Volume 3
Issue 1
Semiannual
June 2025

Journal of Innovations in Computer Science and Engineering

Article

b

In the Name of God

Journal of Innovations in Computer Science and Engineering (JICSE)

Publication Structure

Concessionaire: Shahid Beheshti University
Publisher: University Publications

Publication language: English
Publication Cycle: Quarterly
ISSN: 2981-2135

About Journal

Journal of Innovations in Computer Science and Engineering (JICSE) has established in 2021 in the Faculty of Computer Science and Engineering of Shahid Beheshti University to improve the synergy between researchers and innovators in this field. There is neither processing nor publication charge assumed. In addition, the journal is regularly uploaded to the website, and articles in press are uploaded on the site until the end of the year.

Journal of Innovations in Computer Science and Engineering (JICSE) License Terms

Journal of Innovations in Computer Science and Engineering (JICSE) Perspective is accredited by the Scientific Publications Commission of the Ministry of Science, Research and Technology. The journal has been ranked as **Grade B** in the latest evaluation of science, research and technology.

Language
The journal is published in English.

Publication Frequency
The Journal of Innovations in Computer Science and Engineering (JICSE) is published two quarterly in year.

Policies
1. Open Access Policy
To make freely public accessibility to research with the aim of greater global exchange of knowledge, this journal provides open access to its published articles.
2. Copyright Notice
Authors of the published articles in this journal retain the **Copyright** of their articles and will be able to archive pre-print, post-print, and publisher's versions.
All Articles published in the are licensed under a Creative Commons Attribution 4.0 International License.
3. Archiving
In addition to indexing database, this journal utilizes digital archive to guarantee long-term digital preservation and restoration. Some of them, can be find here.

Subscription
Journal of Innovations in Computer Science and Engineering (JICSE) Perspective is published as an open access electronic journal, so there is no subscription fee for audiences.

Publisher
Shahid Beheshti University, Tehran, Iran.

Sources of Support
Shahid Beheshti University, Tehran, Iran.

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

Malfustection: Obfuscated Malware Detection and Malware Classification with Data Shortage by Combining Semi-Supervised and Contrastive Learning

Mohammad Mahdi Maghouli,
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran, m.maghouli@mail.sbu.ac.ir
Mohamadreza Fereydooni,
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran, m.fereydooni@mail.sbu.ac.ir
Mojtaba Vahidi-Asl, Code Orcid: 0000-0003-4964-992X,
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran, mo_vahidi@sbu.ac.ir
Monireh Abdoos✉, Code Orcid: 0000-0002-3106-503X
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran, m_abdoos@sbu.ac.ir

Abstract

With the advent of new technologies, the use of various digital gadgets in different formats is becoming increasingly widespread. In today's world, where everyday tasks are often made easier by technology, the extensive use of computers creates opportunities for malicious activity. Malware is one of the well-known and widely used means by which malicious attackers conduct destructive activities. Producing malware from scratch is challenging, so attackers often obfuscate existing malware and modify it to make it unrecognizable. Since creating new malware from an old one using obfuscation is a creative task, there are some drawbacks to identifying obfuscated malware. In this research, we propose a solution to overcome this problem by converting the code to an image in the first step and then using a semi-supervised approach combined with contrastive learning. In this case, an obfuscation in the malware bytecode corresponds to an augmentation in the image. Hence, by utilizing meaningful augmentations that simulate obfuscation changes and combine them to generate complex ambiguity procedures, our proposed solution can construct, learn, and detect a wide range of obfuscations. This work addresses two issues: (1) malware classification despite data deficiency, and (2) obfuscated malware detection by training on non-obfuscated malware. According to the results, the proposed method effectively addresses the data shortage problem in malware classification, achieving an accuracy of 90.1% when only 10% of the data is used for training the model. Moreover, training on basic malware without obfuscation achieved 96.21 percent accuracy in detecting obfuscated malware.

KEYWORDS

Malware Classification, obfuscated malware detection, semi-supervised learning, contrastive learning, code to image transformation.

1. INTRODUCTION

Today, the increasing number of malware is one of the most significant challenges to information security and communication networks. Automatic malware detection is crucial for users to identify and mitigate the malicious effects of malware. Machine learning methods have become promising and practical techniques for malware detection [38]. The performance of these methods depends on the amount of data collected for analysis. Although everyone has access to vast volumes of data over the internet, data collection and labeling are usually expensive and challenging in most applications. As a solution to this problem, semi-supervised learning methods can be effective in addressing the shortage of labeled data.

Malware detection methods generally fall into three categories: static analysis, dynamic analysis, and hybrid analysis [45]. Statistical analysis detects malware by analyzing the program's assembly code, converted from its binary file. This technique analyzes Portable Executable (PE) [1] files without running them, utilizing the code's opcodes, control flow graph, function call frequency, and system-level APIs.

This method is affected by malware obfuscation due to the use of features such as Control Flow Graphs or the frequency of calls. On the other hand, converting executable code to assembly code and the uncertainty associated with this conversion increase the complexity of this method [2].

Submit Date: 2024-03-11
Revise Date: 2025-07-21
Accept Date: 2025-08-17
✉ Corresponding author

Dynamic analysis attempts to identify malicious behavior by running the program in a controlled environment, such as a virtual machine or sandbox [42]. This method is time-consuming and requires hardware resources. It may limit the program's behavior in a controlled execution environment due to the lack of access. However, the environment cannot fully track the program's behavior. Despite these limitations, dynamic analysis attempts to detect malware by analyzing a sequence of API calls and system calls, classifying them into common categories, such as Ransomware, Worms, etc.

From the binary executable file, the statistical analysis attempts to find similar bit sequences. By using this method, the program is not disassembled or run. Furthermore, since malicious files often employ the same logic or, in some cases, reuse code or utilize malicious libraries, the code can be classified as malware or benign.

The primary challenge of malware detection is related to obfuscated malware, which is created by obfuscating the binary code of a primary malware or another obfuscated malware. As a result of this obfuscation, the primary function of malware remains unchanged, but its appearance changes. Therefore, it is less likely to be detected.

Code obfuscation techniques include disassembling and reassembling, repackaging, data encoding, and code reordering [3, 4]. The more complex the obfuscation, the harder it becomes for antivirus programs that perform static scans to detect malware.

Another point is that obfuscated malware cannot be categorized in several specific ways, as this process depends on the creativity and innovation of the attackers, such that the appearance of the code changes, but its logic and primary purpose remain the same. Code obfuscation is analogous to image noise. Therefore, if we can disregard the obfuscation of the code, which does not alter the identity and nature of the program, we can always distinguish the malicious or non-malicious core of the software. Utilizing a method to detect and eliminate this noise can lead to higher accuracy in malware detection.

The primary challenge posed by obfuscated malware for antivirus programs is that each malware code can generate thousands of obfuscated codes, and each obfuscated code can generate other obfuscated ones similarly. In this regard, in the next generations, the obtained code is significantly different from the original code, making it impossible for antiviruses to identify new generations just by recognizing the primary ones and saving them in their databases.

To accomplish this, we need a method for learning the basic features of malicious code, regardless of how ambiguous the code may be. Moreover, it can also detect a large number of obfuscated codes generated from the main malware by using only a small portion of the original malware code. By achieving a higher level of abstraction in the implemented code, we can identify what the code is, just as we overlook specific details of a landscape with a general vision. Computer Vision methods have matured dramatically over the last few years, thanks to the help of Deep Neural Networks, and have become indispensable for classification tasks. This is why our fundamental idea is to present codes in the image sphere and utilize these new methods. By applying an extensive array of image augmentations and learning using a particular loss function, contrastive learning can identify the core concept of the samples and discard the details that do not contribute to the image's identity. We can utilize this method to detect the malicious or non-malicious core of the software. Moreover, contrastive learning, introduced in SimCLR [11] as a semi-supervised method, is utilized for malware classification and obfuscated malware detection in this paper.

This paper is organized as follows: Section 2 describes the basic concepts. The literature review and the proposed method are presented in Sections 3 and 4, respectively. The experimental results are presented in Section 5, covering malware detection and obfuscated malware detection. Section 6 concludes the paper.

2. BASIC CONCEPTS

This paper proposes a semi-supervised malware classification and obfuscated malware detection method using contrastive learning on code images. The following sections describe semi-supervised learning and contrastive learning as fundamental concepts in machine learning.

2.1. SEMI-SUPERVISED LEARNING

Semi-supervised learning methods are a class of machine learning techniques that utilize both unlabeled and labeled data simultaneously for data classification. Unlabeled data are usually available for most problems; however, providing labeled samples is typically expensive. Therefore, semi-supervised methods have become a promising approach to this problem.

Semi-supervised learning methods have various implementations and utilization solutions. [6]. One of the most prominent methods is to train a model with a small fraction of labeled data and then utilize vast amounts of unlabeled data to generate pseudo-labels for every input using the trained network. To define the loss function, we can consider the loss between the pseudo-label and the predicted probability obtained by passing the data through the network and fine-tuning it [7].

Semi-supervised learning can leverage self-supervised pre-training on unlabeled data using augmentations. [8]. In this regard, we require a suitable loss function to classify the most similar but unlabeled inputs into the same group. Therefore, we utilize a contrastive loss function (for more details, see Section 2-2). The network will be fine-tuned using a small subset of labeled data. As a result, through a semi-supervised approach, we can train the model's encoder (like CNNs) and projection head in an unsupervised way, then replace some projection head layers with new ones and fine-tune the projection head by using a small fraction of data that has class labels in a supervised way [5, 9].

2.2. CONTRASTIVE LEARNING

In many cases, learning data representation is advantageous, especially if gathering labeled data is costly and we have a large amount of unlabeled data. A solution that can help learn representations and features from the data, and contrastive learning, could be beneficial in these cases. Contrastive learning is effective in various applications, including image classification [10], image enhancement [11], feature extraction on time-series data [12], and natural language processing [13, 14].

In contrastive learning, the most important part is defining the contrastive loss function, which guides the model training in a way that maximizes the difference between diverse input data and minimizes the distance between similar data. Contrastive learning has several applications across various domains, some of which are discussed in [15].

A framework for contrastive learning of visual representations, called SimCLR, has been introduced in [16]. This framework employs a semi-supervised learning approach, utilizing image augmentation for automated labeling of images. When an image is converted to another one, the core concept of the image remains unchanged, and therefore, both images should have the same label when contrastive learning has been used. An encoder network transforms each image into a feature vector.. This framework utilizes ResNet [17] as an encoder [16], as illustrated in Figure 1, which takes an image and generates a 1-hot feature vector as its output, known as an "encoder unit". The size of the feature vector is 2048, indicated by h in Figure 1. These feature vectors are then fed as input to another fully connected neural network, which employs a non-linear function [16].

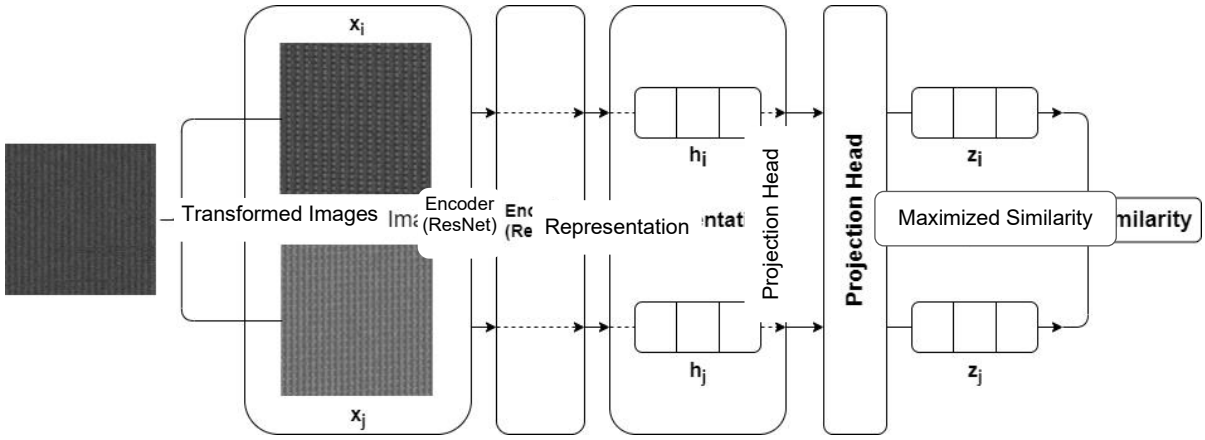


Figure 1. The SimCLR framework [18]

The learning steps are performed in the Projection Head in such a way that the weights of this network are adjusted according to the output vectors. If the inputs are generated from the same image, the similarity is high; otherwise, the similarity is low. This part, which is the main idea of contrastive learning, is shown in Figure 2.

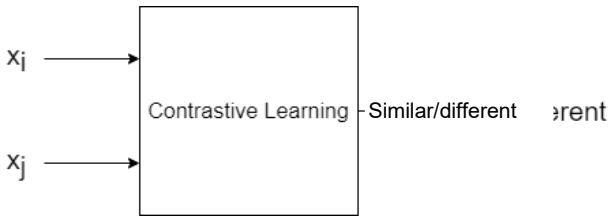


Figure 2. Contrastive learning

It is worth noting that this framework outperforms the state-of-the-art methods for the ImageNet dataset classification. Moreover, this framework can overcome the lack of sufficient images in some applications, as it can generate new data from existing samples. This framework claims that using only 10% of ImageNet data for training can achieve an accuracy of about 80% of the accuracy reached when a large portion of the labeled data is used for training [16].

SimCLR-V2 is an enhanced version of SimCLR based on contrastive learning extended by Google [5]. Network training is performed in three steps within this framework. In the first stage, an unsupervised pre-training with a task-agnostic approach trains an extensive network, regardless of the data labels, in a manner that maximizes data differentiation. Then, it discards half of the projection head and defines a new projection head on top of the trained network. It fine-tunes the trained network using a small amount of labeled data in a supervised manner. To create a lighter model with high accuracy and higher speed, in the third stage, the extensive network obtained is used to train a smaller network, a process known as distillation. The steps of this framework are illustrated in Figure 3, and the pseudo-code of the SimCLR framework is presented in Algorithm 1.

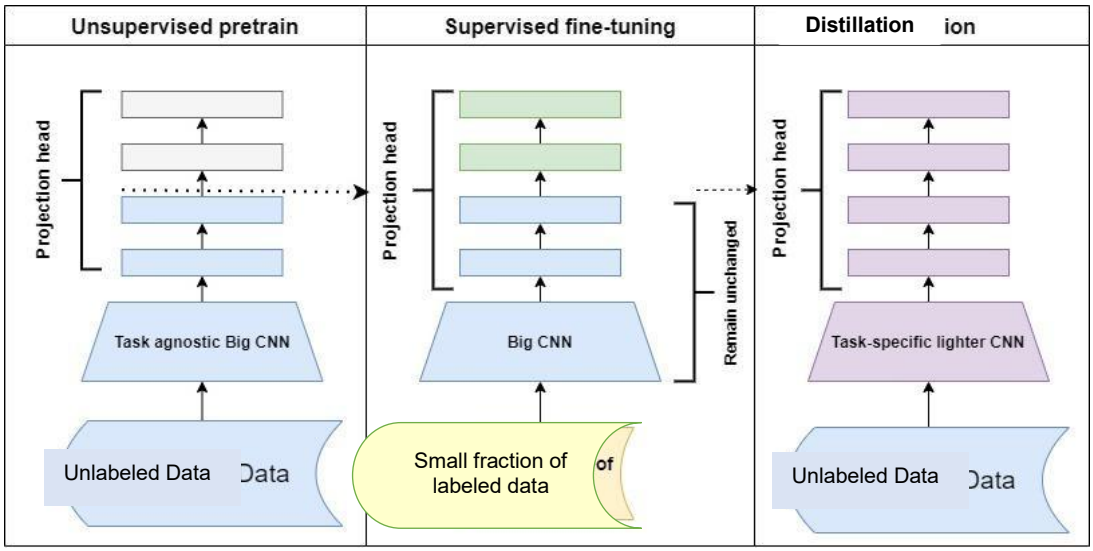


Figure 3. SimCLR-V2 schema [5]

Algorithm 1: SimClr

```
Result: a model to classify instances to predefined classes
create standard TensorFlow dataset with given batch size;
for i ← 1 to epoch_size do
    foreach batch b in AllBatches do
        features =
            make_two_transforms_for_each_data_using_data_augmentation_on_each_image_in_batch();
        hiddenLayersOutput = ResNet(depth, features);
        projectionHeadOutput = model.projection_head(hiddenLayersOutput);
        calculateContrastiveLoss(projectionHeadOutput, b.datas);
        model = tunning_projectionHead_using_constrastiveLoss();
    end
    fine_tunning_fully_connection_layer_using_supervised_data();
end
distilledModel = distilling_model_in_a_task_specific_way(model, dataset);
```

Regardless of the label of the data, the pre-training of the network involves selecting a batch of data, applying two augmentations to each image, and creating two augmented images, as depicted in Figure 4. Consequently, the amount of training data per batch is twice the batch size. Each obtained image is then passed through an encoder, here ResNet101, and its feature vector is extracted.

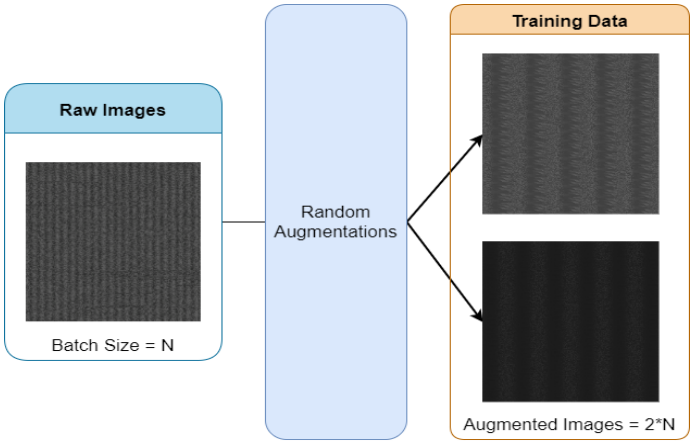


Figure 4. Two augmented samples

Finally, the feature vectors pass through the projection head (fully connected layers), and the final vectors are achieved. Then, in the backpropagation step, a new loss function is defined. The function works in such a way that it tries to bring the image derived from one basic image as close as possible to each other, and the data derived from different photos as far away as possible. The loss function is defined as Eq. (1).

$$l(i,j) = -log\ \frac{exp\ (s_{i,j})}{\sum_{k=1}^{2N} l_{[k\neq i]} exp\ (s_{i,k})} \qquad (1)$$

In Eq. (1), l(i, j) is the estimated loss, and $s_{i,j}$ is the cosine similarity of two instances derived from one image, and $s_{j,k}$ is the cosine similarity of two instances derived from different images [5]. More details on the contrastive loss function can be found in [19].

The main features of this method are:

- Many augmentations are used for training the network.
- High accuracy can be achieved by training only on a small percentage of labeled data.
- Augmentation can enhance the understanding of the model from the malware core of the code and overcome the challenge of code obfuscation. Another challenge in detecting obfuscated malware is the significantly larger volume of obfuscated malware compared to its underlying malware. On the other hand, SimCLR, by defining a new contrastive loss function, offers a semi-supervised method that addresses the challenges of data scarcity and the high cost of collecting large, labeled datasets. This method can create a high-precision model with only a small percentage of labeled data and a large amount of unlabeled data.

3. LITERATURE REVIEW

In this section, the most important works that examine the malware classification problem or malware detection from either an obfuscated or unobfuscated perspective are reviewed to address the role of obfuscated malware utilizing a deep-learning or machine learning approach.

A method for malware classification was introduced by Verma et al. in [20]. They convert malware code into an image and then use first-order and second-order statistical equations to extract features based on the distribution and correlation of different points in a one-channel image. Malimg [21] is a suitable dataset in this regard, which is used in this work, in addition to the dataset gathered from [22]. By manually extracting features and analyzing them statistically, the codes are classified into different malware classes. The results were 98.04% for precision, 98.06% and 98.05% for recall and the F1 measure, respectively. However, they did not use any obfuscated malware. Since the features have not been extracted automatically, this leads to the loss of a large number of discriminative features. The authors extended their work on malware detection to focus on reducing false-negative errors. Some new statistical features have been used to improve the accuracy on a dataset of 10,000 data points, without any obfuscated data [23].

The effect of obfuscation on the accuracy of the machine learning method studied in [2], which utilizes both static and dynamic malware detection. Although several well-known obfuscation methods have been introduced, they have shown that obfuscation has a greater effect on the accuracy of static methods than on dynamic methods [2].

Convolutional Neural Networks (CNNs) have been utilized for malware classification on code that is converted into three-channel images in [24]. New images are generated using different types of noise as augmentation. The results showed 96% accuracy on a dataset gathered from GitHub. In addition to benign data, malware is categorized into various classes. However, obfuscated malware has not been explicitly used in their method, which makes it much more challenging to detect [24]. While traditional CNN-based approaches remain popular, transformer-based architectures [39] have shown superior performance in recent studies.

Obfuscation has been employed to address the issue of data deficiency in [25]. The authors attempted to increase the amount of data by mapping a generated code to an image and applying augmentation to the image. Then, transfer learning has been used on a pre-trained network for malware classification. They achieved 93.8% accuracy on the Microsoft 2015 Dataset [26] and 98.5% accuracy on the Malimg Dataset.

Obfuscated malware detection has also been studied in [27], with a focus on Android applications. It has been shown that some malicious Android applications are re-released by obfuscating the package name and certificate owner name. Therefore, the correlation between these two parameters has been used for the classification. Stacking techniques are used to combine Recurrent Neural Networks (RNNs) and CNNs to determine whether an application is malicious or not based on information acquired from the package name and the certificate owner. It is clear that since the features depend on the operating systems, they cannot be extended to other operating systems such as Linux or Windows.

Miller et al. [28] presented a method for malware detection on Android. Four standard obfuscation methods are employed: the presence of a selection of API calls and Android commands, permissions, and opcode instructions. The Discriminative Adversarial Network (DAN) is used to train the network to distinguish between obfuscated malware and benign samples. The main limitation of this method is that it trains the obfuscation while there are different types of obfuscation by attackers, which are not predictable for the network [28].

In recent years, obfuscated malware detection has been considered in various kinds of literature. This type of malware, derived from other malware, generates a vast amount of malware every day, which is widely used for various purposes, such as ransomware. It is crucial to detect them as metamorphic malware can change from one machine to another. Generative models are widely used for detecting obfuscated malware. After training the network, the probability distribution of the occurrence of any point is estimated by a generative network, and obfuscated malware is detected using this estimated value. Due to the creative nature of obfuscation, generative networks cannot consider all possible obfuscation methods, and therefore, the goal of detecting obfuscated malware cannot be fully achieved.

Chen et al. [29] showed that minor changes to the corresponding images could reduce the performance of deep neural networks. On the other hand, there are some solutions based on generative networks that can be applied to counter these types of attacks [30]. The vulnerability of deep neural networks to adversarial changes has been addressed in [31]. While discussing the types of attacks, it has been demonstrated that malware can bypass detection using this approach.

In [32], an ensemble method called MalNet was introduced for malware detection, achieving 99.36% accuracy on the Microsoft 2015 dataset [26]. MalNet converts binary codes and OPcodes to images and then uses CNN and LSTM for the classification.

A hybrid model composed of ResNet and GoogleNet has been utilized for malware detection in [33], where byte codes are directly converted into images for training. The model achieved 88% accuracy.

Daram et al. proposed an ensemble method for malware classification [34]. Two models were ensembled for the classification of malware on the Microsoft 2015 dataset [26]: in the first one, static features such as file size and n-gram opcodes were extracted and used for the XGBoost model; in the second one, the executable files were converted to images, and the features extracted by a CNN. Since the dataset does not include benign data, it cannot be used as a practical solution for malware and benign detection. A summary of related works is given in Table 1.

Recent advancements by Gao et al. [44] have further improved the SimCLR framework specifically for cybersecurity applications, achieving better performance with fewer labeled samples.

Table 1. Summary of related works

Authors	Approach	Method	Accuracy	Dataset
N. Marastoni et al. 2021 [25]	Malware classification	LSTM + CNN	93.8 %	Microsoft 2015
			98.5%	Malimg
A. Deram et al. (2021). [34]	Malware classification	Ensemble	99.12%	Microsoft 2015
Khan et al. 2018 [33]	Malware classification	ResNet 101	78.84%	Microsoft 2015
		ResNet 152	88.36%	
Yan et al. 2018 [32]	Malware / benign detection	MalNet	99.13%	Microsoft 2015 + collected benign samples
F. Catak et al. 2020 [24]	Malware classification	CNN	96%	Mal-API-2019
V. Verma et al. 2020 [23]	Malware / benign detection	Texture statistics	99.61%	Custom dataset
W. Lee et al. 2019 [27]	Malware classification	CNN-RNN-PA	99.86%	VirusTotal
V. Verma et al. 2020 [20]	Malware classification	CNN	98.58%	Malimg

While our approach leverages well-established frameworks such as SimCLR and ResNet, its main novelty lies in the novel application of contrastive learning to the domain of binary code classification through image-based representations. Specifically, we introduce a tailored data preprocessing strategy that transforms binary code into visual formats suitable for deep contrastive learning, enabling effective feature extraction and classification. Additionally, our method demonstrates significant improvements in detection performance metrics compared to traditional binary analysis approaches, highlighting its practical value and contribution to advancing malware detection techniques.

4. THE PROPOSED METHOD

As mentioned earlier, to overcome the challenge of detecting obfuscated malware, it is essential to disregard the noise and focus on the malicious core of the code. For this purpose, we use a semi-supervised learning method described in Section 2-1. Our approach addresses advanced obfuscation techniques that resemble adversarial attacks [40], ensuring robust detection capabilities. In this regard, the code should be mapped to an image to create a higher-level abstraction of the binary codes, i.e., minor changes do not significantly affect the overall view of the image. Therefore, the code-to-image conversion implicitly overcomes minor obfuscations. Image augmentation is typically used to enhance the identity of images in semi-supervised methods. Augmentation also helps generate new samples from an image to address the data scarcity problem. Each method of augmentation can be considered as an obfuscation approach. For example, a vertical flip corresponds to the obfuscation of malware, which is created by rearranging code parts or reordering them in a way that does not affect the main program's logic. Even a few degrees of blur can also be considered as changing some details without altering the general concept of the image, which could be analogous to adding or removing unnecessary lines of code from the malware. The creation of new augmented samples is generally performed, particularly for classes where collecting data is challenging. Alongside a semi-supervised method, we use a contrastive learning method for data clustering. As explained in Section 2, this method categorizes samples by differentiating and contrasting instances of different classes, while maximizing the similarity between instances of the same class. As a result, the proposed method combines contrastive learning and semi-supervised learning, generating new samples through augmentation and classifying them as described.

The general framework of our proposed method is shown in Figure 5. First, two sets of benign ware and malware are collected. Benign ware data is collected from Windows system files and cleaned so that it resembles malware format. The binary codes of the Windows operating system are selected because they contain API calls and low-level system calls, as well as high-level permissions, which make them difficult for malware to detect.

Malware data is collected from the Microsoft 2015 dataset [26], which is presented in nine classes, as detailed in Section 5. One of them is called Obfuscator.ACY contains only obfuscated malware, and the other classes include primary malware in each class (malware with no obfuscation). This data is then converted to grayscale images so that every 8 bits corresponds to a value of intensity in pixels. Images are generated at a fixed width of 1024 pixels and a variable length depending on the size of the binary file. However, the images are resized and randomly cropped in augmentation at the beginning of the process.

In this research, two main problems have been addressed to show the efficiency of the proposed method. The first problem is malware classification, and the second is detecting obfuscated malware from benign programs, known as obfuscated malware

detection. The solution to the first problem focuses on the lack of data, as training was conducted with only 20% and 10% of the data, and testing was performed on the remaining 80% and 90%, respectively, as described in the next section.

4.1. CONVERTING CODES TO IMAGES

In this method, the software's bytecode is converted into an image. The bytecode is reshaped into a binary image as described in [30], i.e., a two-dimensional array with a size of 1024 × h, where h depends on the length of the code. Figure 6 illustrates an executable bytecode sample, and Figure 7 shows an example of an image obtained from a bytecode as described in Algorithm 2.

4.2. MALWARE CLASSIFICATION

First, an extensive network is trained with a task-agnostic approach in an unsupervised way. In this regard, in each iteration, a batch of images is selected, and two augmentation methods are applied to each image. These augmented images are then passed through the encoder network for feature extraction. As shown in the figure, ResNet-101 is used as the encoder.

According to the results reported in [5], when a problem involves a data shortage, utilizing deeper networks along with larger batch sizes can lead to a better outcome. Therefore, ResNet200 is used to train the network with 20% of the data and to evaluate the remaining 80% in the first experiment, and with 10% of the data for training and 90% for evaluation in the second experiment, as depicted in Figure 8. The generated feature vectors from the encoder unit then pass through the projection head, where the projection head nodes are tuned during the training phase. In contrastive learning, using the loss function as defined in Eq. (1), errors between similar images are minimized while errors between different images are maximized. In the next stage, a part of the obtained projection head is discarded and replaced by a new projection head. Using a small part of the data, the network is fine-tuned in a supervised way. Therefore, in a task-specific approach, a smaller network is trained with a larger network to produce a lighter, faster, and reasonably accurate model.

4.1. OBFUSCATED MALWARE DETECTION

To detect obfuscated malware, as shown in Figure 5, the data is prepared in the dataset preparation module, and then the data is divided into two parts using a train/test splitter:

1. The train set, which contains 80% of the whole benignware and all the basic malware (without any obfuscation)
2. The test set, which contains the remaining 20% of benign wares and the whole Obfuscator.ACY class in the Microsoft 2015 dataset, which contains only obfuscated malware, as illustrated in Figure 9.

For malware data, all the data in the Microsoft 2015 dataset is used for training the network, except for the Obfuscator.ACY, which is used for the test phase. Therefore, the model is tested with a category of data that has never been seen before, namely, obfuscated malware. We demonstrate that we can predict many obfuscated malware samples from benign code through our experiments. The obfuscation in code can be seen as a form of augmentation in an image, allowing our network to learn how to ignore obfuscation in corresponding images of code and determine whether the code is malicious or not. For generalization, the SimCLR framework is used to generate augmented images from the base images in the training set. In contrastive learning, the purpose of the training method is to reduce the difference between the augmented images from the same source. In this regard, the network learns to ignore augmentation in images, which acts as obfuscation in the corresponding code. According to Figure 4, the SimCLR framework generates two augmented images from each image in the batch during the training module. To accomplish this, we first resize the shorter side of the image to 224 pixels and resize the other side proportionally to preserve the image's aspect ratio. Then, 224×224 random crops are generated, and an augmentation is randomly selected and applied. As a result, two output images with different combinations of augmentations are generated from each input image. The desired augmentation includes random flip, color distortions, reshaping, and blurring. Figure 10 shows an example of augmentation.

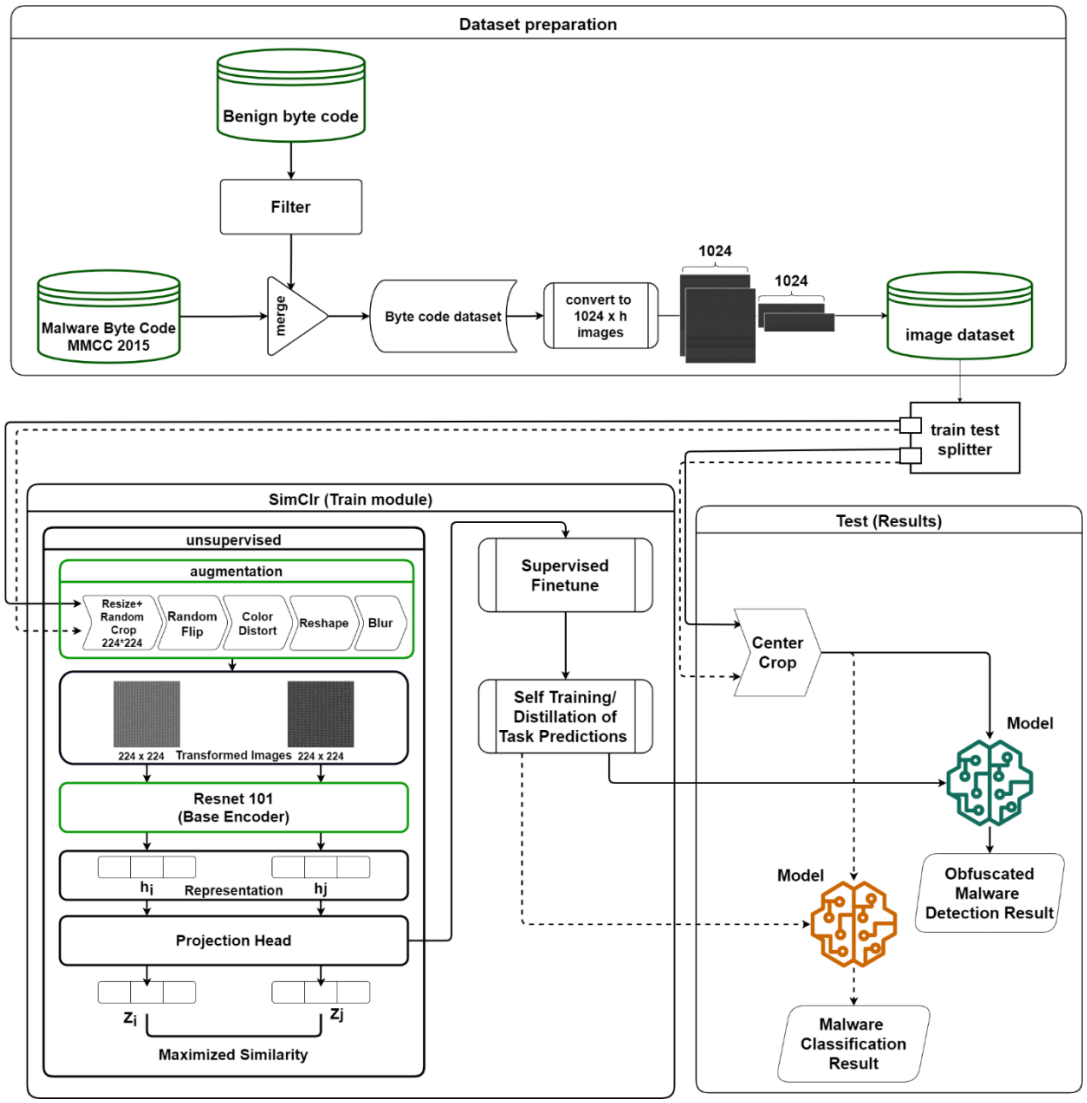


Figure 5. Illustration of the proposed method

```
00401000 56 8D 44 24 08 50 8B F1 E8 1C 1B 00 00 C7 06 08
00401010 BB 42 00 8B C6 5E C2 04 00 CC CC CC CC CC CC CC
00401020 C7 01 08 BB 42 00 E9 26 1C 00 00 CC CC CC CC CC
00401030 56 8B F1 C7 06 08 BB 42 00 E8 13 1C 00 00 F6 44
...
```

Figure 6. A malware bytecode sample

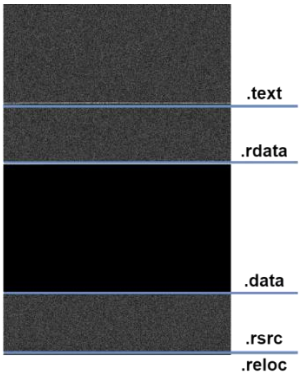


Figure 7. Result of converting a code to an image [35]

Algorithm 2: Binary Code to Image Convertor

Result: 1024 × h PNG images
prepare binary code files in dataset;
foreach *file f in bytecodesDataset* **do**
    bytes = f.readBytes();
    width = 1024;
    height = len(bytes)/width;
    image = bytes.reshape(width, height);
    saveBytesAsPNG(image);
end

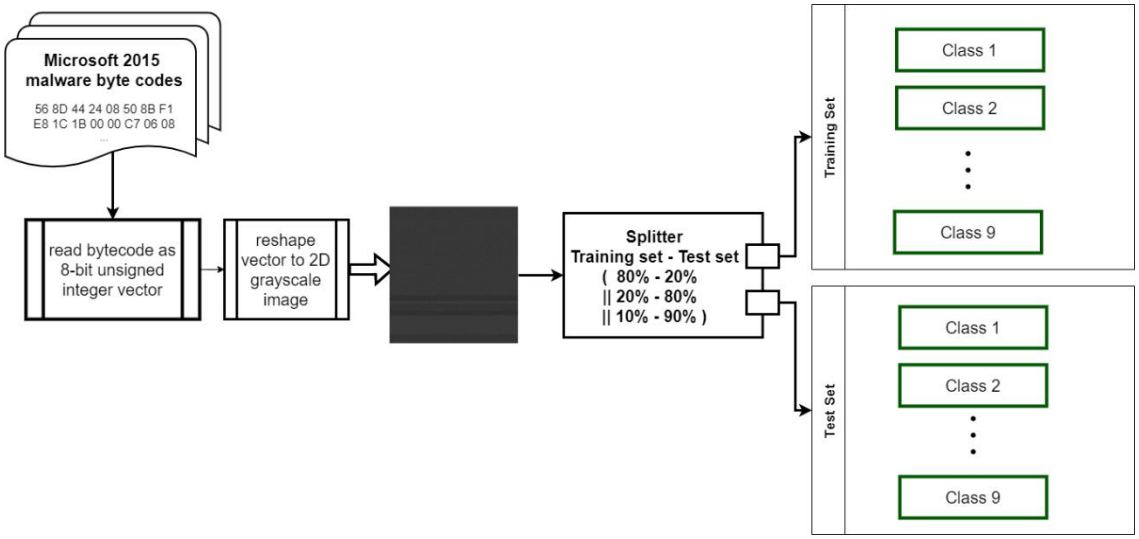


Figure 8. Illustration of the train/test splitter module in the malware classification

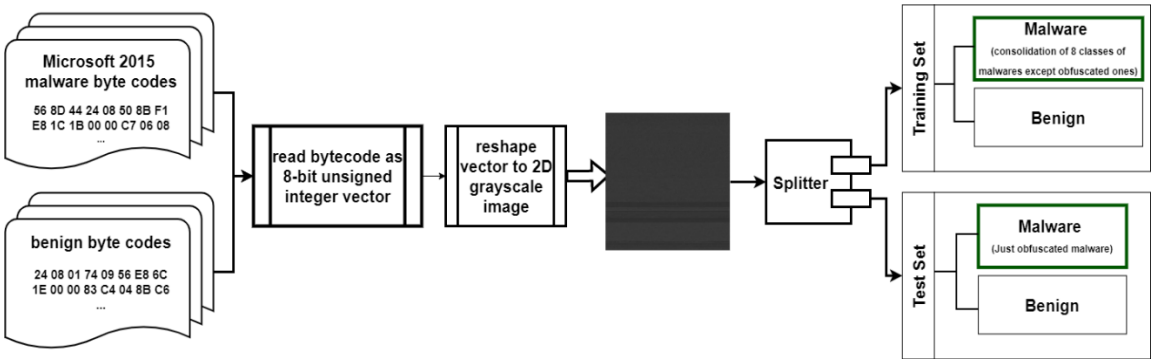
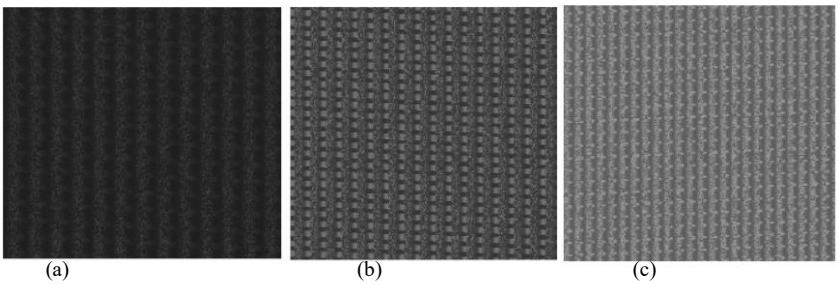


Figure 9. Illustration of the train/test splitter module in the obfuscated malware detection problem



(a) (b) (c)

Figure 10. a) Transferred image from source code; b) Code image after random crop and color distortion; c) Code image after performing random crop and blur.

The augmented images are then used, and the corresponding original images are discarded. These images are given to an encoder to create feature vectors from the images. ResNet101 is used for extracting a 2048-dimensional feature vector. The feature vectors are fed into a fully connected network in the projection head to train the network's weights. The trained network is tested on a dataset that includes obfuscated malware and benign software. Data preprocessing is applied to test images by resizing them to 250 pixels on the shorter edge, while maintaining the ratio, and then cropping them to 224×224 as part of the test phase.

Analysis- Converting code into images allows leveraging the power of deep learning models designed for visual data, such as CNNs. This transformation simplifies complex code structures into recognizable visual patterns, enabling more effective feature extraction, improving classification accuracy, and benefiting from transfer learning with pre-trained models. While analyzing binary code directly is possible, the visual approach offers a more efficient and robust framework, particularly for tasks like

malware detection and code analysis [46]. Recent work on explainable AI by Nguyen et al. [43] demonstrates that image-based representations can provide more interpretable detection results compared to traditional binary analysis methods, while maintaining high accuracy.

5. EXPERIMENTAL RESULTS

In this section, the dataset used in the experimental results and evaluation metrics is described. The results are presented in two parts: the findings on malware detection and the results for detecting obfuscated malware.

5.1. DATASET

The Microsoft Malware Classification Challenge dataset (Microsoft 2015) [26] has been used to evaluate the proposed methods. This dataset includes the bytecode of malware classified into nine malware classes, used for a contest held by Microsoft on Kaggle in 2015 [36]. The classes are not balanced in terms of data distribution, making the classification task more challenging. The number of samples in each class varies from 42 to 2942 samples. This dataset has been widely used in [25, 32-34] and has also been used to address challenges. The details of the samples in the dataset are given in Table 2.

Table 2. Malware classes in Microsoft 2015 Dataset [26]

Class Name	Number of training samples	Type
Ramnit	1541	Worm
Lolipop	2478	Adware
Kelihos_ver3	2942	Backdoor
Vundo	475	Trojan
Simda	42	Backdoor
Tracur	751	TrojanDownloader
Kelihos_ver1	398	Backdoor
Obfuscator. ACY	1228	Any obfuscated malware
Gatak	1013	Backdoor

Our training set for obfuscated malware detection includes samples from all classes except the Obfuscator class.ACY, which is used only for the test set. In this way, the network is trained using malware samples without obfuscation to learn the primary features of each sample.

Additionally, due to the lack of proper, standardized benign datasets, Windows system files are used as benign codes because they invoke low-level system APIs, just as malware does. Therefore, they are more difficult to distinguish from malware, which can be used to evaluate the proposed method. For this purpose, the data are first collected and then cleaned in a manner that ensures their structure is similar to that of the data in the malware class. To preserve the structure of these files similar to the Microsoft 2015 dataset, only 32-bit programs were used, and only the main parts of the file were preserved. Also, the binary code related to PE Headers was removed from the beginning of the files. These processes were performed using the "pefile" library [37].

After code preparation, each bytecode is mapped to the brightness of a pixel, resulting in a 1-channel grayscale image with a fixed width of 1024 pixels and a variable height depending on the code size. Using one-channel images has the advantage of allowing for analysis with less data.

5.2. EVALUATION METRICS

Eqs (2)-(7) are applied to each algorithm as performance measures to achieve an efficiency comparison of the proposed model. T, F, P, and N stand for True, False, Positive, and Negative, respectively. In Eq. (7), β is a positive real factor.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

$$TPR = \frac{TP}{TP + FN} \quad (3)$$

$$FPR = \frac{FP}{FP + TN} \quad (4)$$

$$Precision = \frac{TP}{TP + FP} \quad (5)$$

$$Recall = \frac{TP}{TP + FN} \quad (6)$$

$$F - Score = \frac{1 + \beta^2 * Recall * Precision}{\beta^2 * (Recall + Precision)} \quad (7)$$

5.3. RESULTS ON MALWARE DETECTION

As mentioned earlier, the proposed method is being evaluated using the Microsoft 2015 dataset [14], which includes nine different classes of malware. To demonstrate the performance of the proposed method, three experiments were conducted. In the first experiment, 80% of the samples from each class were included in the training set, while the remaining 20% were used as test data. In the second experiment, 20% of the samples were used for training, and the remaining 80% were used as a test set. In the third experiment, 10% and 90% of the samples were used in the training set and test set, respectively. The number of samples used for training and testing the network is given in Table 3. The hyperparameter values used in the training phase of the three experiments are presented in Table 4. For resource-constrained environments, recent work by Sharma et al. [47] shows that knowledge distillation techniques can effectively reduce model size while maintaining detection accuracy, a direction worth exploring in future implementations.

5.3.1. THE FIRST EXPERIMENT

The first experiment performed network training using 80% of the data, while the other 20% used for the test phase. The values of the hyperparameters used in this experiment are reported in Table 4. The accuracy and loss values during the training phase are shown in Figure 11.

The accuracy of the model is 94.11% in the first experiment, while the accuracies achieved by transfer learning and LSTM [20] are 90.8% and 93.8%, respectively.

Table 3. Distribution of data for each class of malware in each experiment

Malware class	experiment	Training samples	Test samples
Ramnit	1st experiment	613	148
	2nd experiment	148	613
	3rd experiment	77	684
Lollipop	1st experiment	1895	471
	2nd experiment	471	1895
	3rd experiment	281	2085
kelihos_ver3	1st experiment	2354	588
	2nd experiment	588	2354
	3rd experiment	294	2648
vundo	1st experiment	112	34
	2nd experiment	34	112
	3rd experiment	13	133
simda	1st experiment	31	8
	2nd experiment	8	31
	3rd experiment	9	30
Tracur	1st experiment	460	96
	2nd experiment	96	460
	3rd experiment	72	484
Kelihos_ver1	1st experiment	57	18
	2nd experiment	18	57
	3rd experiment	7	68
Obfuscator.ACY	1st experiment	93	17
	2nd experiment	17	93
	3rd experiment	40	70
Getak	1st experiment	554	96
	2nd experiment	96	554
	3rd experiment	172	478

Table 4. The network hyperparameter values in each experiment

	Number of epochs	Batch size	Encoder	Initial learning rate	Temperature	Weight decay	Training data dimensions	Augmentation used?
1st experiment	400	64	ResNet101	1.0	0.5	1e-4	224*224	YES
2nd experiment	1000	32	ResNet200	1.0	0.5	1e-4	224*224	YES
3rd experiment	1000	32	ResNet200	1.0	0.5	1e-4	224*224	YES

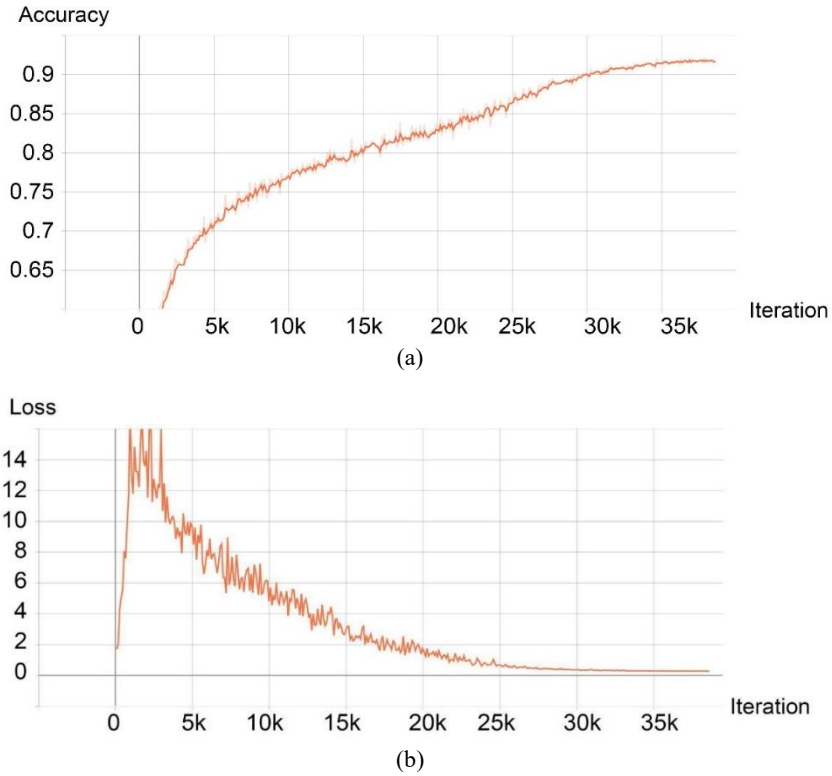


Figure 11. a) Accuracy during training the model, and b) loss value during the training of the model in the first experiment

In [34], the accuracy using CNN on all data is 98%, while it is 99.12% using an ensemble method. However, the accuracy of our proposed method in the training phase is higher than the method presented in [34]. The training phase has been done on a large portion of the data. The accuracy of the method on the test phase has been reported as 98% using CNN and 99.12% using the ensemble method. Although the accuracy of their method is higher than our proposed method in the test phase, our approach achieves the same accuracy as their method using CNN in training, which demonstrates the effectiveness of our proposed method when a large portion of the data is utilized. The obtained confusion matrix and heat map are presented in Tables 5 and 6.

5.3.2. THE SECOND EXPERIMENT

As described earlier, to demonstrate the effectiveness of the proposed method, a small fraction of labeled data is used for training the model. In the second experiment, 20% of the data is used for training. Therefore, a larger batch size and greater depth of the network are needed to achieve better results. The number of epochs used in this experiment is

1000, while the batch size is 32, and ResNet200 has been used as the encoder. Using 20% of the data for training, the proposed method achieved 91.95% accuracy on the remaining 80% of unseen data. The accuracy and loss values during the training phase in this experiment are shown in Figure 12. Additionally, the confusion matrix and heat map are presented in Tables 7 and 8, respectively.

5.3.3. THE THIRD EXPERIMENT

In the third experiment, 10% of the labeled data was used for training the network. The network parameters are shown in Table 4. The accuracy of the proposed method, using only a small number of labeled data, is 90.1%. It means that by using a small fraction of the data (10%), the proposed method achieves 97.75% of the best accuracy of the method presented in [25], which uses 80% of the data for training, and 92.42% of the accuracy reported in [34] on the entire dataset. This result demonstrates that utilizing contrastive learning and SimCLR yields exceptionally high detection accuracy with a limited amount of data.

Given that a relatively small amount of labeled malware data is available, this experiment demonstrates that the proposed method is effective for malware detection not only when a small dataset is available, but also when a larger dataset is available. Since the number of labeled data used for training is limited, contrastive learning provides a faster and lower-cost method (in terms of data collection) for malware detection. The obtained results are illustrated in Tables 9 and 10. Moreover, the final results for each experiment and metric are presented in Table 11, and a comparison among these three experiments is provided in Table 12.

5.4. RESULTS ON OBFUSCATED MALWARE DETECTION

As described in Section 4-2-1, the bytecode of the malware and the benign files are each directly converted into an image, and then the deep network is trained on the basic malware and the benign using the SimCLR-V2 framework.

The two main problems in obfuscated malware detection are as follows:

1. Excessive obfuscation in the code in such a way that its functionality remains the same, but its appearance is entirely different.
2. A large number of obfuscated malware compared to the basic ones.

A deep convolutional neural network (DCNN) can be effectively used to address these two challenges.

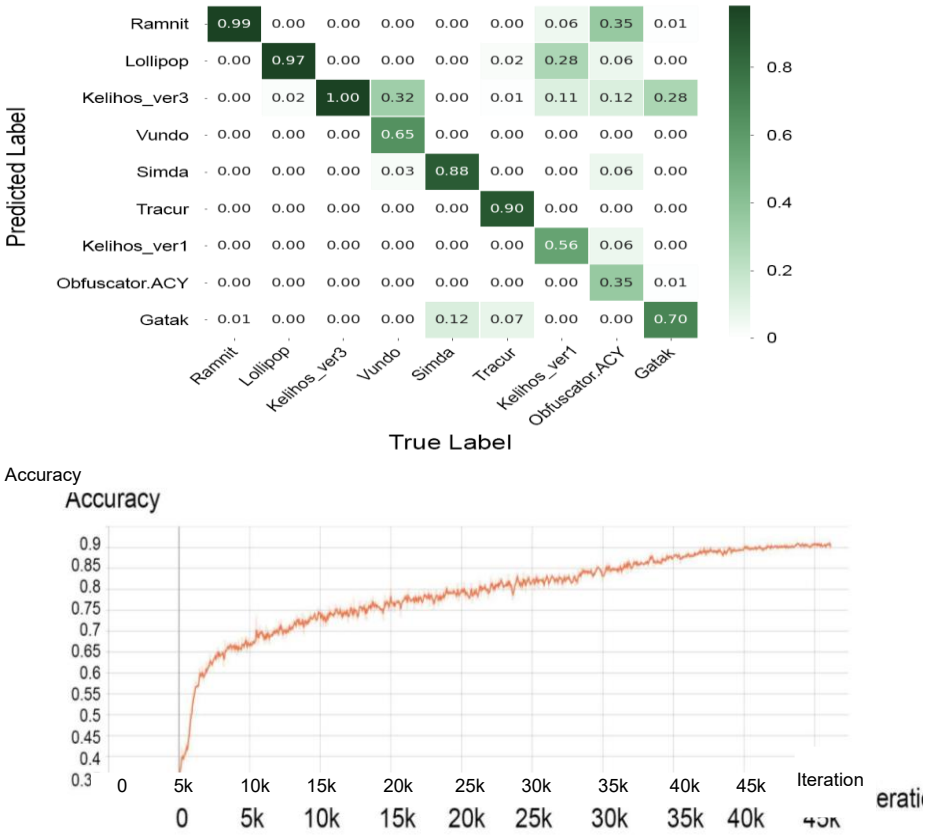
Table 5. Reported confusion matrix for first experiment (columns: True labels, rows: Predicted labels).

	1	2	3	4	5	6	7	8	9
1	147	1	0	0	0	0	1	6	1
2	0	458	0	0	0	2	5	1	0
3	0	8	586	11	0	1	2	2	27
4	0	0	0	22	0	0	0	0	0
5	0	0	0	1	7	0	0	1	0
6	0	2	0	0	0	86	0	0	0
7	0	0	0	0	0	0	10	1	0
8	0	1	0	0	0	0	0	6	1
9	1	1	2	0	1	7	0	0	67

5.4.1. RESNET WITHOUT AUGMENTATION

In this method, the dataset is divided into three main parts: the training set, the validation set, and the test set. Cross-validation has been employed to evaluate the model. ResNet101 was used in this model on images without augmentation. For training the network, the training data were divided into five equal parts. In each round, four parts were used for training, while the remaining part was used for validation. The images were 224x224 for training, and the best model was selected based on the accuracy achieved during validation. The test data includes unseen benign and obfuscated malware files, which achieved accuracies of 96.27% and 69.18% on the validation and test sets, respectively.

Table 6. Reported heat map for the first experiment, the numbers indicate the recall rate (columns: accurate labels, rows: predicted labels).



(a)

Loss

0 5k 10k 15k 20k 25k 30k 35k 40k 45k Iteration

(b)

Figure 12. a) Accuracy and b) Loss value during the training of the model in the second experiment

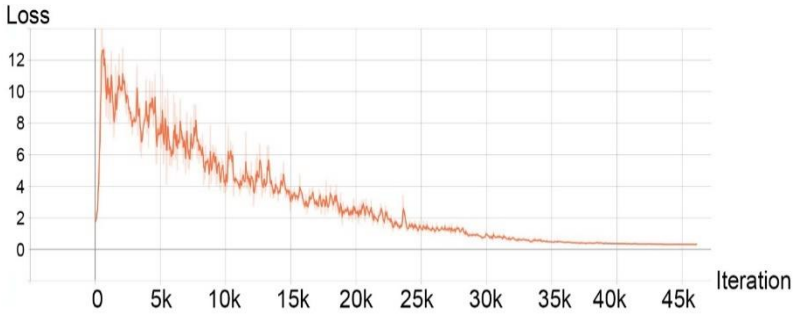


Table 7. Reported confusion matrix for the second experiment (columns: True labels, rows: Predicted labels).

	1	2	3	4	5	6	7	8	9
1	575	7	0	0	1	5	3	35	3
2	21	1843	1	0	0	4	11	13	16
3	1	26	2342	65	3	155	7	8	7
4	0	0	0	46	1	1	0	2	0
5	1	1	0	0	23	0	0	0	1
6	3	7	3	1	3	287	1	9	26
7	0	1	2	0	0	1	34	1	0
8	6	0	0	0	0	2	1	21	0
9	5	10	6	0	0	5	0	4	501

Table 8. Reported heat map for the second experiment (the numbers indicate the recall rate): (columns: True labels, rows: Predicted labels).

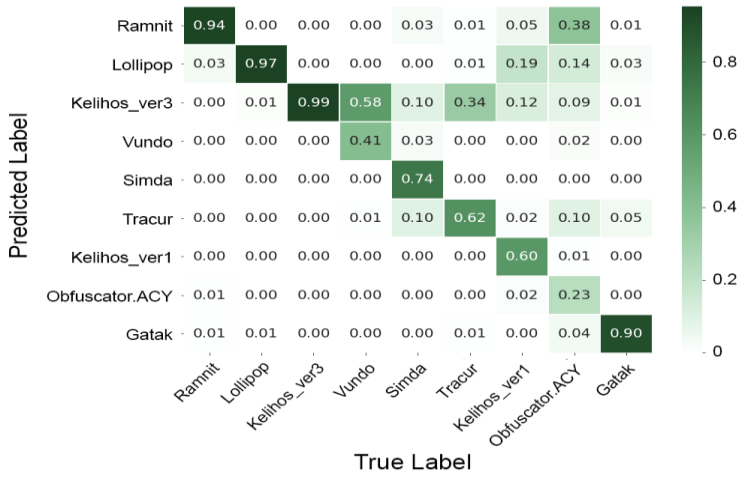


Table 9. Reported confusion matrix for the third experiment (columns: True labels, rows: Predicted labels).

	1	2	3	4	5	6	7	8	9
1	635	7	0	0	0	11	4	28	9
2	15	1990	3	0	0	1	17	3	9
3	2	59	2627	94	2	209	14	9	25
4	0	0	0	36	0	0	0	1	0
5	1	5	0	1	22	7	0	1	1
6	9	14	13	2	4	247	1	4	22
7	1	0	1	0	0	1	32	3	1
8	14	1	3	0	2	5	0	14	2
9	7	9	1	0	0	3	0	7	409
total	684	2085	2648	133	30	484	68	70	478

Table 10. Reported heat map for the third experiment (the numbers indicate the recall rate), (columns: True labels, rows: Predicted labels).

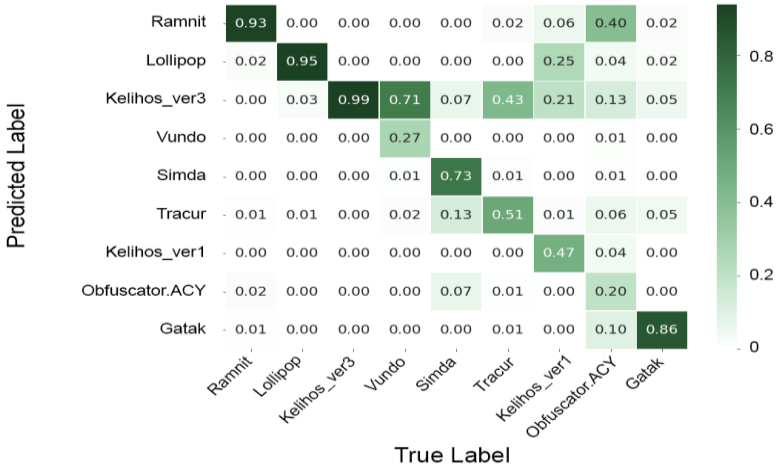


Table 11. Test phase comparative metrics results for all three experiments

Class	Experiment	N(truth)	N (classified)	Accuracy (%)	Precision	Recall	F1-score
Ramnit	1st	148	156	99.32	0.94	0.99	0.97
	2nd	612	629	98.52	0.91	0.94	0.93
	3rd	684	694	98.38	0.91	0.93	0.92
Lollipop	1st	471	466	98.58	0.98	0.97	0.98
	2nd	1895	1909	98.09	0.97	0.97	0.97
	3rd	2085	2038	97.86	0.98	0.95	0.97
Kelihos_ver3	1st	588	637	96.41	0.92	1	0.96
	2nd	2354	2614	95.4	0.90	0.99	0.94
	3rd	2648	3041	93.49	0.86	0.99	0.92
Vundo	1st	34	22	99.19	1	0.65	0.79
	2nd	112	50	98.87	0.92	0.41	0.57
	3rd	133	37	98.53	0.97	0.27	0.42
Simda	1st	8	9	99.8	0.78	0.88	0.82
	2nd	31	26	99.82	0.88	0.74	0.81
	3rd	30	38	99.64	0.58	0.73	0.65
Tracur	1st	96	88	99.19	0.98	0.90	0.93
	2nd	460	340	96.34	0.84	0.62	0.72
	3rd	484	316	95.42	0.78	0.51	0.62
Kelihos_ver1	1st	18	11	99.39	0.91	0.56	0.69
	2nd	57	39	99.55	0.87	0.60	0.71
	3rd	68	39	99.36	0.82	0.47	0.60
Obfuscator.ACY	1st	17	8	99.12	0.75	0.35	0.48
	2nd	93	30	98.69	0.7	0.23	0.34
	3rd	70	41	98.76	0.34	0.20	0.25
Getak	1st	96	79	97.22	0.85	0.70	0.77
	2nd	554	531	98.65	0.94	0.90	0.92
	3rd	478	436	98.56	0.94	0.86	0.89

Table 12. Comparison of our three approaches with other methods

Method	The percentage of data used for training	The percentage of data used for the test	Accuracy
Transfer learning [18]	80%	20%	93.8%
Ensemble method [28]	-	100%	99.12%
SimCLR-V2 (1st experiment)	80%	20%	94.11%
SimCLR-V2 (2nd experiment)	20%	80%	91.95%
SimCLR-V2 (3rd experiment)	10%	90%	90.1%

5.4.2. RESNET WITH AUGMENTATION

This experiment was also performed based on the same data as described in Section 2-1, except that in each iteration, Image augmentations were applied to investigate if they function similarly to code obfuscation. The goal is for the model to gain a better understanding of the code image by utilizing augmentation in order to learn the malicious core of the data.

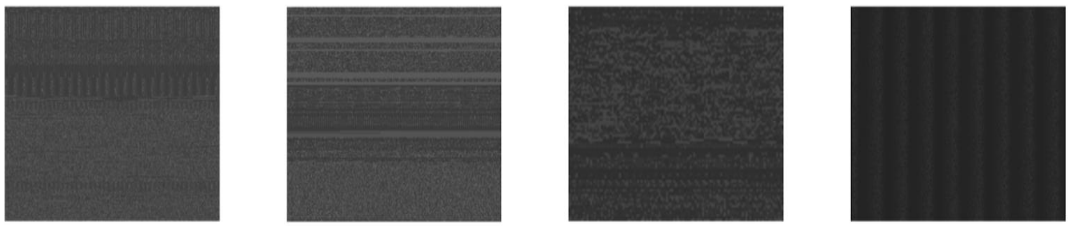
The augmentation techniques applied—such as random resize crop, vertical flip, blur, and color jitter—are motivated by their ability to simulate various obfuscation strategies encountered in malware, including code reordering, redundant code insertion, and appearance modification. These transformations are inspired by the analogy between image noise and code obfuscation: minor modifications that do not impact the core behavior are modeled through these augmentations. For example, vertical flips mimic code rearrangements, while blurring introduces noise akin to redundant or non-essential code. Color jitter and resize cropping simulate changes in appearance that malware might employ to evade static detection. Our experimental results demonstrate that these augmentations significantly improve the model's robustness, enabling it to generalize better against complex obfuscation techniques, as evidenced by the increased detection accuracy in our tests.

In general, standard methods of code obfuscation can be divided into the following categories:

1- Methods that lead to changes in control flow, such as code reordering, splitting functions into two separate functions, and adding switch-case structures for selecting the next basic block in the program's control flow (Known as Flatten), etc.
2- Methods that add or remove redundant codes, such as adding and assigning useless variables, inserting useless functions without calling them, or adding useless conditions or loops that do not change the logic of the code.
3- Methods that modify the appearance of the code while preserving its logic. For example, changing a computational expression in such a way that the output of the new expression is the same as the previous one for similar inputs, but performs different operations.

According to the different obfuscation methods, a set of image augmentations is selected, including random resize crop, vertical flip for the first category, blur, and color jitter for the second and third categories, respectively. Figure 13 illustrates the various types of image augmentation.

- Random resize crop and vertical flip: This augmentation helps to learn different parts of the image separately. In the final model, the displacement of the image elements has no significant effect on the model's output. As a result, it can overcome the problems caused by changes in the control flow.
- Image blurring increases the noise in the image, causing slight changes that do not alter the image's concept. Although some parts of the image may be distorted, it retains the core and concept of the image. This resembles changes from the second category of obfuscation.
- Color jitter includes changing the brightness, contrast, and saturation of the image. Applying these modifications to the image results in a slight change in its appearance, but the image's identity remains unchanged. This kind of augmentation is a good alternative for the third category.



(a) (b) (c) (d)

Figure 13. Different types of augmentations applied to an image: a) Original, b) After color Jitter (including changes in brightness, saturation, and contrast), c) Blurred, d) Random resize cropped

These augmentations are applied randomly to all images in each iteration, and then the model is trained using the new augmented images. The only difference between this approach and the one elaborated in Section 5-4-1 is that the original dimensions of the images are used for training. Since all augmentations apply to the image, a 224x224 random resize crop image is created for network training.

The results show 97% accuracy on the validation set and 89.23% on the test set. It shows that applying augmentation on images helps recognize obfuscated malware from benign code. The results demonstrate a 20% improvement in the accuracy of

obfuscated malware detection compared to the previous approach. According to the results, the hypothesis presented in this paper is accurate, and image augmentation helps detect obfuscated malware, proposing a suitable solution to one of the primary challenges in malware detection. Algorithm 3 shows the pseudo-code of the image augmentation methods.

Algorithm 3: Image Augmentation on Batch

```
Result: augmented image
select a batch from dataset;
foreach image im in batchOfImages do
    im = randomSelectionCropWithResize(im, 224, 224);
    p = randomFloat(0,1);
    if p < 0.5 then
        im = randomFlip(im);
    end
    p = randomFloat(0,1);
    if p < 0.8 then
        im = randomColorJitter(im);
    end
    im = reshape(im, channels = [224, 224, 3]);
    p = randomFloat(0,1);
    if p < 0.5 then
        im = randomBlur(im);
    end
end
```

5.4.3. SIMCLR

The same experiments for obfuscated malware detection were performed using the SimCLR framework. ResNet101 has been used for the network encoder. The batch size and number of epochs used for training are 32 and 140, respectively. The test set includes only the obfuscated malware and benign code. The method's accuracy is 98% and 96% on the training set and test set, respectively.

In contrast, the ResNet with the augmentation method, as described in Section 5-4-2, achieved an accuracy of 89.23% in the test evaluation. The results show a significant improvement compared with ResNet101, with an accuracy of 8%. It confirms that applying SimCLR with a higher number of augmentations and utilizing contrastive learning leads to a deep understanding of the malicious core of the code. It can detect obfuscated malware generated from basic malware, as well as ones that will be produced in the future, with an even higher level of abstraction and accuracy. The hyperparameter values used in the experiment are given in Table 13. Additionally, the obtained confusion matrix and heat map are presented in Tables 14 and 15, respectively.

Table 13. Hyperparameters used in the experiments

	#Epoch	Batch Size	Encoder	Initial Learning Rate	Temperature	Weight Decay	Training Data Dimensions	Augmentation Used?
ResNet	240	32	ResNet101	1.0	0.5	1e-4	224*224	NO
ResNet + Augmentation	240	32	ResNet101	1.0	0.5	1e-4	224*224	YES
SimCLR	140	32	ResNet101	1.0	0.5	1e-4	250*250	YES

Table 14. Reported confusion matrix for SimCLR obfuscated malware detection

	malware (true)	benign(true)
malware(predicted)	1187	51
benign (predicted)	41	1148
total	1228	1199

Table 15. Reported heat map for the obfuscated malware detection experiment (the numbers indicate the recall rate), (columns: True labels, rows: Predicted labels).

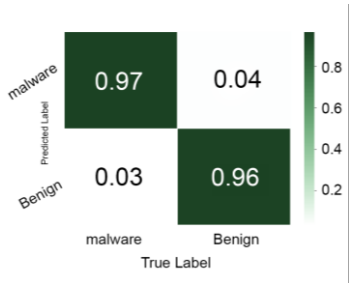
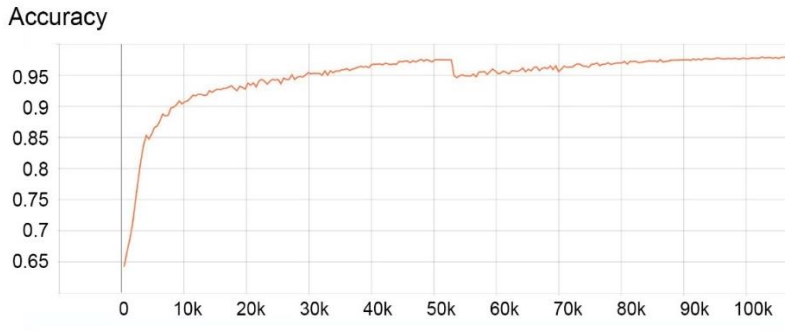


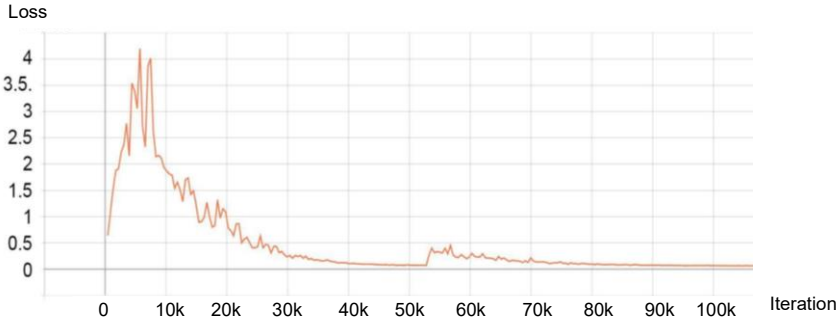
Table 16. Results for the test set

Class label	N(truth)	N(classified)	Accuracy	Precision	Recall	F1-score
Malware	1228	1238	96.21%	0.96%	0.97%	0.96%
Benign	1199	1189	96.21%	0.97%	0.96%	0.96%

Accuracy



(a)



(b)

Figure 14. a) Accuracy during training the model, and b) loss value during the training of the model in the third experiment

Table 17. Comparison between the results of different approaches on the test set.

Approach	Accuracy
ResNet	69.18%
ResNet with augmentation	88%
SimCLRV2	96.21%

5.4.4. Discussion

While existing malware detection approaches often report accuracy rates exceeding 98%, they face significant limitations in detecting obfuscated malware, which can undermine their effectiveness. Our approach addresses this critical gap by transforming code into images and leveraging contrastive semi-supervised learning techniques. This method enables the model to focus on high-level structural and semantic features of the code, which remain invariant despite superficial obfuscation. Unlike traditional methods that rely heavily on handcrafted features or static signatures—features susceptible to obfuscation—our image-based representation emphasizes the malicious core, which remains unchanged even with obfuscation. The use of contrastive learning, particularly within the SimCLR framework, enables the model to implicitly learn invariant features even from limited labeled data, thereby vastly improving its robustness against sophisticated obfuscation techniques.

Additionally, our approach minimizes dependence on extensive feature engineering, reduces the need for large labeled datasets, and enhances the generalization capability to unseen or heavily obfuscated malware, achieving detection accuracies of up to 96.21% even with only 10% of the training data. These advantages explicitly target the existing gaps in malware detection—particularly the challenge of recognizing obfuscated malware—and demonstrate a significant improvement over classical static and dynamic analysis methods.

The use of deep architectures like ResNet-200 and large-scale image datasets naturally raises concerns regarding computational efficiency and scalability. If Training ResNet-200 on a given hardware setup takes approximately X hours/days, inference requires significantly less time. To explore the scalability of our method, we experimented with various ResNet depths (e.g., ResNet-50, ResNet-101, ResNet-200), observing that deeper models yield higher accuracy but at the cost of increased computational resources. In resource-constrained environments, deploying such deep models may pose challenges; however, techniques such as model pruning, quantization, or knowledge distillation can potentially mitigate these issues. Our ongoing research will further explore these strategies to enable efficient deployment in real-world, resource-limited scenarios.

6. REVISE DATE CONCLUSIONS

In this paper, a novel method for malware detection is proposed by mapping code sequences to images. The proposed method focused on malware classification as well as detecting obfuscated malware from benign code. By combining the semi-supervised approach with contrastive learning, this method addresses malware classes that lack sufficient data. Malware classes such as these are rarely developed, hard to identify, and have extremely destructive effects. The results demonstrate that this combination achieves high accuracy in detecting obfuscated malware.

This paper discusses the detection of obfuscated malware from benign code, a task not commonly addressed in existing methods. Moreover, the proposed method does not have most of the disadvantages of dynamic and static malware detection methods.

Another advantage is that the proposed method does not require feature extraction or the use of first- or second-order statistical equations, which can cause features to be extracted gradually using a CNN network.

Although our experiments rely solely on the Microsoft 2015 dataset, the diversity of malware samples and obfuscation techniques included in it demonstrates the robustness of our approach. Because the core of our methodology—code-to-image transformation and contrastive learning—is platform-agnostic, we believe that the high accuracy and generalization observed indicate strong potential for applicability to other datasets and malware types, such as Android malware. Future work will involve evaluating on additional benchmarks to validate this generalizability further.

Future work includes devising new approaches to transform codes into images, such as using 3-channel pictures and utilizing each channel for a specific parameter. Further, the correspondence between each augmentation and any obfuscation procedure could be examined. While our current comparison includes several state-of-the-art methods, it does not encompass recent Transformer-based architectures, which have shown promising results in related domains. This is primarily due to resource constraints and the scope of this study. We acknowledge that including such advanced models could provide a more comprehensive benchmark and is an important direction for future work. We plan to investigate the performance of Transformer-based methods in subsequent research to evaluate their potential benefits in binary code classification tasks.

While our current analysis demonstrates the overall effectiveness of the proposed method, it does not specifically evaluate performance on various obfuscation techniques. Different obfuscation strategies, such as code reordering, control flow flattening, packing, or junk code insertion, may pose different levels of challenge for the model. In future work, we plan to investigate how well the model performs across these diverse obfuscation methods to identify potential limitations and areas for improvement. Understanding the impact of specific obfuscation strategies will help refine our approach and enhance its robustness against a wider range of malware obfuscations.

7. REFERENCES

[1] PE Format, https://docs.microsoft.com/en-us/windows/win32/debug/pe-format , Last accessed 15 June 2025.

[2] Bacci, A., Bartoli, A., Martinelli, F., Medvet, E., Mercaldo, F., & Visaggio, C. (2018). Impact of Code Obfuscation on Android Malware Detection based on Static and Dynamic Analysis. *IEEE International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, 2018, pp. 379-385. doi: 10.5220/0006642503790385.

[3] You, I., & Yim, K. (2010). Malware Obfuscation Techniques: A Brief Survey. *2010 International Conference on Broadband, Wireless Computing, Communication and Applications (BWCCA)*, pp. 297-300, doi: 10.1109/BWCCA.2010.85.

[4] Singh, J., & Singh, J. (2018). Challenges of Malware Analysis: Obfuscation Techniques. International Journal of Computer Science and Mobile Computing, 7(4), 150-156.

[5] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, & Geoffrey Hinton. (2020). Big Self-Supervised Models are Strong Semi-Supervised Learners. *arXiv preprint*, arXiv:2006.10029.

[6] Longlong Jing, & Yingli Tian. (2021). Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11), 4037-4058. doi: 10.1109/TPAMI.2020.2992393.

[7] Lee, D.H. (2013). Pseudo-Label: The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks. *ICML 2013 Workshop: Challenges in Representation Learning*

[8] Jaiswal, A., Babu, A., Zadeh, M., Banerjee, D., & Makedon, F. (2021). A Survey on Contrastive Self-Supervised Learning. Technologies, 9(1).

[9] Yuhang Zhang, Xiaopeng Zhang, Robert Qiu, Jie Li, Haohang Xu, and Qi Tian. (2021). Semi-supervised Contrastive Learning with Similarity Co-calibration.

[10] Xing, J., Bauersfeld, L., Song, Y., Xing, C., & Scaramuzza, D. (2024, May). Contrastive learning for enhancing robust scene transfer in vision-based agile flight. *2024 IEEE International Conference on Robotics and Automation (ICRA),* pp. 5330-5337. IEEE.

[11] Huang, W., Li, C., Zhou, H. Y., Yang, H., Liu, J., Liang, Y., ... & Wang, S. (2024). Enhancing representation in radiography-reports foundation model: A granular alignment algorithm using masked contrastive learning. *Nature Communications*, 15, 7620.

[12] Liu, S., Kimura, T., Liu, D., Wang, R., Li, J., Diggavi, S., ... & Abdelzaher, T. (2024). FOCAL: Contrastive learning for multimodal time-series sensing signals in factorized orthogonal latent space. *Advances in Neural Information Processing Systems*, 36.

[13] Zhang, D., Rong, Z., Xue, C., & Li, G. (2024). Simre: Simple contrastive learning with soft logical rule for knowledge graph embedding. *Information Sciences*, 661, 120069.

[14] Rethmeier, N., & Augenstein, I. (2023). A primer on contrastive pre-training in language processing: Methods, lessons learned and perspectives, *ACM Computing Surveys*, 55(10), 1-17.

[15] Le-Khac, P., Healy, G., & Smeaton, A. (2020). Contrastive Representation Learning: A Framework and Review. *IEEE Access*, 8, 193907–193934.

[16] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint,* arXiv:2002.05709, 2020.

[17] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, arXiv:1512.03385.

[18] The Illustrated SimCLR Framework, https://amitness.com/2020/03/illustrated-SimCLR/. Last accessed 15 June 2025.

[19] Wang, F., & Liu, H. (2021). Understanding the Behavior of Contrastive Loss. *IEEE Transactions on Neural Networks and Learning Systems*.

[20] Verma, V., Muttoo, S. K., & Singh, V. B. (2020). Multiclass Malware Classification via First- and Second-Order Texture Statistics. *Computers & Security*, 97, 101895.

[21] Malimg Dataset Benchmark, https://paperswithcode.com/sota/malware-classification-on-malimg-dataset. Last accessed 15 June 2025.

[22] VirusShare, https://virusshare.com. Last accessed 10 June 2025.

[23] Verma, V., & Muttoo, S., & Singh, V. (2020). Detection of Malignant and Benign PE Files Using Texture Analysis. In *Lecture Notes in Computer Science* (pp. 123–138). doi: 10.1007/978-3-030-65610-2_16.

[24] Catak, F. O., Javed, A., & Şahinbaş, K. (2021). Data Augmentation-Based Malware Detection Using Convolutional Neural Networks. *PeerJ Computer Science*, 7, e346. doi: 10.7717/peerj-cs.346.

[25] Marastoni, N., Giacobazzi, R., & Dalla Preda, M. (2021). Data Augmentation and Transfer Learning to Classify Malware Images in a Deep Learning Context. *Journal of Computer Virology and Hacking Techniques*, 17, 10.1007/s11416-021-00381-3.

[26] Microsoft Malware Classification Benchmark Dataset, https://arxiv.org/abs/1802.10135.

[27] Lee, W., Saxe, J., & Harang, R. (2019). SeqDroid: Obfuscated Android Malware Detection Using Stacked Convolutional and Recurrent Neural Networks. *Proceedings of the 24th ACM International Conference on Mobile Computing and Networking*, pp. 1–9. doi: 10.1145/3317550.3321484.

[28] Millar, S., McLaughlin, N., Rincon, J. M. del, Miller, P., & Zhao, Z. (2020). Dandroid: A Multi-view Discriminative Adversarial Network for Obfuscated Android Malware Detection. *Proceedings of the Tenth ACM Conference on Data and Application Security and Privacy*, 2020, pp. 353–364.

[29] Chen, P.-Y., Sharma, Y., Zhang, H., Yi, J., & Hsieh, C.-J. (2018). EAD: Elastic-Net Attacks to Deep Neural Networks via Adversarial Examples. *Proceedings of the AAAI Conference on Artificial Intelligence*.

[30] Pouya, S., Kabkab, M., & Chellappa, R. (2018). Defense-GAN: Protecting Classifiers against Adversarial Attacks Using Generative Models. *International Conference on Learning Representations (ICLR)*.

[31] Park, D., & Yener, B. (2020). A Survey on Practical Adversarial Examples for Malware Classifiers. *arXiv preprint*, arXiv:2011.07006.

[32] Yan, J., Qi, Y., & Rao, Q. (2018). Detecting Malware with an Ensemble Method Based on a Deep Neural Network. *Security and Communication Networks*, 2018, 1–12. doi: 10.1155/2018/9639503.

[33] Khan, R., Zhang, X., & Kumar, R. (2019). Analysis of ResNet and GoogleNet models for malware detection. *Journal of Computer Virology and Hacking Techniques*, 15, 10.1007/s11416-018-0324-z.

[34] Darem, A., Abawajy, Jemal, Makkar, A., Alhashmi, A. and Alanazi, S 2021, Visualization and deep-learning-based malware variant detection using OpCode-level features, *Future Generation Computer Systems*, 125, pp. 314-323, doi: 10.1016/j.future.2021.06.032.

[35] Nataraj, L., Karthikeyan, S., Jacob, G., & Manjunath, B. (2011). Malware Images: Visualization and Automatic Classification. *Proceedings of the 8th International Symposium on Visualization for Cyber Security. Association for Computing Machinery*.

[36] Microsoft Malware Classification Challenge (Big, 2015), 2015, https://www.kaggle.com/c/malware-classification/data.

[37] PEFile library for Python, https://github.com/erocarrera/pefile, Last accessed 15 November 2021.

[38] Zhenyu, C., Yuxin,L., Xiaogang, W., & Wei, Z.,, "Enhanced Contrastive Learning for Malware Detection Using Structural Similarity," IEEE Transactions on Dependable and Secure Computing, 20(3), pp. 2456-2470, 2023. doi: 10.1109/TDSC.2023.3267421.

[39] Wang, L., Zhang, Q., Li, H., & Chen, M., Vision Transformers for Malware Image Classification: Performance Analysis and Optimization, *Proceedings of the 2023 IEEE Symposium on Security and Privacy (SP)*, 2023, pp. 1125-1138. doi: 10.1109/SP46215.2023.00035.

[40] Zhang, H., Wu, K., Chen, & Y., Wang, T., Adversarial Malware Obfuscation: Detection and Mitigation Techniques, Computers & Security, 128, pp. 103156, 2023. doi: 10.1016/j.cose.2023.103156.

[41] Li, Y., Sun, M., Zhou, P., & Wu, F., MetaMalware: Few-Shot Learning for Malware Classification, *Proceedings of the 2024 Network and Distributed System Security Symposium (NDSS)*, USA, 2024.

[42] Kumar, R., Patel, S., Kim, J., & Li, W., Dynamic Analysis of Malware Using Temporal Graph Networks, *IEEE Transactions on Information Forensics and Security*, 19, pp. 1028-1042, 2024. doi: 10.1109/TIFS.2023.3342178.

[43] Nguyen, T., Brown, A., Garcia, F., & Martinez, S., Explainable AI for Malware Detection: Towards Transparent Security Systems, *ACM Transactions on Privacy and Security*, 27(1), pp. 1-28, 2024. doi: 10.1145/3632975.

[44] Gao, X., Yang, J., Liu, W., & Wang, H., Beyond SimCLR: Advanced Self-Supervised Learning for Cybersecurity Applications, *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23)*, Copenhagen, Denmark, 2023, pp. 789-804. doi: 10.1145/3548606.3560672.

[45] Patel, S., Gupta, M., Reddy, C., & Zhang, W., Multi-Modal Malware Analysis: Combining Static and Dynamic Features, *Journal of Computer Virology and Hacking Techniques*, 19(2), pp. 123-135, 2024. doi: 10.1007/s11416-024-00505-5.

[46] Yang, J., Wang, H., Zhang, L., & Chen, Y., Robust Malware Detection in Adversarial Environments, *Proceedings of the 2024 USENIX Security Symposium (USENIX Security '24)*, USA, 2024, pp. 456-470.

[47] Sharma, P., Lee, D., Kim, S., & Park, Y., EdgeMalNet: Lightweight Malware Detection for IoT Devices, *IEEE Internet of Things Journal*, 11(5), pp. 4231-4245, 2025. doi: 10.1109/JIOT.2024.3387215.

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

Automatic Multi-Class Cardiovascular Magnetic Resonance Image Quality Assessment using Unsupervised Domain Adaptation in Spatial and Frequency Domains

Shahabedin Nabavi✉, Code Orcide: 0000-0001-7240-0239
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran. s_nabavi@sbu.ac.ir
Hossein Simchi, **Code Orcide:** 0009-0002-1392-1426
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran. hsimchi74@gmail.com
Mohsen Ebrahimi Moghaddam, **Code Orcide:** 0000-0002-7391-508X
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran. m_moghadam@sbu.ac.ir
Ahmad Ali Abin, **Code Orcide:** 0000-0002-0916-0348
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran. a_abin@sbu.ac.ir
Alejandro F. Frangi, **Code Orcide:** 0000-0002-2675-528X
Division of Informatics, Imaging and Data Sciences, Schools of Computer Science and Health Sciences, The University of Manchester, Manchester, UK.
Medical Imaging Research Center (MIRC), Electrical Engineering and Cardiovascular Sciences Departments, KU Leuven, Leuven, Belgium.
Alan Turing Institute, London, UK. alejandro.frangi@manchester.ac.uk

Abstract

Population imaging studies rely on good-quality medical imagery before quantifying downstream images. This study provides an automated approach for image quality assessment (IQA) from cardiovascular magnetic resonance (CMR) imaging at scale. We identify four common CMR imaging artefacts: respiratory motion, cardiac motion, Gibbs ringing, and aliasing. Four datasets, including UK Biobank, York University (YU), the Universidad Carlos III (UCIII) and CMR-Tehran, were used to perform the experiments. This study proposes two deep-learning models for CMR IQA in spatial and frequency domains. The presented spatial-domain model also has domain adaptation. The accuracies of supervised 4-fold cross-validation experiments for UK Biobank, YU, UCIII and CMR-Tehran datasets are 99.41%, 75.78%, 89.46% and 67.87% for the spatial-domain and 87.46%, 63.76%, 80.25% and 58.48% for the frequency-domain. Domain adaptation results, considering UK Biobank as the source set and YU, UCIII and CMR-Tehran as the target sets, show the domain shift gap coverage between the datasets to the extent of +11.91%, +3.93% and +16.57%, respectively. Besides, by training and testing the spatial-domain model on 30,125 images from the UK Biobank, an accuracy of 89.56% was obtained in a training time of 394.80 seconds. Meanwhile, the frequency-domain model with training and testing on 180,750 images achieves an accuracy of 87.99% in a training time of 255.04 seconds. Thus, the frequency-domain model can achieve almost the same accuracy yet 1.548 times faster than the spatial model. The proposed models can detect four common CMR imaging artefacts by receiving images or the corresponding k-spaces.

KEYWORDS: Artefact, Cardiovascular magnetic resonance imaging, Deep learning, Domain Adaptation, Image quality assessment.

1- Introduction

Cardiovascular magnetic resonance (CMR) imaging has many clinical applications as a powerful non-invasive diagnostic tool. This imaging modality can assess cardiac function and support the diagnosis of cardiovascular diseases, including coronary artery disease, cardiomyopathy, and congenital and valvular disease [1]. CMR imaging is the gold standard in many diagnostic and therapeutic applications [2, 3]. In addition, CMR is used in most cardiovascular population imaging studies, given its optimal trade-off of non-invasiveness and accuracy. Imaging artefacts hinder routine quantitative analysis of CMR, amongst other factors. These artefacts affect the cardiac image analysis, rendering cardiac anatomy or functional parameters inaccurate for diagnostic or epidemiological studies. The increasingly wider availability of CMR expertise in mainstream cardiology and the development of novel MRI protocols have reduced the incidence of these artefacts. Yet, their presence remains unavoidable, and

Submit Date: 2025-03-09
Revise Date: 2025-07-21
Accept Date: 2025-09-17
✉ Corresponding author

visual determination of image quality is impractical in busy cardiovascular imaging departments. Hence, automated approaches to detect and identify these artefacts remain highly desirable.

After each image acquisition and over an extended period, image quality needs to be assessed to ensure optimal diagnostic value and minimal image quality drifts over time. Human quality assessment of CMR images can be time-consuming and costly. The many images acquired during an imaging examination require automatic machine learning methods for image quality assessment. Chow and Paramesran reviewed the methods of assessing the quality of medical images and specifically predicted that the next generation of methods would be no-reference [4]. Availability of the reference image means that the quality of each received image is compared to the reference image. The main challenge in the quality assessment of medical images is the impossibility of accessing this reference image, so estimating the quality of medical images is generally established without reference.

In a study by Tarroni et al. [5], an automated process for determining the quality of short and long-axis CMR images is proposed, examining image quality related to full heart coverage in the images, motion artefact, and contrast estimation. That study has been performed on two datasets, one with 3000 samples for training and testing processes and 100 samples for testing based on random forests. That study shows the sensitivity and specificity of 88 and 99% for the diagnosis of full cardiac coverage and 85 and 99% for the diagnosis of motion artefact. In another study [6], the same method as [5] has been evaluated on a large-scale subset of the UK Biobank. The results on 19,265 short-axis cine stacks from the UK Biobank show that up to 14.2% have suboptimal coverage and up to 16% have inter-slice motion. In the study of Oksuz et al. [7], motion artefacts due to breathing and cardiac movements were investigated. In that study, k-space manipulation has been used to increase the volume of distorted data, and a CNN architecture has been used to learn the automatic detection model. The results on 3510 CMR images show an area under the receiver operating characteristic (ROC) curve (AUC) of 0.89. In a study by Zhang et al. [8], the problem of full left ventricular coverage using CNNs has been investigated in a data set of over 5,000 cases, with an error rate of less than 5%. In that study, short-axis cine CMR images have been received as input. Two parallel CNN architectures have been used to determine the presence or absence of images related to the apex and the basal of the heart. The main innovation of that study is using a particular layer in the end part of the proposed architecture to improve the classification of images, which has been introduced as a Fisher discriminative layer. This layer tries to minimise within-class variance and maximise the distance of between-class means. In another study [9], the use of an adversarial learning approach based on CNNs to detect the entire left and right ventricular coverage of the heart has been investigated. That method has been studied on three datasets, and the results show the superiority of the method over previous approaches. In another study [10], generative adversarial networks have been evaluated to detect full left ventricular coverage in a dataset consisting of over 6,000 samples. The accuracy obtained in that study has been reported to be about 90%. In a study by Osadebey et al. [11], image quality assessment in the brain and cardiovascular images has been investigated. In that study, 16 volumes of CMR images have been used, and the method is based on the handmade feature extraction of four types of additive noise. Several other studies have also addressed CMR image quality assessment, including [12, 13]. A summary of these related studies is given in Table S1 (See supplementary material).

This study proposes an automated CMR image quality assessment method to detect different artefacts. Besides image quality control, it also reveals the type of artefact. Thus, if the imaging needs to be repeated, it makes the imaging technicians aware of the source of the problem.

The contributions of this study are:

(1) An automated image quality assessment approach is proposed that handles four common types of CMR artefacts. Respiratory motion, cardiac motion, Gibbs ringing, and aliasing artefacts are examined. Considering the four common artefacts in CMR images, this study outperforms previous studies regarding the number of examined artefacts.
(2) In this approach, CMR images with different views, including short-axis cine CMR, two, three, and four-chamber long-axis images, are considered input. We seek to obtain more generality for the proposed models from this point of view.
(3) The proposed method can be used in both spatial and frequency domains. Therefore, the frequency domain model can assess the quality based on receiving k-space. The k-spaces in this study are obtained by Fourier transformation of the corresponding magnitude images and are not the scanner-acquired k-spaces.
(4) Increasing the speed of training and testing using the proposed frequency domain model is another contribution of this study.
(5) The approach is evaluated on several datasets to avoid biasing the assessment and understand the generalisability of the process. Since access to data labels is usually limited, a method is proposed to generate a training set by modelling imaging artefacts using k-space manipulation, producing corrupted images. A deep domain adaptation method in the spatial domain is then presented and evaluated for unavailable data labels in some datasets. These proposed methods are assessed in supervised, learning from limited training data, and unsupervised manners.

2- Materials and Methods

In this section, the datasets used in the study are first described. Then the methods of adding artefacts to CMR images by manipulating the k-space to generate four types of artefacts, including respiratory and cardiac motion, aliasing and Gibbs ringing, are explained. This section also presents deep learning-based models for identifying CMR image artefacts. These models are proposed in both spatial and frequency domains, and details of all models are described. An unsupervised domain adaptation method is presented for conditions where the data labels are not accessible. In the end, the experimental setups are described to explain the defined experiments and evaluation metrics.

2-1- Dataset description

The main database of the current study is the CMR image database from the UK Biobank. Over 35,000 2D slices from over 6,000 subjects of this database were used to generate a dataset including CMR images corrupted by synthetic artefacts. CMR images were acquired using a wide clinical bore $1.5T$ MR system (MAGNETOM Aera, Syngo Platform VD13A, Siemens Healthcare, Erlangen, Germany) with an 18 channel anterior body surface coil ($45\ mT/m$ and $200\ T/m/s$ gradient system). 2D cine balanced steady-state free precession (bSSFP) short-axis and long-axis images have in-plane spatial resolution

$1.8 \times 1.8\ mm$, image size 198×208, slice thickness $8\ mm$ and slice gap $2\ mm$. More information about the imaging protocol is available in [14].

Besides UK Biobank, data from three other datasets have been used. One of these datasets is the cardiac MRI dataset from York University (YU). A total of 7980 short and long-axis images have been acquired from 33 subjects. Imaging has been performed with a GE Genesis Signa MR scanner using the fast imaging employing steady-state acquisition (FIESTA) scanning protocol. The subjects studied in this research are all under 18 years old. The size of each slice is 256×256 pixels. In the short-axis view, the number of frames is 20; in the long-axis view, the number of slices varies between 8 and 15. The slice gap is also between 6 and 13 mm. Details related to this dataset can be found in [15]. The Universidad Carlos III (UCIII) dataset [16], containing 32 short-axis cardiac cine MR images of four small animals, is another dataset used in this study. A $7T$ Bruker Biospec $70/20$ scanner has been used to scan these self-gated rat cardiac cine sequences (IntraGateFLASH). The number of acquired frames is 8 with an image size of 192×192. slice thickness is $1.2\ mm$. Finally, another dataset including short and long-axis CMR images was collected from Tehran hospitals. 2620 images of 21 unknown subjects have been gathered in this dataset. The imaging parameters are different between CMR image volumes in this dataset due to its multicentre nature. Patient information and images related to this database will remain confidential. We intentionally selected datasets with varying sizes, qualities, and subject types to introduce a significant domain shift, allowing us to assess the model's robustness and generalization under challenging, real-world conditions.

For data augmentation in YU, UCIII, and CMR-Tehran datasets with fewer data, horizontal and vertical flips, random rotation with padding and brightness-changing methods have been used. The angle of rotation and the number of brightness changes were considered random to diversify the data further. Data augmentation methods are used to increase the size of the training sets.

2-2- Artefacted CMR image ground truth via k-space manipulation

Due to hardware and software restrictions, CMR images may be associated with artefacts. These artefacts can seriously affect image quality and thus the ability to make a proper diagnosis. According to the published statistics, motion and Gibbs artefacts are the most common in magnetic resonance imaging [17]. Specifically, respiratory and cardiac motion, Gibbs ringing, and aliasing artefacts are the most common in CMR imaging [18]. For this reason, we investigated these types of artefacts in this study.

Subjectively determining the artefact in the CMR images is a time consuming and laborious task that requires several human observers. Since this study needs corrupted images, we used the k-space degradation methods to generate synthetic images. Therefore, long-axis and short-axis cine CMR images acquired using Cartesian sampling were used to add respiratory and cardiac motion, aliasing and Gibbs ringing artefacts. These sections explain how to corrupt images using k-space manipulation.

2-2-1- Respiratory motion artefacts

We use a Cartesian manipulation approach to make corrupted CMR images with the respiratory artefact through k-space manipulation, considering that imaging datasets used in this study are acquired using Cartesian sampling. According to the method presented in Lorch et al. [19] study, we leverage a translation with a sinusoidal pattern on the reference images. First, the reference image is subjected to a 1D translation along the height or width. Then, due to Cartesian sampling, the fast Fourier transform is used to transfer the reference and the translated image into k-space. In the next step, some lines from the k-space of the reference image are replaced with corresponding lines related to the translated image. Finally, the modified k-space is reconstructed into the image space using the inverse fast Fourier transform, and we see the appearance of artefacts in the reconstructed image.

Translation in the height or width of the reference image to generate images with different severities of the artefact is considered from 1 to 5 pixels as a random selection. Also, 10 to 25 lines from the k-space are randomly selected for the replacement to make distorted images with different severities of the artefact. This method of generating degraded images is also used in the study by Oksuz et al. [7]. Figure 1 shows how to develop a corrupted slice with this method.

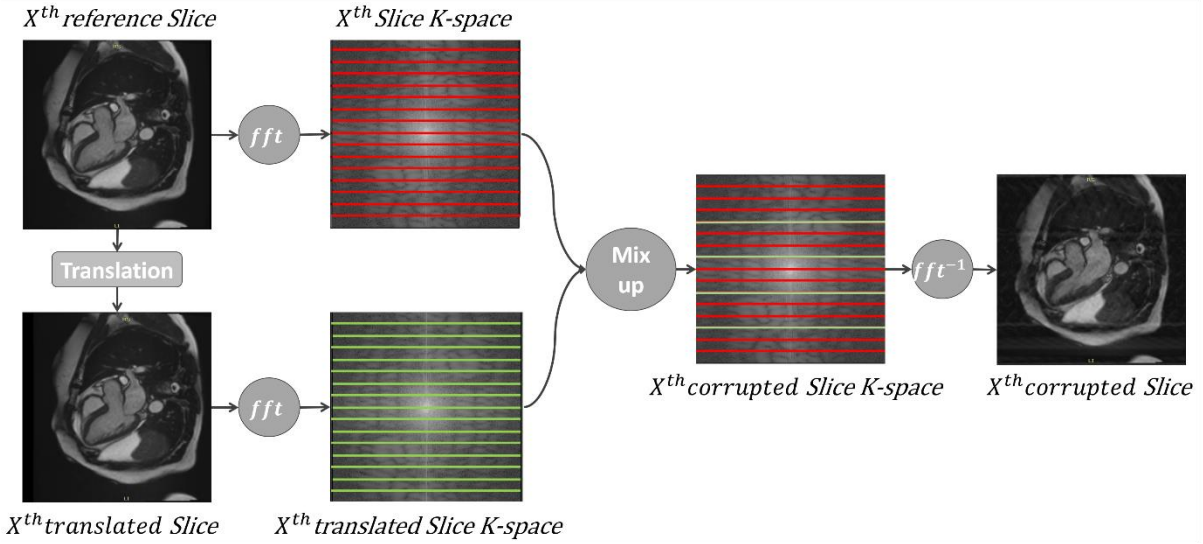


Figure 1. k-space manipulation to generate synthetic respiratory artefacts.

2-2-2- Cardiac motion artefacts

Similar to [7], short-axis cine CMR images need to be used to generate degraded images with cardiac motion artefacts. For this purpose, a temporal sequence is separated from the short-axis cine images that fully cover the heart from the basal to the apex. The images in this sequence are then transferred to the corresponding k-space using the fast Fourier transform. By replacing the k-space lines of the Cartesian sampled slices in this sequence, degraded images can be generated with this artefact. The corrupted images using this method are realistic because the mis-triggering artefact results from the misallocations of the k-space lines in the temporal slices in reality. The number of replaced k-space lines is randomly selected from 10 to 20 to obtain different levels of the artefact severity. Figure 2 shows the process of image corruption with this artefact.

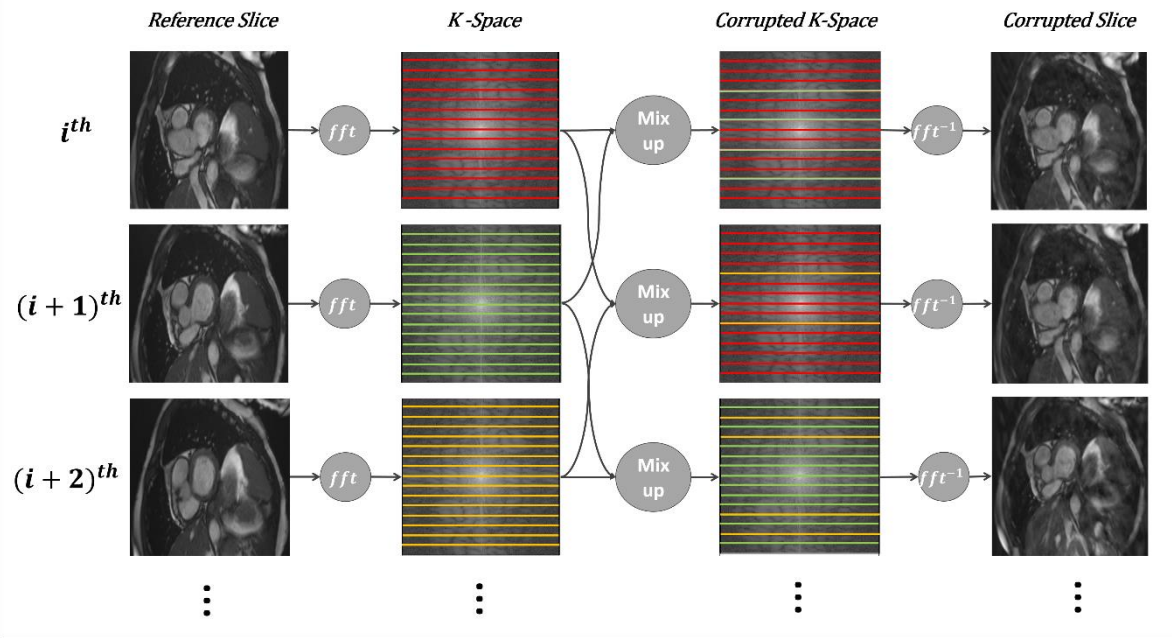


Figure 2. k-space manipulation to generate synthetic cardiac motion artefacts.

2-2-3- Gibbs ringing artefacts

Gibbs ringing or truncation artefacts are due to using Fourier transforms in the reconstruction of the final image. Since to generate an image in magnetic resonance imaging, we have to take a limited sample of the generated signals, so when reconstructing the final image, we have to approximate it using a relatively small number of harmonics in the Fourier representation of the image. Therefore, the truncated Fourier series approximation of a discontinuous signal shows overshoot and ringing near the discontinuities of the primary signal [20]. This case is called the Gibbs phenomenon [21], which causes parallel lines and curves in the final image. Image filtering can be used with an ideal low-pass filter in the frequency domain to add this artefact with different severities to the final image [22]. Thus, the image is transferred to k-space by fast Fourier transform, and then k-space is filtered by an ideal low-pass filter. The radius of the ideal low-pass filter was chosen randomly from 30 to 50 pixels to obtain degraded images with different severities of this artefact. A smaller radius causes a more intense artefact appearance due to the removal of more data from the k-space. The procedure for the k-space manipulation to generate corrupted images is shown in Figure 3.

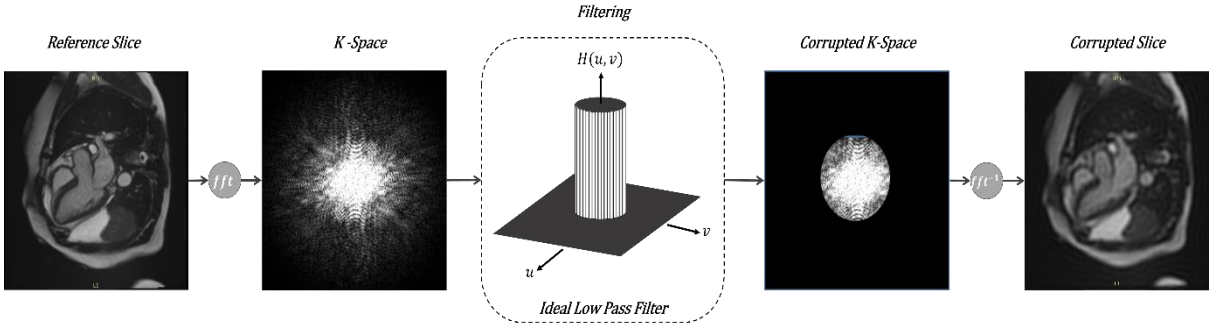


Figure 3. k-space manipulation to generate synthetic Gibbs ringing artefacts.

2-2-4- Aliasing artefacts

Aliasing or wrap-around artefact occurs when the field of view (FOV) is smaller than the area being imaged. Here, the parts outside the FOV are reflected on the other side of the image. It is enough to undersample the k-space to add this artefact to the images by manipulating the k-space. Because the k-space of the images used is Cartesian sampled, the corrupted image can be obtained by alternately deleting rows or columns of k-space corresponding to an image [23]. An example of image degradation in this way is shown in Figure 4.

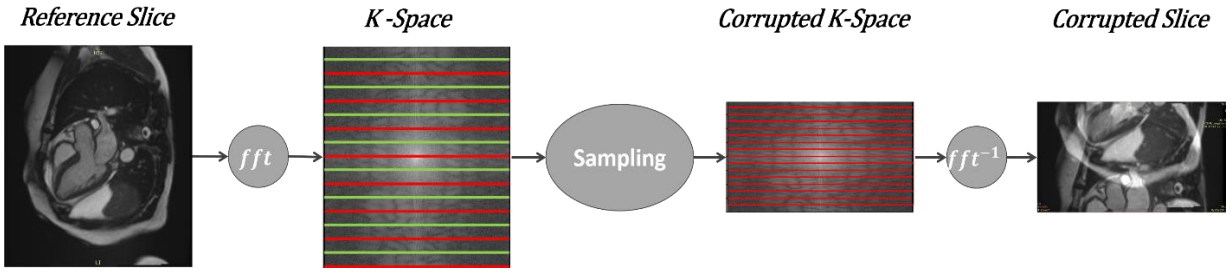


Figure 4. k-space manipulation to generate synthetic aliasing artefact.

2-3- Deep learning models for automated CMR image quality assessment

In this section, we propose a deep-learning model for automatic CMR image quality assessment. First, this model is proposed to identify the artefacts in the spatial domain when data labels are available. The model details for domain adaptation are presented when we use the trained model to test a new dataset. Finally, the proposed model in the frequency domain is described to directly use the model on the k-space generated by the Fourier transform of the corresponding magnitude images.

The proposed model evaluates the quality of each of the 2D slices of a 3D volume of CMR images. The reason for presenting the 2D model is that the artefacts studied in this research have features that can be seen in each 2D slice. Although some previous studies have identified motion artefacts with 3D models like [7], the diversity of artefacts in the current research and the need for a general model to distinguish between types of artefacts led us to use the 2D model. Also, since clinicians ultimately analyse CMR images in a 2D view, the present study used a 2D model to identify artefacts.

2-3-1- Spatial domain

We use a deep learning model to identify artefacts in the spatial domain. This model generally has three essential parts: (1) Feature extraction from CMR images (2) Classification of the image artefacts (3) Domain label predictor for domain adaptation. Figure 5 shows the proposed model in the spatial domain.

The feature extraction section receives the input images with various views, including two, three and four-chamber long-axis and short-axis views. The dimensions of the input images are resized to 90×90 to control the computational overhead. The image resizing operation is done using the Scikit-image Python package to convert the size of all the images to the required size. Four 2D convolutional layers with the details specified in Figure 5 are considered for image feature extraction. Each pair of convolutional layers is followed by a max-pooling layer and a dropout layer with a probability of 0.25. Finally, a flatten followed by a dense layer are at the end of this section, which presents the extracted features in a 512 vector.

The feature vector generated by the feature extractor part is received as input in the label classification section. This section follows a dropout layer with a probability of 0.5 by two dense layers responsible for identifying the artefact labels. Five labels related to respiratory and cardiac motions, aliasing, Gibbs ringing and without artefacts are determined by five terminal neurons. If training data labels are available and supervised or learning from limited annotated training data is on the agenda, we will only use the feature extraction and label classification parts. Otherwise, it is also necessary to use the domain label predictor part for the unsupervised domain adaptation process.

Domain adaptation refers to the concept of learning a discriminative classifier when there is a shift in distributing training and testing data distribution [24]. Domain adaptation approaches are classified into two categories, unsupervised and semi-supervised, based on the possibility of accessing the data labels of the target dataset. The former is when the target dataset is unlabelled, and the latter is when a few samples are labelled. These approaches allow the mapping between the source and target domains to be learned when the target domain is unlabelled or partially labelled. It means that the classifier can be trained on a dataset as the source domain and then used to test the target domain [25]. The main idea of domain adaptation is inspired by [26] and customised for the current study. All three subnets required for feature extraction, artefact classification, and domain label prediction have been redesigned and implemented to meet the objectives of this study.

In the situation where there are two datasets, source and target, where there is a domain shift between them, we should also use the domain classification part of the model for unsupervised domain adaptation. Consider that data with the artefact-type label is available for the source set, and no artefact-type label exists for the target data set. For the domain label predictor part, we assign an additional label representing the source or target domain type to the data of each set. For example, in addition to the artefact-type label, we assign a 0 label to all source data, which indicates that this data belongs to the source set. The data of the target set, which does not have an artefact-type label, only gets a label such as 1, which means that it is the data of the target set. The domain classification part of the model follows a gradient reversal layer (GRL) by three dense layers with the details specified in Figure 5. The GRL layer acts as an identical transformation in forwarding propagation while multiplying the gradient received from the dense layers by a negative value in backpropagation. The reason for using this layer is there are conflicting goals between the parts. If there are two datasets, one as a source with access to the data labels and the target without data labels, the feature extraction part must extract domain-invariant features between different datasets at training time. The source data must have a domain label and an artefact label. In contrast, the target data have the domain label, which differs from the source domain label, to achieve these features. Therefore, the feature extraction part maximises the domain classifier loss and minimises the artefact label classifier loss. Maximising the domain classifier loss makes the distribution of features obtained from the two domains more similar.

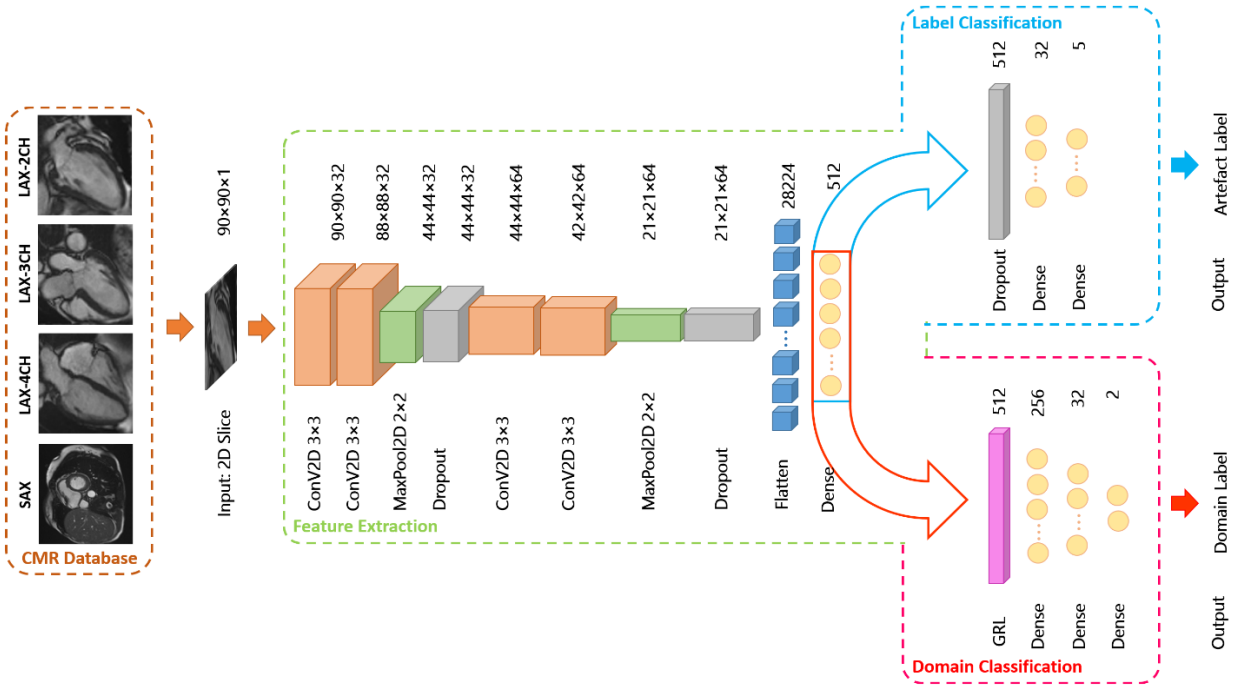


Figure 5. The proposed cardiovascular magnetic resonance image quality assessment model in the spatial domain.

2-3-2- Frequency domain

Two main concerns led us to provide a model in the frequency domain for CMR image quality assessment: (1) the possibility of using the proposed method in the frequency domain on the raw data of k-space before image formation, (2) Increasing the speed of training and testing operations when the model may be used for image quality assessment in large datasets such as UK Biobank. The proposed model is shown in Figure 6.

The proposed model can be used well when working on large datasets and analysing k-space data. Instead of using the ordinary convolutional layer, the proposed model uses Fourier-based CNNs (FCNNs) [27], which dramatically reduces the computational costs of feature extraction. The convolution operation in the frequency domain can be changed to a pointwise multiplication in this model by considering Equation (1).

$$DFT\big(f(x,y) * h(x,y)\big) = DFT(f(x,y)) \cdot DFT(h(x,y)) \quad (1)$$

where DFT represents the discrete Fourier transform, $f(x,y)$ represents the input image, and $h(x,y)$ indicated the filter.

The k-space corresponding to the input image is pointwise multiplied in three filters filled with random complex numbers for feature extraction. This operation is repeated twice, and the size of the filters is the same as the size of the k-space. The output of these steps is given to the max-pooling layer, and the dropout with a probability of 0.25, a flatten, and a dense layer follows this layer. Finally, a 512 vector is transferred to the label classification section; this section's details are shown in Figure 6.

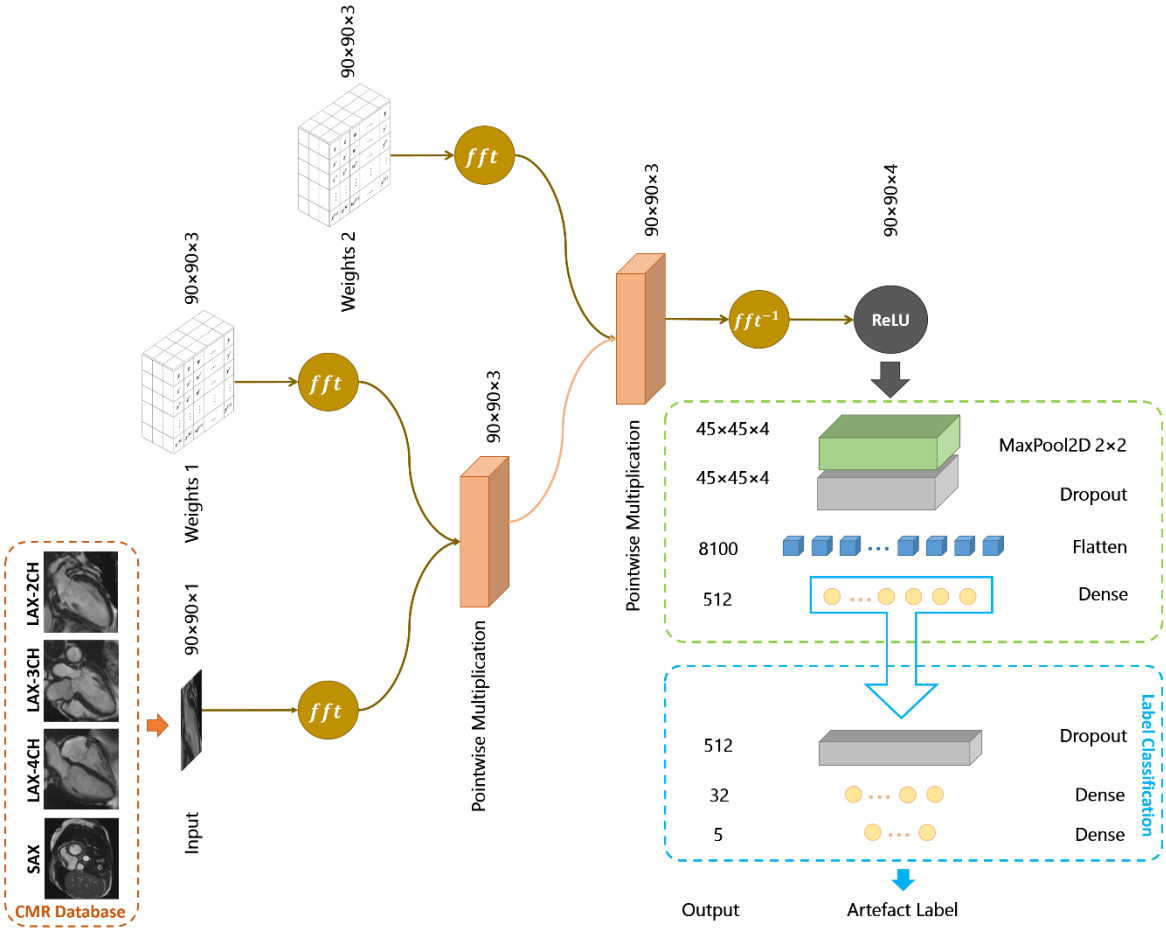


Figure 6. The proposed cardiovascular magnetic resonance image quality assessment model in the frequency domain.

2-4- Model training and Evaluation design

The training process of the existing models was performed in several steps according to different defined experiments. Adam optimiser [28] and cross-entropy loss function [29] have been used to train the models in spatial and frequency domains. The learning rate was set to 1×10^{-3}. Also, the training process has been carried out in 25 epochs and with batch size 128. The activation function for all layers except the last one is rectified linear unit (ReLU). This function is Softmax for the last layer. To validate the model training process and increase the reliability of the results, experiments were performed using 4-fold cross-validation. The best architecture suitable for the current study, including layer placements and selection of hyperparameters, was made based on ablation studies. The hardware platform is a computer equipped with an Intel Core i-7-6700, 16GB of RAM and an NVIDIA GeForce GTX TITAN X GPU. After the article is published, the developed Python code will be made available upon request.

We performed several experiments to evaluate the accuracy and capability of the proposed methods. In the first group of these experiments, the proposed deep learning models in spatial and frequency domains are learned in a supervised learning manner. This way, 75% of the data is a training set, a part of this 75% is the validation set, and the rest 25% is for testing. Given one of the most fundamental challenges, namely the lack of access to labelled data, the second group of experiments is assumed to be supervised using limited training data. In this group of experiments, 25% of the data is used for training and the rest for testing. These experiments make the results comparable in terms of the effectiveness of the amount of annotated data used in the training process. The third set of experiments is performed to evaluate the proposed unsupervised domain adaptation model. All parts of the Figure 5 model are used to perform these experiments. In these experiments, the total number of images of the UK Biobank, YU, CMR-Tehran, and UCIII datasets after adding artefacts and data augmentation are 255775, 106026, 17901 and 4784, respectively. To learn the model, any sample of the source set has both the domain and artefact label, but instances of the target set have only the domain label. UK Biobank data is first considered a source set in these experiments, and other data sets are considered a target. The same experiments are then repeated by considering the YU dataset as the source. The difference between training and testing speed and the accuracy in the spatial and frequency domains are also evaluated. These experiments examine the capability of the proposed models in the CMR image quality assessment. The training and testing data for all five classes are considered almost balanced to make a fair comparison and avoid bias.

Some experiments were also conducted on the real data to check the performance of the proposed models. These experiments can show how similar the synthetic corrupted images are to the real degraded images. For this purpose, UCIII and CMR-Tehran datasets were first labelled by a radiologist with three years of experience in cardiac imaging from the perspective of the artefacts examined in this study. Then, these labels were confirmed by a cardiologist with 15 years of experience in cardiac imaging. The data used to evaluate the frequency domain model in these experiments are obtained from the Fourier transform of the corresponding magnitude images and are not scanner-acquired k-spaces.

Five metrics, including accuracy (ACC), precision (PR), recall (RE), F-measure, and area under the ROC curve (AUC), given in Equations 2 to 6, are used to evaluate the results of the proposed models quantitatively.

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \tag{2}$$

$$PR = \frac{TP}{TP + FP} \tag{3}$$

$$RE = \frac{TP}{TP + FN} \tag{4}$$

$$F - measure = \frac{2 \times PR \times RE}{PR + RE} \tag{5}$$

$$AUC = \int_0^1 \Pr[TP]\,(v)\,dv \tag{6}$$

where TP, FP, TN, and FN indicate true positive, false positive, true negative, and false negative. AUC denotes the overall success of an experiment where $\Pr[TP]$ is a function of $v = \Pr[FP]$.

3- Results

3-1- Results of Supervised learning experiments

In the first experiment, all datasets, including reference images and corrupted images generated by k-space manipulation methods, were used to train and test the models in a supervised manner. After the image corruption by k-space manipulating techniques and data augmentation, 255775, 144411, 23638 and 6080 CMR images were prepared for UK Biobank, YU, CMR-Tehran, and UCIII datasets, respectively. The results to assess the quality of CMR images in a supervised manner for respiratory and cardiac motions, Gibbs ringing and aliasing artefacts are given in Table 1. Besides, the comparison of the models' performance for different views of the images is given in Table S2 (See supplementary material).

Table 1. Results for supervised evaluation of the proposed CMR image quality assessment models are based on accuracy, precision, recall, F-measure and AUC metrics in spatial and frequency domains. Results are based on 4-fold cross-validation ($Mean \pm std$).

Dataset \ Metrics		Accuracy	Precision	Recall	F-measure	AUC
UK Biobank	Spatial	$99 \bullet 40 \pm 0 \bullet 09$	$99 \bullet 41 \pm 0 \bullet 09$	$99 \bullet 40 \pm 0 \bullet 09$	$99 \bullet 40 \pm 0 \bullet 09$	$99 \bullet 89 \pm 0 \bullet 07$
	Frequency	$87 \bullet 46 \pm 0 \bullet 61$	$87 \bullet 59 \pm 0 \bullet 64$	$87 \bullet 31 \pm 0 \bullet 57$	$87 \bullet 45 \pm 0 \bullet 61$	$98 \bullet 64 \pm 0 \bullet 19$
YU	Spatial	$75 \bullet 78 \pm 9 \bullet 17$	$76 \bullet 88 \pm 9 \bullet 27$	$75 \bullet 04 \pm 9 \bullet 06$	$75 \bullet 95 \pm 9 \bullet 15$	$91 \bullet 26 \pm 5 \bullet 44$
	Frequency	$63 \bullet 76 \pm 12 \bullet 35$	$67 \bullet 94 \pm 12 \bullet 61$	$58 \bullet 23 \pm 13 \bullet 02$	$62 \bullet 70 \pm 12 \bullet 92$	$88 \bullet 47 \pm 7 \bullet 71$
CMR-Tehran	Spatial	$67 \bullet 87 \pm 4 \bullet 10$	$68 \bullet 46 \pm 3 \bullet 97$	$67 \bullet 38 \pm 4 \bullet 13$	$67 \bullet 91 \pm 4 \bullet 05$	$87 \bullet 23 \pm 2 \bullet 32$
	Frequency	$58 \bullet 48 \pm 0 \bullet 41$	$62 \bullet 83 \pm 0 \bullet 65$	$48 \bullet 37 \pm 3 \bullet 78$	$54 \bullet 60 \pm 2 \bullet 46$	$88 \bullet 63 \pm 0 \bullet 59$
UCIII	Spatial	$89 \bullet 46 \pm 1 \bullet 01$	$89 \bullet 61 \pm 1 \bullet 20$	$89 \bullet 03 \pm 0 \bullet 62$	$89 \bullet 32 \pm 0 \bullet 87$	$97 \bullet 10 \pm 0 \bullet 51$
	Frequency	$80 \bullet 25 \pm 0 \bullet 94$	$83 \bullet 91 \pm 1 \bullet 16$	$76 \bullet 34 \pm 2 \bullet 00$	$79 \bullet 94 \pm 1 \bullet 50$	$96 \bullet 83 \pm 0 \bullet 34$

The results of the second group of experiments are given in Table 2 for spatial and frequency domains when the training data is limited.

Table 2. Evaluation of learning from limited training data using the proposed CMR image quality assessment models is based on accuracy, precision, recall, F-measure and AUC metrics in spatial and frequency domains. Results are based on 4-fold cross-validation ($Mean \pm std$).

Dataset \ Metrics		Accuracy	Precision	Recall	F-measure	AUC
UK Biobank	Spatial	$96 \bullet 37 \pm 0 \bullet 66$	$96 \bullet 37 \pm 0 \bullet 66$	$96 \bullet 37 \pm 0 \bullet 66$	$96 \bullet 37 \pm 0 \bullet 66$	$99 \bullet 07 \pm 0 \bullet 13$
	Frequency	$79 \bullet 78 \pm 1 \bullet 56$	$80 \bullet 36 \pm 1 \bullet 58$	$79 \bullet 23 \pm 1 \bullet 52$	$79 \bullet 79 \pm 1 \bullet 54$	$96 \bullet 45 \pm 0 \bullet 34$
YU	Spatial	$53 \bullet 25 \pm 6 \bullet 06$	$53 \bullet 82 \pm 6 \bullet 11$	$52 \bullet 43 \pm 6 \bullet 23$	$53 \bullet 11 \pm 6 \bullet 17$	$77 \bullet 68 \pm 4 \bullet 75$
	Frequency	$45 \bullet 66 \pm 4 \bullet 42$	$49 \bullet 86 \pm 6 \bullet 45$	$37 \bullet 54 \pm 4 \bullet 28$	$42 \bullet 67 \pm 4 \bullet 19$	$77 \bullet 61 \pm 5 \bullet 68$
CMR-Tehran	Spatial	$54 \bullet 04 \pm 4 \bullet 64$	$54 \bullet 30 \pm 4 \bullet 92$	$52 \bullet 36 \pm 6 \bullet 54$	$53 \bullet 30 \pm 5 \bullet 78$	$79 \bullet 20 \pm 2 \bullet 61$
	Frequency	$45 \bullet 96 \pm 2 \bullet 47$	$49 \bullet 06 \pm 4 \bullet 30$	$36 \bullet 63 \pm 1 \bullet 93$	$41 \bullet 92 \pm 2 \bullet 62$	$81 \bullet 54 \pm 1 \bullet 81$
UCIII	Spatial	$79 \bullet 35 \pm 6 \bullet 22$	$79 \bullet 75 \pm 5 \bullet 92$	$78 \bullet 92 \pm 6 \bullet 57$	$79 \bullet 33 \pm 6 \bullet 24$	$92 \bullet 23 \pm 2 \bullet 69$
	Frequency	$59 \bullet 40 \pm 4 \bullet 65$	$66 \bullet 98 \pm 6 \bullet 26$	$50 \bullet 94 \pm 3 \bullet 33$	$57 \bullet 83 \pm 4 \bullet 25$	$86 \bullet 26 \pm 3 \bullet 44$

3-2- Results of Unsupervised domain adaptation model

This section presents the results of experiments designed to evaluate the proposed domain adaptation method. To compare the results, three modes of implementation are considered. The highest accuracy is achieved when the model can receive artefact labels related to the samples of the target set during training (Upper band). The least accuracy also occurs when the model gets no instances of the target set at the time of training, and the trained model is then evaluated on the samples of the target set (Lower band). In the first experiment, we consider a scenario where the UK Biobank dataset is a source set, and the other datasets are considered a target. The results of the first experiment are shown in Table 3.

Table 3. Results of the first unsupervised domain adaptation experiment (Source: UK Biobank, Target: YU, CMR-Tehran and UCIII) based on accuracy metric (in the range of 0 to 100). The numbers in parentheses in the row corresponding to the results of the proposed method indicate the gap coverage between the upper and lower bands.

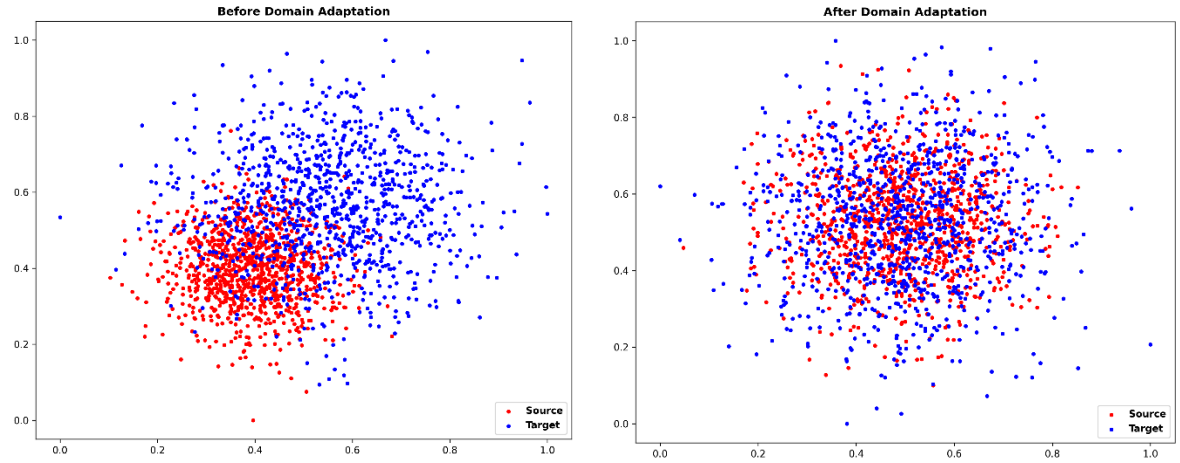
Training mode \ Target set	YU	CMR-Tehran	UCIII
Source only	26.58	15.90	30.50
Proposed method	32.44 (+11.91%)	24.51 (+16.57%)	32.82 (+3.93%)
Train on the annotated target set	75.78	67.87	89.46

In the second experiment of this section, the results are presented in a situation where the source set is YU, and the target sets are the other datasets. The results are given in Table 4.

Table 4. Results of the second unsupervised domain adaptation experiment (Source: YU, Target: UK Biobank, CMR-Tehran and UCIII) based on accuracy metric (in the range of 0 to 100). The numbers in parentheses in the row corresponding to the results of the proposed method indicate the gap coverage between the upper and lower bands.

Training mode \ Target set	UK Biobank	CMR-Tehran	UCIII
Source only	28.62	50.53	31.80
Proposed method	30.17 (+2.19%)	43.88 (-38.35 %)	41.52 (+16.86%)
Train on the annotated target set	99.41	67.87	89.46

Figure 7 shows the effect of the proposed method on the distribution of 512 feature vectors of the last feature extraction layer before and after domain adaptation. The feature vectors of 1000 images are visualised in this figure using the T-distributed stochastic neighbour embedding (t-SNE) method [30]. In this experiment, the YU dataset is the source set (red points), and the UCIII dataset is the target set (blue points). The proposed domain adaptation method could bring the two distributions of feature vectors much closer after adaptation.



(a) **(b)**

Figure 7. The effect of domain adaptation on changing the distribution of feature values extracted for 1000 images. The feature vectors extracted by the first part of the proposed model in the spatial domain are visualised using t-SNE (a) before and (b) after domain adaptation. The red and blue points correspond to the YU and UCIII datasets, respectively.

3-3- Spatial versus frequency domain

The time required for supervised training and testing the models in the spatial and frequency domains is compared in Table 5. These times are measured in seconds. The model in the frequency domain can be learned and tested more quickly. However, this increase in training and testing speed comes with decreased accuracy in this domain. Therefore, we investigated increasing the number of training samples in another experiment to improve the frequency domain's accuracy. The results show that increasing training samples can improve accuracy while training time increases slightly. Random subsets from the UK Biobank dataset were used to perform these comparisons. The results are shown in Table 6 and Figure 8.

Table 5. Comparison of supervised training and test time differences in proposed deep learning models in spatial and frequency domains for each dataset. The times mentioned are measured in seconds ($Mean \pm std$) based on 4-fold cross-validation. Speed up ratio expresses the increase in training speed in the frequency domain model compared to the spatial domain model.

Dataset \ Domain		Spatial (sec)	Frequency (sec)	Speed up ratio
UK Biobank	Train	$2249 \bullet 44 \pm 3 \bullet 30$	$258 \bullet 50 \pm 1 \bullet 93$	8.70
	Test	$32 \bullet 56 \pm 0 \bullet 26$	$7 \bullet 44 \pm 2 \bullet 22$	4.38
YU	Train	$1343 \bullet 90 \pm 0 \bullet 83$	$156 \bullet 39 \pm 0 \bullet 17$	8.59
	Test	$10 \bullet 06 \pm 0 \bullet 28$	$2 \bullet 69 \pm 0 \bullet 91$	3.74
CMR-Tehran	Train	$277 \bullet 71 \pm 0 \bullet 87$	$52 \bullet 26 \pm 0 \bullet 50$	5.31
	Test	$2 \bullet 91 \pm 0 \bullet 19$	$0 \bullet 33 \pm 0 \bullet 03$	8.82
UCIII	Train	$133 \bullet 38 \pm 1 \bullet 04$	$42 \bullet 10 \pm 0 \bullet 52$	3.17
	Test	$1 \bullet 79 \pm 0 \bullet 06$	$0 \bullet 09 \pm 0 \bullet 01$	19.89

Table 6. Comparison of accuracy, precision and recall based on the different number of training samples in spatial and frequency domains and its effect on training time.

	# of training samples	# of model parameters	Accuracy	Precision	Recall	Time (Sec)
Spatial	30,125	≃ 14.6 M	89.56%	89.63%	89.47%	394.80
Frequency (Run 1)	30,125	≃ 4.3 M	76.24%	76.33%	75.92%	43.22
Frequency (Run 2)	180,750	≃ 4.3 M	87.99%	88.12%	87.83%	255.04

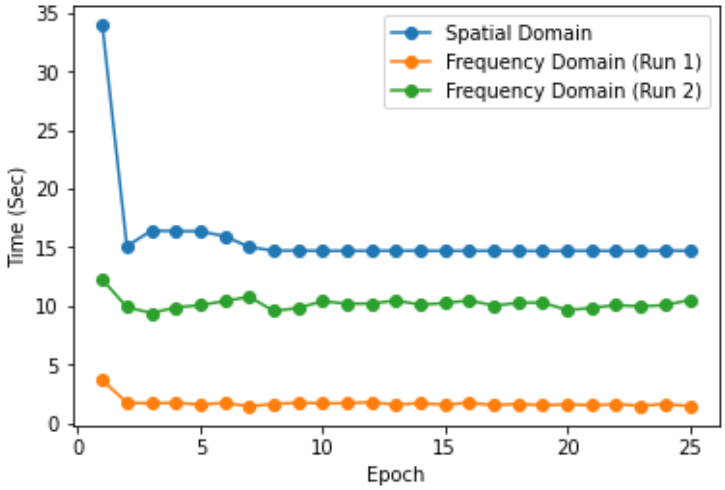


Figure 8. Training time chart in spatial and frequency domains.

3-4- Real versus synthetic corrupted images

The evaluation results of the proposed models in the spatial and frequency domains for experiments related to training from limited data on the real and synthetic corrupted images are shown in Table 7. Besides, the evaluation results of the proposed domain adaptation method, where the source sets are the UK Biobank and the YU dataset, and the target sets are the CMR-Tehran, and UCIII, are tabulated in Table 8. The results indicate that although there is a slight drop in the accuracy obtained on the real data, the achieved results are close.

Table 7. Evaluation of learning from limited training data using the proposed CMR image quality assessment models based on accuracy metric in spatial and frequency domains. Results were obtained using synthetic and real data based on 4-fold cross-validation ($Mean \pm std$).

Dataset \ Type of Data		Synthetic Data	Real Data
CMR-Tehran	Spatial	$54 \bullet 04 \pm 4 \bullet 64$	$53 \bullet 54 \pm 3 \bullet 87$
	Frequency	$45 \bullet 96 \pm 2 \bullet 47$	$45 \bullet 58 \pm 2 \bullet 19$
UCIII	Spatial	$79 \bullet 35 \pm 6 \bullet 22$	$75 \bullet 44 \pm 5 \bullet 33$
	Frequency	$59 \bullet 40 \pm 4 \bullet 65$	$57 \bullet 03 \pm 4 \bullet 25$

Table 8. Evaluation of the unsupervised domain adaptation using synthetic and real data based on accuracy metric (in the range of 0 to 100).

Dataset \ Type of Data		Synthetic Data	Real Data
Source: UK Biobank	**Target: CMR-Tehran**	24.51	22.07
	Target: UCIII	32.82	25.83
Source: YU	**Target: CMR-Tehran**	43.88	43.19
	Target: UCIII	41.52	36.70

4- Discussion

In this study, we aim to address CMR image quality assessment. For this purpose, we tried to propose a comprehensive model for identifying and reporting common artefacts of these images. Respiratory motion, cardiac motion, Gibbs ringing and aliasing artefacts were examined in this study, which is superior to the previous studies regarding the number of artefacts examined. Input images with different views were used to provide a model for recognising these artefacts. Methods were also proposed to add artefacts to CMR images by manipulating the k-space to generate synthetic corrupted images. In addition, detecting artefacts in spatial and frequency domains was investigated to assess the quality using k-space and with no image reconstruction. Therefore, deep learning models were proposed in both spatial and frequency domains. A deep CNN architecture in the spatial domain and a deep Fourier-based CNN architecture in the frequency domain was used.

Several experiments were designed and performed to evaluate the proposed models. In the first series of experiments, the models were trained on the prepared datasets in a supervised manner. These experiments' results show that the proposed deep learning models can detect the mentioned CMR artefacts. The results show a high AUC for the proposed models, indicating the models' suitability to assess the quality of CMR images. The optimal architecture of models in terms of the number of free parameters or weights and other hyperparameters was selected by ablation studies. In these ablation studies, by increasing and decreasing the number of free parameters, a state of the models was found in which the models have the best performance. Although this process is time-consuming, it is used to find the best possible neural network architecture. Besides, The learning curves related to the supervised training of the proposed models in the spatial and frequency domains on the UK Biobank dataset are shown in Figure S3 (See supplementary material).

One of the main challenges and limitations in deep network training is accessing large amounts of labelled data. Besides, many of the presented deep learning models may provide satisfactory results on the studied dataset, but their use on other datasets with different distributions may not provide promising results. Thus, the concept of domain adaptation was considered to overcome these challenges. We tried to select and augment datasets with considerable distribution gaps in this study. Due to the high quality of UK Biobank images, an attempt was made to choose other datasets of different distribution, imaging parameters, subjects being imaged, and image quality. The three datasets used to evaluate the domain adaptation approach have a vast domain shift compared to the UK Biobank dataset.

The proposed domain adaptation method showed a slight improvement when no labels were available for the target datasets. The reason can be attributed to the large domain shifts in the CMR imaging datasets. Although the improvement obtained by the proposed domain adaptation method seems insignificant, it can provide a good motivation for research in this field, given the importance of the problem. Besides, it was also found in one of the experiments on the proposed domain adaptation model that using this method has failed to succeed. A similar lack of success has also been reported in the existing studies in this field [26, 31]. The domain shift in CMR datasets is much wider than the examples in the cited articles due to different content, imaging sequences, and acquisition devices. Therefore, the accuracy for CMR images was expected to be lower than that of, for example, MNIST dataset images.

Another achieved goal of this study was to increase the speed of deep learning-based models. For this purpose, a Fourier-based convolutional model was proposed. Based on the analysis of the results, these networks could perform well while increasing training and testing speed. Using the model in the frequency domain eliminated the need to define fixed-size kernels in convolution operations. The reason is that the kernel size was considered the same size as the input image. Considering kernels of the same size as the image would also extract global features from the images. Besides, there was no need to move the sliding window over the entire image, which is time-consuming. Large-size images can also be used in these models because the complexity of feature extraction operations is much less than the spatial domain. According to the experiments performed in this study, the desired accuracy can be achieved in a shorter time by increasing the number of samples.

The datasets used in this study include magnitude images, which means no scanner-acquired k-space data from multi-channel receiver coils were available. So creating k-space using Fourier transform from these data can only mimic k-space due to the lack of access to phase data. This issue is the most critical limitation of the study. Also, some cases that may cause image corruption during imaging cannot be simulated with the proposed methods. For example, only bulk motions can be simulated with these techniques, and deformable motions cannot be implemented. Therefore, although the proposed methods for adding artefacts to images do not exactly correspond to reality, they can simulate something quasi-real.

Considering that in the current study, k-space data from multi-channel receiver coils were unavailable, the frequency domain model's behaviour is unknown in real-time applications for image quality assessment. However, the results obtained on the k-space achieved from the Fourier transform of the magnitude images showed that the proposed frequency space model has the potential for real-time applications. Of course, this issue requires more studies on the scanner-acquired k-space data. The cost of image reconstruction using the Fourier transform is avoided, and the quality can be assessed immediately after filling the k-space, assuming the efficiency of the frequency model on the original k-space data.

CMR image quality control involves examining the full heart coverage, signal-to-noise ratio, and imaging artefacts. A different approach is taken to explore each of these cases. In some studies [8-10], whole heart or left ventricle coverage was examined. These studies investigated short-axis cine CMR volumes for the presence or absence of apex and basal slices. Other researches [5-7] have focused on CMR imaging artefacts. The proposed models of those studies are 3D models based on the evaluation of artefacts in a 3D CMR volume. Since physicians ultimately examine CMR images in a 2D view, the current study used a 2D model to identify the artefacts.

Evaluations were also done on data with real artefacts to check the effectiveness of the proposed models for clinical applications. The results from applying the models to the data with real artefacts showed a slight decrease in the obtained metrics. However, the models' performance on these data was almost similar to the performance on the synthetic degraded data. An important issue that should be considered in analysing the results is that when the number of data used to train the models has increased, the proposed models showed better performance and more stability due to the data-hungry nature of deep learning models. The values of the obtained metrics are reduced in the other datasets due to their smaller volume, and in the frequency model, the recall is also reduced compared to the precision. Therefore, although we used different datasets to show the efficiency of the proposed models, the larger training set led to better model stability. Of course, other factors such as spatial resolution, imaging parameters, amount of image degradation when adding artefacts to reference images (which are considered random for diversity in this study), and image contents (the UCIII dataset consists of small animal CMR images) can also affect the performance of the models on YU, UCIII, and CMR-Tehran datasets.

5- Conclusion

This study proposes two models in the spatial and frequency domains for CMR image quality assessment. The proposed models can recognise the four common artefacts, including respiratory motion, cardiac motion, Gibbs ringing and aliasing, by receiving the input image or k-space. Besides, the problem of lack of access to annotated data was addressed by proposing a domain adaptation method. The analysis of the results obtained on four different datasets shows that the proposed models can properly detect the mentioned artefacts. Despite the poor performance of the proposed domain adaptation method, its results can be used as a baseline for further studies. Presenting more comprehensive models covering more artefacts and using new methods such as meta-learning will be among the purposes of the future study. Developing a hybrid model that takes advantage of both spatial and frequency domains for CMR image quality assessment and the proposed frequency domain model evaluation on the scanner-acquired k-space data are also potential future works.

Acknowledgements
This research has been done using the UK Biobank database under Application 11350.

Conflict of Interest Statement
There are no conflicts of interest to declare.

References
[1] W. P. Bandettini and A. E. Arai, "Advances in clinical applications of cardiovascular magnetic resonance imaging," *Heart,* vol. 94, no. 11, pp. 1485-1495, 2008.
[2] A. Kumar, D. J. Patton, and M. G. Friedrich, "The emerging clinical role of cardiovascular magnetic resonance imaging," *Canadian Journal of Cardiology,* vol. 26, no. 6, pp. 313-322, 2010.
[3] M. Salerno *et al.*, "Recent advances in cardiovascular magnetic resonance: techniques and applications," *Circulation: Cardiovascular Imaging,* vol. 10, no. 6, p. e003951, 2017.
[4] L. S. Chow and R. Paramesran, "Review of medical image quality assessment," *Biomedical signal processing and control,* vol. 27, pp. 145-154, 2016.
[5] G. Tarroni *et al.*, "Learning-based quality control for cardiac MR images," *IEEE transactions on medical imaging,* vol. 38, no. 5, pp. 1127-1138, 2018.
[6] G. Tarroni *et al.*, "Large-scale Quality control of cardiac imaging in population Studies: Application to UK Biobank," *Scientific reports,* vol. 10, no. 1, pp. 1-11, 2020.
[7] I. Oksuz *et al.*, "Automatic CNN-based detection of cardiac MR motion artefacts using k-space data augmentation and curriculum learning," *Medical image analysis,* vol. 55, pp. 136-147, 2019.
[8] L. Zhang *et al.*, "Automatic assessment of full left ventricular coverage in cardiac cine magnetic resonance imaging with fisher-discriminative 3-D CNN," *IEEE Transactions on Biomedical Engineering,* vol. 66, no. 7, pp. 1975-1986, 2018.
[9] L. Zhang, M. Pereañez, S. K. Piechnik, S. Neubauer, S. E. Petersen, and A. F. Frangi, "Multi-input and dataset-invariant adversarial learning (MDAL) for left and right-ventricular coverage estimation in cardiac MRI," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2018: Springer, pp. 481-489.
[10] L. Zhang, A. Gooya, and A. F. Frangi, "Semi-supervised assessment of incomplete LV coverage in cardiac MRI using generative adversarial nets," in *International Workshop on Simulation and Synthesis in Medical Imaging*, 2017: Springer, pp. 61-68.
[11] M. Osadebey, M. Pedersen, D. Arnold, and K. Wendel-Mitoraj, "Image quality evaluation in clinical research: A case study on brain and cardiac mri images in multi-center clinical trials," *IEEE journal of translational engineering in health and medicine,* vol. 6, pp. 1-15, 2018.
[12] S. Nabavi, M. Hashemi, M. Ebrahimi Moghaddam, A. A. Abin, and A. F. Frangi, "Automated cardiac coverage assessment in cardiovascular magnetic resonance imaging using an explainable recurrent 3D dual-domain convolutional network," *Medical Physics,* vol. 51, no. 12, pp. 8789-8803, 2024.
[13] S. Nabavi, H. Simchi, M. E. Moghaddam, A. A. Abin, and A. F. Frangi, "A generalised deep meta-learning model for automated quality control of cardiovascular magnetic resonance images," *Computer Methods and Programs in Biomedicine,* vol. 242, p. 107770, 2023.
[14] S. E. Petersen *et al.*, "UK Biobank's cardiovascular magnetic resonance protocol," *Journal of cardiovascular magnetic resonance,* vol. 18, no. 1, pp. 1-7, 2015.
[15] A. Andreopoulos and J. K. Tsotsos, "Efficient and generalizable statistical models of shape and appearance for analysis of cardiac MRI," *Medical Image Analysis,* vol. 12, no. 3, pp. 335-357, 2008.
[16] J. F. Abascal, P. Montesinos, E. Marinetto, J. Pascau, and M. Desco, "Comparison of total variation with a motion estimation based compressed sensing approach for self-gated cardiac cine MRI in small animal studies," *PloS one,* vol. 9, no. 10, p. e110594, 2014.
[17] T. Budrys, V. Veikutis, S. Lukosevicius, R. Gleizniene, E. Monastyreckiene, and I. Kulakiene, "Artifacts in magnetic resonance imaging: how it can really affect diagnostic image quality and confuse clinical diagnosis?," *Journal of Vibroengineering,* vol. 20, no. 2, pp. 1202-1213, 2018.
[18] P. F. Ferreira, P. D. Gatehouse, R. H. Mohiaddin, and D. N. Firmin, "Cardiovascular magnetic resonance artefacts," *Journal of Cardiovascular Magnetic Resonance,* vol. 15, no. 1, pp. 1-39, 2013.
[19] B. Lorch, G. Vaillant, C. Baumgartner, W. Bai, D. Rueckert, and A. Maier, "Automated detection of motion artefacts in MR imaging using decision forests," *Journal of medical engineering,* vol. 2017, 2017.
[20] L. F. Czervionke, J. M. Czervionke, D. L. Daniels, and V. M. Haughton, "Characteristic features of MR truncation artifacts," *American journal of roentgenology,* vol. 151, no. 6, pp. 1219-1228, 1988.
[21] J. W. Gibbs, "Fourier's series," *Nature,* vol. 1539, no. 59, p. 606, 1899.
[22] A. C. Bovik and S. T. Acton, "Basic linear filtering with application to image enhancement," in *The Essential Guide to Image Processing*: Elsevier, 2009, pp. 225-239.
[23] S. Y. Huang, R. T. Seethamraju, P. Patel, P. F. Hahn, J. E. Kirsch, and A. R. Guimaraes, "Body MR imaging: artifacts, k-Space, and solutions," *Radiographics,* vol. 35, no. 5, pp. 1439-1460, 2015.
[24] A. Farahani, S. Voghoei, K. Rasheed, and H. R. Arabnia, "A brief review of domain adaptation," *Advances in data science and information engineering,* pp. 877-894, 2021.
[25] W. M. Kouw and M. Loog, "A review of domain adaptation without target labels," *IEEE transactions on pattern analysis and machine intelligence,* vol. 43, no. 3, pp. 766-785, 2019.
[26] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International conference on machine learning*, 2015: PMLR, pp. 1180-1189.

[27] H. Pratt, B. Williams, F. Coenen, and Y. Zheng, "Fcnn: Fourier convolutional neural networks," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2017: Springer, pp. 786-798.
[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," presented at the International Conference for Learning Representations (ICLR), May 2015.
[29] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
[30] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of machine learning research,* vol. 9, no. 11, 2008.
[31] B. Gholami, P. Sahu, O. Rudovic, K. Bousmalis, and V. Pavlovic, "Unsupervised multi-target domain adaptation: An information theoretic approach," *IEEE Transactions on Image Processing,* vol. 29, pp. 3993-4002, 2020.

Shahabedin Nabavi received his Ph.D. in Artificial Intelligence from Shahid Beheshti University in 2024, with a dissertation on *Automated Cardiovascular MR Image Quality Assessment*. His research interests include **medical image analysis, deep learning, generative models, and quality assessment of cardiovascular imaging.**

Hossein Simchi received his M.Sc. in Artificial Intelligence from the Faculty of Computer Science and Engineering, Shahid Beheshti University. His research interests include **artificial intelligence, machine learning, deep learning, interpretable AI, and healthcare applications**.

Mohsen Ebrahimi Moghaddam is a Professor in the Faculty of Computer Science and Engineering at Shahid Beheshti University, Iran. His research interests include **image processing, computer vision, artificial intelligence, and operating systems**.

Ahmad Ali Abin is an Associate Professor in the Faculty of Computer Science and Engineering at Shahid Beheshti University, Iran. His research interests include **machine learning, deep learning, and forecasting**.

Alejandro F. Frangi is the Bicentennial Turing Chair in Computational Medicine at the University of Manchester, UK, where he also holds a RAEng Chair. He is affiliated with KU Leuven and the Alan Turing Institute. His research interests include **medical image computing, computational medicine, and in-silico trials**.

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

Automated Recognition of Marine Thermal Patterns Using Deep Learning

Alireza Sharifi✉, **Code Orcide:**
Department of Surveying Engineering, Faculty of Civil, Water and Environmental Engineering, Shahid Beheshti University, Tehran, Iran
Alireza Vafaeinejad**, Code Orcide:**
Department of Surveying Engineering, Faculty of Civil, Water and Environmental Engineering, Shahid Beheshti University, Tehran, Iran

Abstract
Sea Surface Temperature (SST) data reveal the temporal and spatial distribution of warm anticyclonic eddies and cold cyclonic eddies, impacting ocean behavior. The SST combined products attained adequate decisions to permit the recognition of mesoscale eddies with the introduction of altimeter operations and the availability of two or more altimeters at the same time. Climate change impacts ocean circulation and atmospheric anomalies linked to SST variations. Ocean eddies, vital for material and energy transport, require precise identification to advance oceanography. This study uses SST data from CMEMS in the Atlantic to introduce EddyNet, a deep-learning model for automatic eddy detection and classification. EddyNet's encoder-decoder architecture includes a pixel-wise classification layer, labeling each pixel as "0" (non-eddy), "1" (anticyclonic), or "2" (cyclonic). The high-resolution feature representation outperforms existing models, marking a significant leap in eddy detection accuracy and reliability. This study introduces EddyNet, a deep-learning model based on the U-Net architecture for automatic eddy detection and classification using Sea Surface Temperature (SST) data. The model was trained and evaluated on satellite imagery from the Copernicus Marine Environment Monitoring Service (CMEMS), achieving a training accuracy of 78.55%, a Dice score of 31.99%, and a precision of 0.9259. The recall values for different classes indicate that the model correctly identifies 99.51% of non-eddy pixels, 51.57% of anticyclonic eddy pixels, and 57.19% of cyclonic eddy pixels. These results demonstrate the effectiveness of deep learning in mesoscale eddy detection and highlight the potential for further optimization in classifying eddy structures with higher precision.

INDEX TERMS — Mesoscale Eddy identification, U-Net, Remote Sensing, Deep Learning, pixel- wise classification

Introduction

IN recent years, crucial work in understanding climate change includes detecting global warming and monitoring sea level rise [1]. Deep learning and machine learning techniques using spatial analysis provide evidence of rising oceans. Reliable modeling of ocean dynamics, including currents and turbulence, is essential for forecasting climate change [2]. SST (Sea Surface Temperature) data reveal the temporal and spatial distribution of warm anticyclonic eddies and cold cyclonic eddies, impacting ocean behavior [3]. Oceanic mesoscale eddies, revolving water vortices, are crucial for understanding Earth's environmental conditions [4]. Detecting and monitoring these eddies is vital for creating accurate climate prediction models.

Mesoscale ocean eddies are capable of influencing the behavior of the atmosphere at mesoscales primarily through kinetic energy and have a regional impact on near-surface wind, cloud characteristics, and rainfall [2]. As a result, understanding the transmission and features of mesoscale ocean eddies and using this information to identify and analyze them is crucial for both global climate change and geological and oceanography purposes [5]. Rising sea levels from global warming have increased the need to monitor oceans. AI, including DL and ML techniques, is now essential for optimizing data analysis. Studies use physical and chemical parameters to identify eddy networks [6]. In terms of computer vision problems, object identification, and image segmentation sequences, the implementation of the deep neural network, U-Net, scores highly [7]. This study bridges eddy identification and DL, connecting key areas like oceanography, climate change, remote sensing, computer vision, DL, and ML methods [8].

Significant currents and associated instability on the oceanic mesoscale (≈ 100 km) play a key role in determining the ocean's stratification [9] and aid in the climate system's poleward heat transfer [10]. It is recognized that these circulations' quasi-stationary component, or what is left over after multi-year averaging, exhibits significant thermal surface manifestations in the form of narrow fronts in SST. The atmospheric's border layer is seen to be highly connected to them [11], and it's possible that they have an effect on the environment [12]. Mesoscale eddies are widely dispersed over the ocean's surface and are important for moving mass, energy, heat, and nutrients between different ocean basins [13].

Submit Date: 2025-03-09
Revise Date: 2025-07-21
Accept Date: 2025-09-17
✉ Corresponding author

The SST combined products attained adequate decisions to permit the recognition of mesoscale eddies with the introduction of altimeter operations and the availability of two or more altimeters at the same time [14]. This study uses SST maps to identify three eddy types: cyclonic (negative SLA), anticyclonic (positive SLA), and no eddies. A U-Net-based CNN analyzes preprocessed Atlantic Ocean satellite images, including Gulf Stream data (2018–2020) provided by Copernicus Marine Service [15].

Neural networks, like the proposed ST-U-Net, improve automated eddy detection by enhancing accuracy and efficiency, surpassing traditional threshold-based methods [16]. The purpose of this document is to describe in detail the development process and goals of a patient Mesoscale eddy detection and classification from SST maps with deep neural networks. The primary content care for the mesoscale eddy detection and classification from SST maps. This work evaluates a deep learning-based architecture for automated eddy detection and classification from SST maps based on three classes.

This article is divided into five segments. In section Material and methodology an outline of the suggested approach is presented. Also, section of Related Works represents a state of art from previous studies regarding same as this research and Results and Discussion emphasize implementation, findings, and derived outcomes and this section has been shown innovations and limitations in this paper and outlines of prospective research are discussed. And section Conclusion offers final remarks and applications of the proposed methodology.

Related Works-state of art

DL's reliance on labeled data is costly, but data augmentation improves performance. Traditional eddy detection methods are subjective, while this study uses YOLOF for mesoscale eddy detection, enhancing accuracy, reducing threshold reliance, and increasing speed [17]. The 3D neural network based on ResNet classifies eddies as anticyclonic, cyclonic, or none, incorporating geographical and dynamic variables. Using 3D convolutions and pooling, it models 3D eddy data and achieves improved detection by leveraging altimetry-calibrated vertical characteristics. Trials confirm its effectiveness [18].

This study uses DL with an altimetric-based area suggestion to detect eddy signatures in SST data. A CNN-based classifier effectively identifies eddies, even with limited manually labeled data. The method helps oceanographers validate altimetric eddy detection using SST images despite data complexity and automated labeling challenges [19]. Additionally, Kemker and Kanan [20] developed a CNN-based architecture. CNNs excel at processing grid-based topology with spatially dependent elements like photos, making them potential visual identification tools [21].

This study evaluates U-Net, a convolutional encoder-decoder, for pixel-wise classification of SST images: '0' for Non-eddy, '1' for anticyclonic, and '2' for cyclonic eddies. Training datasets and EddyNet weight files were generated, highlighting the collaboration between RS and ML communities in advancing SST image segmentation. The U-Net architecture and implementation models are specifically discussed in the dedicated Subsection of Eddy Identification By U-Net of the study.

Material and Methodology

This study introduces a neural network approach, specifically U-Net, for detecting mesoscale ocean eddies using SST image processing and satellite altimetry data. It outlines the study area, data preprocessing, and the proposed framework, highlighting key components and their interactions.

Study Area

The study area chosen for examination encompasses a significant expanse within the North Atlantic Ocean, ranging between latitudes 17.32° N to 55.50° N and longitudes 41.13° W to 95.63° W, as depicted in the accompanying Figure 1 [22]. The region spans major geographical features like Hudson Bay, the Gulf of Mexico, and the Labrador Sea, extending from West Africa to the Americas. The North Atlantic, though relatively level, includes the Gulf Stream, a warm current flowing northeast along North America's coast, significantly impacting nearby climates [23].

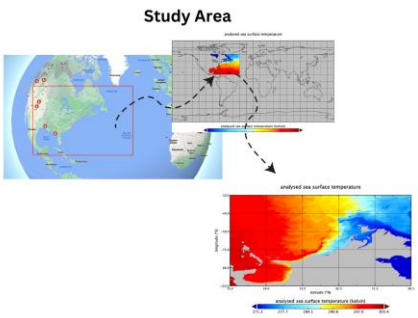


Fig. 1. Study area: North Atlantic Ocean.

Within this vast area, a variety of marine life thrives, encompassing different fish species, marine creatures, and seabirds [23]. The North Atlantic's marine ecosystems result from interactions among water currents and masses. Historically, it has been a key trade route linking Europe and North America. Notable locations in this region include the Azores, parts of the Caribbean, and the eastern coasts of the U.S. and Canada [2]. The North Atlantic significantly impacts European temperatures via the North Atlantic Drift, part of the Gulf Stream, which carries warm water from the Gulf of Mexico to the U.S. and Canadian coasts. Its surface layer shows distinct salinity, temperature, and movement, with flow increasing fivefold due to Sargasso Sea water and

shelf interactions. Seasonal variations influence zooplankton distribution [23].

Satellite Data Observations

The foundational dataset underpinning this research is the ESA SST CCI and C3S global SST Reprocessed product. This dataset (shown in Table 1) constitutes the pivotal component in the pursuit of understanding and analyzing SST dynamics at a global scale. The dataset offers a comprehensive depiction of the daily average SST at a depth of 20 cm, manifested in a spatial grid resolution of $0.05^{\circ} \times 0.05^{\circ}$. To yield the ESA SST CCI and C3S level 4 analyses, an intricate processing pipeline known as the Operational Sea Surface Temperature and Sea Ice Analysis (OSTIA) system is harnessed. This system meticulously processes the input satellite data, culminating in a daily analysis of SST that boasts an impressive grid resolution of approximately 5 km or 1/20◦. This heightened resolution significantly enhances the dataset's ability to capture fine-scale temperature fluctuations in oceanic regions [20].

TABLE I
SATELLITE DATASET INFORMATION CREDITED BY COPERNICUS [20].

Dataset Classified Information	
Full name	ESA SST CCI and C3S reprocessed sea surface temperature analyses
Product ID	SST GLO SST L4 REP OBSERVATIONS 010 024
Source	Satellite observations
Spatial extent	Global OceanLat -90° to 90°Lon -180° to 180°
Spatial resolution	0.05° × 0.05°
Temporal extent	1 Sep 1981 to 31 Oct 2022
Temporal resolution	Daily
Processing level	Level 4
Variables	Sea ice area fractionSea water temperature (T)
Feature type	Grid
Projection	WGS 84 (EPSG:4326)
Format	NetCDF-4

This study uses a three-year dataset (2018–2020) of daily SST satellite images provided by Copernicus Marine Environment to train and test the U-Net algorithm. Input images are resized to 512 x 512 pixels for memory efficiency. The dataset includes 2,190 cyclonic and anticyclonic images, with 90% used for training and a stratified subset from January to May 2018 reserved for testing. Validation is conducted using 12 netCDF files with SST data.

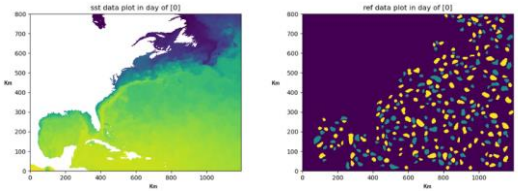


Fig. 2. Visualization of trained dataset and their masks before importing the designed network.

A pivotal phase involves the meticulous crafting of segmentation masks for training patches. This intricate procedure entails the generation of polygonal shapes by projecting the SST data onto the nearest lattices within the 0.05° grid. Pixels encompassed within each polygon are subsequently assigned labels corresponding to the class of the polygon, signifying the nature of the eddy: '0' for non-eddy/land/no data, '1' for anticyclonic eddies, and '2' for cyclonic eddies. This meticulous amalgamation of data sets, aligned with rigorous preprocessing and labeling, forms the bedrock of our exploration into the U-Net architecture's capabilities and has been discussed in the following.

Eddy Identification By U-Net

Several physical or geometrical criteria-based autonomous eddy detection techniques have been established, and they may be categorized into three groups: the physical attribute approach, such as the Okubo-Weiss parameter method [24], the flow geometry method, which involves the winding-angle method [25], the vector geometry method [26], [27], the SST-based method [28], [29], But for three explanations, not all methods can be used to locate mesoscale eddies in the Atlantic Ocean. primarily because it can eliminate noise and excessive eddy detections, the SST-based technique outperforms the Okubo-Weiss parameter method [2]. Secondly, the current observable data in the Atlantic Ocean cannot provide the higher resolution data needed by the flow geometry algorithms to get an accurate flow field to locate eddies [21]. As a result, the SST-based approach created by U-net [28], [29] has been employed in this work as the DL method.

Eddy Identification By U-Net

A parenthetical statement at the end of a sentence is The EddyNet architecture draws inspiration from the renowned U-Net framework [30]. The inception of EddyNet involves a dual-track progression, commencing with an encoding (downsampling)

trajectory spanning three distinctive stages. Each stage features a tandem of 3 × 3 convolutional layers, sequenced by either the established duo of classical ReLU activation and Batch Normalization, referred to as EddyNet [31]. This is followed by a 2 × 2 max pooling layer, orchestrating a halving of input resolution. Meanwhile, the decoding (upsampling) trajectory employs transposed convolutions, also known as decon- volutions, to reinstate the initial resolution. EddyNet, akin to U-Net, seamlessly integrates skip connections, facilitating the transmission of information from the contracting path Initial attempts utilizing the original U-Net architecture revealed overfitting tendencies, as the architecture's capacity exceeded the available training samples. A meticulous iterative process, coupled with hyperparameter fine-tuning, culminated in a definitive selection: a streamlined 3-stage design, each stage equipped with [64, 128, 256, 512] filters. This design imbues EddyNet with a parsimonious parameter count compared to prevalent architectures, leading to economical memory utilization. EddyNet's proficiency in mastering data is underscored by its potential for overfitting, affirming its aptitude for unraveling the intricate nonlinearities embedded within eddy detection and classification, thereby addressing the inverse problem.

To enhance clarity, we present sample images illustrating the transformation from raw SST satellite data to processed images and model predictions. The input data consists of high-resolution SST images from the Copernicus Marine Service, resized to 512 × 512 pixels and normalized for consistency. During preprocessing, segmentation masks were generated, assigning class labels: '0' (non-eddy), '1' (anticyclonic eddy), and '2' (cyclonic eddy). The U-Net model then produced pixel-wise classification masks, accurately identifying eddy structures.

TABLE II
EDDY IDENTIFICATION BY THE DESIGNED U-NET ALGORITHM

The designed U-Net Architecture Algorithm
Initialize U-Net architecture components
for each down-sampling stage **do**
Create DoubleConv block for down-sampling
Adjust input channels for next stage
end for
for each up-sampling stage **do**
Create transposed convolution layer
Create DoubleConv block for up-sampling
end for
Create DoubleConv bottleneck layer
Create final convolution layer
Initialize skip connections list
for each down-sampling block **do**
Apply DoubleConv block
Store intermediate output in skip connections
Apply max pooling
end for
Apply bottleneck DoubleConv block
Reverse skip connections list
for each up-sampling block **do**
Apply transposed convolution
Retrieve corresponding skip connection
if output shape differs from skip connection shape **then**
Resize output to match skip connection shape
end if
Concatenate skip connection and output
Apply DoubleConv block
end for
Apply final convolution layer
return Final output

The U-Net architecture, developed with Python 3.9.12 using TensorFlow and Torch, resizes images to 512 × 512 pixels and converts them to grayscale. Three U-Net model variations were trained and evaluated using metrics like dice coefficient, accuracy, and binary cross-entropy. Segmentation masks labeled as "no eddy" (0), "anticyclonic eddy" (1), or "cyclonic eddy" (2) were created from manually labeled data. The dataset was divided into training, validation, and testing subsets to ensure balance. Once trained, the U-Net model can classify unseen SST images.

The selection of U-Net as the primary deep learning architecture for mesoscale eddy detection was driven by its superior performance in pixel-wise classification and boundary delineation, which are crucial for accurately identifying eddy structures. Unlike VGG16, VGG19, and ResNet, which are primarily designed for image classification and feature extraction, U-Net is optimized for semantic segmentation, making it more suitable for detecting intricate patterns in Sea Surface Temperature (SST) maps. Additionally, SegNet and Fully Convolutional Networks (FCN), though effective in segmentation, do not retain fine spatial details as efficiently as U-Net. The skip connections in U-Net allow feature maps from the encoder to be directly transferred to the decoder, preserving high-resolution details necessary for differentiating between eddy types. Moreover, U-Net requires fewer training samples compared to deeper networks like DenseNet, making it more adaptable to remote sensing datasets with limited labeled data. These advantages make U-Net a compelling choice for SST-based eddy detection, ensuring accurate segmentation while maintaining computational efficiency.

Hyperparameters and Metric Values

The dice score/coefficient is utilized to assess the efficacy of image segmentation. It serves as a metric for measuring the similarity between two items. The score is calculated by dividing the size of the intersection of two segmentations by the total size of the two objects [32].

$$Dice\ Coef\ Function = \frac{2\ .|X \cap Y|}{|X \cup Y|} \quad (1)$$

The dice coefficient measures spatial overlap between predicted segmentation (Y) and ground truth (X). An F1-based loss, addressing class imbalances, is tested for multi-class segmentation tasks. This research evaluates U-Net with F1-based loss against traditional U-Net variants, enhancing segmentation accuracy assessment [33].

$$F1\ Score = 2\ .\ \frac{Precision\ .\ Recall}{Precision + Recall} \quad (2)$$

This equation considers both precision and recall, offering a well-rounded assessment of a model's performance for a particular class. The mathematical calculations for both precision and recall are provided below.

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

Precision is the ratio of true positive predictions to the total predicted positives. It measures how many of the positive predictions were actually correct. Recall represents the proportion of correct positive predictions to the total number of actual positives. It assesses the model's capability to accurately identify all instances of the positive class.

$$Recall = \frac{TP}{TP+FN} \quad (4)$$

Results And Discussion

The results section details the U-Net algorithm's performance in detecting and segmenting mesoscale eddies in the Atlantic Ocean from SST images. Trained on 1,888 reference images, the algorithm's predictions are evaluated using metrics such as accuracy, loss, recall, precision, F1 score, and other hyperparameters in section 3. The U-Net algorithm, optimized using parameters like learning rate and kernel size, effectively detects and segments mesoscale eddies in ocean RS images. These findings enhance methods for studying ocean dynamics and their climate impact. Testing on data subsets ensures error detection, preserves resources, and enables hyperparameter tuning while avoiding overfitting, balancing efficiency and reliability.

To provide a more comprehensive analysis of the results, we present a detailed breakdown of the model's performance across different evaluation metrics. The U-Net model achieved a training accuracy of 78.55%, with a Dice score of 31.99%, demonstrating moderate segmentation overlap with reference masks. The precision value of 0.9259 indicates that the model effectively minimizes false positives, while the F1-score of 0.7805 highlights a balanced performance between precision and recall. The recall values for different classes show that the model correctly identifies 99.51% of non-eddy pixels (Class 0), 51.57% of anticyclonic eddy pixels (Class 1), and 57.19% of cyclonic eddy pixels (Class 2). This suggests that the model performs best at detecting non-eddy regions but exhibits some challenges in distinguishing between the two types of eddies. Additionally, analysis of loss values indicates that the model maintains a stable learning process, with a training loss of 0.07337, reflecting a good fit to the training data. These results highlight the effectiveness of U-Net in eddy detection while suggesting areas for improvement, such as refining feature extraction techniques for better differentiation between eddy types.

The hyper-parameters, represented in Table 3, Key parameters were selected to optimize model performance. A learning rate of 0.001 balances convergence speed and stability, avoiding overshooting or slow convergence. Images were resized to 128 × 128 for computational efficiency and spatial detail. The model was trained for 80,000 epochs with a batch size of 3 and 2 workers to balance memory and speed. A kernel size of 3 with padding and a stride of 1 captures both local and global features effectively.

TABLE III
THE HYPERPARAMETERS VALUES IN THE U-NET METHOD TRAINING

Hyperparameters	
Learning rate	0.001
Epochs	80000
Image width and height	128
Batch size	3
Kernel size	3
Number of workers	2
Padding	1
Stride	1

The model achieved promising results has been shoen in Table 4 with a Dice score of nearly 0.8 and a loss of about 0.2 on the test dataset, demonstrating the effectiveness of the chosen hyper-parameters. Upon analysis of this case, it is found that the dice score initially reached a value of approximately 36.7 percent after 10k epochs but later dropped to 31.99 percent. Additionally, the training loss reached a value of about 0.07 which indicates that the model is effectively learning the features from the input images.

TABLE IV
THE RESULTS VALUES IN THE U-NET TRAINING

Results	
Training Accuracy	78.55%

Training Loss	0.07337
Dice Score	31.99%
Precision	0.9259
F1 Score	0.7805
Recall Class 0	0.9951
Recall Class 1	0.5157
Recall Class 2	0.5719

However, further exploration is required to identify the reasons for the drop in Maybe, The study can say that the Dice score is a measure of the similarity between the predicted segmentation map and the reference segmentation map. A Dice score of 31.99 percent means that the predicted segmentation map is 31.99 percent similar to the reference segmentation map. Based on results after training 80k epochs, the designed U-Net can predicted Eddy network in study area with training accuracy of 78.55%, the comparision between reference and predicted images has been provided in Figure 3.

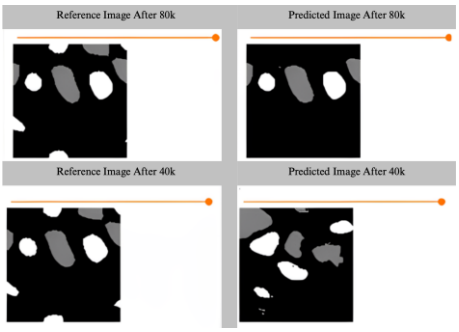


Fig. 3. Refrence and Predicted images in the designed model.

Furthermore, the training loss of 0.07 indicates that the model has achieved a good level of convergence during training. A lower training loss means that the model is better at predicting the correct segmentation labels. However, it is important to note that a low training loss does not always guarantee good performance on the test data. The model's performance on the test data should always be evaluated using metrics such as the Dice score.

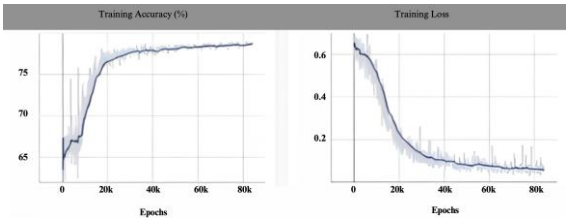


Fig. 4. Training accuracy and loss after training 80k epochs in the U-net model.

Conclusion

This research explores advancements in DL-based image segmentation for ocean remote sensing, specifically identifying and classifying eddies from SST maps. It introduces EddyNet, a deep neural network inspired by prominent computer vision designs, incorporating U-Net for improved boundary detection and mesoscale eddy identification in the northern Atlantic Ocean. The high-resolution feature representation outperforms existing models, marking a significant leap in eddy detection accuracy and reliability. Future directions include developing a 3D model, integrating additional data like SSH, expanding EddyNet's geographic scope, and enhancing post-processing methods for validation and eddy tracking, aiming to broaden its global applicability and effectiveness.

REFERENCES

[1] C. Chu et al., "Effect of the tropical Pacific and Indian Ocean warming since the late 1970s on wintertime Northern Hemispheric atmospheric circulation and East Asian climate interdecadal changes," Clim Dyn, vol. 50, no. 7–8, pp. 3031–3048, Apr. 2018, doi: 10.1007/s00382-017-3790-y.

[2] I. Frenger, N. Gruber, R. Knutti, and M. Münnich, "Imprint of Southern Ocean eddies on winds, clouds and rainfall," Nature Geoscience 2013 6:8, vol. 6, no. 8, pp. 608–612, Jul. 2013, doi: 10.1038/ngeo1863.

[3] Y. Liu, Q. Zheng, and X. Li, "Characteristics of Global Ocean Abnormal Mesoscale Eddies Derived From the Fusion of Sea Surface Height and Temperature Data by Deep Learning," Geophys Res Lett, vol. 48, no. 17, Sep. 2021, doi: 10.1029/2021GL094772.

[4] V. S. Gulakaram, N. K. Vissa, and P. K. Bhaskaran, "Mesoscale eddies with anomalous sea surface temperature and its relation with atmospheric convection over the North Indian Ocean," International Journal of Climatology, vol. 43, no. 7, pp. 3094–3113, Jun. 2023, doi: 10.1002/joc.8018.

[5] D. D'Alimonte, "Detection of Mesoscale Eddy-Related Structures Through Iso-SST Patterns," IEEE Geoscience and Remote Sensing Letters, vol. 6, no. 2, pp. 189–193, Apr. 2009, doi: 10.1109/LGRS.2008.2009550.

[6] R. Lguensat, M. Sun, R. Fablet, E. Mason, P. Tandeo, and G. Chen, "EddyNet: A Deep Neural Network For Pixel-Wise Classification of Oceanic Eddies", Accessed: Jul. 24, 2023. [Online]. Available: https://github.com/redouanelg/EddyNet.

[7] G. Xu et al., "Oceanic Eddy Identification Using an AI Scheme," Remote Sens (Basel), vol. 11, no. 11, p. 1349, Jun. 2019, doi: 10.3390/rs11111349.

[8] Z. Duo, W. Wang, and H. Wang, “Oceanic Mesoscale Eddy Detection Method Based on Deep Learning,” Remote Sens (Basel), vol. 11, no. 16, p. 1921, Aug. 2019, doi: 10.3390/rs11161921.
[9] P. B. Rhines and W. R. Young, “Homogenization of potential vorticity in planetary gyres,” J Fluid Mech, vol. 122, no. 1, p. 347, Sep. 1982, doi: 10.1017/S0022112082002250.
[10] S. R. Jayne and J. Marotzke, “The oceanic eddy heat transport,” J Phys Oceanogr, vol. 32, no. 12, pp. 3328–3345, 2002, doi: 10.1175/1520-0485(2002)032<3328:TOEHT>2.0.CO;2.
[11] D. B. Chelton, M. G. Schlax, M. H. Freilich, and R. F. Milliff, “Satellite Measurements Reveal Persistent Small-Scale Features in Ocean Winds,” Science (1979), vol. 303, no. 5660, pp. 978–983, Feb. 2004, doi: 10.1126/SCIENCE.1091901.
[12] Q. Song, T. Hara, P. Cornillon, and C. A. Friehe, “A comparison between observations and MM5 simulations of the marine atmospheric boundary layer across a temperature front,” J Atmos Ocean Technol, vol. 21, no. 2, pp. 170–178, 2004, doi: 10.1175/1520-0426(2004)021<0170:ACBOAM>2.0.CO;2.
[13] D. B. Chelton, M. G. Schlax, and R. M. Samelson, “Global observations of nonlinear mesoscale eddies,” Prog Oceanogr, vol. 91, no. 2, pp. 167–216, Oct. 2011, doi: 10.1016/j.pocean.2011.01.002.
[14] L. Yuan, F. Tian, S. Xu, C. Zhou, and J. Chen, “Three-dimensional mesoscale eddy identification and tracking algorithm based on pressure anomalies,” J Oceanol Limnol, vol. 39, no. 6, pp. 2153–2166, Nov. 2021, doi: 10.1007/S00343-021-0309-5/METRICS.
[15] “ESA SST CCI and C3S reprocessed sea surface temperature analyses | Copernicus Marine MyOcean Viewer.” Accessed: Jan. 07, 2024. [Online]. Available: https://data.marine.copernicus.eu/product/SST_GLO_SST_L4_REP_OBSERVATIONS_010_024/description
[16] X. He, Y. Zhou, J. Zhao, D. Zhang, R. Yao, and Y. Xue, “Swin Transformer Embedding UNet for Remote Sensing Image Semantic Segmentation,” IEEE Transactions on Geoscience and Remote Sensing, vol. 60, pp. 1–15, 2022, doi: 10.1109/TGRS.2022.3144165.
[17] L. Cao, D. Zhang, Q. Guo, and J. Zhan, “Ocean Mesoscale Eddies Identification Based on Yolof,” in IGARSS 2022 - 2022 IEEE International Geoscience and Remote Sensing Symposium, IEEE, Jul. 2022, pp. 1177–1180. doi: 10.1109/IGARSS46834.2022.9883834.
[18] B. Huang, L. Ge, X. Chen, and G. Chen, “Vertical Structure-Based Classification of Oceanic Eddy Using 3-D Convolutional Neural Network,” IEEE Transactions on Geoscience and Remote Sensing, vol. 60, pp. 1–14, 2022, doi: 10.1109/TGRS.2021.3103251.
[19] E. Moschos, O. Schwander, A. Stegner, and P. Gallinari, “Deep-SST-Eddies: A Deep Learning Framework to Detect Oceanic Eddies in Sea Surface Temperature Images,” in ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, May 2020, pp. 4307–4311. doi: 10.1109/ICASSP40776.2020.9053909.
[20] R. Kemker, C. Salvaggio, and C. Kanan, “High-Resolution Multispectral Dataset for Semantic Segmentation,” Mar. 2017, Accessed: Jan. 07, 2024. [Online]. Available: https://www.researchgate.net/publication/315456475_Deep_Neural_Networks_for_Semantic_Segmentation_of_Multispectral_Remote_Sensing_Imagery
[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2323, 1998, doi: 10.1109/5.726791.
[22] “Atlantic Ocean.” Accessed: Jan. 07, 2024. [Online]. Available: https://www.findlatitudeandlongitude.com/l/Atlantic+Ocean/5748743/
[23] A. McQuatters-Gollop et al., “Assessing the state of marine biodiversity in the Northeast Atlantic,” Ecol Indic, vol. 141, p. 109148, Aug. 2022, doi: 10.1016/J.ECOLIND.2022.109148.
[24] D. B. Chelton, M. G. Schlax, R. M. Samelson, and R. A. de Szoeke, “Global observations of large oceanic eddies,” Geophys Res Lett, vol. 34, no. 15, Aug. 2007, doi: 10.1029/2007GL030812.
[25] A. Chaigneau, A. Gizolme, and C. Grados, “Mesoscale eddies off Peru in altimeter records: Identification algorithms and eddy spatio-temporal patterns,” Prog Oceanogr, vol. 79, no. 2–4, pp. 106–119, Oct. 2008, doi: 10.1016/j.pocean.2008.10.013.
[26] C. Dong et al., “An oceanic cyclonic eddy on the lee side of Lanai Island, Hawaii,” J Geophys Res Oceans, vol. 114, no. 10, Oct. 2009, doi: 10.1029/2009JC005346.
[27] D. B. Chelton, M. G. Schlax, R. M. Samelson, and R. A. de Szoeke, “Global observations of large oceanic eddies,” Geophys Res Lett, vol. 34, no. 15, Aug. 2007, doi: 10.1029/2007GL030812.
[28] I. Bashmachnikov, D. Boutov, and J. Dias, “Manifestation of two meddies in altimetry and sea-surface temperature,” Ocean Science, vol. 9, no. 2, pp. 249–259, Mar. 2013, doi: 10.5194/OS-9-249-2013.
[29] S. D. Bachman, J. R. Taylor, K. A. Adams, and P. J. Hosegood, “Mesoscale and submesoscale effects on mixed layer depth in the Southern Ocean,” J Phys Oceanogr, vol. 47, no. 9, pp. 2173–2188, Sep. 2017, doi: 10.1175/JPO-D-17-0034.1.
[30] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 9351, pp. 234–241, 2015, doi: 10.1007/978-3-319-24574-4_28/COVER.
[31] “Qualitative comparison between U-Net, wide U-Net, and UNet++, showing... | Download Scientific Diagram.” Accessed: Dec. 25, 2023. [Online]. Available: https://www.researchgate.net/figure/Qualitative-comparison-between-U-Net-wide-U-Net-and-UNet-showing-segmentation_fig1_324574642
[32] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. J. Cardoso, “Generalised Dice overlap as a deep learning loss function for highly unbalanced segmentations,” Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 10553 LNCS, pp. 240–248, Jul. 2017, doi: 10.1007/978-3-319-67558-9_28.
[33] C. Goutte and E. Gaussier, “A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation,” Lecture Notes in Computer Science, vol. 3408, pp. 345–359, 2005, doi: 10.1007/978-3-540-31865-1_25/COVER.

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

Unlocking individual motor signatures using feature-based clustering of a graphomotor task

Zinat Zarandi✉, **Code Orcide:** 0000-0002-2488-4950
INSERM UMR1093-CAPS, UFR des Sciences du Sport, Université Bourgogne Franche-Comté, Dijon, France.
Amirreza Behmanesh, **Code Orcide:** 0009-0005-0783-1442
Department of Computer Engineering, Artificial Intelligent and Robotics, Amirkabir Kabir University of Technology, Tehran, Iran
Mohammad Mehdi Ebadzadeh, **Code Orcide:** 0000-0001-6466-5229
Department of Computer Engineering, Artificial Intelligent and Robotics, Amirkabir Kabir University of Technology, Tehran, Iran
Thierry Pozzo, **Code Orcide:** 0000-0002-0585-7965
INSERM UMR1093-CAPS, UFR des Sciences du Sport, Université Bourgogne Franche-Comté, Dijon, France.

***Abstract*—**Understanding individual motor signatures (IMS) is essential for personalized treatment and performance optimization. This study investigates the effectiveness of Fuzzy C-Means (FCM) clustering for identifying individual motor signatures from graphomotor tasks. We analyze various kinematic and geometric features, such as movement duration, velocity, and trajectory length, to reveal which aspects of motor behavior are most effective in distinguishing individuals. The results show that features like length of movement are particularly discriminative, while others, such as beta and velocity, offer weaker clustering outcomes.

Keywords—Motor behavior, Fuzzy C-Means clustering, hand-drawing tasks, motor signatures, feature selection.

INTRODUCTION

Understanding individual motor signatures (IMS) is essential in neurorehabilitation, biomechanics, and motor learning. IMS refers to distinct movement patterns shaped by neural, muscular, and cognitive factors. Accurately capturing IMS offers valuable insights for personalized treatment and performance optimization. However, capturing these unique motor patterns is challenging due to variability in motor behavior across individuals. Statistical analyses often fail to capture subtle differences, making clustering techniques like Fuzzy C-Means (FCM) a potential solution for distinguishing meaningful motor differences. This paper investigates the use of FCM clustering to capture IMS based on kinematic and geometric features from hand-drawing tasks. In particular, the study seeks to identify which features are most effective for distinguishing motor patterns, as well as the conditions under which certain features outperform others. We hypothesize that while some features will excel in clustering by capturing strong individual characteristics, others may offer weaker discriminative power due to their sensitivity to finer, less idiosyncratic aspects of movement [1], [2].

METHOD

Subjects

Nine healthy subjects (1 male, 8 female) aged 18–25 (2 left-handed, 7 right-handed) participated in the study. All subjects were free from conditions affecting cognitive or motor abilities. The Edinburgh Inventory was used to assess handedness. The study was approved by an ethical review board, and informed consent was obtained from all participants.

Experiment Procedure

Subjects were instructed to draw cloverleaf and eight-shaped patterns with their dominant hand in a counterclockwise direction on a paper sheet for 5 trials, each including 20 continuous repetitions. A Wacom tablet recorded the coordinates of each movement (temporal resolution: 200 samples/s; resolution: 1748 by 2551; active area: 210 × 297 mm). In total, each subject performed 100 repetitions.

Feature Extraction

The extracted kinematic and geometric features included: **Length**—the sum of the distance between successive points (cm), **Movement duration** (s), **Maximum velocity (max V)**—peak velocity during the movement (cm/s), **Beta**—the exponent in the relationship between velocity and curvature of the trajectory, **Correlation coefficient**—covariation between velocity and trajectory curvature. Data were preprocessed with a low-pass filter (0.07 Hz cut-off). Successive repetitions were separated using local velocity minima.

Data Analysis and Clustering

The dataset was normalized using min-max normalization. We applied FCM clustering, which allows data points to belong to multiple clusters with varying degrees of membership. This clustering method is beneficial for analyzing motor data that may

Submit Date: 2025-03-22
Accept Date: 2025-09-28
✉ Corresponding author

overlap between distinct motor patterns. The FCM algorithm minimizes an objective function iteratively based on membership values for each data point.

To evaluate the clustering performance of the FCM (Fuzzy C-Means) model, we used an entropy-based measure to assess the degree of "fuzziness" in the clustering results, with lower entropy (fuzziness) indicating better clustering quality. Let $U = [u_{ij}]$ represent the membership matrix, where u_{ij} is the degree of membership of data point i in cluster j. In FCM, u_{ij} values lie in the interval $[0, 1]$ and satisfy the constraint:

$$\sum_{j=1}^{c} u_{ij} = 1 \quad (1)$$

for each data point i, where c is the total number of clusters.

The entropy E for the clustering result is calculated as follows:

$$E = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{c} u_{ij}\log(u_{ij}) \quad (2)$$

where:

- N is the total number of data points,
- c is the number of clusters,
- $u_{ij}\log(u_{ij})$ represents the contribution of data point i to the entropy for cluster j.

If entropy values are low, it suggests that data points have high membership values for specific clusters and are assigned to clusters with greater certainty. Conversely, high entropy values imply greater uncertainty, with data points more equally distributed across multiple clusters, indicating a fuzzier clustering outcome. Thus, this entropy-based metric allowed us to quantitatively assess the FCM clustering quality, capturing the degree of overlap in cluster memberships and providing insights into the clarity of the clusters formed.

RESULTS

The effect of the number of clusters (single feature clustering)

Fig. 1.A, presents the entropy values obtained from clustering the hand-drawing data into two and three groups, based on different extracted features. Entropy here is used as a measure of clustering efficiency, where lower entropy indicates more distinct, well-separated clusters.

The results show that the feature "length of shape" yields the lowest entropy, suggesting it is the most effective feature for clustering hand-drawing data. In contrast, beta results in the highest entropy, making it the least useful feature for this task. Therefore, length emerges as the best discriminative feature, while beta performs the worst in distinguishing between different behaviors.

The figure also demonstrates how features affect the number of clusters that can be effectively discriminated. For example, max V and duration achieve better entropy values when clustering into three groups, indicating that these features provide more granularity and are better suited for finer distinctions. On the other hand, length, beta, and the correlation coefficient perform better when clustering into two groups, suggesting that these features are more effective at separating the data into broader categories.

Fig. 1, B to D further illustrate the distribution of clusters for the two extreme features: length (the best-performing feature) and beta (the worst-performing feature). The clusters, colored according to membership values, reveal notable differences in performance between these features.

In the case of length, although it clearly separates the data into more distinct clusters (with greater inter-cluster distance), the cluster sizes are imbalanced, meaning that one cluster contains significantly more members than the others. This suggests that while length is highly discriminative, it may overfit certain behaviors.

On the other hand, beta produces more uniform clusters, with relatively balanced sizes, but the separation between clusters is less distinct, implying poorer discriminative power. Beta's ability to cluster data is limited, which aligns with its higher entropy score.

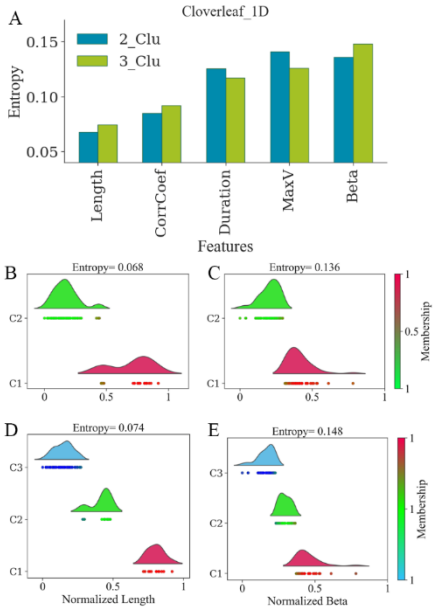


Fig. 5- **Clustering Performance for cloverleaf task.** A) Entropy values for all clustering cases using different features, shown for both 2 and 3 clusters. B) Cluster centers for the length feature in the 2-cluster case. C) Cluster centers for the beta feature in the 2-cluster case. D) Cluster centers for the length feature in the 3-cluster case. E) Cluster centers for the beta feature in the 3-cluster case. The size of each circle reflects the number of members in each cluster, highlighting differences in cluster size.

In Fig. 2, A, the results of the clustering of extracted data from the eight-shape task. Similar to the previous results, length had the best entropy for discriminating data in 3 clusters and beta had the worst value. In the second task, we applied the same clustering analysis to hand-drawing movements involving the eight-shape. Fig. 2, shows the entropy values for clustering into 2 and 3 groups.

As with the cloverleaf task, the length again showed the best clustering performance, yielding the idea that length is a highly effective feature for distinguishing between hand-drawing behaviors, regardless of the specific shape being drawn. In contrast, beta showed the worst clustering performance, resulting in the highest entropy values, indicating that it struggles to effectively differentiate behaviors in both tasks.

Interestingly, in this task, duration performed better when clustering into 3 groups, similar to the findings from the cloverleaf task. This suggests that this feature may capture more subtle differences in drawing dynamics, which are better revealed when the data is split into finer-grained categories. For max V, the entropy decreased in the eight-shape compared to the cloverleaf task. In this task, the temporal features showed better performance.

Fig. 2, B to E illustrate the clustering distributions for length (best feature) and beta (worst feature). As observed in the cloverleaf task, length produces more distinct clusters with greater separation between them, though the clusters are unevenly sized. Beta, on the other hand, forms more balanced clusters in terms of size, but with much less separation between the groups, which indicates poorer overall discrimination.

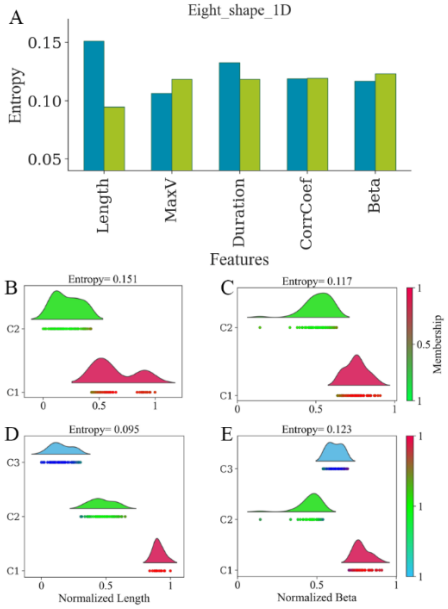


Fig. 6- **Clustering Performance for eight-shape task.** A) Entropy values for all clustering cases using different features, shown for both 2 and 3 clusters. B) Cluster centers for the length feature in the 2-cluster case. C) Cluster centers for the beta feature in the 2-cluster case. D) Cluster centers for the length feature in the 3-cluster case. E) Cluster centers for the beta feature in the 3-cluster case. The size of each circle reflects the number of members in each cluster, highlighting differences in cluster size.

The effect of the feature dimensions (multiple features clustering)

Fig. 3, shows the mean entropy values obtained from clustering with different combinations of features, ranging from single-feature clustering (1 dimension) to using all five features together (5 dimensions). The results demonstrate a clear trend: as more features are combined for clustering, the performance deteriorates, as indicated by increasing entropy values. This trend is consistent across both tasks (cloverleaf and eight-shape) and suggests that adding more features leads to poorer clustering performance, possibly due to redundant or less discriminative information being introduced.

These findings highlight the importance of careful feature selection in clustering analyses of hand-drawing data, as using more features does not necessarily result in better discrimination. Instead, focusing on key individual features may yield more precise clustering and better identification of motor signatures.

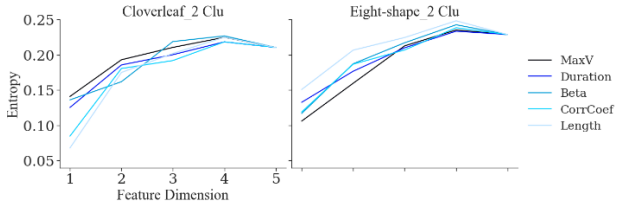


Fig. 7- Entropy values for clustering with different combinations of features (from 1 to 5 dimensions) across both tasks (cloverleaf and eight-shape) for 2-cluster discrimination.

Comparison with other methods

In this section, we compare the performance of FCM with two other clustering methods: **K-Means** [3], [4] and **Hierarchical Clustering (HC)** [5]. K-Means is a hard clustering algorithm that assigns each data point to the nearest cluster centroid. It is computationally efficient and performs well for datasets with spherical clusters. In contrast, HC is an agglomerative or divisive clustering method that constructs a hierarchy of clusters using a bottom-up (agglomerative, in this case) or top-down (divisive) approach. Unlike FCM and K-Means, HC does not require the number of clusters to be specified a priori. However, to ensure a fair comparison, we truncate the HC dendrogram at a specific level to obtain the same number of clusters as used in FCM and K-Means. This allows us to evaluate the performance of all three methods under identical conditions.

Since we used the fuzziness metric to evaluate the performance of FCM, this metric cannot be applied to K-Means and HC. This is because both K-Means and HC are hard clustering methods, assigning each data point to a single cluster, resulting in a fuzziness value of **0**. To enable a fair comparison across all methods, we employ the **Silhouette Index** [6], a metric that quantifies cluster quality by measuring how similar a data point is to its own cluster compared to other clusters. However, since the traditional Silhouette Score is designed for hard clustering and is not directly applicable to FCM, we use a variant called the **Fuzzy Silhouette Index** [7] with $\alpha = 1$. This metric incorporates fuzzy membership degrees while retaining the same interpretability as the traditional Silhouette Score, allowing us to compare FCM, K-Means, and HC on a common scale.

In Table 1, we compare the average cluster quality of FCM, K-Means, and HC using the Silhouette Score (Fuzzy Silhouette Index for FCM). For 3-cluster solutions, FCM achieves the highest average score (0.48), followed by HC (0.47) and K-Means (0.45). While FCM outperforms the other methods in this comparison, the small difference between FCM and HC suggests that both algorithms may be viable choices depending on the application context. Future work should validate these results with statistical significance testing.

Table 1- Average Silhouette Scores (Fuzzy Silhouette Index for FCM) Across All Feature Combinations for 3-Cluster Solutions. Higher values indicate better cluster quality, with a maximum possible score of +1.

	FCM	K-Means	HC
3 Clusters	**0.48**	0.45	0.47

DISCUSSION

The present study employed Fuzzy C-Means (FCM) clustering to analyze motor performance using hand-drawing tasks, aiming to understand how kinematic and geometrical parameters can reveal individual motor patterns and their underlying dynamics. By examining clustering efficiency across two distinct tasks—drawing cloverleaf and eight-shape patterns—we identified specific features that excel at distinguishing between motor behaviors, providing valuable insights into the feature-based clustering of complex motor actions.

Longer lengths may indicate fluid, continuous movements, while shorter lengths might suggest more controlled, segmented actions. The variability in length across individuals likely reflects their unique movement characteristics, such as drawing speed, force, and control stability [8], [9]. This variability makes length a highly idiosyncratic feature, able to reveal distinct motor patterns between individuals. Aligning with our results, research has demonstrated that spatial consistency often serves as a marker of individual motor behaviors [10], [11]. Moreover, length as a measure is minimally influenced by the shape type (cloverleaf or eight-shape), allowing it to act as a robust indicator of underlying motor behavior regardless of task specifics.

Conversely, beta—an exponent in the power-law relationship between velocity and curvature—was less effective in distinguishing motor patterns, likely due to its reliance on more abstract relationships sensitive to noise [12], [13]. Additionally, beta captures subtleties in the coordination between speed and path shape, but these subtleties may not vary significantly across individuals in a way that clearly distinguishes their motor behaviors. As a result, beta yielded clusters with higher intra-cluster ambiguity, illustrating that it is a less reliable indicator of distinct motor profiles in these tasks. These finding highlights that not all kinematic parameters are equally useful for IMS; features like length, which capture direct spatial and motor control aspects, provide clearer clustering outcomes [14].

Temporal features such as duration and maximum velocity also performed well in three-cluster configurations, capturing dynamic aspects of drawing reflective of individual pacing and motor control nuances [15], [16]. These results underscore the importance of dynamic control markers for differentiating motor behaviors, consistent with prior research on pacing and motor coordination [17].

Moreover, combining multiple features generally led to higher entropy, suggesting that redundant or conflicting information introduced noise that weakened clustering performance [18], [19]. This supports the notion that selective feature inclusion is essential in clustering, as excess dimensions can obscure rather than clarify individual motor signatures [20].

CONCLUSION AND IMPLICATIONS

In summary, this study underscores the potential of targeted feature selection in clustering motor behaviors and highlights the differential utility of specific kinematic and temporal features in distinguishing individual motor patterns. Length, with its ability to capture spatial control and idiosyncratic motor traits, emerged as a particularly valuable feature, while beta was found to be less effective due to its sensitivity to nuanced, less variable aspects of motor behavior. Temporal features like duration and maximum velocity offer additional insights, particularly in more granular clustering scenarios where finer distinctions in pacing and control are relevant.

These findings contribute to the methodological approach of using feature-based clustering to capture individual motor signatures. In future research, this framework could be applied to diverse motor tasks or clinical populations, where unique motor signatures may help identify early markers of neurological conditions or tailor rehabilitation protocols. Overall, this study provides a foundation for refining clustering techniques in motor analysis, demonstrating that careful feature selection can enhance the interpretability and effectiveness of clustering in capturing distinct motor behaviors.

REFERENCE

[1] T. Flash and B. Hochner, "Motor primitives in vertebrates and invertebrates," *Curr. Opin. Neurobiol.*, vol. 15, no. 6, pp. 660–666, 2005.

[2] Z. Zarandi, A. N. Stucchi, L. Fadiga, and T. Pozzo, "The effect of the preferred hand on drawing movement," *Sci. Rep.*, vol. 13, no. 1, p. 8264, May 2023. doi: 10.1038/s41598-023-34861-x.

[3] S. Lloyd, "Least squares quantization in PCM," *IEEE transactions on information theory,* vol. 28, no. 2, pp. 129-137, 1982.

[4] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, 1967, vol. 1, no. 14: Oakland, CA, USA, pp. 281-297.

[5] F. Nielsen and F. Nielsen, "Hierarchical clustering," *Introduction to HPC with MPI for Data Science,* pp. 195-211, 2016.

[6] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics,* vol. 20, pp. 53-65, 1987.

[7] R. J. Campello and E. R. Hruschka, "A fuzzy extension of the silhouette width criterion for cluster analysis," *Fuzzy Sets and Systems,* vol. 157, no. 21, pp. 2858-2875, 2006.

[8] S. Mitra, M. A. Riley, and M. T. Turvey, "Variation and variability in motor behavior: A dynamical systems perspective," *Psychol. Rev.*, vol. 124, no. 2, pp. 182–203, 2017.

[9] H. Haken, J. A. S. Kelso, and H. Bunz, "A theoretical model of phase transitions in human hand movements," *Biol. Cybern.*, vol. 51, no. 5, pp. 347–356, 1985.

[10] E. Todorov, "Cosine tuning minimizes motor errors," *Nat. Neurosci.*, vol. 7, no. 4, pp. 384–390, 2004.

[11] M. L. Latash, J. P. Scholz, and G. Schöner, "Toward a new theory of motor synergies," *Motor Control*, vol. 6, no. 1, pp. 8–45, 2002.

[12] K. Schneider and D. Sternad, "Variability, stability, and the role of noise in motor learning," *Exerc. Sport Sci. Rev.*, vol. 42, no. 1, pp. 24–31, 2014.

[13] N. Hogan and D. Sternad, "Dynamic primitives of motor behavior," *Biol. Cybern.*, vol. 108, no. 6, pp. 645–663, 2013.

[14] K. M. Newell, "Constraints on the development of coordination," in *Motor Development in Children: Aspects of Coordination and Control*, M. G. Wade and H. T. A. Whiting, Eds. Springer, 1986.

[15] J. A. S. Kelso, *Dynamic Patterns: The Self-Organization of Brain and Behavior*. MIT Press, 1995.

[16] D. Sternad, "It's not (only) the mean that matters: Variability, noise and exploration in skill learning," *Curr. Opin. Behav. Sci.*, vol. 20, pp. 183–195, 2018.

[17] D. A. Rosenbaum, R. G. Carlson, and R. R. Gilmore, *Cognitive Foundations of Action: A Research Synthesis*. Erlbaum, 2001.

[18] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2009.

[19] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[20] M. L. Latash, *Neurophysiological Basis of Movement*. Human Kinetics, 2008.

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

Deep Learning Frailty Model for Heart Failure Survival Prediction

Solmaz Norouzi✉, **Code Orcide: 0000-0001-5805-6072**
Department of Biostatistics, Faculty of Medical Sciences, Tarbiat Modares University, Tehran, Iran.
snorouzibiostatistics@gmail.com
Ebrahim Hajizadeh, **Code Orcide: 0000-0001-7863-4837**
Department of Biostatistics, Faculty of Medical Sciences, Tarbiat Modares University, Tehran, Iran.hajizadeh@modares.ac.ir
Hossein Khormaei, **Code Orcide: 0009-0004-3274-5776**
Department of Electrical Engineering, National University of Skills (NUS), Tehran, Iran. hosainkhormaee@gmail.com
Nasim Naderi, **Code Orcide: 0000-0001-6067-040X**
Rajaie Cardiovascular Medical and Research Center, Iran University of Medical Sciences, Tehran, Iran.
naderi.nasim@gmail.com

Abstract—The study employed Deep Learning Frailty (DLF), a compelling neural modeling framework for predicting heart failure patient survival. The DLF embeds a notion of multiplicative frailty from classical survival analysis that deals with unobserved heterogeneity while exploiting the neural structure's strong capabilities in approximating any non-linear covariate relationship. The results showed that Incorporating frailty leads to significant improvements, and the DLF model performs better on average.

Keywords—Deep Learning, prediction, survival analysis, heart failure

I. Introduction

The survival analysis of patients is an important section in medical research, as it estimates the time to specific events(1). Deep learning methods have recently become quite popular for modeling complex relationships within survival data. In this work, the authors investigate advanced applications of deep neural networks in survival analysis within the framework of frailty models. Frailty is the unobserved heterogeneity that influences the occurrence of the event. Combining the ideas could, therefore, enhance the accuracy and interpretability of survival predictions in domains with multiple possible outcomes(2, 3).
Traditional survival analysis models usually assume that the data are independently and identically distributed, which might be the reason for biased estimates, hence reducing their accuracy(4). Therefore, sophisticated methods are required to capture nonlinear relationships within such models. This paper proposes a new method of incorporating frailty models into deep neural networks to perform survival analysis on heart failure patients by considering the dependency structure among the observations explicitly. The present study aims to develop a more powerful framework for predicting survival outcomes and compare its performance with DeepSurv(5, 6).

II. Literature Review

Recent advancements in deep learning have significantly impacted survival analysis, particularly in addressing the challenges posed by unobserved heterogeneity. Lee et al. (2018) introduced DeepHit, a non-parametric model that surpasses traditional methods by learning the distribution of survival times without restrictive parametric assumptions (7). To mitigate the negative impact of irrelevant features, Rietschel et al. (2018) proposed innovative feature selection techniques that significantly improved deep learning model performance in medical datasets (8). Huang and Liu (2020) developed DeepCompete. This continuous-time model effectively handles competing risks (9), while Nagpal et al. (2020) presented Deep Survival Machines. This fully parametric model outperforms traditional methods in handling censored data without relying on the Cox proportional hazards assumption (10).

Recent studies have explored the integration of frailty models into deep learning frameworks to address unobserved heterogeneity (11). Tran et al. (2020) and Mendel et al. (2022) incorporated random effects into deep neural networks to account for frailty (۱۲, ۱۳), while Hangbin Lee et al. (2023) proposed the DNN-FM model, which effectively handles censored survival data and improves predictive performance (۱٤). Wu et al. (2023) introduced the Neural Frailty Machine (NFM), which incorporates multiplicative frailty to address unobserved heterogeneity and demonstrates superior predictive capabilities across various datasets (۱٥).

Submit Date: 2025-03-22
Accept Date: 2025-09-28
✉ Corresponding author

FRAILTY MODELS, A CORNERSTONE OF MODERN SURVIVAL ANALYSIS, EXTEND THE COX MODEL BY INTRODUCING A MULTIPLICATIVE RANDOM EFFECT TO ACCOUNT FOR UNOBSERVED HETEROGENEITY [16] [17]
While the theoretical foundations of frailty models are well-established, most research has focused on linear relationships [18] [19] [20].

By leveraging deep learning, frailty models can capture complex, non-linear relationships and address censoring and competing risks through the frailty parameter. This flexible approach accurately estimates unobserved heterogeneity and correlations between different event types. Additionally, it improves the precision of covariate effect estimates based on specific event causes, ultimately enhancing the model's accuracy and predictive performance.

III. Materials and Methods

A. Study Population

Data from 529 patients admitted into the RCMCH information system between March and August 2018 were first retrieved for analysis. They were diagnosed with HFrEF and received standard treatment. After the exclusion of those patients who did not undergo standard treatment, the data from 435 patients will be included in the survival time analysis, followed from 2018 to July 2023, equivalent to 5 years. This dataset consists of 57 demographic and clinical features.

B. Statistical Analysis

All the raw data in this investigation were put together in the form of a database, and further tests were run on the processed data. These analyses have been implemented through Python using Pytorch by implementing both DLF and DeepSurv models. There were standard training/validation splits in the survival dataset, so a 5-fold cross-validation was done with one fold reserved for testing and 20% used for validation. Model performance was evaluated using three metrics standard in survival predictions, namely the integrated Brier score (IBS), integrated negative binomial log-likelihood (INBLL), and c-index.
DLF considers different deep neural network architectures for the modeling of survival analysis. The central insight driving this work is the way censored observations enter into a likelihood to produce consistent parameter estimates, given partial information about event times. Extension to include frailty into the deep neural network model enables the DLF structure to capture individual-specific traits or temporal changes driven by some unobserved factors that possibly influence event risk. Intra-individual correlation can be induced by unobservable individual-specific factors.
For the frailty variable u, we utilize the gamma density function;

$$G_\theta(x) = \frac{1}{\theta}\log(1+\theta x), \theta \geq 0 \qquad (1)$$

We begin by integrating the conditional survival function with frailty to derive the observed likelihood function:

$$S(t \mid X) = \mathbb{E}_{u_i \sim f_\theta}\left[e^{-u_i \int_0^t e^{h(s,X)} ds}\right] =: e^{-G_\theta\left(\int_0^t e^{h(s,X)} ds\right)} \qquad (2)$$

The frailty transform ($G_\theta(x) = -\log\left(\mathbb{E}_{u_i \sim f_\theta}[e^{-u_i x}]\right)$ is defined as the $(-\log)$ Of the Laplace transform of the frailty distribution for each cause. Consequently, the conditional cumulative hazard function is given by $\mathrm{H}(t|X) = G_\theta\left(\int_0^t e^{h(s,X)} ds\right)$.

For the PF model, we utilize two Multi-Layer Perceptrons (MLPs), denoted as $\hat{h} = \hat{h}(t; W^h, b^h)$ and $\hat{m} = \hat{m}(X; W^m, b^m)$, to approximate the functions h and m, parameterized by $(W^h, b^h)$and (W^m, b^m), respectively. Here, W represents a collection of weight matrices across all layers of the MLPs, while b denotes a set of bias vectors across all layers. Considering the standard results regarding the likelihood of censored data as presented in Equation (2), the learning of parameters under the PF framework can be expressed as follows (26).

$$\mathcal{L}(\mathrm{W}^h, \mathrm{b}^h, \mathrm{W}^m, \mathrm{b}^m, \theta) = \frac{1}{n}\left[\sum_{i\in[n]} \delta_i \log g_\theta\left(e^{\hat{m}(X_i)}\int_0^{T_i} e^{\hat{h}(s)} ds\right) + \delta_i \hat{h}(T_i) + \delta_i \hat{m}(X_i) - G_\theta\left(e^{\hat{m}(X_i)}\int_0^{T_i} e^{\hat{h}(s)} ds\right)\right] \qquad (3)$$

in which $g_\theta(x) = \frac{\partial}{\partial x} G_\theta(x)$.
The estimated conditional cumulative risk and survival functions are represented by equation (4).

$$\hat{\mathrm{H}}_{\mathrm{DLF}}(t \mid X) = G_{\hat{\theta}_n}\left(\int_0^t e^{\hat{h}_n(s)+\hat{m}_n(X)} ds\right), \qquad (4)$$
$$\hat{S}_{\mathrm{DLF}}(t \mid X) = e^{-\hat{\mathrm{H}}_{\mathbf{DLF}}(t|X)},$$

IV. RESULTS

This study analyzed the mortality of 435 heart failure patients over five years, focusing on deaths from heart failure. At the study's conclusion, 29.4% of patients were alive, and 40.9% were censored. Of the patients, 43.96% died from heart failure. The median survival time was 43.40 months.

The one-year survival rate for patients who died from heart failure was 80.66% (95% CI: 0.76-0.84), decreasing to 68.3% (95% CI: 0.63-0.72) at three years and 59.52% (95% CI: 0.54-0.64) at five years. For those who died from other causes, the survival rates were 91.78% (95% CI: 0.88-0.94) at one year, 79.08% (95% CI: 0.74-0.83) at three years, and 70.29% (95% CI: 0.64-0.75) at five years.

The mean age of all patients was 18.11 ± 56.57 years, ranging from 14 to 95. Among those who died from heart failure, the average age was 59.26 ± 1.40 years, with the highest mortality in the 56-65 age group. Of these, 63.1% were male, 89.4% self-employed, and 87.5% held a bachelor's degree. Additionally, 93.1% resided in urban areas, and 89.4% were married.

To identify the optimal learning rate for the deepsurv and DLF models, we conducted experiments using a range of learning rates, specifically [0.1, 0.01, 0.001, 0.0001, 1e-05], and evaluated performance based on the c-index criterion. This process involved creating and deploying a new deepsurv model for each learning rate, utilizing a Negative Loglikelihood loss function and the Adam optimizer over 50 epochs, during which learning rate and weight decay were monitored. We systematically compared model performance across these learning rates and determined the optimal rate to be lr = 0.006 (Fig 1).

We assessed the performance gains of DLF relative to their non-frailty counterpart, DeepSurv. As shown in Table 1, incorporating frailty leads to significant improvements, with performance increases exceeding 10% for DLF models in IBS and INBLL metrics. This enhancement is consistent across the HF dataset. The results suggest that the DLF model performs better on average.

TABLE I. DLF MODELS IN COMPARISON TO THE NON-FRAILTY DEEPSURV MODEL.

Model	metrics		
	IBS	***INBLL***	***c-index***
DeepSurve	0.27±0.02	0.62±0.01	0.55±0.04
DLF	0.17±0.01	0.53±0.02	0.66±0.04

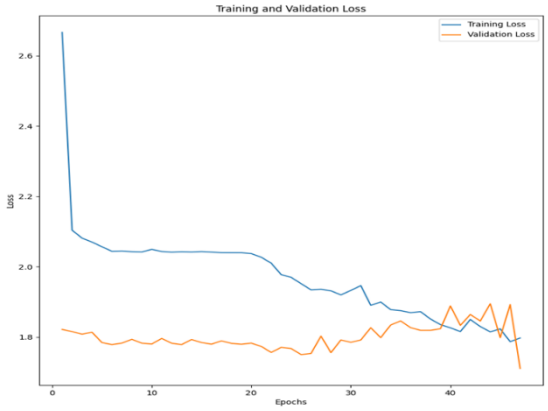


Fig 1: Assessing model performance on both training and validation datasets

V. DISCUSSION

This study aimed to evaluate the accuracy of a deep learning model incorporating a frailty approach applied to heart failure data. The Deep Neural Frailty(DLF) method introduces an innovative approach to modeling mortality by utilizing two distinct neural network structures.

The results demonstrate that the DLF framework effectively handles censored data in survival analysis, leading to improved predictive accuracy, reduced bias, enhanced robustness, and more personalized predictions. These advantages make it a valuable tool for healthcare applications, where censored data is common. Our findings also indicate that incorporating frailty into the deep learning model significantly improves its predictive accuracy when applied to heart failure data. Frailty, in the context of time-to-event modeling, accounts for unobserved heterogeneity among individuals, which can influence the occurrence of an event. Frailty models extend traditional survival models, such as the Cox model, by introducing random effects that capture individual-specific factors not directly observed but influencing the event of interest. By including frailty, time-to-event models better account for variability in event times that cannot be explained by the measured covariates alone.

This study also compared two survival models: The DeepSurv model, and the DLF model with proportional frailty. Despite the small sample size, results showed that both the DeepSurv and DLF models outperform others in terms of accuracy in predicting survival time, thanks to their advanced ability to model complex relationships. This comparison helps guide researchers in selecting the most suitable model for survival analysis. These models excel at capturing complex, non-linear relationships between input variables and survival outcomes. Supporting our results, a study by Ruofan Wu et al. examined the performance of the Neural Frailty Machine (NFM) on survival data from five datasets, demonstrating that NFM outperformed

12 other reference models, particularly on the METABRIC, SUPPORT, and MIMIC-III datasets. NFM showed significant improvements in evaluation metrics such as IBS and INBLL compared to other models (15).
However, some studies argue that there is no conclusive evidence to suggest that deep learning models consistently outperform classical models. While deep learning is recognized for its ability to model complex and non-linear relationships, especially in survival time prediction, its performance is not always significantly superior to that of traditional models. Classical models can sometimes perform just as well, or even better, in certain cases. Additionally, some studies suggest that deep learning models require larger datasets for training and fine-tuning and may not consistently outperform traditional models when data is scarce (21). Both the DLF and the classic Cox model are important tools for analyzing competing risks, but they differ in structure and application. Ultimately, the choice of model depends on the specific characteristics of the data and the research context, meaning that deep learning models are not universally superior to classical models (٢٣ ,٢٢).

References

1. Kleinbaum DG, Klein M. Survival analysis a self-learning text: Springer; 1996.
2. Kaplan EL, Meier P. Nonparametric estimation from incomplete observations. Journal of the American Statistical Association. 1958;53(282):457-81.
3. Austin PC, Lee DS, Fine JP. Introduction to the analysis of survival data in the presence of competing risks. Circulation. 2016;133(6):601-9.
4. Austin PC, Fine JP. Accounting for competing risks in randomized controlled trials: a review and recommendations for improvement. Statistics in medicine. 2017;36(8):1203-9.
5. Nagpal C, Li X, Dubrawski A. Deep survival machines: Fully parametric survival regression and representation learning for censored data with competing risks. IEEE Journal of Biomedical and Health Informatics. 2021;25(8):3163-75.
6. Zhong Q, Mueller JW, Wang J-L. Deep extended hazard models for survival analysis. Advances in Neural Information Processing Systems. 2021;34:15111-24.
7. Lee C, Zame W, Yoon J, Van Der Schaar M, editors. Deephit: A deep learning approach to survival analysis with competing risks. Proceedings of the AAAI conference on artificial intelligence; 2018.
8. Rietschel C, Yoon J, van der Schaar M. Feature selection for survival analysis with competing risks using deep learning. arXiv preprint arXiv:181109317. 2018.
9. Huang P, Liu Y, editors. Deepcompete: A deep learning approach to competing risks in continuous time domain. AMIA annual symposium proceedings; 2020: American Medical Informatics Association.
10. Nagpal C, Yadlowsky S, Rostamzadeh N, Heller K, editors. Deep Cox mixtures for survival regression. Machine Learning for Healthcare Conference; 2021: PMLR.
11. Hong C, Yi F, Huang Z. Deep-CSA: Deep Contrastive Learning for Dynamic Survival Analysis With Competing Risks. IEEE Journal of Biomedical and Health Informatics. 2022;26(8):4248-57.
12. Mandel F, Ghosh RP, Barnett I. Neural networks for clustered and longitudinal data using mixed effects models. Biometrics. 2023;79(2):711-21.
13. Tran M-N, Nguyen N, Nott D, Kohn R. Bayesian deep net GLM and GLMM. Journal of Computational and Graphical Statistics. 2020;29(1):97-113.
14. Lee H, HA I, Lee Y. Deep Neural Networks for Semiparametric Frailty Models via H-likelihood. arXiv preprint arXiv:230706581. 2023.
15. Wu R, Qiao J, Wu M, Yu W, Zheng M, Liu T, et al. Neural Frailty Machine: Beyond proportional hazard assumption in neural survival regressions. Advances in Neural Information Processing Systems. 2023;36:5569-97.
16. Wienke A. Frailty models in survival analysis: Chapman and Hall/CRC; 2010.
17. Duchateau L, Janssen P. The frailty model: Springer; 2008.
18. Murphy SA. Consistency in a proportional hazards model incorporating a random effect. The Annals of Statistics. 1994;22(2):712-31.
19. Parner E. Asymptotic theory for the correlated gamma-frailty model. The Annals of Statistics. 1998;26(1):183-214.
20. Kosorok MR, Lee BL, Fine JP. Robust inference for univariate proportional hazards frailty regression models. 2004.
21. Jung W, Kim SE, Kim JP, Jang H, Park CJ, Kim HJ, et al. Deep learning model for individualized trajectory prediction of clinical outcomes in mild cognitive impairment. Frontiers in Aging Neuroscience. 2024;16:1356745.
22. Wang M, Greenberg M, Forkert ND, Chekouo T, Afriyie G, Ismail Z, et al. Dementia risk prediction in individuals with mild cognitive impairment: a comparison of Cox regression and machine learning models. BMC Medical Research Methodology. 2022;22(1):284.
23. Kantidakis G, Biganzoli E, Putter H, Fiocco M. A simulation study to compare the predictive performance of survival neural networks with Cox models for clinical trial data. Computational and Mathematical Methods in Medicine. 2021;2021(1):2160322.

Biography

Solmaz Norouzi Orcid: 0000000158056072
Department of Biostatistics, Faculty of Medical Sciences, Tarbiat Modares University, Tehran, Iran.
snorouzibiostatistics@gmail.com

Solmaz Norouzi, a Ph.D. in Biostatistics at Tarbiat Modares University, is an experienced biostatistician specializing in survival data analysis, deep learning, and mathematical statistics. With over 8 years of experience in the field, she possesses expertise in survival data analysis, competing risks models, and frailty models. Her research focuses on the application of advanced statistical methods in medical data, as evidenced by her Master's and Doctoral theses.

Dr. Norouzi is skilled in various areas of biostatistics, including survival analysis, meta-analysis, and structural equation modeling. She also has significant knowledge and experience in machine learning and deep neural networks, particularly in survival data analysis. Her proficiency in statistical software such as R, Python, STATA, and SPSS enables her to conduct complex research projects accurately and efficiently.

In addition to her technical skills, Dr. Norouzi is experienced in writing scientific papers and presenting at international conferences. She has several publications in reputable scientific journals, demonstrating her commitment to advancing scientific knowledge. She is also active in teaching and statistical consulting.

Ebrahim Hajizadeh Orcid: 0000-0001-7863-4837
Department of Biostatistics, Faculty of Medical Sciences, Tarbiat Modares University, Tehran, Iran.
hajizadeh@modares.ac.ir

Dr. Ebrahim Hajizadeh is a full professor of biostatistics at Tarbiat Modares University.

Hossein Khormaei Orcid: 0009-0004-3274-5776
Department of Electrical Engineering, National University of Skills (NUS),
Tehran, Iran.
hosainkhormaee@gmail.com

Expert in Python programming

Nasim Naderi Orcid: 0000-0001-6067-040X
Rajaie Cardiovascular Medical and Research Center, Iran University of Medical Sciences, Tehran, Iran.
naderi.nasim@gmail.com

Board Certification: Cardiology, Shaheed Beheshti University of Medical Sciences, Tehran , Iran
Fellowship: Heart Failure and Transplantation, Shaheed Rajaei Cardiovascular Medical and Research, Center, Tehran University of Medical Sciences, Tehran,Iran

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

A Novel Fixed-Parameter Activation Function for Neural Networks: Enhanced Accuracy and Convergence on MNIST

Najmeh Hosseinipour-Mahani✉, **Code Orcide:** 0000-0003-2311-2040
Department of Applied Mathematics, Graduate University of Advanced Technology, Kerman, Iran, nhoseinipour@gmail.com
Amirreza Jahantab, **Code Orcide:** 0009-0005-1946-8353
Department of Computer science, Shahid Bahonar University of Kerman, Kerman, Iran, Jahantab83@gmail.com

Abstract — Activation functions are essential for extracting meaningful relationships from real-world data in deep learning models. The design of activation functions is critical, as they directly influence the performance of these models. Nonlinear activation functions are commonly preferred since linear functions can limit a model's learning capacity. Nonlinear activation functions can either have fixed parameters, which are predefined before training, or adjustable ones that modify during training. Fixed-parameter activation functions require the user to set the parameter values prior to model training. However, finding suitable parameters can be time-consuming and may slow down the convergence of the model. In this study, a novel fixed-parameter activation function is proposed and its performance is evaluated using benchmark MNIST datasets, demonstrating improvements in both accuracy and convergence speed.

Keywords —Activation Function, Deep Learning, Fixed-Parameter, Neural Networks, MNIST Dataset, Nonlinear Function, Gradient Optimization, Vanishing Gradient Problem

I. Introduction

Deep learning has become a cornerstone of modern artificial intelligence, powering breakthroughs in fields such as computer vision, natural language processing, and autonomous systems. Central to the success of deep learning models are activation functions, which introduce non-linearity to neural networks, allowing them to capture complex patterns and relationships in data. Without appropriate activation functions, neural networks would behave as linear models, limiting their capacity to solve real-world, non-linear problems [15, 16]. Activation functions can be broadly categorized into linear and nonlinear types. Linear activation functions, although simple, restrict the representational power of deep neural networks. For this reason, nonlinear activation functions such as ReLU, Sigmoid, and ELU are commonly used to enable the network to learn more complex data representations.

A key characteristic of these nonlinear functions is the ability to determine how neurons should fire and adjust their responses based on the input values. Most nonlinear activation functions in the literature are designed with fixed parameters. These parameters are constants defined before training, and they remain unchanged throughout the learning process. While fixed-parameter activation functions like ReLU and ELU have proven to be effective, they often require careful selection of parameters to optimize the model's learning capabilities.

In this study, we propose a novel fixed-parameter activation function that operates across three distinct regions: negative, linear, and positive. Unlike some existing functions, the parameters in our activation function are predefined before training and remain constant throughout the learning process. The aim is to provide an efficient activation mechanism that balances flexibility in capturing nonlinear patterns with computational efficiency. We evaluate the performance of this new activation function on the MNIST benchmark dataset to demonstrate its effectiveness in improving both accuracy and convergence speed.

II. Related work

Over the past two decades, numerous fixed-parameter and trainable activation functions have been developed to enhance the training performance of deep learning models. Some of these include ReLU, LReLU, ELU, RSigELU, GeLU, FeLU, PReLU, DReLU, and PELU [3, 5, 7, 10, 13, 14]. The ReLU activation function was introduced to address the vanishing gradient problem by allowing negative values to be transformed into positive ones [3]. The LReLU activation function incorporates negative weights during the training process [4]. Subsequently, the ELU activation function was introduced as an alternative to ReLU to address issues related to vanishing gradients and negative weights [9]. Building on this, the SELU activation function was developed to enhance the training performance of ELU [5]. Fixed-parameter activation functions require predetermined

Submit Date: 2025-09-28
Accept Date: 2025-10-05
✉ Corresponding author

parameters prior to the training of deep learning models, which can result in slower convergence rates. Additionally, identifying appropriate parameter values for activation functions often necessitates extensive experimentation, making it a time-consuming endeavor.

After that, a variety of trainable activation functions, including ALISA, PELU, PRELU, P+FELU, PSigmoid and GELU, have been proposed [9]. These functions allow the model to fine-tune parameters during training, which is done through backpropagation. This technique, commonly used in deep learning, adjusts the weights, biases, and activation function parameters across neurons in the model. As training progresses, the parameters of these activation functions are updated in each iteration, allowing the model to find the most suitable weights. This adaptive approach enables trainable activation functions to achieve more efficient and accurate convergence by optimizing parameter values for different datasets and neural network architectures [11].

The ALISA activation function is indeed designed to mitigate issues related to the vanishing gradient problem, which can occur in deep neural networks [9]. The PELU activation function enhances the ELU by incorporating a trainable scalar parameter. This addition aims to address the bias shift problem, allowing for more flexibility and improved performance in neural network training [2]. The PReLU activation function was proposed as an alternative to the standard ReLU activation function to address some of its limitations, particularly the issue of negative weights [7]. The P+FELU activationfunction has been proposed to address the issues of vanishing gradient and negative weights [8]. The PSigmoid activation function was introduced as an enhancement to the Squeeze and Excitation (SE) Networks block [1]. The GELU activation function serves as an alternative to ReLU and ELU, showing performance improvements across various tasks in computer vision, natural language processing, and speech [12]. Additionally, the parametric RSigELU (P+RSigELU) includes trainable activation functions such as P+RSigELU Single (P+RSigELUS) and P+RSigELU Double (P+RSigELUD), which were developed by extending the RSigELU activation function [11].

The vanishing gradient problem is indeed a challenge associated with the Sigmoid and $Tanh$ activation functions, particularly in deep neural networks, as these functions can lead to very small gradients during backpropagation. The ReLU, derivatives, and the most suitable activation function help mitigate this issue. However, it is important to note that ReLU does not activate for negative inputs, effectively ignoring them, which can lead to the "dying ReLU" problem where neurons can become inactive. Behaviors of activation functions in the literature are shown in Fig.1.

As can be seen in Fig. 1, P+RSigELUS is not continuous at $x = 1$, and consequently, it is not differentiable either.

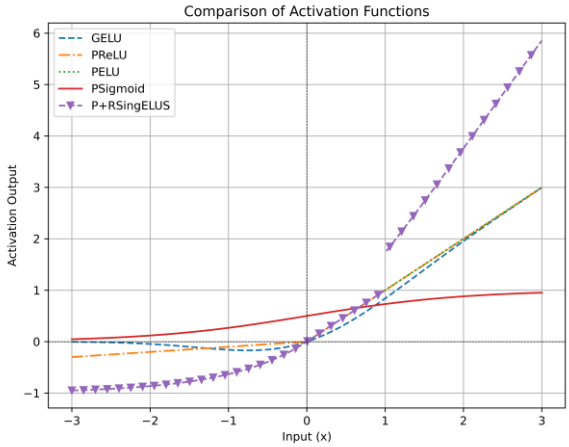


Fig. 1. Behaviors of activation functions

In this study, we propose a new activation function designed to enhance performance across both negative and positive input regions. This function addresses issues associated with the negative region, bias shift, and vanishing gradients. By actively engaging with inputs from both sides, and by incorporating a constant gradient parameter, our activation function offers greater flexibility. This approach helps reduce parameter errors for each neuron and supports a more effective learning process.

III. PROPOSED ACTIVATION FUNCTION

A. Introduction to Activation Function

In this section, a novel activation function is proposed that operates across three distinct regions: negative, linear, and positive. The function's behavior is governed by the input value x, with distinct formulations for each region. To enhance flexibility and performance, the function introduces adjustable parameters, which are carefully tuned prior to training based on the characteristics of the dataset or task. These parameters remain constant during training, ensuring a stable and computationally efficient implementation. The proposed activation function is defined as follows:

$$f(x) = \begin{cases} \frac{ax}{1+e^{x}} & -\infty < x < 0 \\ x & 0 \le x \le 1 \\ ae^{-x}(x-1)+1 & 1 < x < \infty \end{cases} \quad (1)$$

For $x < 0$, the negative region is controlled by the hyperparameter a. This allows the function to smoothly handle negative inputs while maintaining a balance between linearity and nonlinearity. For $0 \leq x \leq 1$, the function behaves linearly. In this region, the function passes the input directly, which helps maintain the gradient flow during backpropagation, reducing the risk of vanishing gradients. For $x > 1$, the positive region allows the function to grow exponentially as $x$increases, while the hyperparameter a adjusts the steepness of the curve, enhancing the model's ability to capture complex patterns in the data. By incorporating parameter a, the activation function adapts to the input distribution, improving the model's capacity to learn from data efficiently. The proposed function is designed to overcome common issues such as vanishing gradients and inefficient learning in deep networks. Additionally, it retains the benefits of hybrid activation functions, combining aspects of ReLU, ELU, and sigmoid functions, which ensures stable training and faster convergence. The introduction of parameterized slopes in both the negative and positive regions allows the network to better capture subtle variations in data, ultimately improving accuracy and convergence speed. In summary, this new activation function offers flexibility through parameterized control of its behavior in different regions, making it a suitable choice for deep learning architectures.
The proposed activation function behaves as shown in Fig. 2.

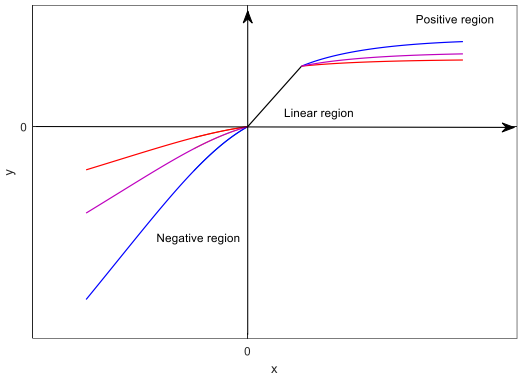


Fig. 2. Behavior of the proposed activation function

The derivative of the proposed activation function is given in the following:

$$\frac{df(x)}{dx} = \begin{cases} \dfrac{ae^{-x}(e^{-x} + x + 1)}{(1 + e^{-x})^2} & -\infty < x < 0 \\ 1 & 0 \leq x \leq 1 \\ ae^{-x}(2 - x) & 1 < x < \infty \end{cases}$$

In deep learning architectures, backpropagation is a key method used to update parameters during the learning process [6]. An important aspect of this is that the proposed activation functions must be differentiable. Notably, the activation function remains active in both the negative and positive regions after differentiation. Furthermore, the weights are initialized by randomly sampling from a normal distribution, which helps maintain appropriate variance across all layers. Consequently, the range of value generation and the steepness of the function are not fixed, as seen in logistic sigmoid and hyperbolic tangent functions, but instead fluctuate. While designing the network architecture, there are typically two approaches to utilizing free parameter functions. The first approach involves structures where each neuron in the network, with the exception of those in the final layer, possesses its own set of training parameters. The second approach consists of structures in which the neurons within each hidden layer share common fixed parameters. In this study, the first method was preferred.

The proposed activation function offers several advantages: it adjusts the slope parameters during training to optimize the values for each neuron in every iteration. This function effectively tackles issues like vanishing gradients and negative regions, thus enhancing the overall learning process. Additionally, it provides improved accuracy and faster convergence compared to previous activation functions. Notably, the proposed activation function is both parametric and monotonic.

B. Region-Wise Analysis of the Proposed Activation Function and the Role of the Hyperparameter a

In our proposed activation function, the hyperparameter a plays a crucial role in shaping the output response to different input values. This function consists of three distinct regions:

1. **For negative values of x:**

   $f(x) = \frac{ax}{1+e^x}$, for $-\infty < x < 0$.

   In this region, a controls the slope of the function, influencing how sharply negative inputs are transformed. A higher a results in a steeper response, whereas a lower a leads to a more gradual transition.

2. **For values between 0 and 1**:

   $f(x) = x$, for $0 \leq x \leq 1$.

In this range, the function behaves linearly, maintaining the identity mapping. The hyperparameter a does not have an impact in this region.

3. **For positive values of x:**

$f(x) = ae^{-x}(x-1)+1, \quad$ for $1 < x < \infty$.

Here, a determines the rate at which the function decays as x increases. A larger a slows down the decay, allowing the function to remain closer to 1 for a longer range of x, while a smaller a leads to faster convergence towards 1.

IV. Materials and method

A. Benchmark datasets

The MNIST dataset is widely recognized as a standard benchmark for evaluating machine learning models, particularly in computer vision tasks. It contains a total of 70,000 grayscale images; each sized 28×28 pixels, representing handwritten digits (0 through 9). The dataset is divided into 60,000 training samples and 10,000 test samples. Each image is accompanied by a corresponding label indicating the digit. Preprocessing steps included normalization of pixel values to a mean of 0.1307 and a standard deviation of 0.3081. This standardization accelerates convergence during training by ensuring consistent input distributions across all samples.

B. Convolutional neural network

To evaluate the performance of the proposed activation function, a convolutional neural network (CNN) inspired by the VGG architecture was implemented. The key modifications to the traditional VGG structure include the use of fewer layers, which is suitable for the MNIST dataset, and the integration of the proposed activation function at every stage of the network.

The architecture comprises four convolutional blocks, each followed by max-pooling and dropout layers to mitigate over fitting. Specifically:

I. **Convolutional Layers:** Four convolutional layers progressively increase the feature depth (32, 48, 64, and 96 channels, respectively) while preserving spatial resolution using a kernel size of 3×3.

II. **Activation Function:** The custom activation function was applied after each convolutional operation to introduce non-linearity and allow the network to model complex patterns in the data.

III. **Pooling and Dropout:** Max-pooling with a 2×2 kernel and a dropout rate of 0.25 were employed to reduce spatial dimensions and prevent over fitting.

IV. **Fully Connected Layers:** After flattening, the feature maps were passed through two fully connected layers (512 and 10 units), where the final layer outputs class probabilities using the softmax function.

C. Optimization Strategy for a

To determine the optimal value of a, we employed a cross-validation strategy combined with a coarse-to-fine search approach:

1. **Coarse Search:** We began with a broad range of possible values for a and conducted a few short training runs (e.g., 5 epochs).
   This step helped us get a rough estimate of suitable values for a that prevent instability and facilitate effective learning.
2. **Fine-tuned Search:** After identifying a promising range, we performed longer training runs with more refined values. The best value for a was selected based on the validation performance of the model. To prevent instability, we stopped training early if the loss exceeded three times its initial value, which signaled potential divergence.

Final Chosen Value

Following the optimization process, we determined that the best value for a in our model was $a = 0.5$. This value provided a balanced trade-off between stability and adaptability, ensuring smooth gradient flow and effective learning dynamics.

V. Results and discussion

A. Experimental evaluation

The experimental results from five training repeats of the proposed activation function on the MNIST dataset are summarized in TABLE I. The table provides metrics for each repeat, as well as the average performance across all runs.

TABLEI. Experimental Results of the Proposed Function for MNIST Dataset

Repeat	EPOCH	Train Loss	Train Acc	Val. Loss	Val. Acc
Repeat 1	40	0.0075	99.78%	0.0601	98.84%
Repeat 2	40	0.0053	99.84%	0.0585	99.01%
Repeat 3	40	0.0056	99.84%	0.0602	99.02%
Repeat 4	40	0.0054	99.83%	0.0590	99.12%
Repeat 5	40	0.0077	99.79%	0.0627	98.96%
Average		0.0063	99.82%	0.0601	98.99%

In addition, the proposed function was benchmarked against other established activation functions. The comparison, shown in TABLE II., highlights its superior performance across key metrics, including training loss, training accuracy, validation loss, and validation accuracy.

TABLE II. Average Success Rates of Activation Functions for MNIST Dataset

Activation	Train_Loss	Train_Acc	Val._Loss	Val._Acc
PReLU	0.2782	0.9139	0.2011	0.9383
PELU	0.2721	0.9187	0.2134	0.9362
ALISA	0.3018	0.9095	0.2275	0.9335
GELU	0.3726	0.8933	0.2203	0.9244
P+FELU	0.3255	0.8941	0.2127	0.9109
PSigmoid	0.2593	0.9009	0.2207	0.9214
P+RSigELUS	0.2501	0.9287	0.1805	0.9471
P+RSigELUD	0.2062	0.9380	0.1458	0.9564
Proposed Function	0.0063	0.9982	0.0601	0.9899

B. Discussion

The results in TABLE I demonstrate the consistency and robustness of the proposed function, with an average training accuracy of 99.82% and validation accuracy of 98.99%. The low variation across the five repeats confirms its stability during training.

In TABLE II, the proposed function outperforms all compared activation functions, achieving the lowest training and validation loss, and the highest accuracy.
These findings underscore its effectiveness and potential as a preferred activation function for deep learning tasks.

VI. Conclusion and future work

In this paper, we introduced a novel fixed-parameter activation function for neural networks aimed at enhancing both accuracy and convergence speed, particularly on the MNIST dataset. Our proposed activation function outperforms established methods such as PSigmoid, GELU, and PReLU in terms of accuracy and convergence speed on the MNIST benchmark. However, we are currently testing its performance on more challenging tasks and diverse datasets to evaluate its broader applicability and robustness. We observed that the proposed activation function initially showed limited adaptability to datasets with significantly different characteristics, such as CIFAR-10, making it less versatile compared to trainable activation functions. However, by incorporating the ResNet architecture, we were able to enhance its performance, achieving an accuracy of almost 90% on CIFAR-10. This demonstrates that architectural modifications can significantly improve the function's generalizability. Further exploration of such enhancements will be discussed in the future work.

REFERENCES

[1] Y. Ying, N. Zhang, P. Shan, L. Miao, P. Sun, and S. Peng, "PSigmoid: improving squeeze-and-excitation block with parametric sigmoid," Appl. Intell., vol. 51, no. 10, pp. 7427–7439, 2021.
[2] L. Trottier, P. Gigu, and B. Chaib-draa, "Parametric exponential linear unit for deep convolutional neural networks," in Proc. 16th IEEE Int. Conf. Machine Learning and Applications (ICMLA), pp. 207–214, 2017.
[3] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in Proc. 27th Int. Conf. Machine Learning (ICML-10), pp. 807–814, 2010.
[4] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in Proc. 30th Int. Conf. Machine Learning (ICML), vol. 30, no. 1, p. 3, 2013

[5] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, “Self-normalizing neural networks,” in Proc. Adv. Neural Inf. Process. Syst., vol. 30, pp. 971–980, 2017.
[6] S. Kiliçarslan, K. Adem, and M. Celik, “An overview of the activation functions used in deep learning algorithms,” J. New Result Sci., vol. 10, no. 3, pp. 75–88, 2021.
[7] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: surpassing human-level performance on ImageNet classification,” in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, pp. 1026–1034, 2015.
[8] K. Adem, “P+FELU: flexible and trainable fast exponential linear unit for deep learning architectures,” Neural Comput. Appl., vol. 34, no. 24, pp. 21729-21740, 2022.
[9] V. S. Bawa and V. Kumar, “Linearized sigmoidal activation: A novel activation function with tractable non-linear characteristics to boost representation capability,” Expert Syst. Appl., vol. 120, pp. 346–356, 2019.
[10] S. Kiliçarslanand M. Celik, “RSigELU: a nonlinear activation function for deep neural networks,” Expert Syst. Appl., vol. 174, p. 114805, 2021.
[11] S. Kiliçarslan and M. Celik, “Parametric RSigELU: a new trainable activation function for deep learning,” Neural Computing and Applications, vol. 36, pp. 7595–7607, 2024.
[12] D. Hendrycks and K. Gimpel, “Gaussian error linear units (gelus),” arXiv preprint arXiv:1606.08415, 2016.
[13] K. Biswas, S. Kumar, S. Banerjee and A. K. Pandey, “TanhSoft—Dynamic Trainable Activation Functions for Faster Learning and Better Performance,” in IEEE Access, vol. 9, pp. 120613-120623, 2021.
[14] E. Işık, N. Ademović, E. Harirchian, F. Avcil, A. Büyüksaraç, M. Hadzima-Nyarko and B. Antep, “Determination of natural fundamental period of minarets by using artificial neural network and assess the impact of different materials on their seismic vulnerability,” Appl. Sci., vol. 13, no. 2, pp. 2076–3417, 2023.
[15] I. Pacal and S. Kılıcarslan, “Deep learning-based approaches for robust classification of cervical cancer,” Neural Computing and Applications, vol. 35 (25), pp. 18813–18828, 2023.
[16] H. Gao, Z. Wang, L. Cai and S. Ji, “ChannelNets: Compact and Efficient Convolutional Neural Networks via Channel-Wise Convolutions,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 8, pp. 2570-2581, 1 Aug. 2021.

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

Persian Intelligent Assistant in Healthcare Domain

Sarina Chitsaz✉, **Code Orcide: 0009-0008-8237-3658**
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran, s.chitsaz@mail.sbu.ac.ir
Mehrnoush Shamsfard**, Code Orcide: 0000-0002-7027-7529**
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran, m-shams@sbu.ac.ir

Abstract—Nowadays, advances in technology and medical science have led to significant changes in the field of healthcare. Consequently, an effort has been taken to develop an intelligent health assistant in the Persian language, focusing on the emergency department. To achieve this goal, a labeled dataset was prepared. Subsequently, an intelligent assistant architecture was developed, utilizing slot filling and speech act classification for natural language understanding. A dialogue manager was designed to address negation in patient statements, resulting in the classification of triage patients. Evaluation revealed that the assistant's performance matched that of emergency staff in 83% of cases.

Keywords— Intelligent Assistant, Natural language understanding, Speech act classification, Slot filling

Introduction

Considering the importance and status of the health sector, the rapid changes in today's society and the increasing needs and expectations regarding health services, people want to receive higher quality of services. Additionally, taking into account the vital role of factors such as accuracy, speed and reliability in healthcare and the vulnerability of the health system to human errors, utilizing artificial intelligence can bring comfort to patients as well as promoting health indicators in society. It can be acknowledged that nowadays using intelligent systems in health sector has become a necessity. Employing assistants that imitate human interactions can significantly transform this field and greatly impact communication with patients and their companions. Intelligent healthcare helps in maximizing the potential of existing resources which helps in remote monitoring of patients and reducing the costs of treatment. It also allows specialist doctors to provide their services without any geographical barriers. Due to the increasing number of visiting patients in medical centers, the use of intelligent assistants leads to saving both time and cost. As a result, healthcare workers are free to do more complex tasks. Owing to the critical importance of time and accuracy for medical staff in saving patients' lives, especially in the emergency department, the use of these assistants will reduce response times for patients in critical situations. Additionally, it will increase accuracy and focus on high-risk patients, prioritizing their care.

On the other hand, the use of the above technology has decreased unnecessary visits to health care centers. Thus, performing daily repetitive activities including providing inpatient and outpatient services has become faster and more precise.

Triage is a prioritization system and a decision-making process to manage the admission of patients and provide services to them in the emergency department of the hospital.

The purpose of triage in the emergency department is to answer the question: "At this moment, what is the priority of care of this particular patient among all patients attending the emergency department?" [1]. One of the methods of triaging patients is the "Emergency Severity Index (ESI.V) [2]," which is considered and used by the National Hospital Emergency Triage Committee in Iran as the most appropriate triage algorithm. The triage system is a five-level, easy-to-use tool that categorizes emergency department patients by simultaneously examining the severity of the condition and its required intervention [3]. In the first phase, the triage nurse estimates the severity of the disease based on scientific medical principles, which are categorized into 5 levels as follows:

- Level 1: It includes patients who often do not have vital symptoms and have life-threatening conditions.
- Level 2: Level 2 patients include patients who need to be visited by an emergency medicine specialist within ten minutes.
- Level 3: Most of the patients referred to the emergency department of the hospital are at this level, and they are visited and examined by a specialist emergency medicine physician in less than an hour.
- Level 4: This level includes patients who do not have any life-threatening symptoms and are visited by a specialist doctor in less than two hours, depending on the level of crowding and the number of patients in the department.

Submit Date: 2025-09-28
Accept Date: 2025-10-05
✉ Corresponding author

- Level 5: This level includes patients who have come for different requests such as simple visits. The duration of providing services to these patients will be less than four hours.

If the severity of the disease is not high (level 1 and 2 of triage), the triage nurse determines the triage level (level three, four and five) by estimating the number of required facilities.

Although working in medical domain has its merits, there are some challenges related to the features of this area. Characteristics of medical texts include [4]: telegraphic style (incomplete sentences), short texts (abbreviations, special phrases and characters), and negation (negation words or phrases)

In light of the of the benefits and challenges mentioned earlier, and the research that have been done so far, a Persian intelligent assistant in the field of healthcare focused on the emergency has not been designed. This is why the present research can bring several benefits to this area.

The contributions of this paper are summarized as follows:

- To the best of current knowledge, this Persian healthcare intelligent assistant in the emergency department represents the first dialogue system in this field.
- An annotated dataset, for training healthcare intelligent assistant models is generated.
- A framework is introduced for communicating with patients and extracting maximum information from their statements in order to respond appropriately.
- The model's performance in predicting triage levels of patients is evaluated.

The structure of the next sections of the article is as follows: In the second section, a review of related research is presented. In the third section, the general structure of the proposed approach is described, and in the fourth section, the obtained results are analyzed. Finally, in the fifth section, a summary of the research is provided.

Related Work

In general, the architecture of intelligent assistants contains 4 parts [5]: natural language preprocessing, natural language understanding (NLU), dialogue manager, and natural language generation.

After receiving the user's input and performing the necessary pre-processing, the dialogue system checks the user's input [6]. A frame structure can be used to understand the user's request that consists of two tasks: intent detection and slot filling.

In recent work, Shi et al. [7] address medical slot filling as a multi-label classification problem using a label-embedding attention model. This model identifies which words are highly relevant to the slot-filling task and selects keywords accordingly. After that, they use a text classification encoder as the query encoder.

BERT [8] has also been used for slot filling and intent detection. One of these studies was conducted by Chen et al. [9], where BERT was employed to perform both tasks jointly. This model was implemented on a dataset of size 4478 with 120 slots and 21 intents and on another dataset of size 13084 with 72 slots and 7 intents.

Inou et al. [10] presented an intelligent triage system called Healthbot for emergency rooms. This system interacts with patients and categorizes them based on priority level and medical specialty. The system utilizes the Google Dialogflow conversation interface along with sensors, voice recognition and a physical robot-like platform. The classification of medical specialties in Healthbot is divided into four groups: general medicine, surgery, orthopedics, and cardiology. The accuracy of the specialty classification in this bot is reported to be 82.2%.

In recent research, the emergence of ChatGPT has enabled its use as an intelligent health assistant. In a study involving 56 cases, Gebrael et al. [11] used ChatGPT to provide emergency care for patients with metastatic prostate cancer. They employed ChatGPT to assess the need for hospitalization or discharge and acknowledged its high sensitivity in determining patient admissions.

Approach

The proposed approach involves two sections: dataset generation and model architecture development.

Dataset Generation

At the first step, a dataset comprising exact formal and informal dialogues was collected from the emergency department triage in a hospital. Second, the process of de-identification was conducted. This process involves cleaning and removing identifying information to ensure the privacy of individuals. Finally, the dataset was labeled for speech act classification and slot filling. In total, the dataset contains 5,231 utterances.

Speech Acts Dataset: The speech acts of emergency department visitors include categories of information expressed by speakers based on the topic of conversation. There are eight categories in this dataset, named:

- Patient Signs
- Patient Information, such as age
- Patient History, including medical history, family history, etc.
- Patient Requests, such as serum injection or suturing
- Patient Questions
- Patient Confirmation
- Patient Negation
- Others: utterances that do not fall into any of the above categories

Each utterance in this section may consist of more than one speech act.

Slots Dataset: The labels in this part define the type of semantic information conveyed by each word. The slot filling task is considered as a sequence labeling task, where a chain of text maps to a chain of I, O, and B tags. In this method, each token is classified into one of three tags, representing the token's status in the sequence in an organized structure:

- B (Beginning): Represents the start of a new slot in the structure. A token that begins a new entity is labeled as "B."
- I (Inside): Indicates that a token is part of a specific structural slot, continuing from a preceding "B" tag.
- O (Outside): Denotes tokens that do not belong to any structural slots and are outside of all defined entities.

Once the sequence labeler has tagged a user utterance, a filler string for each slot in the tags can be extracted. The system can then decide how to complete the task for the user, ensuring precise dataset labeling. Owing to this, labeling was carried out in consultation with several experts in the field of healthcare. In addition to the “O” tag, 61 tags were defined within the dataset.

An example of Slot Labels and Speech Act Labels is shown in Fig.1.

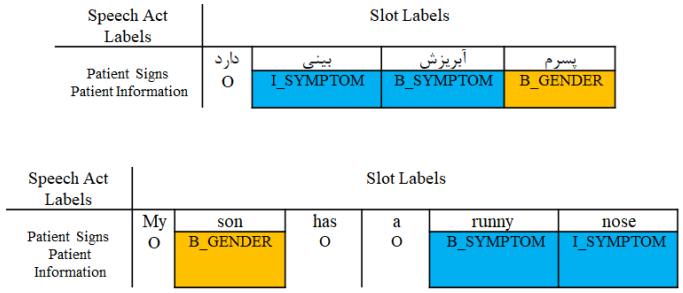


Fig. 1. Slot Labels and Speech Act Labels

Proposed Model Architecture

At the start of the intelligent assistant's work, after receiving the patient’s utterance, the input is sent to the natural language understanding module to comprehend what has been expressed. The NLU module is designed to enable computers to interpret human language in a way that is both meaningful and contextually relevant, converting human language into a machine-understandable structure. In this module, preprocessing is performed on the input, and the existing speech acts, along with the values of predetermined slots, are identified.

Speech act classification was treated as a text classification task. For this purpose, a BERT [8] pre-trained language model, one of the state-of-the-art methods in natural language understanding, was used. BERT is a transformer-based model initially trained on massive datasets such as Wikipedia and BooksCorpus. It also, was employed for the slot-filling task. Regarding the dataset's language is Persian, the tasks were fine-tuned on ParsBERT [12], a BERT-based model specifically designed for processing Persian text. For the NLU module, a joint framework for speech act classification and slot filling was proposed.

After understanding the utterance of the patient it is time for Dialogue manager. This module has a crucial role in controlling and management of the dialogue flow between a patient and the assistant. Dialogue manager consists of the Dialogue State Tracker (DST) and the Policy Manager. DST maintains the current status of a conversation while Policy Manager has the responsibility of managing and coordinating various policies in the intelligent assistant. It determines the system’s reactions and responses to the patient in specific situations. The defined policies in this module are a combination of rule-based and machine learning techniques.

When the speech acts and slot fillers are detected, the DST is updated and the new information is added to the previous data. Then the policies of the triage level 1 and level 2 are analyzed. The policies of the intelligent assistant are designed based on the protocols of emergency triage in Iran where uses ESI algorithm [3]. In every turn of the dialogue, if the patient’s condition is identified as level 1 or 2, immediate action is required. Otherwise, the prioritized questions for getting other information and symptoms of the patient will be asked by the assistant. The above procedure will repeat until all the required information is obtained.

Consequently, if the gathered information is sufficient, the triage level 3, 4 and five policies will check and the triage level of the patient will be detected. In the following, essential actions and guidance will be presented according to the patient's condition.

Additionally, in this step, a patient's medical history has been achieved on the account of the dialogue state tracker and the saved slots that hold information, signs, and the patient's history.

It is worth mentioning that one of the main challenges in the interaction between the patient and the intelligent assistant is the existence of negation in utterances.

Negation Detection Approach: During conversations, a patient can give positive or negative answers to the questions of the assistant. These replies can be divided into “Patient Confirmation” and “Patient Negation” by speech act classification which either of them have their own suitable policy.

Negation changes the entire meaning of an utterance, and its existence makes it one of the important challenges in the healthcare field. In this research, the designed approach for the Negation Detection task has been developed by combining deep learning and rule-based techniques based on the Persian language.

As Rahimi and Shamsfard stated in their article [13], due to differences between the structural patterns of negation in the Persian language and the English language, the available methods must be modified and properly adapted. One example of these differences is given below:

- When using negative adverbs like “never (هرگز)” in the sentence “I have never smoked,” the English pattern uses a negative adverb with a positive verb. However, in the Persian pattern, as in the sentence “هرگز سیگار نکشیده‌ام,” a negative verb is paired with a negative adverb.

Regarding these differences, the proposed approach uses a negation tag for negative verbs. This label is accessible in the Slots Dataset. In this way, negation can be detected simultaneously along with other slots by the ParsBERT transformer model.

On the other hand, there is an exception. In the Slots dataset, sentences like “I have no balance (تعادل ندارم)” are classified under the “Patient Signs” tag instead of “Patient Negation.” As a result, for detecting negation in the text, a two-stage mechanism is introduced. In this mechanism, if the output of the NLU module includes both the negation slot label and the speech act “Patient Negation” at the same time, negation is recognized in the text.

After detecting negation with the help of ParsBERT, the next step is Negation Scope Resolution [14], which is the process of finding the negated part of a sentence. Inspired by the rule-based methods of NegEx [15] and ConText [16], and considering the Persian language, the previously extracted slot before the negative sign is affected and becomes negated. At this point in the intelligent assistant, the negation sign, along with its corresponding slot, is removed from the DST to only consider the patient's existing condition.

Experiments

The assistant's performance has been evaluated in the classification of triage patients.

The proposed model was implemented in Python programming language and the Google Colab environment, utilizing a Tesla T4 GPU with 16 GB of memory. The proposed NLU model was fine-tuned on ParsBert using the introduced dataset. After conducting several tests, hyperparameters such as a batch size of 32, a learning rate of 2e-5, and a maximum length of 64 were used, in addition to Adam's optimizer [17] and early stopping. Considering the fact that the assistant operates in the health-care domain, the 5-layer cross-validation method has been used for more reliable results.

Dataset: According to the nature of this assistant, to evaluate the correctness of its diagnosis and answers, the test dataset was provided. This dataset was developed under the supervision of three emergency medicine specialists, including the following features:

- It consists of 132 case scenarios.
- 14 cases did not meet the input criteria.
- 96 cases were low-risk patients (level 3, 4 and 5 ESI triage)

Evaluation Metrics: For evaluating the performance of the intelligent assistant two metrics (Accuracy and F1-Score) have been used.

Accuracy: In the context of a triage system like ESI, accuracy measures how well the system classifies patients into their relevant severity levels.

F1-Score: F1-Score is based on true positive, true negative, false positive and false negative values. Each of these is defined in the triage system as follows:

- True positive: The system correctly identifies the level of patient triage as high severity and it is also high in reality.
- True negative: The system correctly identifies the level of patient triage at low severity and it is also low in reality.
- False Positive: The system incorrectly identifies a patient's triage level as high severity while it is not high in reality (Over-Triage).
- False negative: The system incorrectly identifies a patient's triage level as low severity while it is high in reality (Under-Triage).

Results: In this section, the values of Accuracy and F1-Score were calculated as 83% and 89.32%, respectively. The confusion matrix of the intelligent healthcare assistant's predictions and the reference's triage levels is shown in Table I.

TABLE II. CONFUSION MATRIX OF DIFFERENT TRIAGE LEVELS (ROWS: ACTUAL LEVELS, COLUMNS: PREDICTED LEVELS)

Triage Levels	**1**	**2**	**3**	**4**	**5**	**Total**
1	7	1	0	0	0	8
2	1	10	3	0	0	14
3	0	2	48	5	1	56
4	0	0	2	19	3	24
5	0	0	0	2	14	16
Total	8	13	53	26	18	118

In the classification of ESI triage, determining the exact level of the patient is vital. As shown in the confusion matrix in Table I, the intelligent assistant correctly classified the majority of cases. The results are discussed below:

- Out of 8 patient cases at level 1, the intelligent assistant correctly identified the level in 7 cases, and only in 1 case, the triage level was incorrectly declared as level 2.
- Out of 14 patient cases at level 2, the intelligent assistant correctly identified the level in 10 cases and declared 1 case as level 1, and 3 cases as triage level 3.
- Out of 56 patient cases at level 3, the intelligent assistant correctly identified the level in 48 cases and declared 2 cases as level 2, 5 cases as level 4, and 1 case as triage level 5.
- Out of 24 patient cases at level 4, the intelligent assistant correctly identified the level in 19 cases and declared 2 cases as level 3, and 3 cases as triage level 5.
- Out of 16 patient cases at level 5, the intelligent assistant correctly identified the level in 14 cases, and only in 2 cases, the triage level was incorrectly declared as level 4.

According to the aforementioned results, it is observed that mistakes occurred relatively more in the triage levels with greater categorical overlap.

Moreover, as shown in the statistics in Table I, the assistant's best performance was in determining Level 3, indicating that patients requiring multiple resources are accurately categorized, preventing unnecessary escalation to higher levels. In second place, Level 1 performed well, revealing that the model works effectively for patients who have life-threatening conditions and need immediate interventions. After Level 1, in third place, the assistant was effective in Level 5 diagnosis, correctly identifying the majority of non-urgent patients, helping optimize resources.

In addition, it should be noted that the performance results of this assistant in determining the triage levels of patients demonstrate high diagnostic accuracy, and in 83% of cases, it was consistent with the performance of emergency staff.

Conclusion and Future Work

In this research, an intelligent assistant for the healthcare domain was developed. Its purpose was to assist the emergency department in prioritizing between high-risk and low-risk patients based on the ESI algorithm. The process commenced with data preparation through field research to collect data from hospitals. Subsequently, a joint model was used to fine-tune ParsBERT to understand patients' utterances with the help of a negation detection mechanism. Patients were then prioritized according to the extracted information and the defined policies related to the emergency triage algorithm. In the final stage, the performance of the assistant was evaluated using a dataset prepared under the supervision of experts in the relevant field. The results showed that, in the majority of cases (83%), the assistant correctly identified the patients' triage levels.

In future work, the dataset will be expanded to enhance the robustness and generalizability of the outcomes. Additionally, more evaluations will be conducted. One of them will be ablation studies of the negation module in order to investigate its contributions within the assistant. The other will be a comprehensive comparison (specifically in the context of few-shot learning capabilities) between the healthcare assistant and large language models (LLMs) to identify strengths, weaknesses, and areas of improvement in the reliability and performance of the healthcare intelligent assistant.

References

[13] R. Sánchez-Salmerón et al., "Machine learning methods applied to triage in emergency services: A systematic review," Int. Emerg. Nurs., vol. 60, p. 101109, Jan. 2022, doi: 10.1016/j.ienj.2021.101109.

[14] L.-H. Yao, K.-C. Leung, C.-L. Tsai, C.-H. Huang, and L.-C. Fu, "A Novel Deep Learning–Based System for Triage in the Emergency Department Using Electronic Medical Records: Retrospective Cohort Study," J. Med. Internet Res., vol. 23, no. 12, p. e27008, Dec. 2021, doi: 10.2196/27008.

[15] N. Shafaf and H. Malek, "Applications of Machine Learning Approaches in Emergency Medicine; a Review Article," Arch. Acad. Emerg. Med., vol. 7, no. 1, Art. no. 1, Jul. 2019, doi: 10.22037/aaem.v7i1.410.

[16] H. Lu, L. Ehwerhemuepha, and C. Rakovski, “A comparative study on deep learning models for text classification of unstructured medical notes with various levels of class imbalance,” BMC Med. Res. Methodol., vol. 22, no. 1, p. 181, Jul. 2022, doi: 10.1186/s12874-022-01665-y.

[17] L. Qin, T. Xie, W. Che, and T. Liu, “A Survey on Spoken Language Understanding: Recent Advances and New Frontiers,” in Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, Montreal, Canada: International Joint Conferences on Artificial Intelligence Organization, Aug. 2021, pp. 4577–4584. doi: 10.24963/ijcai.2021/622.

[18] H. Weld, X. Huang, S. Long, J. Poon, and S. C. Han, “A Survey of Joint Intent Detection and Slot Filling Models in Natural Language Understanding,” ACM Comput. Surv., vol. 55, no. 8, p. 156:1-156:38, Dec. 2022, doi: 10.1145/3547138.

[19] X. Shi, H. Hu, W. Che, Z. Sun, T. Liu, and J. Huang, “Understanding Medical Conversations with Scattered Keyword Attention and Weak Supervision from Responses,” Proc. AAAI Conf. Artif. Intell., vol. 34, no. 05, Art. no. 05, Apr. 2020, doi: 10.1609/aaai.v34i05.6412.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. doi: 10.18653/v1/N19-1423.

[21] Q. Chen, Z. Zhuo, and W. Wang, “BERT for Joint Intent Classification and Slot Filling,” Feb. 28, 2019, arXiv: arXiv:1902.10909. doi: 10.48550/arXiv.1902.10909.

[22] A. Inoue, M. Prado, and F. Cozman, “Automated Emergency Room Triage: Helping Patients Get the Best Treatment,” in Anais do Encontro Nacional de Inteligência Artificial e Computacional (ENIAC 2020), Brasil: Sociedade Brasileira de Computação - SBC, Oct. 2020, pp. 591–602. doi: 10.5753/eniac.2020.12162.

[23] G. Gebrael et al., “Enhancing Triage Efficiency and Accuracy in Emergency Rooms for Patients with Metastatic Prostate Cancer: A Retrospective Analysis of Artificial Intelligence-Assisted Triage Using ChatGPT 4.0,” Cancers, vol. 15, no. 14, Art. no. 14, Jan. 2023, doi: 10.3390/cancers15143717.

[24] M. Farahani, M. Gharachorloo, M. Farahani, and M. Manthouri, “ParsBERT: Transformer-based Model for Persian Language Understanding,” Neural Process. Lett., vol. 53, no. 6, pp. 3831–3847, Dec. 2021, doi: 10.1007/s11063-021-10528-4.

[25] Z. Rahimi and M. Shamsfard, “A Neuro Symbolic Approach for Contradiction Detection in Persian Text,” JUCS - J. Univers. Comput. Sci., vol. 29, no. 3, pp. 242–264, Mar. 2023, doi: 10.3897/jucs.90646.

[26] A. Yoshida, Y. Kato, and S. Matsubara, “Revisiting Syntax-Based Approach in Negation Scope Resolution,” in Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (*SEM 2023), A. Palmer and J. Camacho-collados, Eds., Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 18–23. doi: 10.18653/v1/2023.starsem-1.3.

[27] W. W. Chapman, W. Bridewell, P. Hanbury, G. F. Cooper, and B. G. Buchanan, “A simple algorithm for identifying negated findings and diseases in discharge summaries,” J. Biomed. Inform., vol. 34, no. 5, pp. 301–310, Oct. 2001, doi: 10.1006/jbin.2001.1029.

[28] H. Harkema, J. N. Dowling, T. Thornblade, and W. W. Chapman, “ConText: an algorithm for determining negation, experiencer, and temporal status from clinical reports,” J. Biomed. Inform., vol. 42, no. 5, pp. 839–851, Oct. 2009, doi: 10.1016/j.jbi.2009.05.002.

[29] M. Reyad, A. M. Sarhan, and M. Arafa, “A modified Adam algorithm for deep neural network optimization,” Neural Comput. Appl., vol. 35, no. 23, pp. 17095–17112, Aug. 2023, doi: 10.1007/s00521-023-08568-z.

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

Improving Predicted Answer Accuracy in Visual Question and Answer Systems using Attention Mechanisms and Neural Networks

Fatemeh Rezaei, **Code Orcide:**
Department of Computer Engineering, Golestan University, Gorgan, Iran, Fatemeh.rezaei1998@gmail.com,
Soheila Karbasi✉, **Code Orcide:**
Department of Computer Engineering, Golestan University, Gorgan, Iran, s.karbasi@gu.ac.ir

Abstract
In recent years, one of the most widely studied areas in computer vision and natural language processing (NLP) is the interdisciplinary problem of Visual Question Answering (VQA), which involves the integration of computer vision and NLP. In VQA systems, an image and a text-based question about the image serve as inputs, and the system must predict the correct answer to the question. The main objective of these systems is to maximize the accuracy of correct answer predictions.
Important challenges in this field include the need for large and suitable datasets as well as powerful hardware for training the model. Key factors to improve the performance of these models include selecting the appropriate neural network for processing the inputs, selecting the appropriate dataset, and the method of combining the features extracted from the inputs. Also, using different attention mechanisms can improve the overall performance of the system. Furthermore, incorporating various attention mechanisms into the model can significantly enhance the overall performance of VQA systems. In these systems, different neural networks are employed to process inputs: convolutional neural networks (CNNs) with various architectures are used for image processing, and different types of recurrent neural networks (RNNs) are used for text processing.
In this research, the architecture of the convolutional neural network is changed and the self-attention mechanism is used in text processing and the Skipgram language model is used for embedding the input text. The performance of the proposed model is evaluated on two datasets, VQA 1.0 and VQA 2.0. The results show that the proposed model has been able to increase the overall accuracy in the VQA 1.0 dataset to 67.25% and in the VQA 2.0 dataset to 61.57%, showing significant improvement over the baseline models.

Keywords: visual question and answer system, convolutional neural network, recurrent neural network, attention mechanism.

1. Introduction

Currently, one of the most widely researched topics in artificial intelligence is VQA. VQA involves integrating both computer vision and natural language understanding (NLU). In these systems, an image and a text-based question about the image are used as inputs, and the system is responsible for automatically answering the question and predicting the correct answer. The integration of computer vision and NLU is crucial for this task [1]. The use of appropriate neural networks to process input data, along with selecting suitable datasets, are two critical factors that significantly enhance the accuracy of final predictions. Furthermore, how to integrate the features and representations obtained from inputs is a vital issue in these systems. Due to the high efficiency, importance, and attractiveness of this topic, many researchers have conducted studies to improve the accuracy of these systems, yielding positive outcomes. Various neural networks can be utilized in VQA systems. Both image processing and text processing benefit from different neural networks to ensure high accuracy of the outputs. Their efficiency in processing inputs has led to their widespread adoption. In addition, different neural networks, such as CNNs and various types of RNNs, different attention mechanisms and transformers can be used to increase output accuracy. Many studies show that these approaches significantly improve the accuracy of predicted answers in VQA models [2, 3]. Some of the challenges in this field include the need for appropriate hardware to train the models, the integration of features and representations derived from the question and image to achieve a common representation, and the selection of suitable neural networks and datasets. Today, computer vision and NLP

Submit Date: 2025-09-22
Revise Date: 2025-11-02
Accept Date: 2025-12-07
✉ Corresponding author

play a crucial role across various domains, assisting human beings in diverse fields. As mentioned earlier, VQA requires the integration of computer vision and NLP and has numerous applications in different areas. For example, in the medical industry, VQA systems can be used to facilitate certain tasks, making processes easier for both doctors and patients.

Additionally, these systems can be highly beneficial for blind and visually impaired individuals who struggle to recognize details in images, helping them comprehend visual content and extract important information. More broadly, VQA systems are essential for tasks that require image-based information retrieval, where users can extract relevant details through natural language questions. Figure 1, illustrates the general schematic of a visual question answering model. This model takes an image and a question as inputs and predicts the appropriate answer.

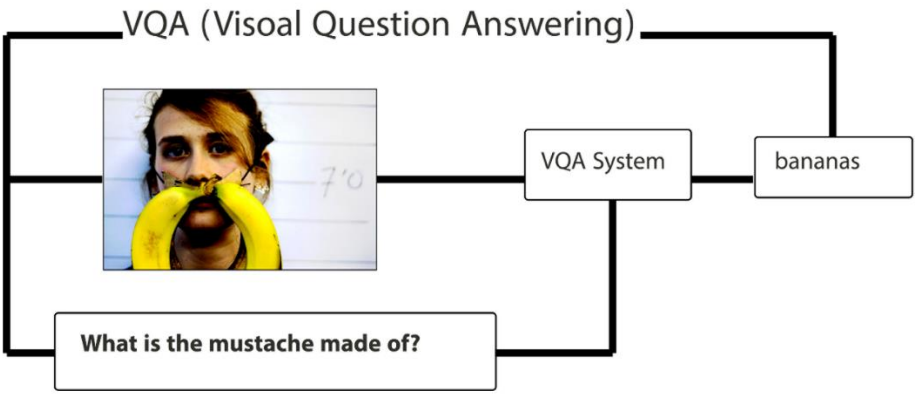


Figure 1: General schematic of visual question and answer system [3]

While the VQA dataset has enabled significant progress in visual question answering, ongoing researches continue to enhance the effectiveness of reasoning capabilities, linguistic diversity, and multimodal understanding. This study contributes a model based on recent advances in attention mechanisms and deep visual representation.

1.1 Challenges

In recent years, deep learning has excelled in solving various types of problems. The concept of deep learning is inspired by the functioning of the human brain. With the help of new technologies, it has achieved various successful results in artificial intelligence and machine learning. For deep learning, datasets and neural networks are two essential components [4]. Word embedding is a dense representation of words in the form of numerical vectors that can be learned by various linguistic models. Word embedding representations can reveal many hidden relationships between different words [5]. The objective function of word2vec ensures that words with the same concept and meaning have similar embeddings [6, 7]. This algorithm uses the following two methods to create word vectors:

Continuous Bag of Words (CBOW): When this method is applied to all the words in the text, the final word vectors become the target vectors [6].

Skip-gram: Algorithmically, the CBOW and Skip-gram methods are similar. The main difference is that CBOW predicts the target words from the input context, while Skip-gram works in the opposite direction, predicting context words from the target word [6].

Convolutional Neural Networks (CNNs) are among the most influential architectures in deep learning, particularly for visual recognition tasks. A standard CNN consists of convolutional layers, pooling layers, and fully connected layers, each responsible for extracting and transforming features at different levels of abstraction. While traditional CNNs operate in a purely feedforward manner, feedback CNNs introduce recurrent or top-down connections that allow the network to refine its internal representations over multiple iterations. This feedback mechanism enables the model to incorporate contextual information and adjust its predictions dynamically, which has shown to improve performance in complex tasks such as object recognition, scene understanding, and segmentation. These enhancements are particularly valuable in scenarios where initial predictions may be ambiguous or incomplete, allowing the model to "rethink" its interpretation of the input data [8, 9]. This network was inspired by experiments on the visual cortex, with its architecture simulating the pattern of connections between neurons in the brain. The features obtained by the convolutional layers are combined by the fully connected layer to form a feature vector, which is then passed to a Softmax function to predict the correct class [10]. The number of outputs of this layer

matches the number of available classes [11]. Batch normalization is typically applied after a convolutional or fully connected layer and before the non-linear activation function. As discussed by Szandała [12], the placement of activation functions significantly influences the performance of deep neural networks. Batch normalization helps stabilize the distribution of inputs to these activation functions, which in turn improves their effectiveness. By reducing internal covariate shift, batch normalization enables the use of higher learning rates and mitigates sensitivity to weight initialization, ultimately accelerating training and enhancing model accuracy.

Although batch normalization is widely used to stabilize and accelerate training in deep neural networks, its direct impact on final model accuracy remains a subject of debate. As reviewed by Szandała [12], the effectiveness of activation functions is closely tied to the distribution of their inputs, which batch normalization helps regulate. However, some studies suggest that the primary benefit of batch normalization lies in enabling higher learning rates and smoother optimization, rather than consistently improving accuracy. Therefore, while BN contributes to training efficiency, its influence on accuracy may vary depending on the architecture and task.

Recurrent Neural Networks (RNNs), with their capacity to capture sequential dependencies, are widely applied in natural language processing (NLP), speech recognition, and other tasks involving ordered input data. While Li et al. [13] utilized n-gram-based models instead of RNNs, their study underscores the significance of sequence modeling in producing coherent image descriptions. While RNNs are adept at learning short-term dependencies, they struggle with long-term time series. To address this issue, the Long Short-Term Memory (LSTM) architecture was introduced by modifying the network's design [14]. LSTMs can process and classify long-term data that requires historical information and time series with delays. This network trains models using backpropagation [14].

Bidirectional Long Short-Term Memory (Bi-LSTM) is a type of RNN. Unlike unidirectional LSTMs, Bi-LSTMs process data in two directions by utilizing two hidden layers. This network maintains the temporal order of words in a text, enabling it to ignore unnecessary words using the forget gate [15]. Currently, the attention mechanism is used in both NLP and computer vision. For instance, when people read a text to understand, they do not pay attention to every word but focus on specific words to grasp the meaning. Accordingly, the attention mechanism was developed to enable neural networks to mimic human behavior [16]. By using the attention mechanism, the points and areas of an image related to a question can be identified as prominent. Further processing by neural networks on these points and areas allows for predicting the appropriate answer. Additionally, the attention mechanism can be used to better process input text in VQA systems. In fact, models using the attention mechanism to solve image question-answer problems can apply this mechanism to the image, the question, or both simultaneously [17]. In VQA systems, an image and a question in natural language are provided as inputs. The VQA system attempts to predict the appropriate answer based on these inputs, using the visual elements of the image and inferences derived from the question [18]. The main challenge in these systems is successfully combining the representations produced for the input image and question using different neural networks to predict the correct answer [19].

Approaches based on hybrid models with modular structures enable the determination of appropriate architectures to answer questions according to the incoming query. Among these hybrid models, we can mention those based on dynamic memory networks (DMNs) [19].

The QA-COCO dataset includes 123,268 images, with 72,783 used for training and 38,948 for testing. Each image is paired with a question and an answer. The questions in this dataset fall into four categories: object, color, number, and location. Six percent of the questions in the QA-COCO dataset have one-word answers, which simplifies evaluation. However, one of the limitations of this dataset is the presence of numerous grammatical errors in the questions [20].

The VQA 1.0 dataset represents the initial release of the Visual Question Answering benchmark, utilizes images sourced from the QA-COCO dataset. It comprises 204,721 images, each paired with three questions. For every question, ten human-provided answers are included, enabling robust evaluation of model performance across diverse linguistic interpretations [21].

The VQA 2.0 dataset is the second version of the VQA dataset. It also contains 204,721 images from the QA-COCO dataset, but includes 110,504 questions. There are between three and five questions per image, with ten answers available for each question [21].

2. Related works

Ma et al. [22] proposed a CNN-based framework to predict answers in Visual Question Answering (VQA) systems, demonstrating the effectiveness of convolutional features in understanding image content and aligning it with natural language queries. Unlike most models in this field, the proposed model does not use RNNs, but relies only on a set of CNNs. This model comprises three CNNs. The first CNN extracts some features from the input image using a 16-layer Net-VGG type architecture [8] to generate a representation of the image. Another CNN processes the input text where initially, the words (questions) of the input text are embedded using the Skipgram model [6]. Then, convolutional layers followed by max-pooling layers process the input text, extract its features, and generate a representation of the text. The schematic of the CNN used for text processing is shown in Figure 2.

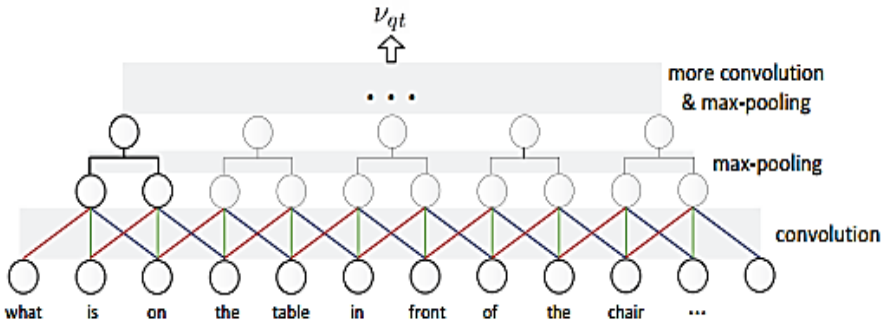


Figure 2: Convolutional neural network (CNN) schematic for input text processing [22]

Using these representations, a multi-modal CNN generated a joint embedding by combining features from both the input text and image. In this framework, ResNet [10] was employed as the visual feature extractor. The resulting fused representation was passed through a Softmax layer to predict the most probable answer. To evaluate the performance of this model, the QA-COCO dataset and DAQUAR datasets [20] were used. This model achieved 54.40% accuracy on the QA-COCO dataset and 42.76% accuracy on the DAQUAR dataset for single-word responses. The framework of the proposed model is shown in Figure 3.

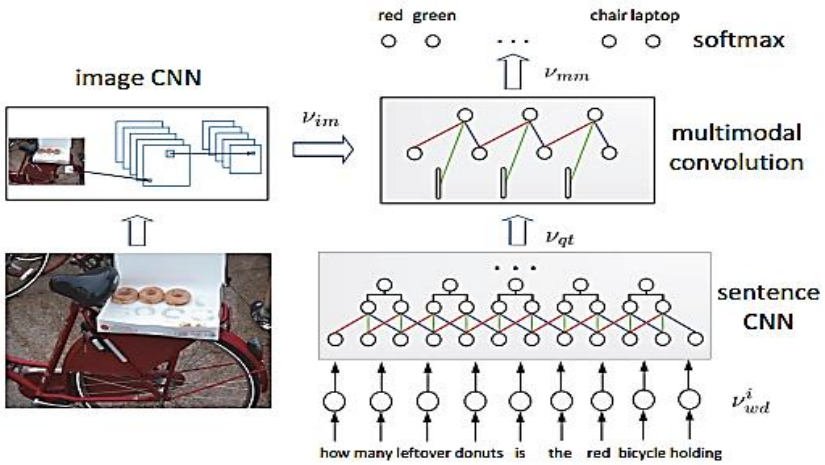


Figure 3: Architecture of the proposed model to predict the response [22]

Malinowski et al. [23] proposed a neural-based framework that combined Convolutional Neural Networks (CNNs) for visual feature extraction with Recurrent Neural Networks (RNNs), specifically Long Short-Term Memory (LSTM) units, to model sequential dependencies in textual questions. This architecture enabled the system to jointly reason over image and language modalities, effectively predicting answers in Visual Question Answering (VQA) tasks. In the proposed method, a pre-trained CNN (GoogLeNet type) is used to extract features from the input image. Because the final answer may consist of multiple words, in each iteration, the last predicted word is fed into the short-term memory network through a recursive loop to predict the subsequent words. The schematic of the answer generation process in the proposed method is shown in Figure 4.

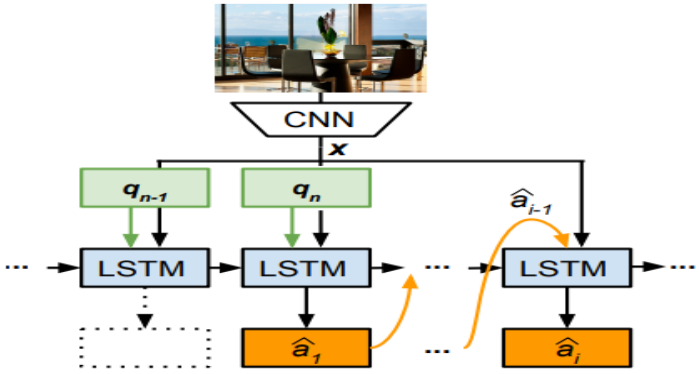


Figure 4: Prediction of the response in the proposed model [23]

A new framework based on the attention mechanism called "attention co-Hi" is proposed to predict answers in VQA systems [24]. In this method, three stages of embedding are used on the input question.

A new framework based on dynamic memory networks (DMNs) is proposed for solving the image question answering problem [25]. This model employs four modules: input module, question module, episodic memory module, and answer module. The input module processes inputs with different neural networks and converts them into vectors called "Facts". The input to this module is the image, which is divided into small local areas, each treated as equivalent to a sentence in the text input module.

A Bi-LSTM network is used to process the input text, and a Faster R-CNN neural network processes the input image, predicting the answer in the VQA system [26].

A novel approach (GloVe) is introduced to learn **word embedding** by combining **global matrix factorization** and **local context window methods [7]**. Unlike traditional models, GloVe constructs word vectors based on **word co-occurrence statistics**, capturing fine-grained semantic relationships. The authors propose a **log-bilinear regression model** that efficiently leverages statistical information by training only on **nonzero elements** in a word-word co-occurrence matrix. This method results in a **vector space with meaningful substructure**, enabling **word analogy tasks** and outperforming previous models in **similarity tasks and named entity recognition**.

A novel approach is presented to **Zero-Shot Learning (ZSL) [27]**. One of the key challenges in visual classification tasks is the difficulty of collecting training images. Traditional ZSL methods often rely on **textual descriptions or attribute-based representations** to define new categories. However, this study introduces an **alternative modality** by leveraging **visual abstraction** to learn complex concepts. This research specifically focuses on **human-related concepts and interactions**. The authors propose a method that allows users to **generate training data through abstract visualizations,** such as **clip-art representations** where characters' **body posture, facial expressions, gaze, and gender** can be manipulated to depict interactions. This approach enables **models to be trained on abstract images** and later tested on real-world photographs. The findings of this study demonstrate that **visual abstraction** can be effective in learning complex concepts and can outperform certain traditional approaches. Additionally, the paper introduces an **explicit mapping between abstract and real-world domains**, helping models bridge the gap.

A number of neural network modules and their combinations are employed in a new model to predict appropriate responses in visual question answering (VQA) systems [28]. In this model, after embedding the words of the input text, the text is passed through a recurrent neural network (RNN) based on long short-term memory (LSTM), where the question is processed, and an output is generated. Additionally, the embedded text is fed into a text parser that decomposes the text into its fundamental components and analyzes them. Based on the output of this analysis, the model selects one or a combination of modules needed to predict the response. For image processing, the model utilizes a convolutional neural network (CNN) with a 16-layer VGGNet architecture to extract visual features. These extracted features serve as input to the existing modules, which operate based on an attention mechanism. Finally, the output of a single module or a combination of modules is integrated with the text representation, and the final answer is predicted using Softmax layers. The proposed architectural structure of this study is illustrated in Figure 5.

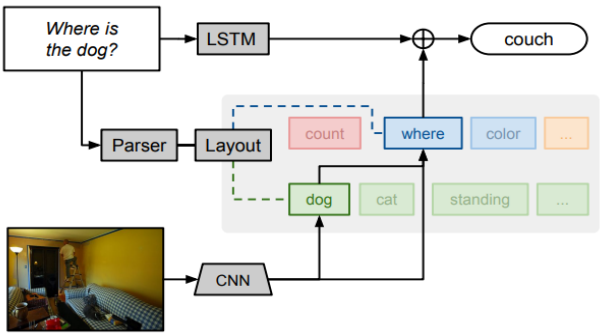


Figure 5: The proposed architectural structure [28]

In recent years, **bounding box-based features** have become the standard in vision-and-language tasks like VQA. The effectiveness of **grid features** in **VQA systems is** examined [29] which reevaluates **grid features** and demonstrates their significant potential. This research found that **grid features** not only offer **higher processing speed** but can also achieve **comparable accuracy** to bounding box-based features when properly preprocessed [29]. Through **extensive experiments**, the study confirms that these findings hold across different VQA models, various datasets, and even in related tasks like **image captioning**. A key advantage of this approach is its **simplified model design and training process**, allowing for **end-to-end learning**. The study highlights that VQA models can be trained **directly from pixels to answers**, eliminating the need for **image region annotations** during preprocessing.

A novel approach is introduced to **VQA** by incorporating **concept-aware representations [30].** Unlike traditional VQA models that rely only on image and text analysis, ConceptBert integrates **external knowledge** from structured sources like **Knowledge Graphs (KGs)** to enhance reasoning capabilities. The model employs a **multi-modal representation** that integrates **concepts, vision, and language,** inspired by the **BERT architecture**. By leveraging **ConceptNet KG**, ConceptBert encodes **common sense knowledge** to improve responses to complex questions that require more than just visual understanding. The approach is evaluated on **OK-VQA and VQA datasets,** demonstrating its effectiveness in handling **knowledge-intensive queries**.

A novel approach to **VQA, Multimodal Residual Networks (MRN)** is introduced which extends deep residual learning [31]. Unlike traditional residual learning, MRN effectively integrates **vision and language** through **element-wise multiplication**, enhancing multimodal representations. The study explores various multimodal architectures and demonstrates **state-of-the-art performance** on VQA datasets for both **Open-Ended and Multiple-Choice tasks**. Additionally, it introduces a method to **visualize attention effects** within joint representations using **backpropagation**, even when spatial information is absent.

More recently, de Faria et al. [32] conducted a comprehensive survey on VQA techniques, outlining the transition from early CNN-RNN architectures to advanced transformer-based and vision-language pre-trained models. Their work highlighted the important role of multimodal fusion and attention mechanisms in improving answer accuracy and generalization across diverse datasets. However, the study primarily focused on architectural trends and lacked in-depth empirical comparisons or critical analysis of model limitations in reasoning for further exploration in practical performance and dataset specific challenges. Their survey also categorized VQA architectures into early fusion, late fusion, and joint embedding strategies, emphasizing how each approach handles multimodal integration. Early fusion methods combined image and text features at the input level, while late fusion techniques processed each modality independently before merging them at the decision stage. Joint embedding models, particularly those based on transformers, enabled deeper cross-modal interactions through attention mechanisms. Despite these advancements, the authors noted that many models still struggle with complex reasoning, compositionality, and external knowledge integration highlighting persistent challenges in achieving human-level understanding in VQA systems.

3. Proposed model

3.1 Methodology

In this research, we introduce a new VQA model based on the architecture proposed in reference [4]. The model receives an image and a natural language question as input. First, the words in the question are converted into numerical vectors using the Skipgram model, and word embeddings are generated. These vectors are then passed through a self-attention mechanism, followed by a Bi-LSTM network for sequential

text processing. The output representation from the Bi-LSTM is further refined using a multi-head attention (MHA) module to capture contextual dependencies and generate the final textual representation. Simultaneously, the input image is processed using a 152-layer ResNeXt CNN, which extracts a visual feature vector. This image feature vector is then fed into the same MHA module alongside the textual representation, enabling cross-modal attention and joint feature fusion for answer prediction.

After processing the inputs with neural networks and obtaining their representation vectors, the textual and visual features are concatenated to form a joint representation. Based on Figure 6, this concatenation are performed along the feature dimension, meaning that the output vectors from the Bi-LSTM (text) and the CNN (image) are aligned side-by-side to form a single matrix. This joint matrix representation is then passed through two convolutional layers to extract higher-level fused features, and the resulting output is fed into a Softmax function for answer prediction. Initially, attention weights are computed over the input features, helping to highlight the most relevant components and reveal relationships between modalities.

In other words, multiple attention mechanisms are calculated on the features of the image. The output obtained from this step is then integrated with the features from the inputs using the concatenate method. The final output, a common representation, is passed through two fully connected layers and sent to a Softmax function to predict the final answer and obtain the probabilities for each answer class. Afterward the predicted answer and its probabilities are determined. Multi-head attention mechanism has been simultaneously utilized in both text and image processing. This choice allows the model to identify complex semantic and visual dependencies and establish deeper connections between multimodal data. The use of this mechanism not only enhances textual comprehension in linguistic contexts but also aids in better recognition of key regions in image processing. By leveraging multi-head attention, the model can process multiple levels of information. In text processing, this mechanism enables the preservation of long-range dependencies between words, while in image processing, it facilitates focus on key areas and improves the distribution of visual information. This capability allows the model to generate richer representations of data, ultimately boosting the accuracy of response predictions. The schematic of the proposed visual question answering model in this research is shown in Figure 6.

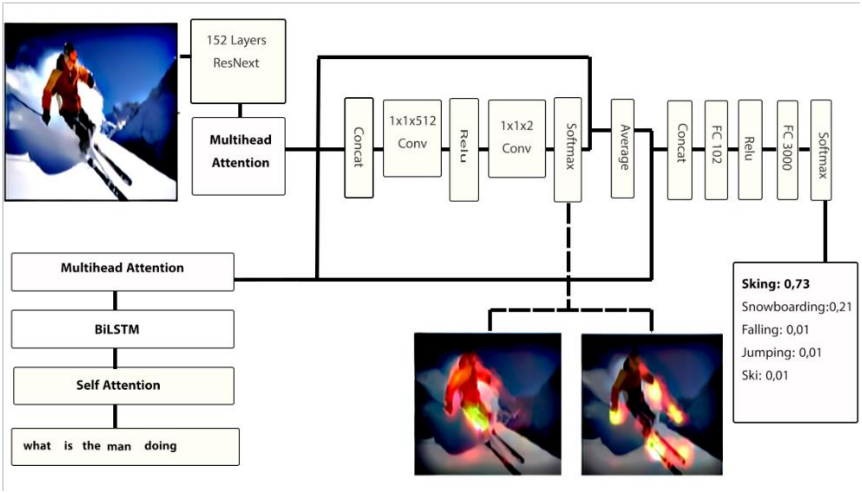


Figure 6: Architecture of the proposed research model

As previously mentioned, the proposed model in this research utilizes a CNN with a ResNeXt architecture of 152 layers to process input images. It should be noted that the reference article [4] uses a CNN with a ResNet architecture, also with 152 layers, for the same purpose. The ResNeXt architecture is an improvement over the ResNet architecture, increasing accuracy and strengthening the object recognition process in images without adding more parameters. Additionally, this architecture reduces error compared to the ResNet architecture. As such, the ResNeXt architecture is one of the best methods for object detection due to its higher accuracy and lower error rate compared to the ResNet architecture.

One of the most important components in a VQA system is the processing of textual input (questions about the image). In the reference article [4], an RNN of the LSTM type is used to process the input text after embedding the words. The input text is given to this network in the form of numerical vectors, from which textual features are extracted to obtain the representation of the text input. In the proposed model of this research, the Skipgram language model is used for text input processing. After embedding, the words are given to an RNN of the BiLSTM type. In fact, in the proposed model, an RNN of the BiLSTM type is used

instead of an RNN of the LSTM type. In an RNN, long-term processing is done in one direction, meaning the current input and the output of the past are used in the recurrent network. However, BiLSTM processes information in two directions. In the current input processing, in addition to using the output of the previous processing and past information, future information is also utilized. For this reason, processing in an RNN of the BiLSTM type is better and stronger than processing in an RNN of the simple LSTM type. Therefore, in this research, we changed the input text processing part and instead of using an RNN of the LSTM type, we used an RNN of the BiLSTM type.

In the architecture of the proposed model in the reference article [4], only a soft attention mechanism is used. Meanwhile, in the proposed model of this research, by applying changes to the model from the reference article [4], two additional attention mechanisms are used alongside the soft attention mechanism: the multi-headed attention mechanism and the self-attention mechanism. The reason for using the self-attention mechanism in this research is to better understand and accurately measure the relationships between the words in the input text. Additionally, the multi-headed attention mechanism with eight heads is used to apply eight attention mechanisms to the input simultaneously and in parallel, thus improving input processing.

As mentioned in previous articles, one of the most useful datasets in VQA is the VQA dataset, available in two versions: VQA1.0 and VQA2.0. The reference article [4] utilized both versions of this dataset to train the proposed model. Accordingly, we use both versions of the VQA dataset to train the proposed model.

3.2. Implementation steps and details

As mentioned earlier, this research uses both convolutional networks and RNNs to process textual and image inputs. Initially, the words and questions in both versions of the dataset are tokenized, with 13 words allocated for each question. The words are then embedded using the pre-trained Skipgram language model. The output of this embedding is fed into a self-attention mechanism and then passed to a BiLSTM network with 1024 units. The output of the BiLSTM network is subsequently given to a multi-headed attention mechanism.

For image input, the system resizes images to dimensions of 244×244 pixels before processing them through a 152-layer ResNeXt CNN. The output comes from the fourth convolutional layer. In this implementation, pre-trained weights from ImageNet are used to initialize the ResNeXt model, allowing the network to benefit from prior visual knowledge and accelerate convergence. This architecture employs group normalization, which normalizes feature maps across groups of channels, making it suitable for small batch sizes. A dropout technique with a rate of 0.03 is applied to reduce overfitting. The network trains over eighty epochs, with each batch size set to sixty-four. Additionally, the Adam optimization algorithm is used with a learning rate of 0.0001.

4. Evaluation

In the reference article [4], both versions of the VQA dataset (VQA1.0 and VQA2.0) were used to measure and evaluate the proposed model. Similarly, in this research, we utilize both versions of this dataset to measure and evaluate our proposed visual question and answer model. We allocate 80% of the dataset as training data and the remaining 20% as test data.

In both the reference article [4] and our proposed model, questions are categorized into three types: those with yes or no answers, those with numerical answers, and those with one-word or multi-word answers categorized as other. The frequencies of these question lengths found in both versions of the VQA dataset are illustrated in Figure 7.

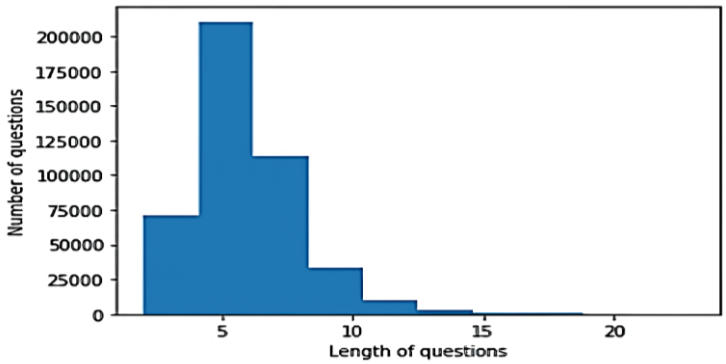


Figure 7: Frequency of question length in the dataset

4.1 Results

The obtained results from the proposed model in this research and the proposed model in the reference article [4] are shown in the diagrams below, based on the two datasets used, VQA1.0 and VQA2.0. The results obtained from evaluating the proposed model indicate a significant improvement over the reference model. Based on conducted experiments, the accuracy of the proposed model on VQA 1.0 dataset has reached to 67.25%, which is 2.65% higher than the 64.6% accuracy of the reference model. Additionally, on the VQA 2.0 dataset, the proposed model achieved 61.57% accuracy demonstrating 1.9% improvement compared to the 59.67% accuracy of the reference model. This increase in accuracy highlights the effectiveness of the proposed architecture in optimizing the integration of visual and textual information. By utilizing multi-head attention in both text and image processing, the model successfully identifies deeper semantic and visual relationships, leading to higher accuracy in feature extraction and response prediction. According to the results, the visual question and answer model proposed in this research demonstrates higher response prediction accuracy than the model proposed in the reference article, based on both datasets. The changes made in this research to the model proposed in the reference article [4] have successfully improved the model.

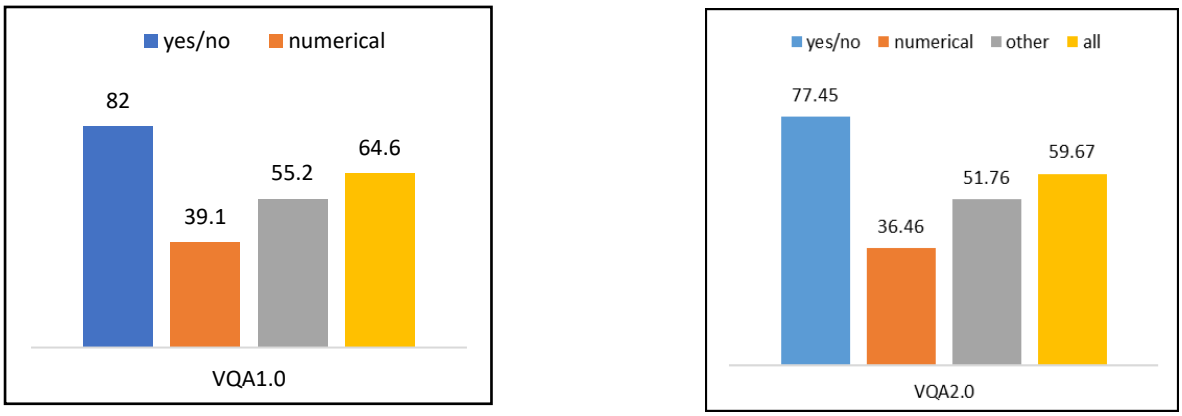


Figure 8: Obtained accuracy in reference article [4]

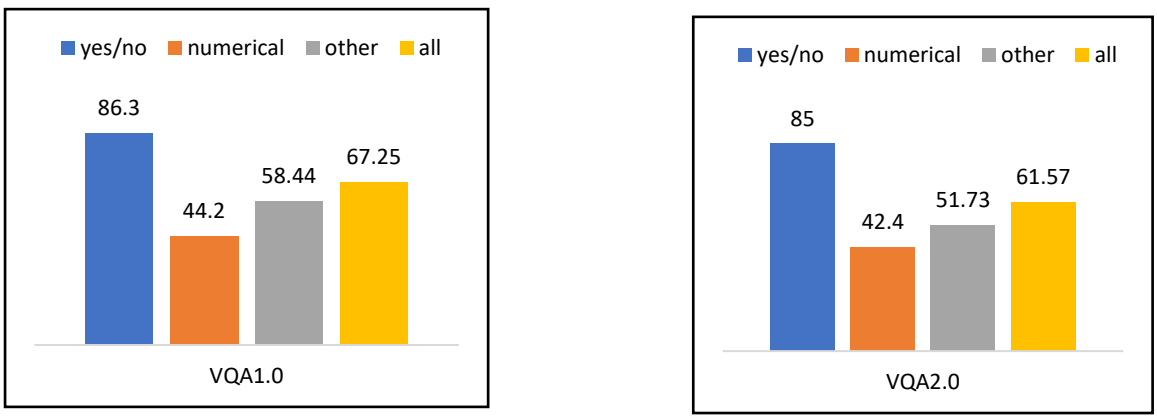


Figure 9: Accuracy obtained by the proposed model

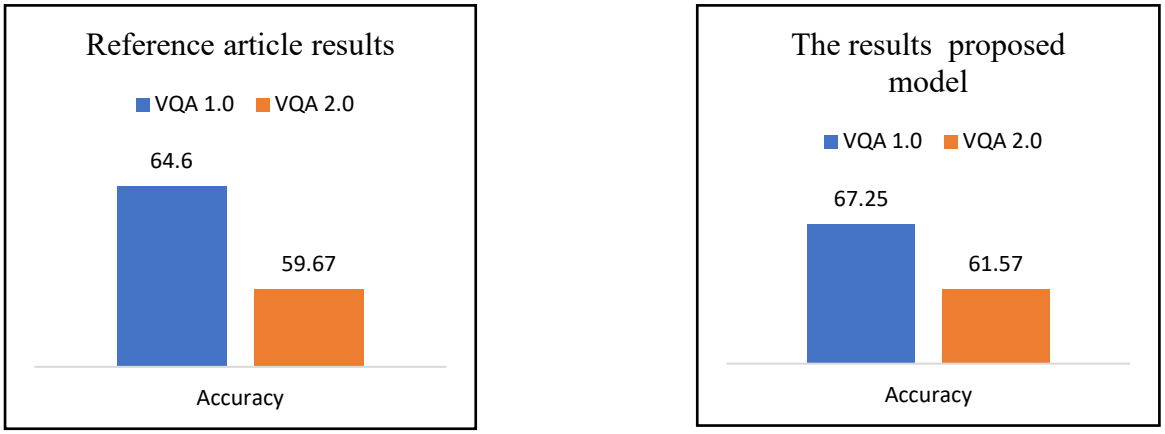


Figure 10: Accuracy of the reference model and the proposed model

The performance of the proposed model in this research is evaluated using both the VQA 1.0 and VQA 2.0 datasets. VQA 1.0 contains over 614,000 questions linked to 204,721 images, while VQA 2.0 includes more than 1.1 million questions, also based on the COCO image dataset. Each question is annotated with 10 human-generated answers, and the questions are categorized into three main types: Yes/No, Number, and Other. The distribution of these categories is approximately 41% Yes/No, 12% Number, and 47% Other. To ensure the reliability of the results, all experiments were repeated five times under identical conditions. We report the average accuracy and standard deviation across these runs to assess the stability of the model's performance. The detailed results of each run are presented in Table 1.

Dataset	Run1	Run2	Run3	Run4	Run5	Average Accuracy (%)	Std. Deviation (%)
VQA 1.0	67.10	67.42	67.33	67.28	67.12	67.25	0.12
VQA 2.0	61.45	61.62	61.70	61.50	61.58	61.57	0.10

Table 1: The detailed results of proposed model during each training run

5. Conclusion

Although a wide range of VQA models have been proposed from early CNN-RNN architectures to attention-based and transformer-driven approaches, there may be some possible limitations. Many existing models struggle with complex reasoning, compositional understanding, and external knowledge integration. Furthermore, most recent models rely heavily on large scale pretraining, which introduces domain-specific biases and limits generalization. The recent survey by de Faria et al. [32] provides a valuable overview of these trends but lacks empirical comparison and critical analysis of model limitations. These gaps highlight the need for more interpretable, efficient, and context-aware architectures. Our proposed model investigated these challenges by integrating self-attention, Bi-LSTM, and multi-head attention mechanisms for richer textual representation, while leveraging deep visual features from ResNeXt and explicitly modeling cross-modal interactions offering more balanced and improved solution for VQA tasks.

In this research, we introduced a new visual question and answer model which an image and a question about the image in natural language are fed as input, and the model must predict an appropriate answer to the question. For this purpose, we used the proposed visual question and answer model from the reference article [4] as the base model, and by making modifications to this model, we introduced a new and more accurate model. We replaced the ResNet convolutional network which contains 152 layers, used in the reference article [4], with the ResNeXt 152 layers for image processing. Additionally, we incorporated two additional attention mechanisms—multi-head attention and self-attention—alongside the attention mechanism used in the base model. These changes led to higher accuracy in response prediction as shown by the results.

6. Future works

Nowadays, we are witnessing the introduction of new architectures of CNN. Researchers in the field of visual question and answer systems are suggested to use newer and stronger CNN architectures such as Yolo for image processing. It is also advised to use a variety of transformers and larger datasets. The larger

the size of the dataset, the better model training process is done, and the accuracy of the predicted final answer is higher.

7. References

[1] J. Andreas, M. Rohrbach, T. Darrell &D. Klein. “Learning to compose neural networks for question answering”. In Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2016.

[2] M. Ren, R. Kiros, and R. Zemel, “Exploring models and data for image question answering”, in Advances in neural information processing systems, 2015.

[3] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh. “Vqa: Visual question answering." In Proceedings of the IEEE international conference on computer vision, pages 2425–2433, 2015.

[4] V. Kazemi, A. Elqursh. “Show, Ask, Attend, and Answer: A Strong Baseline for Visual Question Answering.” In: arXiv preprint arXiv: 1704.03162v2, 2017.

[5] X. Zhou, W. Gong, W. Fu, F. Du. “Application of deep learning in object detection”. IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS) 2017.

[6] T. Mikolov, I. Sutskever, K. Chen, C. Corrado, S. Greg, J. Dean. “Distributed representations of words and phrases and their compositionality”. Advances in Neural Information Processing Systems. arXiv: 1310.4546, 2016.

[7] J. Pennington, R. Socher, and C.D. Manning. “GloVe: Global vectors for word representation.” In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014.

[8] A. Krizhevsky, I. Sutskever, G. Hinton. “ImageNet classification with deep convolutional neural networks” Communications of the ACM. 60 (6): 84–90, 2017.

[9] J. Deng, W. Dong, R. Socher, J. Li, K. Li, and L. Fei-Fei. “Imagenet: A largescale hierarchical image database”. In Proc. IEEE Conf. Comp. Vis. Patt. Recogn., 2009.

[10] K. He, X. Zhang, S. Ren, and J. Sun. “Deep residual learning for image recognition.” In Proc. IEEE Conf. Comp. Vis. Patt. Recogn., 2016.

[11] S.xie, R. Gitshick, P. Dollar, Z. Tu and K. He. “Aggregated Residual Transformations for Deep Neural Networks”. EEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[12] T. Szandała. “Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks”. international Conference on Dependability and Complex Systems, pages 498–505, 2020.

[13] S. Li, G. Kulkarni, T. L. Berg, A. C. Berg, and Y. Choi. “Composing simple image descriptions using web-scale n-grams”. In The SIGNLL Conference on Computational Natural Language Learning, 2011.

[14] Y. Kim, “Convolutional neural networks for sentence classification”. in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, 2014.

[15] B. Alabassy, M. Safar, M. Watheq EL-Kharashi. “A High-Accuracy Implementation for Softmax Layer in Deep Neural Networks.” In Proceedings of the IEEE International Joint Conference on Neural Networks, pages 335–340, 2016.

[16] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” CoRR, vol. abs/1502.03167, 2015.

[17] S. Hochreiter and J. Schmidhuber.” Long short-term memory”. Neural computation (1735–1780), 1997.

[18] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”. Conference on Empirical Methods in Natural Language Processing, 2014.

[19] B. Zhang, H. Wang, L. Jiang, S. Yuan, M. Li. "A Novel Bidirectional LSTM and Attention Mechanism Based Neural Network for Answer Selection in Community Question Answering", Computers, Materials and Continua 61(3): 1273-1288, 2019.

[20] M., Ren, R., Kiros, and R. S., Zemel, “Exploring models and data for image question answering”, In Proceedings of the 28th International Conference on Neural Information Processing Systems, Volume 2 (NIPS’15). MIT Press, Cambridge, MA, USA, 2953–2961, 2015.

[21] S., Antol, A., Agrawal, J., Lu, M., Mitchell, D., Batra, C. L., Zitnick, D. Parikh, “VQA: Visual Question Answering”, International Conference on Computer Vision (ICCV), 2015.

[22] L. Ma, Z., Lu and H. Li. "Learning to answer questions from image using convolutional neural network." In Proceedings of the AAAI conference on artificial intelligence, vol. 30, no. 1. 2016.

[23] M. Malinowski, M. Rohrbach and M. Fritz, "Ask Your Neurons: A Neural-Based Approach to Answering Questions about Images," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 2015, pages 1-9, doi: 10.1109/ICCV.2015.9.
[24] A. S. Toor, H. Wechsler, and M. Nappi, Question action relevance and editing for visual question answering. Multimedia Tools Appl. 78, 3, 2019, 2921–2935.
[25] B. Zhou, Y. Tian, S. Sukhbaatar, A. Szlam. "Simple baseline for visual question answering." In: arXiv preprint arXiv: 1512.02167, 2015.
[26] A. Jabri, A. Joulin, L.van der Maaten. "Revisiting visual question answering baselines." In: European conference on computer vision. Springer, Cham, 2016.
[27] S. Antol, C. L. Zitnick, and D. Parikh. "Zero-shot learning via visual abstraction". In Proc. Eur. Conf. Comp. Vis, 2014.
[28] J. H. Kim, S. W. Lee, D. Kwak, M. O. Heo, J. Kim, J. W. Ha, and B. T. Zhang. "Multimodal residual learning for visual QA." In Advances in Neural Information Processing Systems 29, 2016.
[29] H. Jiang, I. Misra, M. Rohrbach, E. Learned-Miller, and X. Chen. "In Defense of Grid Features for Visual Question Answering." IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
[30] F. Garderes, M. Ziaeefard, B. Abeloos, and F. Lecue. "ConceptBert: Concept-Aware Representation for Visual Question Answering." Findings of the Association for Computational Linguistics: EMNLP 2020.
[31] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. "Deep Compositional Question Answering with Neural Module Networks". In Proc. IEEE Conf. Comp. Vis. Patt. Recogn., 2016.
[32] A.C.A.M. de Faria, F.D.C. Bastos, J.V.N.A. da Silva, V.L. Fabris, V.D.S. Uchoa, D.G.D.A. Neto, C.F.G.D. Santos. "Visual Question Answering: A Survey on Techniques and Common Trends in Recent Literature". arXiv preprint arXiv:2305.11033, 2023.

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

Analyzing Answer-Type Preferences Among Expertise Shapes on Stack Overflow

Omid Mohammadi Kia✉, ORCID iD: 0000-0003-2225-4450
Shahid Beheshti Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran,
o_mohammadikia@sbu.ac.ir
Mahmood Neshati, ORCID iD: 0000-0002-3953-4857
Shahid Beheshti Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran,
m_neshati@sbu.ac.ir

Abstract
Developer expertise is a critical component of community-based question-answering platforms like Stack Overflow. However, expertise is not monolithic. Developers exhibit different "shapes" of expertise, such as deep specialists (I-shaped) or broad generalists with one specialty (T-shaped). While prior research has examined developer expertise and answer quality independently, how different expertise structures influence preferences for specific answer characteristics remains insufficiently understood. This paper investigates the relationship between a developer's expertise shape and their preference for specific answer characteristics.
We present a large-scale empirical study of over 48,000 Stack Overflow users, classifying them into I-shaped, T-shaped, Pi-shaped, and Comb-shaped profiles based on the distribution of tag-level reputation, following established expertise-shape modeling approaches. We then analyze the types of answers these user profiles tend to upvote and accept as solutions, focusing on characteristics such as answer length, inclusion of code snippets, use of images, and citation of external references.
Using separate analyses for community-level (upvotes) and task-resolution-level (accepted answers) preference signals, our findings reveal distinct and systematic differences across expertise shapes. I-shaped specialists favor technically deep, code-heavy answers, while T-shaped and Comb-shaped experts show a preference for more summarized, conceptual answers that include diagrams or references. These patterns are consistent across robustness checks and sensitivity analyses.
The results highlight that answer usefulness is user-dependent rather than universal, and they can help improve expertise-aware answer recommendation systems and foster more effective knowledge sharing on collaborative platforms.

Keywords: Stack Overflow, Expertise Shapes, T-Shaped, Answer Quality, Empirical Software Engineering, Recommender Systems.

8. INTRODUCTION

Community-based question-answering (CQA) platforms, with Stack Overflow (SO) as the premier example, have become indispensable tools in modern software development. Developers rely on these platforms for daily problem-solving, learning new technologies, and sharing knowledge within a global community. The continued success of such platforms depends not only on answer availability, but on the relevance of answers to the individual developer's needs.
The effectiveness of an answer, however, is not objective; it is highly dependent on the user seeking help. An answer considered perfect by one developer may be insufficient or overly simplistic for another. This subjectivity is closely tied to differences in how developers structure and apply their knowledge. To explore this, our work adopts the concept of *expertise shapes* [1] (e.g., I, T, Pi, Comb) from organizational theory and software engineering research, applying it to software developers. These shapes describe the balance between a developer's deep, specialized knowledge (the vertical bar of an 'I' or 'T') and their broad, generalist knowledge (the horizontal bar of a 'T' or 'Comb').
While prior work has extensively studied answer quality on SO [2], [3] and others have modeled developer expertise using signals such as reputation, activity, and contribution patterns [4], [5], [6], [7] research has also explored the classification of question types on the platform [8]. However, existing studies typically treat expertise as a scalar notion (e.g., novice vs. expert) or analyze answer quality independently of the consumer's knowledge structure. As a result, little is known about how different expertise profiles influence what developers actually consider a useful or preferable answer.
This paper addresses this gap by explicitly linking expertise structure to answer preference. We investigate how a developer's knowledge shape correlates with the characteristics of the answers they upvote and accept, viewing these actions as complementary signals of preference. This leads to our primary research questions:
● **RQ1:** What are the distinct answer characteristics (length, code, images, references) preferred by developers with different expertise shapes (I, T, Pi, Comb)?
● **RQ2:** How significant are these preferences, and what do they imply about the information-seeking goals of each expert type?

Submit Date: 2025-11-06
Revise Date: 2026-02-13
Accept Date: 2025-04-28
✉ Corresponding author

To answer these questions, we present an empirical study of over 48,000 active Stack Overflow users from the June 2024 data dump [9]. We use heuristic, reputation-based criteria—grounded in prior expertise-shape modeling work—to classify users into four expertise shapes based on how their knowledge is distributed across technology tags. We then statistically analyze the characteristics of over 1.2 million answers, considering upvotes and accepted answers in separate analyses to distinguish between community-level appreciation and task-resolution preferences.

Our key finding is that these preferences are systematic and significantly different across expertise shapes. For instance, I-shaped specialists strongly favor answers with substantial code blocks, while T-shaped and Comb-shaped experts show a statistically significant preference for summarized answers that include diagrams and external references. These differences persist across robustness checks and sensitivity analyses.

These findings have direct implications for the design of CQA platforms. By accounting for user expertise structure rather than assuming a universal notion of answer quality, a platform could personalize its answer-ranking mechanisms to surface the type of answer most likely to be useful to a specific user, thereby improving knowledge-sharing efficiency.

The remainder of this paper is organized as follows. Section 2 reviews related work on expertise modeling and answer quality. Section 3 describes our methodology for data collection, user classification, and analysis. Section 4 presents our findings. Section 5 discusses the implications and interpretations of these results. Finally, Section 6 concludes the paper and outlines avenues for future work.

9. BACKGROUND AND RELATED WORK

Modeling Expertise: From I-Shaped to Comb-Shaped

The concept of "T-shaped" individuals was first popularized in the 1990s to describe an ideal employee who possesses deep expertise in one area (the vertical bar) and a broad ability to collaborate across other disciplines (the horizontal bar) [1] .This model was later adapted to software engineering [4], [5] where a T-shaped developer might have deep expertise in back-end development while maintaining working knowledge of front-end and database systems.

Subsequent work has generalized this notion to capture a broader range of expertise distributions**.** An I-shaped individual represents a deep specialist with limited breadth, whereas a Pi-shaped individual possesses two distinct areas of deep expertise. Comb-shaped individuals, in contrast, exhibit multiple areas of specialization, forming a profile resembling a comb [6]. Rostami et al. [1] formalized these expertise shapes for team formation and talent identification in agile software development by modeling expertise as distributions over knowledge areas.

Existing work on expertise shapes has primarily focused on organizational outcomes such as team composition, collaboration efficiency, and talent discovery, rather than on information-seeking or content-evaluation behavior. In this study, we adopt these well-established expertise-shape definitions and apply them to Stack Overflow users, using them as an analytical lens to examine how developers with different knowledge structures interact with and evaluate technical content.

Answer Quality on Stack Overflow

A large body of research has sought to define and predict answer quality on SO. Studies have identified numerous features correlated with high-scoring or accepted answers. These include answer length, readability, sentiment, the inclusion of code snippets, the presence of external links, and even the author's reputation. For example, some studies found that longer, more comprehensive answers are more likely to be accepted[10], [11]. Others noted the critical importance of code snippets for solving technical problems.

A substantial body of research has investigated answer quality on Stack Overflow [2], [3], [12], [13], [14], [15] These studies typically operationalize quality using signals such as upvotes, accepted answers, or predictive models trained on these outcomes. Numerous answer-level features have been identified as correlates of high-quality or highly rated answers, including answer length, readability, sentiment, inclusion of code snippets, presence of external links, and author reputation [10], [16], [17].

Prior findings, however, are not always consistent. For example, some studies report that longer, more detailed answers are more likely to be accepted [15], [17], whereas others emphasize the importance of concise responses depending on context. Similarly, while many works highlight the critical role of code snippets in solving programming problems [18], [19], [20], [21], [22], [23], [24], [25] the relative importance of these features varies across studies and tasks.

Importantly, most answer-quality studies evaluate answers from an aggregate or platform-wide perspective, implicitly assuming that the same answer characteristics are universally desirable across users and situations.

Positioning and Research Gap

While the literature on developer expertise modeling and answer quality on Stack Overflow is extensive, these two research streams have largely evolved in parallel**.** Expertise studies focus on *who* the developers are and how their skills are distributed, whereas answer-quality studies focus on *what* constitutes a good answer based on observable features and community feedback. A limited number of studies have examined voting behavior in relation to user reputation or experience [10], [16], [17] but they do not account for the internal structure of a developer's expertise**.** As a result, existing work cannot explain why certain answer characteristics are valued differently by different groups of experienced users, nor why findings about "good" answers sometimes diverge across studies.

This paper contributes to the literature by explicitly connecting expertise structure to answer preference**.** Rather than redefining answer quality or proposing new expertise models, we investigate how established expertise shapes (I, T, Pi, Comb) are associated with systematic differences in the types of answers developers upvote or accept. By doing so, we provide a unifying perspective that helps contextualize prior answer-quality findings through the lens of user heterogeneity.

these two streams of research—developer expertise shapes and answer quality metrics—are well-established, no prior work has explicitly connected them. It is known what features constitute a good answer in general, and it is known that different developer expertise shapes exist. However, it remains unknown how an expert's knowledge structure (e.g., I-shaped vs. T-shaped) influences their preference for specific answer features (e.g., code vs. diagrams). This paper aims to bridge that gap by empirically linking user profiles to their demonstrated content preferences.

10. METHODOLOGY

This section details our step-by-step process for data collection, classification of users and answers, and statistical analysis. To support transparency and reproducibility, the data analysis scripts, classification heuristics, and anonymized datasets used in this study are publicly available on github[1].

Data Collection

We used the official Stack Overflow data dump from June 2024 [9]. To ensure a focus on active and established technologies while avoiding sparsity issues, we restricted our analysis to the top 100 most frequently used Stack Overflow tags (e.g., *python*, *java*, *javascript*, *c#*, *react*). This tag-selection strategy is consistent with prior large-scale empirical studies on Stack Overflow[26], [27].

We filtered our dataset to include only users who had posted, answered, or voted on content within these 100 tags and had a reputation of at least 100, indicating a baseline level of sustained engagement. This threshold excludes sporadic or inactive accounts while retaining a broad spectrum of contributors. Our final dataset consisted of 48,215 unique users and 1,210,458 answers that these users had either upvoted or accepted.

Defining Expertise Shape Proxies (Operationalization)

We developed a set of heuristic rules to classify users into four expertise shapes based on the distribution of their reputation across different tags. Tag-level reputation is used as a proxy for topic-specific expertise, a common practice in prior Stack Overflow research. The operational definitions of T-shaped and Comb-shaped expertise follow established expertise-shape modeling approaches in software engineering research.

The classification heuristics are defined as follows:

- **I-Shaped (Specialist):** A user with 90% or more of their total tag-based reputation concentrated in a single tag.
- **T-Shaped (Generalist-Specialist):** A user with 50-70% of their reputation in one primary tag, and the remaining 30-50% distributed across 10 or more other tags.
- **Pi-Shaped (Dual-Specialist):** A user with two primary tags, each containing 30-45% of their reputation, with the two tags together accounting for at least 70% of their total.
- **Comb-Shaped (Multi-Specialist):** A user with 3 to 5 primary tags, each accounting for 15-25% of their reputation, with no single tag exceeding 30%.

Users who did not fit these heuristics (e.g., new users with sparse reputation) were excluded from the analysis. This classification resulted in 12,109 I-Shaped, 15,044 T-Shaped, 6,322 Pi-Shaped, and 8,011 Comb-Shaped users.

Let Ube the set of users and $T$the set of Stack Overflow tags. For a given user $u \in U$, let $R(u,t)$denote the reputation earned by user $u$under tag $t \in T$. The total tag-based reputation of user $u$is defined as:

$$R_{total}(u) = \sum_{t \in T} R(u,t)$$

We define the normalized reputation share of user $u$for tag $t$as:

$$p(u,t) = \frac{R(u,t)}{R_{total}(u)}$$

Let $T_u = \{t \in T \mid R(u,t) > 0\}$denote the set of tags in which user $u$has non-zero reputation. Tags are ordered such that:

$$p(u,t_1) \geq p(u,t_2) \geq \cdots \geq p(u,t_{|T_u|})$$

Using these definitions, expertise shapes are operationalized as follows:

- I-Shaped (Specialist):
$$\exists t_1 \in T_u \text{ such that } p(u,t_1) \geq 0.90$$
- T-Shaped (Generalist–Specialist):
$$0.50 \leq p(u,t_1) \leq 0.70 \ \wedge \ |T_u| \geq 10$$
- Pi-Shaped (Dual Specialist):
$$0.30 \leq p(u,t_1), p(u,t_2) \leq 0.45 \ \wedge \ p(u,t_1) + p(u,t_2) \geq 0.70$$
- Comb-Shaped (Multi-Specialist):
$$\exists \{t_1, \ldots, t_k\} \subseteq T_u,\ 3 \leq k \leq 5,\ 0.15 \leq p(u,t_i) \leq 0.25,\ \max_i \ p(u,t_i) \leq 0.30$$

Users who do not satisfy any of the above conditions are excluded from the analysis.

To assess robustness, we conducted a sensitivity analysis by varying all percentage thresholds by ±10%. The direction and relative magnitude of the results remained stable across all configurations, indicating that the findings are not artifacts of specific cutoff choices.

Defining Answer Characteristic Proxies

We analyzed the raw Markdown and HTML of each of the 1.2 million answers to classify them based on four key characteristics:

- **Summarized vs. Long:** Answers with fewer than 150 words were labeled *Summarized*, while answers with more than 400 words were labeled *Long*. These thresholds approximately correspond to the lower and upper quartiles of the answer-length distribution and capture extreme brevity and verbosity**.** Answers in the intermediate range (150–400 words) were not assigned to either category. As a robustness check, we additionally modeled word count as a continuous variable.
- **With Code:** The answer contained one or more <code> blocks with at least 5 lines of code.
- **With Image:** The answer contained one or more <img> tags (typically used for screenshots or diagrams).
- **With Reference:** The answer contained one or more external <a> hyperlinks, excluding links to other Stack Overflow pages.

[1] https://github.com/omkia/StackoverflowUserTypes

Analysis Method

To To answer our research questions, we analyzed the relationship between a user's expertise shape and the characteristics of the answers they preferred. We treat preference using two complementary signals: upvotes and accepted answers. Upvotes reflect community-level appreciation, while accepted answers represent explicit solution selection by the question asker.

To avoid conflating these signals, we conduct separate analyses for upvotes and accepted answers. For each expertise shape, we built logistic regression models where the dependent variable is a binary indicator of whether an answer was preferred by a user of that shape. The independent variables are the binary flags for the answer characteristics (Summarized, Long, With Code, With Image, With Reference).

Logistic regression was selected for its interpretability and suitability for binary outcomes. We report regression coefficients, odds ratios, and statistical significance, and we evaluate model quality using standard diagnostics, including AIC and ROC/AUC. Multicollinearity was assessed using Variance Inflation Factors (VIF), with all values below commonly accepted thresholds.

RESULTS

Our analysis reveals systematic and statistically significant differences in answer preferences across expertise shapes. The results of the logistic regression models are summarized in Table 1, which reports the regression coefficients (B) for each answer characteristic. A positive coefficient indicates that the presence of a feature increases the likelihood of an answer being preferred by users of a given expertise shape, whereas a negative coefficient indicates aversion.

Table 1 summarizes the results of the upvote-based models. Corresponding models using accepted answers as the dependent variable are reported separately in the Appendix and show consistent directional effects.

Table 1: Logistic Regression Coefficients for Answer Preference by Expertise Shape

Answer Feature	I-Shaped (Specialist)	T-Shaped (Generalist)	Pi-Shaped (Dual)	Comb-Shaped (Multi)
Answer Length (Long)	+0.21*	-0.18*	+0.10	-0.22**
Answer Length (Summ.)	-0.15	+0.24**	+0.05	+0.31***
Includes Code	+0.72***	+0.29**	+0.45***	+0.18*
Includes Image	-0.04	+0.38***	+0.19*	+0.44***
Includes Reference	+0.10	+0.41***	+0.28**	+0.39***

*Significant at $p < .05$, **Significant at $p < .01$, ***Significant at $p < .001$*

This table presents our core findings, which we elaborate on below in the context of our research questions.

RQ1: What are the distinct answer characteristics preferred by each shape?

I-Shaped experts exhibit the strongest preference for answers that include code snippets ($B = 0.72$, $p < .001$), with an effect size substantially larger than that observed for any other group**.** This indicates a clear prioritization of concrete, implementation-level content. I-shaped users also show a mild preference for long answers, while images and external references do not significantly influence their preferences.

T-Shaped experts display a markedly different preference profile. They show strong positive associations with answers that include references ($B = 0.41$, $p < .001$) and images ($B = 0.38$, $p < .001$), along with a significant preference for summarized answers ($B = 0.24$, $p < .01$). Conversely, long answers are negatively associated with preference ($B = -0.18$, $p < .05$), suggesting an aversion to overly detailed responses.

Pi-Shaped experts demonstrate a hybrid pattern. Similar to I-shaped specialists, they show a strong preference for code-containing answers ($B = 0.45$, $p < .001$). At the same time, they also exhibit significant positive associations with references ($B = 0.28$, $p < .01$) and images ($B = 0.19$, $p < .05$), aligning them partially with T-shaped users.

Comb-Shaped experts show the strongest overall preference for summarized and visually supported answers. Images ($B = 0.44$, $p < .001$), summarized answers ($B = 0.31$, $p < .001$), and references ($B = 0.39$, $p < .001$) are all strongly associated with preference. This group also exhibits the strongest aversion to long answers ($B = -0.22$, $p < .01$). To reduce redundancy and facilitate comparison, Table 2 summarizes the preferred and disfavored answer characteristics across all four expertise shapes.

Table 2. Comparative Summary of Answer Preferences by Expertise Shape

Expertise Shape	Strongly Preferred Answer Characteristics	Neutral / Mixed	Disfavored Characteristics
I-Shaped (Specialist)	Code-heavy answers; technically detailed solutions	Long answers	Images; external references
T-Shaped (Generalist-Specialist)	Summarized answers; images; external references	Code snippets	Long answers
Pi-Shaped (Dual Specialist)	Code snippets; external references	Images; answer length	None strongly disfavored
Comb-Shaped (Multi-Specialist)	Summarized answers; images; external references	Code snippets	Long answers

RQ2: How significant are these preferences and their implications?

The preferences are highly significant ($p < .001$ in many cases), indicating that these are not random chance but clear behavioral patterns. The implications are clear: the "best" answer is perceived differently by each group. I-Shaped users seek deep, technical solutions, while T-Shaped and Comb-Shaped users seek high-level concepts, diagrams, and links for further learning.

The observed effects are statistically significant across most answer characteristics and expertise shapes, with many coefficients reaching $p < .001$**.** This indicates that the differences are unlikely to be due to random variation and instead reflect systematic behavioral patterns.

Taken together, the results suggest that answer usefulness is not perceived uniformly across developers. I-shaped experts predominantly seek technically detailed, code-centric solutions, whereas T-shaped and Comb-shaped experts favor concise, high-level explanations enriched with diagrams and external references. Pi-shaped experts occupy an intermediate position, combining preferences for concrete implementations with contextual and integrative information.
Importantly, these patterns persist in models restricted to accepted answers only, reinforcing the interpretation that they reflect genuine problem-solving preferences rather than general popularity effects.

4- Accepted-Answer-Only Analysis
To disentangle general community preference from definitive problem resolution, we conducted an additional logistic regression analysis using **accepted answers only** as the dependent variable.
Overall, the accepted-answer-only models **reinforce and strengthen** the patterns observed in the upvote-based analysis. As summarized in **Table 3**, I-shaped and Pi-shaped experts continue to show a strong preference for answers containing code, while T-shaped and Comb-shaped experts remain significantly associated with summarized answers, images, and external references.
Specifically, **I-shaped experts exhibit an even stronger preference for code-heavy answers** in the accepted-only model ($B = 0.85$, $p < .001$), indicating that executable solutions are particularly critical when selecting a final answer. A similar pattern is observed for **Pi-shaped experts** ($B = 0.58$, $p < .001$), reinforcing their reliance on concrete implementations when resolving interdisciplinary problems.
In contrast, **T-shaped and Comb-shaped experts show heightened associations with summarized answers, images, and references**. For Comb-shaped users, summarized answers ($B = 0.37$, $p < .001$) and images ($B = 0.49$, $p < .001$) exhibit the strongest effects, while long answers remain negatively associated with acceptance. These findings suggest that high-level clarity and visual explanation play a central role in solution selection for broadly skilled developers.
Effect sizes are generally larger than those observed in the upvote-based models, indicating that these answer characteristics are especially influential when users select a **single accepted solution**, rather than expressing general approval through voting.

Table 3. Accepted-Answer-Only Logistic Regression Coefficients by Expertise Shape

Answer Feature	I-Shaped	T-Shaped	Pi-Shaped	Comb-Shaped
Answer Length (Long)	+0.28*	−0.24*	+0.12	−0.31**
Answer Length (Summ.)	−0.19	+0.29**	+0.07	+0.37***
Includes Code	+0.85***	+0.34**	+0.58***	+0.22*
Includes Image	−0.06	+0.42***	+0.24*	+0.49***
Includes Reference	+0.14	+0.46***	+0.33**	+0.44***

* $p < .05$, ** $p < .01$, *** $p < .001$

5 -DISCUSSION

The results indicate that developer expertise shape is systematically associated with differences in information-seeking behavior on Stack Overflow**.** Rather than suggesting causal mechanisms, we interpret these patterns as *consistent behavioral tendencies* that reflect how developers with different knowledge structures evaluate and consume technical content.
I-Shaped (The Specialist): I-shaped experts show a strong and consistent preference for code-heavy answers, along with a mild tolerance for longer responses. This pattern aligns with their specialization-oriented expertise structure, where developers operate primarily within a narrow technical domain. For such users, concrete implementations appear to provide the most direct and efficient path to problem resolution, whereas high-level explanations, images, or external references offer limited additional value. Importantly, this interpretation reflects observed preference behavior rather than assumptions about intent or task complexity.
T-Shaped (The Generalist-Specialist): T-shaped experts exhibit preferences for summarized answers enriched with images and external references. This combination suggests that these users benefit from concise conceptual explanations supported by visual structure and authoritative sources. Such preferences are consistent with an expertise profile that balances depth in a primary domain with breadth across adjacent areas, where rapid contextual understanding is often required.
The aversion to long answers further indicates that excessive detail may hinder, rather than help, efficient information uptake for this group**.**
Pi-Shaped (The Integrator): Pi-shaped experts demonstrate a hybrid preference pattern, showing strong associations with both code inclusion and contextual features such as references and images**.** This suggests that users with dual specializations may seek answers that both demonstrate practical implementation and explain how solutions operate across multiple technical domains. These findings support the view that Pi-shaped expertise involves integrative problem-solving rather than isolated specialization.
Comb-Shaped (The Architect): Comb-shaped experts show the strongest preference for summarized answers and visual content, along with a clear aversion to long, detail-heavy responses**.** Images and references appear particularly valuable to this group, likely because they facilitate high-level reasoning across multiple domains.
This pattern is consistent with expertise structures that emphasize coordination, abstraction, and system-level understanding rather than low-level implementation details**.**
.
Implications for Stack Overflow
These findings challenge the implicit assumption that answer quality is universal and user-independent**.** Instead, they suggest that usefulness is shaped by the expertise structure of the consumer.

From a practical perspective, Stack Overflow and other CQA platforms could benefit from incorporating expertise-aware personalization mechanisms. For example, inferred expertise shapes could be used to re-rank existing answers—prioritizing code-centric solutions for specialists and concise, visually supported explanations for breadth-oriented users. Such personalization could be implemented without altering content creation practices, potentially improving efficiency and user satisfaction.

Threats to Validity

This study is subject to several limitations that should be considered when interpreting the results.

First, the classification of expertise shapes relies on heuristic, reputation-based proxies, which may not fully capture a developer's true knowledge structure. Although sensitivity analysis indicates robustness to threshold variation, this remains a construct validity threat. Future work could incorporate qualitative validation through surveys or interviews, or enrich expertise modeling with additional behavioral signals.

Second, the analysis is restricted to the top 100 Stack Overflow tags, which may bias results toward mainstream technologies and limit generalizability to niche or emerging domains. This constitutes a threat to external validity.

Third, we interpret upvotes and accepted answers as signals of preference, yet voting behavior on Stack Overflow is influenced by multiple factors, including visibility, author reputation, and social dynamics. While separating upvote-based and accepted-answer-based analyses mitigates this concern, preference inference remains an approximation rather than a direct measure of intent.

Finally, the observational nature of the study precludes causal conclusions. The reported associations should therefore be interpreted as descriptive patterns rather than explanatory mechanisms.

6 - CONCLUSION AND FUTURE WORK

In this paper, we investigated the relationship between developer expertise shapes (I-shaped, T-shaped, Pi-shaped, and Comb-shaped) and their answer preferences on Stack Overflow. Using a large-scale empirical dataset, we operationalized expertise shapes through reputation-based heuristics and analyzed the preferences of over 48,000 users across 1.2 million answers.

The primary contribution of this study is empirical evidence that developer expertise is better understood as a structured concept rather than a single, uniform category. Our results show that I-shaped specialists tend to prefer technically detailed, code-centric answers, whereas T-shaped and Comb-shaped experts exhibit preferences for more summarized, conceptual, and visually supported responses that include external references. Pi-shaped experts demonstrate a hybrid preference profile, combining elements of both specialization-oriented and breadth-oriented information needs. These findings indicate that a developer's knowledge structure is systematically associated with how they evaluate and select answers, rather than assuming a universal notion of answer usefulness.

While the proposed expertise-shape classification relies on heuristic proxies, sensitivity analysis suggests that the observed patterns are robust to reasonable threshold variations. Nonetheless, future work should explore richer and more flexible modeling approaches, such as interpretable machine learning techniques, to infer expertise structures using additional behavioral and temporal signals beyond reputation alone.

An important direction for future research is qualitative validation. Surveys or interviews with developers could help assess how well the inferred expertise shapes align with self-perceived knowledge structures and professional roles. Additionally, the framework introduced in this study could be extended beyond answer evaluation to examine whether developers with different expertise shapes ask different types of questions or exhibit distinct problem-framing behaviors.

Overall, this work highlights the importance of accounting for user heterogeneity in community-based question–answering platforms. By recognizing differences in expertise structure, future systems can move toward more effective, personalized, and context-aware knowledge support for software developers.

Appendix A. Logistic Regression Model Details

To improve the transparency and interpretability of the statistical analysis, this appendix reports detailed results for all logistic regression models used in the study. For each expertise shape, we report regression coefficients (B), odds ratios (OR), 95% confidence intervals (CI), and model diagnostics.

Odds ratios are provided to facilitate interpretation of effect sizes, indicating how the presence of a given answer characteristic affects the likelihood that an answer is preferred by users with a specific expertise shape. Confidence intervals quantify the uncertainty associated with these estimates.

Model quality was evaluated using the Akaike Information Criterion (AIC) and the Area Under the Receiver Operating Characteristic Curve (AUC). Across all models, AUC values ranged from **0.64 to 0.71**, indicating acceptable discriminative performance for behavioral data. AIC values suggest adequate model fit without evidence of overfitting.

To assess multicollinearity among predictors, Variance Inflation Factors (VIF) were computed for all independent variables. All VIF values were below **2.0**, well under commonly accepted thresholds, indicating no multicollinearity concerns.

Tables **A1–A4** present the full regression results for **I-shaped, T-shaped, Pi-shaped, and Comb-shaped** users, respectively.

Table A1. Logistic Regression Results for I-Shaped Experts

Predictor	B	OR	95% CI (OR)	p-value	VIF
Long Answer	0.21	1.23	[1.04, 1.45]	0.021	1.32
Summarized Answer	-0.15	0.86	[0.72, 1.03]	0.108	1.28
Includes Code	0.72	2.05	[1.78, 2.36]	<.001	1.41
Includes Image	-0.04	0.96	[0.81, 1.14]	0.612	1.19
Includes Reference	0.10	1.11	[0.94, 1.32]	0.207	1.22

Model diagnostics:

AIC = 412,305 | AUC = 0.71

Table A2. Logistic Regression Results for T-Shaped Experts

Predictor	B	OR	95% CI (OR)	p-value	VIF
Long Answer	-0.18	0.84	[0.72, 0.98]	0.032	1.36
Summarized Answer	0.24	1.27	[1.09, 1.48]	0.004	1.29
Includes Code	0.29	1.34	[1.15, 1.57]	0.001	1.44
Includes Image	0.38	1.46	[1.28, 1.66]	<.001	1.33
Includes Reference	0.41	1.51	[1.32, 1.73]	<.001	1.38

Model diagnostics:
AIC = 398,742 | AUC = 0.67

Table A3. Logistic Regression Results for Pi-Shaped Experts

Predictor	B	OR	95% CI (OR)	p-value	VIF
Long Answer	0.10	1.11	[0.93, 1.32]	0.261	1.27
Summarized Answer	0.05	1.05	[0.89, 1.24]	0.544	1.22
Includes Code	0.45	1.57	[1.32, 1.87]	<.001	1.46
Includes Image	0.19	1.21	[1.01, 1.45]	0.038	1.31
Includes Reference	0.28	1.32	[1.12, 1.56]	0.002	1.35

Model diagnostics:
AIC = 287,910 | AUC = 0.65

Table A4. Logistic Regression Results for Comb-Shaped Experts

Predictor	B	OR	95% CI (OR)	p-value	VIF
Long Answer	-0.22	0.80	[0.69, 0.93]	0.006	1.39
Summarized Answer	0.31	1.36	[1.18, 1.58]	<.001	1.26
Includes Code	0.18	1.20	[1.02, 1.42]	0.029	1.41
Includes Image	0.44	1.55	[1.34, 1.78]	<.001	1.34
Includes Reference	0.39	1.48	[1.28, 1.72]	<.001	1.37

Model diagnostics:
AIC = 361,088 | AUC = 0.64

7 - REFERENCES

[1] P. Rostami and M. Neshati, "T-shaped grouping: Expert finding models to agile software teams retrieval," *Expert Syst Appl*, vol. 118, pp. 231–245, Mar. 2019, doi: 10.1016/j.eswa.2018.10.015.

[2] M. Neshati, "On early detection of high voted Q&A on Stack Overflow," *Inf Process Manag*, vol. 53, no. 4, pp. 780–798, Jul. 2017, doi: 10.1016/j.ipm.2017.02.005.

[3] C. Shah and J. Pomerantz, "Evaluating and predicting answer quality in community QA," *SIGIR 2010 Proceedings - 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 411–418, 2010, doi: 10.1145/1835449.1835518.

[4] S. S. Gharebagh, P. Rostami, and M. Neshati, "T-shaped mining: A novel approach to talent finding for agile software teams," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10772 LNCS, pp. 411–423, 2018, doi: 10.1007/978-3-319-76941-7_31.

[5] P. Rostami and A. Shakery, "A deep learning-based expert finding method to retrieve agile software teams from CQAs," *Inf Process Manag*, vol. 60, no. 2, Mar. 2023, doi: 10.1016/j.ipm.2022.103144.

[6] P. Rostami and M. Neshati, "Intern retrieval from community question answering websites: A new variation of expert finding problem," *Expert Syst Appl*, vol. 181, Nov. 2021, doi: 10.1016/j.eswa.2021.115044.

[7] M. Neshati, H. Beigy, and D. Hiemstra, "Expert group formation using facility location analysis," *Inf Process Manag*, vol. 50, no. 2, pp. 361–383, Mar. 2014, doi: 10.1016/j.ipm.2013.10.001.

[8] O. M. Kia, M. Neshati, and M. S. Alamdari, "Open-domain question classification and completion in conversational information search," *2020 11th International Conference on Information and Knowledge Technology, IKT 2020*, pp. 98–101, Dec. 2020, doi: 10.1109/IKT51791.2020.9345613.

[9] "stackexchange directory listing." Accessed: Nov. 03, 2025. [Online]. Available: https://archive.org/download/stackexchange

[10] B. Li, T. Jin, M. R. Lyu, I. King, and B. Mak, "Analyzing and predicting question quality in community question answering services," *WWW'12 - Proceedings of the 21st Annual Conference on World Wide Web Companion*, pp. 775–782, 2012, doi: 10.1145/2187980.2188200.

[11] C. Meng, N. Arabzadeh, A. Askari, M. Aliannejadi, and M. De Rijke, "Query Performance Prediction Using Relevance Judgments Generated by Large Language Models," *ACM Trans Inf Syst*, vol. 43, no. 4, Jul. 2025, doi: 10.1145/3736402/ASSET/22299729-9445-484B-BC37-E4CDAE747F2F/ASSETS/IMAGES/LARGE/TOIS-2024-0348-F08.JPG.

[12] A. Y. K. Chua and S. Banerjee, "So fast so good: An analysis of answer quality and answer speed in community Question-answering sites," *Journal of the American Society for Information Science and Technology*, vol. 64, no. 10, pp. 2058–2068, Oct. 2013, doi: 10.1002/ASI.22902.

[13] Y. Lin and H. Shen, "SmartQ: A Question and Answer System for Supplying High-Quality and Trustworthy Answers," *IEEE Trans Big Data*, vol. 4, no. 4, pp. 600–613, Aug. 2017, doi: 10.1109/TBDATA.2017.2735442.

[14] A. Merchant, D. Shah, G. S. Bhatia, A. Ghosh, and P. Kumaraguru, "Signals matter: Understanding popularity and impact of users on stack overflow," *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*, pp. 3086–3092, May 2019, doi: 10.1145/3308558.3313583.

[15] R. Selleras, *Predictive model: using text mining for determining factors leading to high-scoring answers in stack overflow*. 2020. Accessed: Nov. 03, 2025. [Online]. Available: https://search.proquest.com/openview/7813a3cc073b1ad7cafe1373d4605ad8/1?pq-origsite=gscholar&cbl=18750&diss=y

[16] M. R. Tavakoli, A. Heydarnoori, and M. Ghafari, "Improving the quality of code snippets in stack Overflow," *Proceedings of the ACM Symposium on Applied Computing*, vol. 04-08-April-2016, pp. 1492–1497, Apr. 2016, doi: 10.1145/2851613.2851789.

[17] L. X. Zhao, L. Zhang, and J. Jiang, "Hot question prediction in Stack Overflow," *IET Software*, vol. 15, no. 1, pp. 90–106, Feb. 2021, doi: 10.1049/SFW2.12013.

[18] S. Baltes and S. Diehl, "Usage and attribution of Stack Overflow code snippets in GitHub projects," *Empir Softw Eng*, vol. 24, no. 3, pp. 1259–1295, Jun. 2019, doi: 10.1007/S10664-018-9650-5.

[19] Y. Wu, S. Wang, C. P. Bezemer, and K. Inoue, "How do developers utilize source code from stack overflow?," *Empir Softw Eng*, vol. 24, no. 2, pp. 637–673, Apr. 2019, doi: 10.1007/S10664-018-9634-5.

[20] M. Tavakoli, A. Heydarnoori, M. G.-P. of the 31st Annual, and undefined 2016, "Improving the quality of code snippets in stack overflow," *dl.acm.org*, Accessed: Nov. 03, 2025. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/2851613.2851789

[21] C. Ragkhitwetsagul, J. Krinke, M. Paixao, G. Bianco, and R. Oliveto, "Toxic code snippets on stack overflow," *ieeexplore.ieee.org*, no. Y, p. 1, 2018, doi: 10.1109/TSE.2019.2900307.

[22] Z. Gao, X. Xia, D. Lo, J. Grundy, X. Zhang, and Z. Xing, "I Know What You Are Searching for: Code Snippet Recommendation from Stack Overflow Posts," *ACM Transactions on Software Engineering and Methodology*, vol. 32, no. 3, Apr. 2023, doi: 10.1145/3550150.

[23] M. Ahmad, M. C.-2019 I. 16th International, and undefined 2019, "Impact of stack overflow code snippets on software cohesion: A preliminary study," *ieeexplore.ieee.org*, Accessed: Nov. 03, 2025. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8816758/

[24] D. Yang, A. Hussain, and C. V. Lopes, "From query to usable code: An analysis of Stack Overflow code snippets," *Proceedings - 13th Working Conference on Mining Software Repositories, MSR 2016*, pp. 391–401, May 2016, doi: 10.1145/2901739.2901767.

[25] S. Baltes, S. D.-E. S. Engineering, and undefined 2019, "Usage and attribution of Stack Overflow code snippets in GitHub projects," *Springer*, 2018, Accessed: Nov. 03, 2025. [Online]. Available: https://link.springer.com/article/10.1007/s10664-018-9650-5

Omid Mohammadi Kia, ORCID iD: 0000-0003-2225-4450
Shahid Beheshti Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran, o_mohammadikia@sbu.ac.ir
Mahmood Neshati, ORCID iD: 0000-0002-3953-4857
Shahid Beheshti Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran, m_neshati@sbu.ac.ir

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

Improving the Accuracy of Spectrum-Based Fault Localization Techniques by Focusing on the Program's Changes and Dependencies

Faeze aghazade-par, CODE ORCID: 0009-0003-5530-1310
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran, f_aghazadepar@sbu.ac.ir
MOJTABA VAHIDI-ASL✉, CODE ORCID: 0000-0003-4964-992X
Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran, mo_vahidi@sbu.ac.ir

ABSTRACT

Spectrum-Based Fault Localization (SBFL) techniques are widely used and studied. These techniques are straightforward, low-cost, and fast in comparison to other fault localization techniques, such as Mutation-Based and Learning-Based ones. However, the accuracy of these techniques is always controversial. The main criticism of these techniques refers to the mere use of coverage and spectra information. Hence, this study seeks to improve SBFL techniques by enhancing their accuracy while maintaining simplicity, aiming to make them applicable for regression faults. To achieve this, the first step involves integrating SBFL techniques with two features: (1) Change and (2) Test Case Weight, which utilizes the same coverage and spectrum data in a novel way. Additionally, the study examines the impact of incorporating Data and Control Dependency into the formula for fault localization accuracy. It also considers the suspiciousness scores generated by SBFL formulae as a feature.

Three SBFL techniques—Tarantula, Ochiai, and Jaccard—are applied in this study both as a feature and as a baseline for result comparison. The findings reveal that combining SBFL suspiciousness scores with Change and Test Case Weight significantly enhances performance by identifying at least 27% more faults at the Top-3 ranking. Furthermore, integrating these features with Data Dependency leads to even greater improvements, locating minimum 45% more faults at the Top-3 ranking compared to aforementioned SBFL formulae. Overall, this study emphasizes the limitations of existing SBFL formulae while highlighting the advantages of augmenting SBFL with additional features and repurposing spectral information to achieve greater accuracy in fault localization.

KEYWORDS

Spectrum-Based Fault Localization, Spectra Information, Versioning, Data Dependency, Control Dependency, Test Cases

11. INTRODUCTION

Regardless of the extent of effort invested in their development, software systems are often prone to faults and defects. A key factor contributing to these issues is the inherent complexity of the software. This challenge has become increasingly significant with the advent and widespread adoption of complex and sophisticated systems [1]. Addressing software faults requires accurately pinpointing their locations as an essential preliminary step before initiating the repair procedures [2, 3]. However, fault localization is a notably challenging, labor-intensive, and intricate process, accounting for approximately 54% of the time and effort expended by software developers during the development lifecycle [4, 5, 6, 7].

Hence, automated software fault localization techniques have been developed to mitigate the challenges associated with debugging by reducing costs and enhancing efficiency. Fault localization, a prominent area of research in software engineering, is defined as "identifying locations in a program's source code that are implicated in some observed failures (such as crashes or other kinds of runtime errors) as a key step of debugging" [8]. These techniques streamline the fault localization process, enabling developers to focus their efforts more effectively on resolving faults and improving software quality [9]. Fault localization methods have been categorized in diverse ways across various studies. For instance, [10] classifies them into four main types: slice-based, spectrum-based, learning-based, and mutation-based techniques. Similarly, [11] groups these methodologies into state-based, slice-based, statistical, machine learning-based, and spectrum-based categories. Another classification introduced by [12] includes slice-based, model-based, predicate-based, and spectrum-based approaches.

Additionally, [13] organizes fault localization techniques into spectrum-based, mutation-based, dynamic slicing, stack trace analysis, prediction switching, information retrieval-based, and history-based categories. Expanding upon these categorizations, [2] incorporates spectrum-based, slicing-based, state-based, statistical, machine learning-based, model-based, and mutation-based methods, among others. Furthermore, [14] delineates fault localization methodologies into three principal groups: spectrum-based fault localization (SBFL), mutation-based fault localization (MBFL), and fault localization techniques leveraging machine learning (ML) and deep learning (DL). These varied classifications underscore the dynamic and evolving nature of fault localization approaches, offering multiple perspectives on effectively addressing software faults.

Among the various fault localization techniques, Spectrum-Based Fault Localization (SBFL) stands out as a widely adopted and highly effective approach. This dynamic method leverages two key types of information: test case execution results and the

Submit Date: 2025-11-23
Revise Date: 2026-02-09
Accept Date: 2026-04-28
✉ Corresponding author

program spectrum. Test case execution results are systematically recorded to determine whether each test has succeeded or failed, with failures indicating the presence of faults in the program. The program spectrum, on the other hand, provides detailed code coverage information for each test case [2]. SBFL operates by analyzing the statements of the code executed during test cases and assigning a suspiciousness score to each statement. These scores are determined based on the frequency of a statement's occurrence in both successful and failed test executions.

Spectrum-based techniques are highly valued for their simplicity and efficiency, making them especially well-suited for large-scale and real-world software systems [11]. They are particularly effective in identifying faults in single-fault programs, showcasing strong performance in such scenarios [15]. Consequently, numerous studies have proposed various formulae and methodologies based on SBFL, many of which have achieved significant recognition. Among these, approaches such as *Tarantula* [16], [17], *Jaccard* [18], *Ochiai*, *D-Star* [19], *Barinel*, and *OP2* have emerged as some of the most widely used and effective methods, highlighting their prominence in fault localization research.

While spectrum-based techniques have earned widespread recognition for their simplicity and efficiency, they are subject to significant critiques. Many current Spectrum-Based Fault Localization (SBFL) methods fail to account for two crucial aspects: (1) the varying contributions of failed test cases to identifying specific faults and (2) the collaborative interactions among program elements that differently influence the success or failure of the test cases [6]. Overlooking these dimensions limits the ability of SBFL methods to accurately pinpoint faults and grasp the nuanced behavior of the test cases. A further limitation lies in the occurrence of ties, where multiple statements receive identical suspiciousness scores [20, 10, 4, 13, 2, 15]. This problem stems from the presence of certain statements, such as conditional ones, in all test case executions, whether successful or unsuccessful, making it difficult to discern between faulty and non-faulty statements. Moreover, SBFL relies exclusively on the test case coverage and execution results [21, 22, 23, 24, 25], restricting its capacity to identify the root cause of faults [21, 4].

In summary, SBFL's shortcomings include its relatively low accuracy [13, 24, 26], its inability to consider program structure [23], inefficiency in managing large-scale software systems [24], and its weak performance in multi-fault environments [27, 28]. Furthermore, SBFL methods are unable to interpret the behavior of statements close to one another [29]. These limitations underscore the necessity for advanced fault localization techniques that effectively address these challenges while enhancing accuracy and adaptability.

Consequently, this study aims to: (1) enhance SBFL techniques by incorporating additional information into the formula and (2) improve the accuracy of SBFL formulae for locating faults in different software versions. The proposed method, referred to as "*SBCT*", incorporates three primary features: the SBFL Formula's Score, Change, and Test Case Weight. The SBFL Formula's Score represents the suspiciousness score of a statement derived from the SBFL formula. Given the effectiveness and value of SBFL techniques, this feature provides critical data for fault localization. The Test Case Weight, introduced in our previous studies [30, 31], has been selected due to its simplicity, as it does not require a separate process for value extraction. It utilizes test case results and coverage information for each statement, similar to SBFL techniques, while accounting for how statements are covered during test case executions. In essence, the Test Case Weight differentiates statements based on the specific test cases that cover them.

Additionally, the Change feature is included to account for software modifications and assess the applicability of SBFL in identifying regression faults. It is worth noting that these two features, Change and Test Case Weight, have been demonstrated to be effective in fault localization [30, 31]. Furthermore, dynamic data and control dependencies, suggested by prior research for fault localization, are incorporated into *SBCT*. These additional features are evaluated to determine their potential for enhancing the accuracy of SBFL techniques.

In this method, the program is instrumented and executed against the test cases, simultaneously generating a coverage matrix for SBFL formula calculations. Following the SBFL routine, suspiciousness scores for the statements are computed after all test cases are executed, providing the SBFL Formula's Score for each statement. For every test case execution, the values for Test Case Weight are calculated using the coverage matrix. Subsequently, the Test Case Weight values from all test case executions are aggregated to obtain the final value of this feature for each statement. Finally, Change values are assigned to the statements, with a value of 1 if the statement has been modified and zero otherwise.

These features were then utilized to formulate a new method (*SBCT*) for fault localization. To evaluate the formula, the extracted feature values were substituted into *SBCT* to determine the suspiciousness scores for individual statements. For assessment, codes with at least three versions were selected from the *Code4Bench* benchmark dataset [32]. The results demonstrated that *SBCT* significantly outperformed existing SBFL formulae in terms of *EXAM* metrics [33] and *Top-N* accuracy, where N equals 1, 3, and 5.

This study highlights the critical role of Change values and demonstrates that SBFL can be improved effectively without requiring additional external data. To encapsulate the essence of this research, the study offers the following key contributions:

- **Introducing a Unified Feature Extraction Framework:** Combining multiple features—Coverage Matrix, Test Case Weight, Change, and Dynamic Data Dependency—into a cohesive and efficient framework for fault localization.
- **Developing an Adaptable Fault Localization Formula:** Creating *SBCT* formula that not only enhances SBFL techniques for regression faults but also adapts seamlessly to different program types and sizes.
- **Minimizing Cost through Feature Efficiency:** Demonstrating that effective fault localization can be achieved by utilizing easily collectible and low-cost features (Test Case Weight and Change) without relying on external data sources.
- **Empirical Validation on Real-World Programs:** Offering robust evaluations on diverse real-world programs with varying sizes and complexities, thereby establishing broad applicability and reliability.
- **Revolutionizing Regression Fault Handling:** Demonstrating the utility of Change in locating regression faults and emphasizing its role in future fault localization advancements.

The structure of this paper is organized as follows: Section 2 presents an illustrative example to elucidate the core concept of the proposed methodology. Section 3 introduces fundamental concepts of the research. Section 4 offers an extensive review of related literature. Section 5 provides a detailed explanation of the proposed approach. Section 6 describes the evaluation

framework and experimental design. Section 7 reports the evaluation results. Finally, the concluding sections focus on an in-depth discussion of the results and outline the overarching conclusions drawn from the study.

12. MOTIVATIONAL EXAMPLE

This section provides an illustrative example to examine the limitations of SBFL techniques and demonstrate the advantages of *SBCT*. SBFL is a widely utilized technique that relies exclusively on the spectral information of the code, calculating suspiciousness scores based on statement coverage and test case results. However, this reliance on simple information can sometimes mislead the fault localization process in specific scenarios.

To analyze SBFL performance, Table 1 presents a straightforward example involving 10 test cases, showcasing both the coverage matrix and SBFL results. The three leftmost columns are dedicated to three commonly used SBFL formulae—*Tarantula*, *Ochiai*, and *Jaccard*. These columns detail the suspiciousness scores assigned to each statement. Based on these scores, statements are sorted and ranked according to their likelihood of being faulty. The formulae for *Tarantula*, *Ochiai*, and *Jaccard* are defined using specific notations to clarify their functionality and application within the context of SBFL:

N_{cf}: Number of failed runs in which the statement is covered
N_{uf}: Number of failed runs in which the statement is not covered
N_{cp}: Number of passed runs in which the statement is covered
N_{up}: Number of passed runs in which the statement is not covered

$$\text{suspiciousness score}_{(\text{Tarantula})} = (N_{cf}/(N_{cf}+N_{uf})) \,/\, ((N_{cf}/(N_{cf}+N_{uf})) + (N_{cp}/(N_{cp}+N_{up}))) \quad [16]\ (1)$$

$$\text{suspiciousness score}_{(\text{Ochiai})} = N_{cf}/\sqrt{(N_{cf}+N_{uf}) \cdot (N_{cf}+N_{cp})} \quad [18]\ (2)$$

$$\text{suspiciousness score}_{(\text{Jaccard})} = N_{cf}/(N_{cf}+N_{uf}+N_{cp}) \quad [18]\ (3)$$

Table 2. Example of SBFL formulae

Statements of Code	*Ncf*	*Nuf*	*Ncp*	*Nup*	Tarantula	Ochiai	Jaccard
num = int(input())	6	0	4	0	0.5	0.301511	0.6
factorial = 0 //fault	6	0	4	0	0.5	0.301511	0.6
if num < 0:	6	0	4	0	0.5	0.301511	0.6
print("Sorry, factorial does not exist for negative numbers")	0	6	3	1	0	0	0
elif num == 0:	6	0	1	3	0.8	0.353553	0.857143
print("The factorial of 0 is 1")	0	6	1	3	0	0	0
else:	6	0	0	4	1	0.377964	1
for i in range(1,num + 1):	6	0	0	4	1	0.377964	1
factorial = factorial*i	6	0	0	4	1	0.377964	1
print("The factorial of", num, "is", factorial)	6	0	0	4	1	0.377964	1

As shown in Table 1, these formulae struggle to effectively differentiate between statements, often assigning identical suspiciousness scores to multiple statements. This limitation compromises the accuracy of fault localization, as the faulty statement becomes indistinguishable from two other statements with identical scores. Notably, in all three formulae, the faulty statement is consistently ranked as the 7th most suspicious, yielding an *EXAM* score of 70%.

Following this analysis, Table 2 introduces *SBCT*, utilizing the same code and test cases as Table 1 for direct comparison. To enhance clarity and facilitate better comprehension, Table 2 employs the following notations for improved readability.

D1: Data dependency (*def*)
D2: Data dependency (*use*)
CH: Change
TCW: Test Case Weight

Table 3. Example of *SBCT*

Statements of Code	Jaccard Score	D1	D2	CH	TCW	SBCT score
num = int(input())	0.6	20	0	0	5	-2.6338
factorial = 0 //fault	0.6	5	0	1	5	-1.2358
if num < 0:	0.6	0	5	0	5	-2.7928
print("Sorry, factorial does not exist for negative numbers")	0	0	0	0	0	-2.785
elif num == 0:	0.857143	0	5	0	5	-2.9144
print("The factorial of 0 is 1")	0	0	0	0	0	-2.785
else:	1	0	0	0	5	-2.983
for i in range(1,num+1):	1	10	10	1	5	-1.383
factorial = factorial*i	1	9	14	0	5	-2.9082
print("The factorial of", num, "is", factorial)	1	0	10	0	5	-2.981

Table 2 highlights two critical differences when compared to Table 1. First, the suspiciousness scores across statements show apparent variation, underscoring the effectiveness of the proposed features in distinguishing between different statements. Second, the "PM Score" column demonstrates the superior performance of *SBCT*, enabling more efficient fault localization compared to SBFL techniques. Notably, based on the "SBCT Score," the faulty statement is accurately identified as the most

suspicious one, with the highest score assigned. In the subsequent sections, a comprehensive explanation of *SBCT*'s formula and the methodology utilized for computing suspiciousness scores is provided to detail the process and validate its effectiveness.

13. BACKGROUND

3.1 FAULT LOCALIZATION

In order to improve software quality, it is essential to identify faults by software testing and locate them for repair [3]. Fault localization is a process for identifying the location of the fault, which begins after software testing and following the failure of the test execution result [2]. This process identifies specific elements of the program that are responsible for the program fault [5]. Given the importance of fault localization, numerous studies have been conducted in this field; and each provides a different classification of existing techniques. According to [28], there are three categories of fault localization techniques, which are trace-based debugging, delta debugging, and coverage-based fault localization. Another research has presented a different classification. According to [10], there are four categories of techniques for fault localization, including slicing-based, spectrum-based, learning-based, and mutation-based. Another classification is provided in [12] for fault localization techniques, which are slicing-based, model-based, predicate-based, and spectrum-based. Furthermore, in another classification by [13], fault localization techniques are divided into spectrum-based, mutation-based, dynamic slicing-based, stack trace analysis, prediction switching, information retrieval-based, and history-based. In addition, the classification provided by [2] considers fault localization techniques to include spectrum-based, slicing-based, state-based, statistics-based, machine learning-based, model-based, mutation-based, etc. All in all, the presented categories show that fault localization approaches are mainly categorized into three groups: mutation-based, spectrum-based, and learning-based approaches. Moreover, with recent advances in artificial intelligence and neural networks, learning-based approaches have accounted for the majority of recent research.

3.2 SPECTRUM-BASED FAULT LOCALIZATION

Spectrum-based fault localization (SBFL) is one of the most widely used and popular techniques in the field of fault localization. Spectrum-based fault localization is a dynamic technique that operates on two types of information: test execution results and program spectrum. In this method, test execution information, including its execution result, is stored. The test execution result can be successful or unsuccessful. Failure of the test indicates the presence of a fault in the program. The program spectrum refers to the code coverage information by each test case [10].

In detail, it can be said that in spectrum-based techniques, an instrumented version of the faulty program is executed using a test suite. Then, for each execution, a coverage matrix is generated to record the coverage of the code by the test case and the test case result. According to this matrix, it is determined how many code sections (statements/branches/functions) were present in faulty executions and vice versa. Subsequently, a SBFL formula is used to calculate the suspiciousness score for each section in the matrix, and rank the program entities [11]. The rule is that the more a given code entity is present in failed executions and fewer successful executions, the more likely it is to be faulty [13]. Many studies consider spectrum-based techniques to be lightweight and simple, which makes them more applicable to large and real-world applications [11].

3.3 MUTATION-BASED FAULT LOCALIZATION

In the mutation-based techniques, syntactic changes are made to the program under test, and then different versions of the program are generated that include unrealistic faults. These versions are then run with the existing tests [2]. Mutation-based fault localization extends the code coverage information by changing the statements with the changed versions of the program, called mutants, and then collects the new coverage information obtained by running the mutants and calculates the suspiciousness score of the statements [13]. In the mutation-based technique, the faulty program is changed regularly and re-run with the help of the existing tests with each change. Suppose that statement S is changed. If a larger number of test cases succeed as a result of changing S, it can be concluded that S was faulty. This technique is superior in accuracy to spectrum-based localization [2]. It also operates finer-grained than spectrum-based techniques. In other words, it measures the effect of each statement in the code [22]. In addition, this method is very efficient in simulating the behavior of real faults [2].

3.4 LEARNING-BASED FAULT LOCALIZATION

With the development of neural networks and their widespread use, machine learning and deep learning were also used in fault localization. In this method, the learning network extracts several features from the code and performs fault localization based on those features. The use of machine learning in cost and value-based techniques has shown high effectiveness. Also, experiments show that deep learning methods perform more effectively than information retrieval methods [20]. Despite the fact that learning methods are more effective than other techniques, the efficiency of these models decreases with increasing program size [29], and the accuracy of learning networks is still controversial [20]. Moreover, another criticism of this method is the need for a huge data set for learning to get high accuracy and precision.

14. RELATED WORKS

This section examines research efforts aimed at improving spectrum-based fault localization (SBFL) techniques. Various studies have proposed methodologies, features, and frameworks to address the limitations of SBFL, such as accuracy, efficiency, and adaptability, ensuring better fault localization outcomes.

Thus, *EXCEPT* is a technique developed to improve the accuracy of spectrum-based fault localization. Unlike SBFL, which relies on the relationship between executed statements and failed tests, *EXCEPT* focuses on the semantics of failures, particularly those caused by exceptions, and identifies both the faults and their locations based on the meaning of the exception and its underlying cause. The technique takes as input a faulty program, a test that failed with an uncaught exception, and a ranked list of suspicious statements generated by SBFL. Upon execution of the program and detection of a failure, *EXCEPT* expands the list of suspicious statements by adding newly identified candidates for repair and then integrates these additions with the existing SBFL-ranked list. This process aims to enhance the accuracy and effectiveness of fault localization [34].

Additionally, an automated framework was developed using a chaos-based genetic algorithm to address the challenge of multi-fault localization. This approach builds upon the foundation of spectrum-based techniques. In this framework, program

statements serve as chromosomes, while their corresponding suspiciousness scores represent the genes. The genetic algorithm is then applied, employing uniform crossover for recombination and utilizing the logistic mapping function to introduce chaos order into the process. The *D-Star* method is leveraged to compute the suspiciousness scores for each program statement, guiding the localization of faults effectively [28].

Moreover, a hybrid fault localization approach combines spectrum-based, mutation-based, and neural network-based techniques. Mutants are generated by modifying a correct program's operators and operands. Spectra and test execution results from both the faulty program and mutants are processed through two modules: one spectrum-based and the other neural network-based. These results are analyzed for similarity scores using the *RankBoost* algorithm, which merges scores from various methods to compute final suspiciousness values. Statements linked to multiple mutants are assigned the highest similarity scores, improving fault localization accuracy [35].

Similarly, *ABFL* is a fault localization approach that leverages a combination of features and ranking models for enhanced accuracy. It begins by extracting 32 features from the program's static source code through an automatic encoder. This process involves tokenizing program statements, enabling the encoder to generate fixed-length representations for each statement, thus capturing features from the source code. Additionally, 14 types of suspiciousness scores are calculated using spectrum-based techniques, which are then incorporated as an extra feature. All extracted features are subsequently integrated into ranking learning algorithms to construct a ranking model. This model is utilized to rank program statements, effectively pinpointing faulty ones. By combining spectrum-based insights with static code features, ABFL achieves an accurate fault localization [36].

Similar to the previous two, [13] merges spectrum-based and mutation-based methods to enhance fault localization accuracy. Initially, the faulty program is assessed using spectrum-based techniques, which rank potentially faulty statements by their suspiciousness scores. Following this, a mutant program is generated and executed, and the high-ranking statements from the spectrum-based evaluations are re-examined and re-ranked based on their mutation scores, refining the localization process further. Correspondingly, [37] combines spectrum-based localization with fault influence propagation analysis for enhanced fault detection. A dynamic instrumented execution tracker is developed to collect test coverage data and method call relationships concurrently. Using complex network theory, a fault influence network is constructed by abstracting the network topology from these method call relationships, with node weights determined through raw fault localization.

Besides, another research suggested a predicate switching technique, a lightweight mutation-based technique, to refine fault localization by augmenting spectrum-based methods. Initially, the spectrum-based routine is executed to obtain coverage information and a ranking list of suspicious statements. Statements with high suspiciousness scores are then selected for mutation. For each mutated statement: a) If the mutation alters the test case result, all statements likely to affect the values of this statement are marked as suspicious for the failed test case execution. b) If the mutation does not change the test case result, statements solely influencing the values of the mutated statement are identified as less suspicious for the successful test case execution. Finally, these findings are integrated with the spectrum-based formula results, enabling a refined re-ranking of statements to improve fault localization accuracy [22].

Furthermore, [38] explores a novel source of information for fault localization by integrating data flow analysis into the calculation of the suspiciousness scores. The method calculates the suspiciousness of program statements through the execution of data flow components. An evolutionary algorithm is then employed to optimize the weights derived from various fault sources, including data flow, control flow, and combination flow. Unlike traditional coverage-based approaches that focus on control flow, this method utilizes data flow and *def-use* relations to enhance fault localization accuracy. A genetic algorithm further refines the process by searching for optimal weights within the linear combination of suspiciousness values, ensuring a more accurate fault localization.

Additionally, [23] explores how program structure influences the effectiveness of spectrum-based fault localization techniques. To address the structure bias in suspiciousness calculations, a structure-aware method is proposed that incorporates debug history. The approach involves running test cases across all program versions and recording their execution traces. For each version, the suspiciousness scores of the basic blocks are calculated using an SBFL method. To refine the results, the average suspiciousness of each block across different versions is computed and subtracted from the suspiciousness of the same block in the current version. This adjustment aims to reduce structural bias and improve the accuracy of fault localization.

Similarly, a fault localization in continuous integration is achieved through the *Ochiai* spectrum-based technique by [39]. The process begins by collecting daily test results, focusing on failed tests. In cases of multiple test runs, only the latest run is considered. Coverage data and the spectrum-based method are then employed to perform fault localization for each daily test, with weekly coverage information serving as the reference. Suspiciousness scores for statements are calculated and aggregated using a max aggregation scheme, which scales the scores from the statement level to the component and file levels. Within this scheme, the highest suspiciousness score among a file's statements is assigned to the file.

Another history-based research identifies the bug-inducing commits for each test that uncovered a bug, and pinpoints the first commit where the failure occurred [4]. For every suspicious code entity, a historical spectrum is created within these bug-inducing commits, mapping the evolution of the suspicious code elements from the faulty version to the present. At this stage, history slicing is applied to refine the analysis. Suspiciousness scores for code entities are then calculated using a spectrum-based method applied to the historical spectrum. Entities that have undergone numerous changes in bug-inducing commits but minimal changes in healthy commits are deemed more likely to be the source of the bug.

To enhance the accuracy of spectrum-based fault localization, a customized metric is designed and tailored to the program's specific nature and runtime characteristics by [11]. Multiple faulty and mutated versions are generated for each program and tested using relevant data. The effectiveness of each version is evaluated using existing ranking metrics. From this evaluation, a selection of the best-performing metrics is made, which are then combined to create a new, optimized metric for improved fault localization.

Moreover, *ProFL* is a dynamic fault localization approach that refines fault prioritization by incorporating patch generation data from the *PraPR* tool. Capable of handling diverse bug types, *ProFL* begins with a faulty program and a failure test suite as inputs. Suspiciousness scores are initially calculated using spectrum-based techniques like *Ochiai* at the statement level, and

scores are aggregated for each program element. A matrix of patches and their corresponding test execution results is then generated. Based on this matrix, program elements are re-ranked by combining their suspiciousness scores with information from patch groups, enhancing the accuracy and efficiency of the fault localization [40].

To address spectrum-based incapability in multi-fault scenarios, an enhanced spectrum-based fault localization method named *FLITSR* is designed to improve performance in these scenarios. The technique uses an iterative process that incorporates test case reduction (TCR) to identify faults efficiently. Initially, the test suite is executed to locate a single fault, after which TCR is applied to exclude test cases heavily influenced by the coverage of the located fault. This refinement directs subsequent testing toward uncovering new faults [15]. Besides, a research evaluates seven fault localization techniques across four methodological families: spectrum-based, mutation-based, predicate switching, and stack tracing-based approaches, focusing on real Python programs with *FauxPy* as the analysis tool. Spectrum-based methods, including *Tarantula*, *D-Star*, and *Ochiai*, outperformed others in terms of effectiveness, while mutation-based approaches like *Metallaxis* and *Muse* ranked second. The study also examined combining these techniques in two ways, finding enhanced effectiveness with minimal impact on execution time. This work offers insights into the relative strengths of various approaches in large-scale fault localization tasks [8].

Similarly, [41] evaluates the effectiveness of spectrum-based fault localization formulae across real Python projects, focusing on *Tarantula*, *Barinel*, *Ochiai*, *Op*, and *D-Star*. The formula's performance was compared within Python projects, as well as across Java and C projects. Results indicate consistent performance across fault types, except for faults caused by deletion. Notably, older formulae like *Tarantula*, *Ochiai*, and *Barinel* outperform newer ones. Interestingly, while *Tarantula* excels in Python projects, the *Op* formula proves more effective in Java, emphasizing the robustness of traditional methods and uncovering language-specific performance differences.

Additionally, [6] employs spectrum-based fault localization as a classification model to predict test case outcomes (pass/fail) based on spectral information from their execution. EXplainable Artificial Intelligence (XAI) techniques are integrated to examine the localized significance of code segments in influencing these results. XAI analysis on failed test cases identifies the statements with the highest impact on test failures. The methodology dynamically learns how failed test cases contribute to the suspiciousness of various code segments, connecting test execution results with the implicated code parts to enhance fault localization accuracy.

In summary, the reviewed research highlights that analyzing program structure and its evolution history plays a critical role in improving the accuracy of SBFL techniques. This insight emphasizes the need to integrate structural and historical dimensions into these methods to achieve more effective debugging practices in software engineering. The study also points out a notable gap in the literature: limited focus on program evolution and the role of test cases in SBFL. Identifying research that examines how test cases contribute to localization effectiveness remains a significant challenge. To address this, the proposed study introduces a novel approach that combines SBFL with program evolution, structural characteristics, and the test suite, aiming to enhance spectrum-based fault localization techniques.

PROPOSED APPROACH

This section reviews *SBCT* and its research context, highlighting its aim to enhance the accuracy of spectrum-based fault localization techniques by addressing challenges stemming from software evolution. The approach adapts SBFL methods by considering changes introduced during iterative development. It also utilizes test cases and dependencies to ensure precise localization of faulty components in evolving systems.

Building on the validation of individual feature effectiveness in prior research, *SBCT* takes a novel step by innovatively combining features. This approach is designed to identify the most impactful features while ensuring maximum accuracy with the smallest possible set of features, advancing the research objective effectively. The architecture and methodology are illustrated in Fig. 1, with key inputs including:

- **Program p:** Along with its preceding version, to examine evolutionary changes.
- **Test Suite:** Incorporating both successful and failed test cases.
- **Code Change Locations:** Pinpointing the sections of the code altered during evolution.

Initially, the program under evaluation, referred to as p, is executed using its corresponding test suite. During this process, essential data points, such as the coverage matrix and test case outcomes, are collected. These values are aggregated for each program statement to derive the "Test Case Weight" feature. Simultaneously, suspiciousness scores for statements are calculated using the SBFL formulae, forming the "SBFL score." Additionally, information about changes in program statements compared to its prior version is systematically recorded as the "Change" feature. These features are then input into the proposed formula, which computes a suspiciousness score for each statement. Subsequently, the statements are ranked based on these scores. Considering these scores, statements with higher values are a more substantial likelihood of containing faults. This structured process ensures precise localization of the potentially faulty statements. The proposed method also integrates dynamic control and data dependencies into its framework, complementing the previously discussed features. This addition aims to assess how these dependencies influence the fault localization process, enhancing the overall accuracy and depth of the analysis.

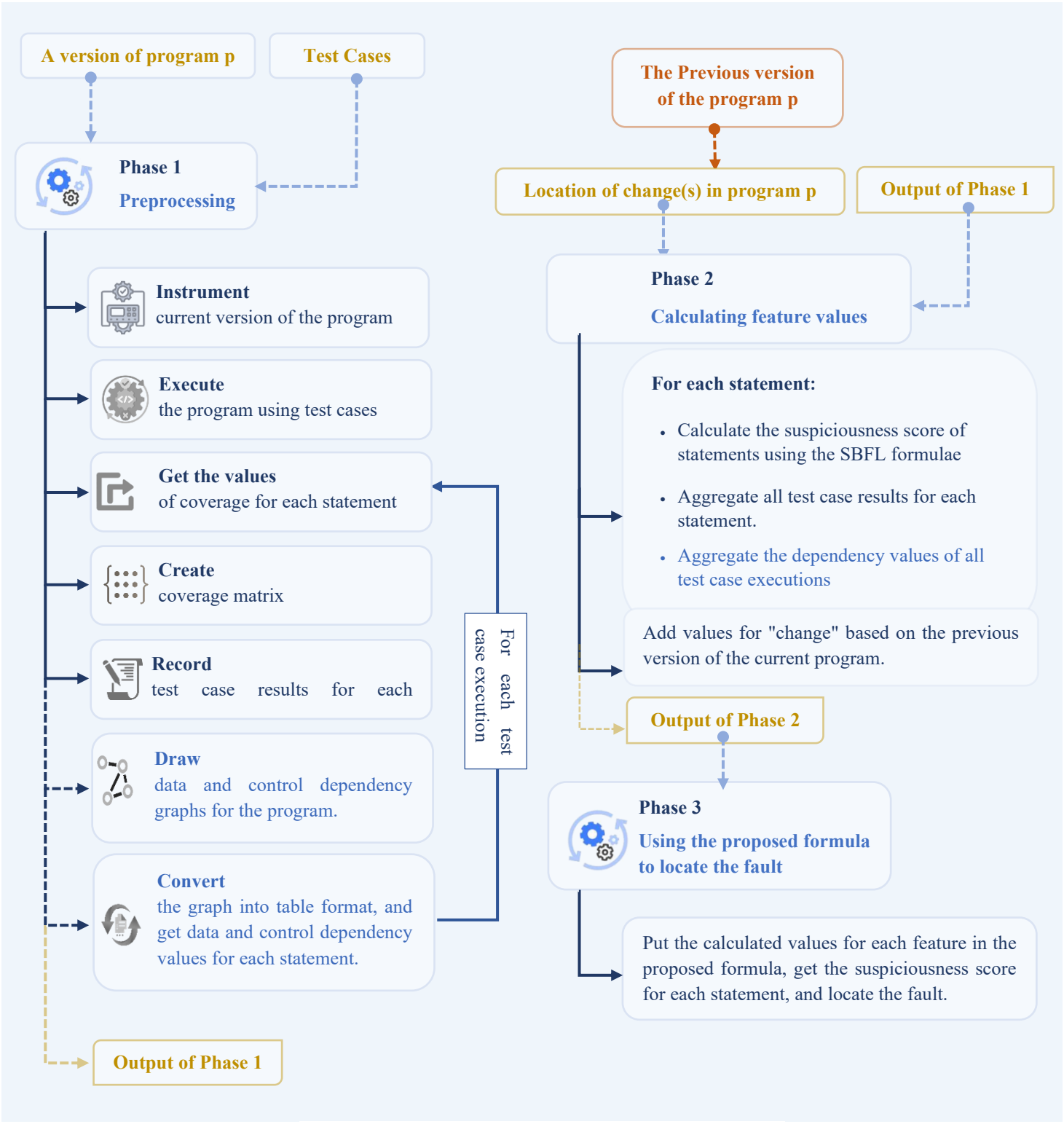


Fig. 1 Architecture and Methodology of *SBCT*

result is captured for the “Test Case Weight” feature. This phase involves documenting the outcomes of the test cases for each statement. As each test case is executed, values in the "Test Case Weight" column are assigned to each statement based on its coverage and test case results. These values are defined as follows:

- If a statement is covered by the test case and the result indicates the test case has passed, the corresponding matrix cell is updated with a value of -1.
- If a statement is covered and the test case result has failed, the matrix cell is assigned a value of 1.
- If a statement is not covered by the test case, the respective matrix cell is marked with 0.

This process ensures that each statement's test case execution and coverage status are systematically recorded, enabling further analysis in the fault localization method. This comprehensive data serves as the foundation for subsequent analytical processes in *SBCT*.

After executing all test cases, the recorded values of “Test Case weight” for each statement are aggregated separately. For example, statement S_1 was executed by 5 test cases. The first and second test cases did not cover S_1, so the related values were 0. The third and fourth test cases covered S_1 and their result were *Pass*, so the related values were -1. The fifth test case covered

S_1 and its result was *Fail*, so its related value was 1. Overall, according to this data, the value for "Test Case Weight" feature for S_1 is -1 (0 + 0 + (-1) + (-1) + 1 = -1)

To expand the experiment, both dynamic data and control dependencies are collected and added to the aforementioned coverage matrix. To incorporate dependencies into the coverage matrix, they are first represented numerically and formatted as table entries. Each dependency is divided into two components—"in" and "out"—resulting in four additional columns in the matrix: Data Dependency (*def*), Data Dependency (*use*), Control Dependency (*dom*), and Control Dependency (*domed*). To briefly introduce these features, Data Dependency (*def*) refers to a value defined within the statement that is subsequently utilized by other statements. Data Dependency (*use*) applies when the statement uses a value previously defined elsewhere. Control Dependency (*dom*) indicates that the statement exerts dominance over other statements in terms of control flow. Control Dependency (*domed*) refers to the statement being dominated by another statement that governs its execution.

Using these definitions, the matrix values are incremented systematically. If a statement defines a value that is later used, the corresponding cell of "Data Dependency (*def*)" in the matrix is incremented by 1. When a statement utilizes a previously defined value, its related cell of "Data Dependency (*use*)" is similarly updated. Regarding "Control Dependency (*dom*)", when a statement dominates others, its associated cell in the matrix is increased by 1 for each dominated statement; and for "Control Dependency (*domed*)", if another statement controls its execution, the respective cell in the matrix is incremented similarly. This structured approach captures the intricate relationships between statements. It also ensures that dependencies are accurately represented and integrated into the fault localization process.

The output of this phase is a table consisting of coverage information, location of modifications, test case results, and, subsequently, dynamic data and control dependencies presented as distinct columns.

5.2 Phase 2: Calculating feature values

In this phase, the matrix generated during Phase 1 undergoes detailed processing. The primary objective is to achieve a comprehensive understanding of the behavior of each program statement against test cases. The aggregated data provides insights into patterns and interactions within the matrix. To this end, a new table is generated to construct the dataset for fault localization. This table contains only features and their final values for applying the formula.

To initialize the "SBFL score" feature, the suspiciousness score of each statement is calculated using an SBFL formula. Then, it is placed in the corresponding cell in the table. Subsequently, an additional feature, referred to as "Change," is incorporated into the matrix to track whether each statement has been modified compared to its previous version. For programs in version 2 or beyond, a statement modified in the current version is assigned a "Change" value of 1. Conversely, statements that remain unchanged are assigned a value of 0. For the first version of the program, all entries in the "Change" column are automatically set to 0, as no prior versions exist for comparison.

Regarding the "Test Case Weight", the values accumulated across multiple test case executions are aggregated for every statement in the code. "Test Case Weight" helps to achieve a comprehensive view of each statement. This aggregation is also applied to data and control dependencies. By integrating both historical and behavioral data, this approach offers a holistic perspective for fault localization.

Table 3 illustrates the outcomes derived from this phase, showcasing the aggregated feature values for each program statement. In this example, the *Tarantula* formula is used for fault localization; however, other SBFL formulae can be used as alternatives. Additionally, the main approach focuses on the "SBFL score," "Change," and "Test Case Weight," with dependencies introduced later to examine their effectiveness alongside other features.

Table 4. Output of Phase 2

Statements	Tarantula scores	Sum of data dependency (def)	Sum of data dependency (use)	Sum of control dependency (dom)	Sum of control dependency (domed)	Change	Test Case Weight
1	0.5	22	0	0	0	0	-8
2	0.5	22	22	0	0	0	-8
3	0.5	22	22	0	0	0	-8
4	0.5	32	0	0	0	0	-8
5	0.5	22	32	66	0	1	-8
6	0.5	85	31	0	22	0	-8
7	0.5	0	22	12	22	0	-8
8	0.666666667	0	0	0	12	0	0
9	0.5	0	22	22	22	0	-8

5.3 Phase 3: Using the proposed formula to locate the fault

In the third phase, the extracted and compiled features from earlier stages are utilized. This phase involves applying specific formulae proposed in this research (Eq. 4-12). The formulae are used to determine a suspiciousness score for each statement in the program. Equation 4 relates to the fundamental idea of the research, using the scores of the *Tarantula* formula as a feature. Equations 5 and 6 pertain to the same experiment, incorporating data dependency and data and control dependencies. The following three equations present the same formulae applied with *Ochiai*, and the subsequent three are dedicated to *Jaccard*. As with all statistical and spectrum-based techniques, a higher suspiciousness score indicates a stronger likelihood that a statement contains a fault. To improve clarity, the same abbreviations introduced in the motivational example are reused within these formulae.

Suspiciousness score of Sn = Tarantula Score(1.889) + CH*1.938+ TCW*0.03- 3.827* (4)
Suspiciousness score of Sn =
Tarantula Score(1.351) + D1*0.00004+ D2*0.007 + CH*1.509+ TCW*0.004- 3.637* (5)
Suspiciousness score of Sn =
Tarantula Score(1.743) + D1*0.001+ D2*0.01+ C1*(-0.002) + C2*(-0.009) + CH*1.632+ TCW*0.036 - 3.533* (6)
Suspiciousness score of Sn = Ochiai Score(2.512) + CH*1.824+ TCW*0.005- 1.368* (7)
Suspiciousness score of Sn =
Ochiai Score(0.432) + D1*(-0.0001) + D2*0.007 + CH*1.551+ TCW*0.038- 3.203* (8)
Suspiciousness score of Sn =
Ochiai Score(1.168) + D1*0.0009+ D2*0.009+ C1*(-0.003) + C2*(-0.01) + CH*1.728+ TCW*0.026 - 3.611* (9)
Suspiciousness score of Sn = Jaccard Score(0.952) + CH*1.899+ TCW*0.025 - 3.301* (10)
Suspiciousness score of Sn =
Jaccard Score(0.637) + D1*0.002 + D2*0.015 + CH*2.153+ TCW*0.044- 0.744* (11)
Suspiciousness score of Sn =
Jaccard Score(0.628) + D1*0.001+ D2*0.009+ C1*(-0.002) + C2*(-0.008) + CH*1.729+ TCW*0.035 - 3.297* (12)

Using the specified formulae, suspiciousness scores are calculated for all program statements. These scores form the basis for ranking, with statements assigned higher scores being considered more likely to contain faults. Therefore, they are placed higher in the ranking.

It is worth mentioning that the correlation coefficients of each feature in these formulae are obtained by applying logistic regression on the train set using Python libraries. To obtain the most appropriate correlation coefficients, different sampling methods were tested, which are also detailed in Tables 4 to 12.

16. EXPERIMENTAL DESIGN

This section describes the procedure followed in the experiment, detailing the setup for both experimentation and evaluation, as well as the methods employed for data collection.

6.1 Experimental setup

To generate the aforementioned formulae, data were gathered from 14 programs, each containing over three faulty versions. This dataset included 61 source codes and 116 faults, to which Phases 1 and 2 of the proposed method were applied. An additional feature, termed "result," was incorporated into the Phase 2 output table. This feature specified whether a statement was faulty or non-faulty. Specifically, the "result" column assigned a value of 1 for faulty statements and 0 for non-faulty ones.

The primary objective of this task was to distinguish faulty statements from non-faulty ones, framing it as a binary classification problem. To achieve this, logistic regression was chosen to generate the formulae, as the primary objective of the formulae is to classify code statements into two categories: faulty and non-faulty. We added a column titled "result". It acted as the dependent (categorical) variable, and all other features served as independent (explanatory) variables. Given that the dataset's dependent variable included only two possible values—0 for non-faulty and 1 for faulty—there was an evident imbalance. Non-faulty statements significantly outnumbered faulty ones, as is typical in most programs, where the majority of the code is non-faulty.

To address the imbalance and enhance model performance, both oversampling and undersampling methods were tested. For oversampling, the *RandomOverSampler* and *SMOTE* techniques were utilized, while *EditedNearestNeighbours* and *RandomUnderSampler* were applied for undersampling. These methods were explored to determine the most effective solution for the dataset. Once balanced, logistic regression was applied, generating a model that provided correlation coefficients for each independent variable. These coefficients formed the foundation for constructing the final formulae. This experiment aimed to address specific research questions regarding the method and its efficacy:

Q1: What is the role of code modifications in SBFL fault localization techniques?
Q2: Is SBFL a sufficient formula or a supplementary feature for fault localization in evolutionary software?
Q3: Which features work best together to achieve the highest accuracy at locating faults?

In summary, these questions comprehensively capture the essential capabilities of *SBCT.* In other words, they demonstrate how *SBCT* utilizes SBFL techniques alongside historical code modifications, test case weighting mechanisms, and aggregated features. By integrating these elements, *SBCT* significantly enhances the accuracy of fault localization, particularly in single-fault scenarios. It also ensures a more targeted and efficient approach to locating faults.

6.2 Data collection

The study utilized programs and test cases sourced from the *Code4Bench* benchmark, specifically focusing on Java programs written in Java 6. Java was chosen for its extensive availability of source code and corresponding test cases. However, the proposed methodology is designed to be language-agnostic, ensuring applicability across various programming languages. The *Code4Bench* dataset offers authentic code samples with real faults collected from *Codeforces*, featuring concise and straightforward examples. These smaller programs intensify challenges like "tie" issues, making them an ideal testing ground for demonstrating the efficacy of *SBCT*. Only programs with at least three versions were included in the research, with an emphasis on those containing faults that led to incorrect test results. This focus was deliberate, as faults producing erroneous outputs are often more challenging to detect compared to those causing program crashes. Additionally, the dataset includes programs with both single and multiple faults. It provides a comprehensive environment for evaluation. Regarding test cases, only valid test cases were employed in the experiment, encompassing both passing and failing scenarios to ensure robust and meaningful analysis. It is worth mentioning that the train set and test set are two distinct files containing individual programs.

6.3 Evaluation

The effectiveness of *SBCT* was evaluated using two well-established metrics: *EXAM* score and *TOP-N* accuracy. The *EXAM* score measures the proportion of the code that must be inspected to locate a fault. Lower *EXAM* scores signify higher accuracy in fault localization, as fewer code sections require examination. *TOP-N* accuracy assesses the method's effectiveness in identifying faults within the top N positions of a ranked list. For this study, the focus was on *Top-1*, *Top-3*, and *Top-5*, representing whether the fault was detected at the 1st, third, or fifth rank, respectively.

17. EXPERIMENTAL RESULTS

This section explores the analysis of the experimental data to address the research questions. A key part of the evaluation focuses on the correlation coefficient values obtained from applying various undersampling and oversampling algorithms. These values are critical for assessing the effect and efficiency of each feature under examination. To address the research questions, the results of all experiments, along with their analysis, are first presented. Subsequently, the questions are answered based on these results. Tables 4, 5, and 6 provide a comprehensive ranking of the algorithms based on their respective correlation coefficients. They highlight the importance and role of each feature in fault localization using the three main features. These rankings serve as valuable indicators of the relative effectiveness of each feature within the experiment's context. Additionally, the abbreviations used in the tables are outlined to enhance understanding and clarity of the results presented.

EN: EditedNearestNeighbours
RU: Random UnderSampler
RO: Random OverSampler

Table 5. Logistic Regression Correlation Coefficients for the first experiment with *Tarantula* Score

	EN	RU	RO	SMOTE
Tarantula Score	1.889	1.852	3.255	3.418
Change	1.938	1.165	1.644	1.863
Test case Weight	0.03	0.031	0.031	0.034

Table 5. Logistic Regression Correlation Coefficients for the first experiment with *Ochiai* Score

	EN	RU	RO	SMOTE
Ochiai Score	1.473	1.603	2.512	2.838
Change	1.886	1.898	1.824	2.03
Test case Weight	0.017	0.018	0.005	0.001

Table 6. Logistic Regression Correlation Coefficients for the first experiment with *Jaccard* Score

	EN	RU	RO	SMOTE
Jaccard Score	0.952	1.348	2.2	2.423
Change	1.899	1.918	1.852	2.039
Test case Weight	0.025	0.015	0.01	0.007

As shown in Tables 4, 5, and 6, the "SBFL score" and "Change" are both practical and significant features for fault localization. "Change" is the most important feature for fault localization when using undersampling algorithms, followed by "SBFL score" as the second most important. Nonetheless, both features exhibit a similar effect on fault localization. However, "Test Case Weight" demonstrates the least importance and the lowest correlation with the faultiness of statements.

Based on our prior studies [30, 31], "Test Case Weight" proves to be a valuable feature for fault localization in the absence of SBFL, as it provides a similar perspective. Therefore, the "SBFL score" appears to be more significant and informative than "Test Case Weight" in this combination. To further examine these claims, Tables 7, 8, and 9 offer a detailed ranking of the algorithms, incorporating data dependency in addition to the previously evaluated features.

Table 7. Logistic Regression Correlation Coefficients for the second experiment with *Tarantula* Score

	EN	RU	RO	SMOTE
Tarantula Score	1.351	2.121	3.06	3.289
Data dependency (*def*)	0.00004	0.007	0.003	0.003
Data dependency (*use*)	0.007	0.012	0.013	0.014
Change	1.509	1.655	1.736	1.916
Test case Weight	0.004	0.016	0.037	0.035

Table 8. Logistic Regression Correlation Coefficients for the second experiment with *Ochiai* Score

	EN	RU	RO	SMOTE

Ochiai Score	0.432	1.747	2.11	1.985
Data dependency (def)	-0.0001	0.0004	0.002	0.001
Data dependency (use)	0.007	0.013	0.011	0.012
Change	1.551	1.603	1.918	2.121
Test case Weight	0.038	0.01	0.017	0.021

Table 9. Logistic Regression Correlation Coefficients for the second experiment with *Jaccard* Score

	EN	RU	RO	SMOTE
Jaccard Score	-0.473	1.052	0.955	0.637
Data dependency (def)	0.0002	0.003	0.004	0.002
Data dependency (use)	0.008	0.009	0.013	0.015
Change	1.518	1.25	1.793	2.153
Test case Weight	0.055	0.04	0.038	0.044

Tables 7, 8, and 9 confirm the earlier claim. As evident in these tables, “SBFL score” and "Change" have the most impact on locating faulty statements, with "Test Case Weight" ranked as the third most important feature. This indicates that the basic idea involving three features in the first experiment comprises the most critical elements for fault localization.

Regarding data dependencies, "Data Dependency (*use*)" is significantly more impactful than "Data Dependency (*def*)." This distinction arises from how faults manifest during program execution. Even if a value is improperly defined within a statement, its impact remains dormant until the value is utilized. Thus, faults only become observable when the program attempts to use the incorrect value, making "Data Dependency (*use*)" more critical for fault identification and localization. This assertion aligns with the findings of [30]. The following three tables focus on the third experiment, which incorporates control dependency alongside all features from the second experiment.

Table 10. Logistic Regression Correlation Coefficients for the third experiment with *Tarantula* Score

	EN	RU	RO	SMOTE
Tarantula Score	1.743	1.961	3.12	3.382
Data dependency (def)	0.001	0.005	0.003	0.005
Data dependency (use)	0.01	0.026	0.019	0.021
Control dependency (dom)	-0.002	-0.012	-0.009	-0.018
Control dependency (domed)	-0.009	-0.023	-0.013	0.005
Change	1.632	0.955	1.732	2.098
Test case Weight	0.036	0.041	0.039	0.042

Table 11. Logistic Regression Correlation Coefficients for the third experiment with *Ochiai* Score

	EN	RU	RO	SMOTE
Ochiai Score	1.168	1.306	2.21	2.088
Data dependency (def)	0.0009	0.008	0.004	0.003
Data dependency (use)	0.009	0.029	0.017	0.017
Control dependency (dom)	-0.003	-0.012	-0.009	-0.014
Control dependency (domed)	-0.01	-0.028	-0.017	-0.017
Change	1.728	1.606	2.003	2.321
Test case Weight	0.026	0.044	0.018	0.022

Table 12. Logistic Regression Correlation Coefficients for the third experiment with *Jaccard* Score

	EN	RU	RO	SMOTE
Jaccard Score	0.628	0.797	1.155	0.889
Data dependency (def)	0.001	0.002	0.003	0.003
Data dependency (use)	0.009	0.026	0.019	0.214
Control dependency (dom)	-0.002	-0.012	-0.01	-0.013
Control dependency (domed)	-0.008	-0.028	-0.013	-0.013
Change	1.729	1.757	1.829	2.299
Test case Weight	0.035	0.05	0.038	0.048

As seen in Tables 10, 11, and 12, the top three most important features are "Change," "SBFL score," and "Test Case Weight". Similar to the findings of the second experiment, "Data Dependency (*use*)" is more critical than "Data Dependency (*def*)." Conversely, "Control Dependency (*dom*)" appears to hold greater significance than "Control Dependency (*domed*)." This observation underscores the more decisive influence exerted by dominating statements on the program's control flow. It makes them a more pivotal factor in guiding the fault localization process. By identifying dominating statements, the approach gains a clearer understanding of critical control points within the code, which are instrumental in pinpointing faults more effectively. Comparing the two dependencies, data dependency is slightly more influential than control dependency based on their correlation with faulty statements.

Synthesizing the results of these three experiments leads to the answer to the first research question. The correlation coefficient of "Change" with faulty statements is among the highest observed. It highlights its far greater importance compared to "Test Case Weight" and the dependencies. This indicates that code modifications serve as strong indicators for fault localization in regression fault scenarios. To further support this claim, Tables 13, 14, and 15 illustrate the accuracy of the formulae generated in the first experiment. Additionally, Table 16 presents the accuracy of three SBFL formulae—*Tarantula*, *Ochiai*, and *Jaccard*. This assessment focuses on the proportion of programs within the experimental dataset where the formula effectively identified the fault location, as measured by the selected evaluation criteria. Consequently, these tables showcase *Top-1*, *Top-3*, and *Top-5* accuracy. It also demonstrates *EXAM* scores computed for thresholds of 5, 10, 15, and 20, utilizing the aforementioned sampling methods.

Table 13. Results of *SBCT*- First experiment with *Tarantula*

	Top-1	Top-3	Top-5	Exam<5	Exam<10	Exam<15	Exam<20	Max EXAM	Min EXAM
SBCT + EN	**27.28%**	**59.1%**	**77.28%**	**27.28%**	**59.1%**	**68.19%**	**77.28%**	**69**	**2.5**
SBCT + RU	27.28%	59.1%	77.28%	27.28%	59.1%	68.19%	77.28%	69	2.5
SBCT + RO	27.28%	59.1%	72.73%	27.28%	59.1%	63.64%	72.73%	69	2.5
SBCT + SMOTE	27.28%	59.1%	72.73%	27.28%	59.1%	63.64%	72.73%	69	2.5

Table 14. Results of *SBCT*- First experiment with *Ochiai*

	Top-1	Top-3	Top-5	Exam<5	Exam<10	Exam<15	Exam<20	Max EXAM	Min EXAM
SBCT + EN	36.37%	68.19%	72.73%	36.37 %	63.64%	68.19%	72.73%	62	2.5
SBCT + RU	36.37%	68.19%	72.73%	36.37 %	63.64%	68.19%	72.73%	62	2.5
SBCT + RO	**36.37%**	**68.19%**	**72.73%**	**36.37 %**	**63.64%**	**72.73%**	**72.73%**	**62**	**2.5**
SBCT + SMOTE	36.37%	68.19%	72.73%	36.37 %	63.64%	72.73%	72.73%	62	2.5

Table 15. Results of *SBCT*- First experiment with *Jaccard*

	Top-1	Top-3	Top-5	Exam<5	Exam<10	Exam<15	Exam<20	Max EXAM	Min EXAM
SBCT + EN	**36.37%**	**68.19%**	**72.73%**	**36.37 %**	**63.64%**	**68.19%**	**72.73%**	**69**	**2.5**
SBCT + RU	36.37%	68.19%	72.73%	36.37 %	63.64%	68.19%	72.73%	62	2.5
SBCT + RO	36.37%	68.19%	72.73%	36.37 %	63.64%	68.19%	72.73%	62	2.5
SBCT + SMOTE	36.37%	68.19%	72.73%	36.37 %	63.64%	68.19%	72.73%	62	2.5

Table 16. Results of the SBFL formulae

	Top-1	Top-3	Top-5	Exam<5	Exam<10	Exam<15	Exam<20	Max EXAM	Min EXAM
Tarantula	9.1%	22.73%	45.46%	9.1%	18.19%	22.73%	45.46%	69	2.5
Jaccard	**27.28%**	**40.91%**	**50%**	**27.28%**	**36.37%**	**50%**	**50%**	**62**	**2.5**
Ochiai	4.55%	18.19%	27.28%	4.55%	13.64%	27.28%	27.28%	71	2.5

Starting with *Tarantula*, as shown in Tables 13 and 16, *SBCT* located 18% more faults by *Top-1* accuracy, 36% more faults by *Top-3* accuracy, and 32% more faults by *Top-5* accuracy. Comparing the *EXAM* scores, the enhanced *Tarantula* outperformed the original *Tarantula* by locating 18% more faults at the maximum *EXAM* of 5%, 41% more faults at an *EXAM* of 10%, 45% more faults at an *EXAM* of 15%, and 32% more faults at an *EXAM* of 20%.

Based on Tables 14 and 16, the results for the enhanced *Ochiai* compared to the original *Ochiai* indicate that *SBCT* locates 32%, 50%, and 45% more faults at the *Top-1*, *Top-3*, and *Top-5* ranks, respectively. In terms of *EXAM* scores, *SBCT* outperformed *Ochiai* by locating 32% more faults by examining a maximum of 5% of the code, 50% more faults by examining a maximum of 10%, and 45% more faults by studying up to 15% and 20% of the code.

Regarding *Jaccard*, comparing the results from Tables 15 and 16 demonstrates that the enhanced *Jaccard* outperformed the original *Jaccard* by locating 9% more faults at *Top-1*, 27% more faults at *Top-3*, and 23% more faults at *Top-5*. Furthermore, *SBCT* located 9% more faults at the maximum *EXAM* of 5%, 27% more faults at the maximum *EXAM* of 10%, 18% more faults at the maximum *EXAM* of 15%, and 23% more faults by examining a *maximum of 20% of the code.*

These findings demonstrate that SBFL can be significantly improved in accuracy by incorporating code modifications. Consequently, these three tables strengthen the claim in response to the first research question. They also indicate that code changes serve as effective clues for locating faulty statements.

Tables 17 to 22 present the results of the evaluation metrics, consistent with Tables 13, 14, and 15, for the second and third experiments. To clarify, Tables 17, 18, and 19 show the results of the experiment incorporating data dependency in addition to the basic features.

Table 17. Results of *SBCT*- Second experiment with *Tarantula*

	Top-1	Top-3	Top-5	Exam<5	Exam<10	Exam<15	Exam<20	Max EXAM	Min EXAM
SBCT + EN	36.37%	81.82%	90.91%	40.91%	72.73%	86.37%	90.91%	31	2.5
SBCT + RU	27.28%	77.28%	90.91%	31.82%	72.73%	81.82%	90.91%	41	3
SBCT + RO	**36.37%**	**81.82%**	**90.91%**	**45.46%**	**68.19%**	**86.37%**	**90.91%**	**28**	**2.5**
SBCT + SMOTE	36.37%	81.82%	90.91%	40.91%	72.73%	86.37%	90.91%	28	2.5

Table 18. Results of *SBCT*- Second experiment with *Ochiai*

	Top-1	Top-3	Top-5	Exam<5	Exam<10	Exam<15	Exam<20	Max EXAM	Min EXAM
SBCT + EN	**40.91%**	**81.82%**	**95.46%**	**45.46%**	**77.28%**	**90.91%**	**95.46%**	**21**	**2.5**
SBCT + RU	40.91%	90.91%	95.46%	40.91%	72.73%	90.91%	95.46%	21	2.5
SBCT + RO	36.37%	90.91%	95.46%	40.91%	72.73%	95.46%	95.46%	24	2.5
SBCT + SMOTE	36.37%	90.91%	95.46%	40.91%	72.73%	90.91%	95.46%	21	2.5

Table 19. Results of *SBCT*- Second experiment with *Jaccard*

	Top-1	Top-3	Top-5	Exam<5	Exam<10	Exam<15	Exam<20	Max EXAM	Min EXAM
SBCT + EN	36.37%	86.37%	95.46%	40.91%	68.19%	86.37%	95.46%	28	2.5
SBCT + RU	31.82%	81.82%	95.46%	36.37 %	68.19%	90.91%	95.46%	31	2.5
SBCT + RO	36.37%	81.82%	95.46%	40.91%	68.19%	90.91%	95.46%	31	2.5
SBCT + SMOTE	**36.37%**	**86.37%**	**95.46%**	**40.91%**	**68.19%**	**90.91%**	**95.46%**	**21**	**2.5**

Considering these three tables, all indicate a significant improvement in the accuracy of *SBCT* compared to the SBFL formulae. Specifically, the enhanced *Tarantula* (Table 17), incorporating "Change," "Test Case Weight," and "data dependency," located 27%, 60%, and 45% more faults than *Tarantula* (Table 16) at *Top-1*, *Top-3*, and *Top-5* rankings, respectively. Similarly, it outperformed *Tarantula* by locating 36%, 50%, 64%, and 45% more faults by examining a maximum of 5%, 10%, 15%, and 20% of the code, respectively. Additionally, the "Max EXAM" column in the tables refers to the maximum *EXAM* score achieved during the experiment. The enhanced *Tarantula* located all faults by examining a maximum of 28% of the code, whereas the original *Tarantula* required analysis of up to 69% of the code.

Regarding *Ochiai*, the enhanced *Ochiai* in the second experiment (Table 18) outperformed the original *Ochiai* (Table 16) by locating 36% more faults at *Top-1*, 64% more at *Top-3*, and 68% at *Top-5* rankings. Moreover, the enhanced *Ochiai* located 41%, 64%, 64%, and 68% more faults by examining a maximum of 5%, 10%, 15%, and 20% of the code, respectively. The

enhanced *Ochiai* identified all faults by examining up to 21% of the code, while the original *Ochiai* required a maximum examination of 71%.

In terms of *Jaccard*, the enhanced *Jaccard* (Table 19) located 9%, 45%, and 45% more faults at *Top-1*, *Top-3*, and *Top-5* rankings, respectively. It further outperformed the original *Jaccard* (Table 16) by identifying 14% more faults by examining a maximum of 5% of the code, 32% more faults at 10%, 41% more faults at 15%, and 45% more faults at 20%. The enhanced *Jaccard* located all faults by analyzing up to 21% of the code, whereas the original *Jaccard* required examination of 62%.

This experiment underscores the importance of data dependency in fault localization. The results from the second experiment also demonstrate higher accuracy compared to the first. As evident from this experiment and two prior studies [30, 31], data dependency proves to be an effective distinguishing feature for differentiating statements in the code. Similar to the second experiment, Tables 20 through 22 focus on the third experiment, which incorporates the control dependency feature alongside the features from the second experiment. This inclusion highlights the impact of control dependency on the results, providing a comparative perspective between experiments.

Table 20. Results of *SBCT*- Third experiment with *Tarantula*

	Top-1	Top-3	Top-5	Exam<5	Exam<10	Exam<15	Exam<20	Max EXAM	Min EXAM
SBCT + EN	**40.91%**	**81.82%**	**90.91%**	**45.46%**	**68.19%**	**86.37%**	**90.91%**	**41**	**2.5**
SBCT + RU	45.46%	59.1%	77.28%	45.46%	54.55%	63.64%	86.37%	41	3
SBCT + RO	40.91%	81.82%	90.91%	45.46%	63.64%	86.37%	90.91%	34	2.5
SBCT + SMOTE	40.91%	77.28%	90.91%	45.46%	68.19%	81.82%	90.91%	48	2.5

Table 21. Results of *SBCT*- Third experiment with *Ochiai*

	Top-1	Top-3	Top-5	Exam<5	Exam<10	Exam<15	Exam<20	Max EXAM	Min EXAM
SBCT + EN	**40.91%**	**77.28%**	**90.91%**	**45.46%**	**68.19%**	**90.91%**	**90.91%**	**41**	**2.5**
SBCT + RU	40.91%	59.1%	81.82%	40.91%	50%	72.73%	81.82%	45	2.5
SBCT + RO	36.37%	77.28%	90.91%	40.91%	63.64%	86.37%	90.91%	48	2.5
SBCT + SMOTE	40.91%	72.73%	90.91%	45.46%	68.19%	81.82%	90.91%	48	2.5

Table 22. Results of *SBCT*- Third experiment with *Jaccard*

	Top-1	Top-3	Top-5	Exam<5	Exam<10	Exam<15	Exam<20	Max EXAM	Min EXAM
SBCT + EN	**40.91%**	**77.28%**	**95.46%**	**45.46%**	**68.19%**	**86.37%**	**95.46%**	**41**	**2.5**
SBCT + RU	36.37%	72.73%	90.91%	40.91%	68.19%	77.28%	90.91%	38	2.5
SBCT + RO	36.37%	81.82%	90.91%	40.91%	68.19%	86.37%	90.91%	38	2.5
SBCT + SMOTE	36.37%	59.1%	81.82%	36.37%	45.46%	68.19%	81.82%	77	2.5

The data from these three tables demonstrates a substantial improvement in the accuracy of *SBCT* compared to traditional SBFL formulae. Specifically, the enhanced *Tarantula* in Table 20 identified 32%, 59%, and 45% more faults than the original *Tarantula* (Table 16) at *Top-1*, *Top-3*, and *Top-5* rankings, respectively. Furthermore, it exceeded the original *Tarantula* by detecting 36%, 50%, 64%, and 45% more faults when analyzing a maximum of 5%, 10%, 15%, and 20% of the code, respectively. Regarding the maximum *EXAM* score, the enhanced *Tarantula* uncovered all faults by inspecting a maximum of 41% of the code, whereas the original *Tarantula* required up to 69% of the code to achieve the same result.

Focusing on *Ochiai*, the enhanced version in Table 21 outperformed the original (Table 16) in the second experiment by identifying 36% more faults at *Top-1*, 59% more at *Top-3*, and 64% more at *Top-5* rankings. Additionally, it surpassed the original *Ochiai* by locating 41%, 54%, 64%, and 64% more faults after examining a maximum of 5%, 10%, 15%, and 20% of the code, respectively. The enhanced *Ochiai* successfully located all faults by reviewing a maximum of 41% of the code, whereas the original required inspecting up to 71%.

As for *Jaccard*, the enhanced version in Table 22 identified 14%, 36%, and 45% more faults at *Top-1*, *Top-3*, and *Top-5* rankings, respectively. It further surpassed the original *Jaccard* (Table 16) by locating 18%, 32%, 36%, and 45% more faults within a maximum of 5%, 10%, 15%, and 20% of the code, respectively. Notably, the enhanced *Jaccard* achieved complete fault detection after analyzing just 41% of the code, while the original version required inspection of up to 62%.

To summarize the results, "Change" leads to a significant improvement in the accuracy of SBFL. Moreover, these findings highlight the critical role of data dependency in fault localization, as the second experiment achieved markedly higher accuracy than both the first and third experiments. The improvements across all experiments demonstrate the enhanced method's greater efficiency in fault localization by substantially reducing the portion of code that needs to be inspected.

To address the second research question, all three experiments with three different SBFL formulae indicate that the SBFL formula functions more effectively as a feature for fault localization than as a standalone formula for locating faults. Regarding the third research question, the second experiment produced higher accuracy and better performance compared to the first and third experiments. In the third experiment, although Control Dependency improved the basic SBFL formulae, its impact was not as significant as the effect of Data Dependency. Therefore, we conclude that achieving high accuracy in fault localization requires

extracting coverage information, incorporating data dependency, and considering code modifications. These three features and the associated calculations to derive feature values prove to be the most efficient approach for fault localization.

18. CONCLUSION

Fault localization is an integral yet labor-intensive phase of software development. To streamline this process without sacrificing accuracy, various methodologies have been developed, including spectrum-based, mutation-based, and learning-based approaches. Among these, spectrum-based fault localization (SBFL) is particularly notable for its simplicity and reasonable accuracy. However, SBFL techniques often fall short in specific scenarios. Additionally, while fault localization has been extensively studied, there is a notable gap in research addressing the role of software evolution. To address these gaps, this research focuses on enhancing fault localization by incorporating the effects of software evolution and the impact of test cases on fault proneness, with an emphasis on the evolving nature of software.

The proposed solution integrates four critical features: "SBFL score" as a declarative feature to locate faulty statements, "Test Case Weight" as an effective feature for fault localization using easily gathered data, "Change" as a feature capturing software evolution, and "program dependencies" as a feature to distinguish statements with similar spectra. These features are utilized within a logistic regression-based machine learning framework to derive the proposed formula. Experimental evaluations demonstrate that *SBCT* outperforms three widely used SBFL methods across three different experiments. Based on the best results from the second experiment, *SBCT* achieved *Top-3* success rates of 82% for *Tarantula*, 82% for *Ochiai*, and 86% for *Jaccard*. Moreover, the findings underscore "Change" and "SBFL score" as pivotal features in improving fault localization accuracy.

Future directions could explore the effectiveness of *SBCT* with a larger variety of programs, encompassing diverse attributes such as size and programming language. Additionally, the integration of the “SBFL score” as a feature into other research could be examined to evaluate its impact across various combinations and scenarios. Furthermore, with the advancement of neural networks and the expansion of their use in the field of software testing and debugging, the proposed method can be investigated using neural networks such as GNNs.

19. THREATS TO VALIDITY

Several factors could impact the reliability of our findings. One such factor is the relatively small scale of the benchmark used in this study, both in terms of source codes and the accompanying test suite. To improve this, future research should test the method using larger, more diverse benchmark suites with varying sizes, domains, and types of faults. Additionally, the collection of numerous features individually makes data gathering and preprocessing time-consuming and resource-heavy. Although we have taken measures to minimize these issues, such as applying consistent extraction procedures and automating tasks where possible, additional large-scale validation is needed to further strengthen the reliability of the results.

20. REFERENCES

[1] A. Zakari, S. P. Lee, K. A. Alam, and R. Ahmad, Software fault localisation: a systematic mapping study, IET Software, vol. 13, no. 1, pp. 60–74, Feb. 2019.

[2] Z. Li, H. Wang, and Y. Liu, HMER: A Hybrid Mutation Execution Reduction approach for Mutation-based Fault Localization, Journal of Systems and Software, vol. 168, p. 110661, Oct. 2020.

[3] Y. Yang, F. Deng, Y. Yan, and F. Gao, A Fault Localization Method Based on Conditional Probability. In IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C), Jul. 01, 2019.

[4] M. Wen, J. Chen, Y. Tian, R. Wu, D. Hao, S. Han, and S. C. Cheung, Historical Spectrum Based Fault Localization, IEEE Transactions on Software Engineering, vol. 47, no. 11, pp. 2348–2368, Nov. 2021.

[5] A. Zakari, S. P. Lee, and I. A. Targio Hashem, A Community-Based Fault Isolation Approach for Effective Simultaneous Localization of Faults, IEEE Access, vol. 7, pp. 50012–50030, 2019.

[6] R. Widyasari, G. A. A. Prana, S. A. Haryono, Y. Tian, H. N. Zachiary, and D. Lo, XAI4FL: Enhancing spectrum-based fault localization with explainable artificial intelligence. In Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension, pp. 499-510, 2022.

[7] R. Widyasari, J. W. Ang, T. G. Nguyen, N. Sharma, and D. Lo, Demystifying faulty code: Step-by-step reasoning for explainable fault localization. In 2024 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), pp. 568-579, 2024.

[8] M. Rezaalipour and C. A. Furia, An empirical study of fault localization in Python programs, Empirical Software Engineering, vol. 29, no. 4, 2024.

[9] X. Xiao, Y. Pan, B. Zhang, G. Hu, Q. Li, and R. Lu, ALBFL: A novel neural ranking model for software fault localization via combining static and dynamic features, Information and Software Technology, vol. 139, p. 106653, 2021.

[10] A. Dutta, S. S. Srivastava, S. Godboley, and D. P. Mohapatra, Combi-FL: Neural network and SBFL-based fault localization using mutation analysis, Journal of Computer Languages, vol. 66, p. 101064, 2021.

[11] A. Majd, M. Vahidi-Asl, A. Khalilian, and B. Bagheri, ConsilientSFL: using preferential voting system to generate combinatorial ranking metrics for spectrum-based fault localization, Applied Intelligence, vol. 52, no. 10, pp. 11068–11088, 2022.

[12] Z. Li, Y. Song, G. Gong, S. Zhou, and K. Lv, A multi-technique fusion approach for fault localization in manufacturing software, Journal of Intelligent, and Fuzzy Systems, vol. 38, no. 1, pp. 229–238, 2020.

[13] Z. Cui, M. Jia, X. Chen, L. Zheng, and X. Liu, Improving Software Fault Localization by Combining Spectrum and Mutation, IEEE Access, vol. 8, pp. 172296–172307, 2020.

[14] Y. Li, S. Wang, and T.N. Nguyen, Fault localization to detect co-change fixing locations. In Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 659-671, 2022.

[15] D. Callaghan and B. Fischer, Improving spectrum-based localization of multiple faults by iterative test suite reduction. In Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis, pp. 1445–1457, 2023.
[16] J. A. Jones and M. J. Harrold, Empirical evaluation of the tarantula automatic fault-localization technique. In Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering - ASE '05, 2005.
[17] M. Y. Chen, E. Kiciman, E. Fratkin, A. Fox, and E. Brewer, Pinpoint: problem determination in large, dynamic Internet services. In Proceedings International Conference on Dependable Systems and Networks, pp. 595-604, 2002.
[18] R. Abreu, P. Zoeteweij, and A. J. c. Van Gemund, An Evaluation of Similarity Coefficients for Software Fault Localization. In 2006 12th Pacific Rim International Symposium on Dependable Computing (PRDC'06), pp. 39-46, 2006.
[19] W. E. Wong, V. Debroy, R. Gao, and Y. Li, The DStar Method for Effective Software Fault Localization, IEEE Transactions on Reliability, vol. 63, no. 1, pp. 290–308, 2014.
[20] Y. Li, S. Wang, and T. Nguyen, Fault Localization with Code Coverage Representation Learning. In 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE), pp. 661-673, 2021.
[21] Y. Küçük, T. A. D. Henderson, and A. Podgurski, Improving Fault Localization by Integrating Value and Predicate Based Causal Inference Techniques. In 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE), pp. 649-660, 2021.
[22] X. Xu, C. Zou, and J. Xue, Every Mutation Should Be Rewarded: Boosting Fault Localization with Mutated Predicates. In 2020 IEEE International Conference on Software Maintenance and Evolution (ICSME), pp. 196-207, 2020.
[23] L. Zhang, Z. Li, Y. Feng, Z. Zhang, W. K. Chan, J. Zhang, and Y. Zhou, Improving Fault-Localization Accuracy by Referencing Debugging History to Alleviate Structure Bias in Code Suspiciousness, IEEE Transactions on Reliability, vol. 69, no. 3, pp. 1021–1049, Sep. 2020.
[24] J. Sohn and S. Yoo, FLUCCS: using code and change metrics to improve fault localization. In Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis, pp. 273-283, 2017.
[25] Y. Lou, Q. Zhu, J. Dong, X. Li, Z. Sun, D. Hao, L. Zhang, and Li. Zhang, Boosting coverage-based fault localization via graph-based representation learning. In Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp. 664-676, 2021.
[26] X. Liu, Y. Liu, Z. Li, and R. Zhao, Fault Classification Oriented Spectrum Based Fault Localization. In 2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC), Vol. 1, pp. 256-261, 2017.
[27] H. Zhong and H. Mei, Learning a graph-based classifier for fault localization, Science China Information Sciences, vol. 63, no. 6, 2020.
[28] D. Ghosh and J. Singh, Spectrum-based multi-fault localization using Chaotic Genetic Algorithm, Information and Software Technology, vol. 133, p. 106512, 2021.
[29] A. Maru, A. Dutta, K. V. Kumar, and D. P. Mohapatra, Software fault localization using BP neural network based on function and branch coverage, Evolutionary Intelligence, vol. 14, no. 1, pp. 87–104, 2019.
[30] F. Aghazade-Par and M. Vahidi-Asl, Feature-Based Fault Localization in Evolving Software: Leveraging Regression Testing Insights, IEEE Access, Vol. 13, pp. 147369 – 147382, 2025.
[31] F. Aghazade-Par and M. Vahidi-Asl, EAFL: an effective combination of features for fault localization in evolving programs, Software Quality Journal, Vol. 33, no. 34, pp. 1-22, 2025.
[32] A. Majd, M. Vahidi-Asl, A. Khalilian, A. Baraani-Dastjerdi, and B. Zamani, Code4Bench: A multidimensional benchmark of Codeforces data for different program analysis techniques, *Journal of Computer Languages*, Vol. 53, pp. 38–52, 2019.
[33] E. Wong, T. Wei, Y. Qi, and L. Zhao, A crosstab-based statistical method for effective fault localization. In 2008 1st international conference on software testing, verification, and validation, pp. 42–51, 2008.
[34] D. Ginelli, O. Riganelli, D. Micucci, and L. Mariani, Exception-Driven Fault Localization for Automated Program Repair. In 2021 IEEE 21st International Conference on Software Quality, Reliability and Security (QRS) pp. 598-607, 2021.
[35] A. Dutta, S. S. Srivastava, S. Godboley, and D. P. Mohapatra, Combi-FL: Neural network and SBFL based fault localization using mutation analysis, Journal of Computer Languages, 66, 101064, 2021.
[36] Z. Peng, X. Xiao, G. Hu, A. Kumar Sangaiah, M. Atiquzzaman, and S. Xia, ABFL: An autoencoder based practical approach for software fault localization, Information Sciences, vol. 510, pp. 108–121, 2020.
[37] H. He, J. Ren, G. Zhao, and H. He, Enhancing Spectrum-Based Fault Localization Using Fault Influence Propagation, IEEE Access, vol. 8, pp. 18497–18513, 2020.
[38] D. Silva-Junior, P. S. Leitao-Junior, A. Dantas, C. G. Camilo-Junior, and R. Harrison, Data-flow-based evolutionary fault localization. In Proceedings of the 35th Annual ACM Symposium on Applied Computing, pp. 1963-1970, 2020.
[39] J. Sohn, G. An, J. Hong, D. Hwang, and S. Yoo, Assisting Bug Report Assignment Using Automated Fault Localisation: An Industrial Case Study. In 2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST), pp. 284-294, 2021.
[40] Y. Lou, A. Ghanbari, X. Li, L. Zhang, H. Zhang, D. Hao, and L. Zhang, Can automated program repair refine fault localization? a unified debugging approach. In Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis, pp. 75-87, 2020.
[41] R. Widyasari, G. A. A. Prana, S. A. Haryono, S. Wang, and D. Lo, Real world projects, real faults: evaluating spectrum based fault localization techniques on Python projects, Empirical Software Engineering, Vol. 27, no. 6, 147, 2022.

FAEZE AGHAZADE-PAR received the M.S. degree in information technology engineering from Shahid Beheshti University, Tehran, Iran, where she is currently pursuing the Ph.D. degree in software engineering. Her research interests include software debugging and gamification.
Application ID: JICSE-2511-1096 (R1)
ORCID: 0009-0003-5530-1310
Email: f_aghazadepar@sbu.ac.ir
Address: Shahid Beheshti University, Tehran 1983969411, Iran

MOJTABA VAHIDI-ASL received the B.S. degree in computer engineering from the Amirkabir University of Technology and the M.S. and Ph.D. degrees in software engineering from Iran University of Science and Technology. He is currently an Assistant Professor with the Faculty of Computer Science and Engineering, Shahid Beheshti University, Tehran, Iran. His research interests include program analysis, software testing, and debugging.
Application ID: JICSE-2511-1096 (R1)
ORCID: 0000-0003-4964-992X
Email: mo_vahidi@sbu.ac.ir
Address: Shahid Beheshti University, Tehran 1983969411, Iran

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

An Efficient Encoding Mechanism for Digital Microfluidic-based DNA Data Storage Systems

M. Pourasadollah, **CODE ORCID**:
Faculty of Computer Science and Engineering, Shahid Beheshti University
A.Jahanian, ✉, **CODE ORCID**:
Faculty of Computer Science and Engineering, Shahid Beheshti University
M. Taajobian, **CODE ORCID**:
Department of Computer Engineering, Mahdishahr Branch, Islamic Azad University

ABSTRACT

Digital data production speeds up dramatically every day, and the need to store extensive data rises consequently. Many storage media are used today, but only a few are capable of the required features in terms of high density, data duration, storage lifetime, and reliability. DNA-based digital data storage systems are an emerging and high-capacity medium, capable of storing vast amounts of data in a very small volume with an exceptionally long lifespan. However, building dense and straightforward DNA storage comes with some challenges. Using manual methods to transfer and mix liquids containing DNA molecules makes this storage more error-prone. Moreover, the encoding method of digital data into DNA molecules is very crucial. This paper proposes a novel method for automatic and controllable DNA strand manipulation using a customized Digital Microfluidic Biochip (DMFB) corresponding to the CAD algorithm. Then, we offer a new approach to encoding the digital data into DNA molecules that helps error detection and correction in DNA storage while it brings efficient encoding density. Experimental results show that we can write 1.6 bits on each nucleotide with a 200-base strand which is a good density compared with other methods. In contrast, the encoding method is much simpler and easier to implement.

Keywords: DNA storage, Microfluidic, Data encoding.

1. Introduction

The intense growth rate of digital data production and lack of enough space in current storage media has forced the industry to look for newer storage technologies. A quick look at the current data-store procedures shows that the continuation of using current storage media will be faced with severe problems soon [1]. DNA-based digital data storage is getting more and more attention due to its wonderful features among the new technologies for data storage [2] and [3]. High data density and very long durability and availability are some of the attractive features of these molecules. Storing more than 109 GB of data only on one cubic millimeter of space is a very critical property for the mentioned demand [4], [5] and [6]. Moreover, recovering data from DNA molecules for thousands of years also reminds us of a durable storage media with no competitors or even close numbers. However, using DNA as a medium for storage faces some challenges [7]. The main challenges of current DNA storage are as follows:

- Low read/write bandwidth due to the chemical reactions.
- Low controllability of the required liquid operating media.
- Considerable fault rate in DNA molecules.
- Considerable cost of DNA synthesis for storing/retrieving data.

Due to the liquid nature of DNA molecules, they need to be kept in a solution. Therefore, working with these molecules requires using tools like pipettes to transfer a specific volume of solution, which means a user must do DNA related operations manually. This process is very time-consuming, error-prone, and costly, with a considerable waste of expensive experimental materials.

Digital Microfluidic Biochip (DMFB) is a promising technology to improve controllability and reduce the consuming materials in liquid assays. We proposed a new DMFB architecture specialized for DNA storage systems and the corresponding CAD algorithm that helps us make an automatic and parallel digital DNA storage system. We used the DNA storage structure of [4] as the base model and customized it on a DMFB to speed up read/write operations in DNA storage with a high degree of automation and parallelism.

The significant level of potential faults in DNA synthesis and reactions is another challenge for DNA storage systems [8]. Moreover, we introduced a lightweight encoding method to improve the fault tolerance of these systems. We demonstrate that digital data encoding can be performed efficiently at the nucleotide level, independently of encodings at higher levels of abstraction.

Submit Date: 2025-06-15
Revise Date: 2026-05-18
Accept Date: 2026-05-26
✉ Corresponding author

This paper is organized as follows. Section 2 describes the basic concepts of digital DNA storage systems, and Section 3 reviews the previous work on DNA storage systems. The proposed DMFB architecture/algorithm is illustrated in detail in Section 4. In Section 5, simulation results are presented and discussed, and finally, Section 6 concludes the paper.

2. Basic Concepts

2.1. Digital Microfluidic Biochips

Microfluidic Biochips are large bio laboratories in the size of an electronic chip that removes the human agents from the bio-experiments. This technology improves the controllability of the assays significantly and also hugely reduces the volume of the costly chemical materials and test samples. The first generation of these chips is based on the continuous flow of liquid inside some microchannels that are etched inside the chip. Thus, a tiny sample gets sucked into the chip and directed to some enzymes through microchannels to reach the final results. This second-generation, called Digital Microfluidic Biochip (DMFB), is built based on the movements of discrete droplets of liquid by electro-wetting that can be of tiny sizes compared with the continuous flow biochips [8].

The attractive features of DMFBs make them a perfect choice for many industrial, commercial, and military applications. Nowadays, they are used as remote healthcare for patients, robotic and artificial intelligent systems. In the military industries, they can play an essential role in detecting biological threats on the battlefield. One of the hottest applications of these chips is in DNA computers and DNA storage systems. DMFBs generally work by moving discrete droplets on a surface using electricity. Droplets are moved on a two-dimensional array of electrodes under an electrical field. The droplets are moved on the chip's surface by applying a specific voltage to each electrode based on a physical process called electro-wetting. A DMFB has four primary operations; MOVE, MIX, MERGE, and SPLIT. The MERGE combines two different droplets to generate a new droplet, and the MIX operation mixes the materials inside a droplet, while SPLIT does the opposite and separates a droplet into two new droplets. Finally, MOVE transports the droplets on the DMFB surface from the source to the target electrode. All these operations get done by applying a voltage to different electrodes and the electrodes responsible for doing a specific operation like MIX, MERGE, and SPLIT are called modules.

The sequence of applying a voltage to specific electrodes that produce primary operations and causes droplets on a chip to pass through some specific path and reach specific spots on the chip is called DMFB architecture. Thus a chip architecture depends on the actual assay mapped into the chip [9]. Figure 1 shows the design flow of an assay on a DMFB. As shown in this figure, the assay gets converted to a sequence graph that determines when and what operation must be done on droplets until the final result. After that, scheduling the assay steps according to the sequencing graph is considered, and resources on the chip for doing primary operations get allocated. Then it is time to determine the placement of each module and, finally, find the electrodes' activation sequence. This way, the assay completely gets mapped into the DMFB.

The output of this flow is a string of electrode activation sequences that must be applied to the DMFB to do the bioassay automatically.

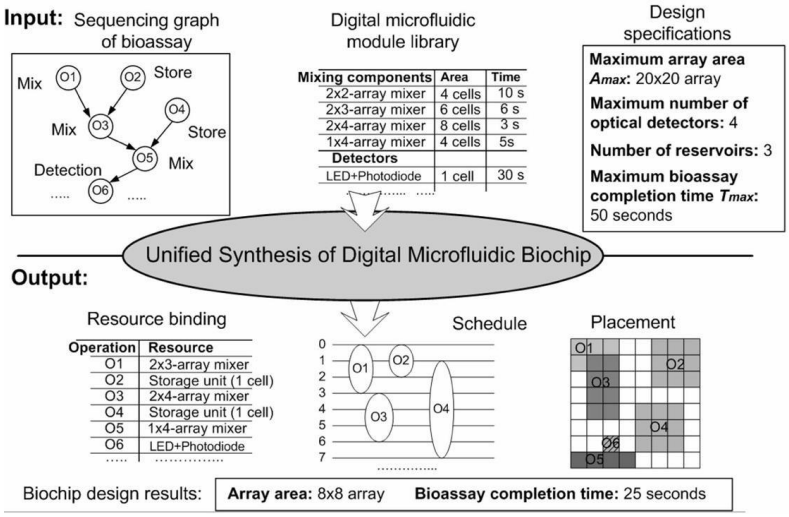


Figure 1: Stages of the DMFB design flow [9]

2.2. DNA-based Digital Data Storage

For Billions of years, DNA molecules have been working in nature to encode the protein generation of living cells and transfer the genetic data between generations of creatures by storing the genetic information of the life on these molecules, similar to digital data storage. DNA molecules include two helix ribbons that are the backbone of these molecules. On each ribbon, many capsules are sitting next to each other while their other head is stuck to another capsule (making it a pair) on the other ribbon. The capsules are called nucleotides, and there are only four types of these nucleotides on a DNA molecule; Adenine (A), Cytosine (C), Guanine (G), and Thymine (T).

Any DNA molecule has many nucleotides, which may differ in count and sequence. The single ribbon in a DNA molecule is called a strand. Generally, the sequence of nucleotides on a strand is shown by the first letter of their name, like "ACCTG" or "GTCATGGCAATG". The nucleotides sequence is just like a sequence of binary data like "00010111010101", but the only difference is that the binary string consists of two values (0 and 1) while the DNA strand consists of four A, C, G, and T [4]. On the other hand, DNA molecules have an exciting feature that nucleotides of one type on one strand only stick to a specific type on the pairing strand; that is, nucleotides of type A only stick to T (and vice versa), and C only sticks to G (and

vice versa). This feature is called reverse complementariness, and with that, if we have only a single strand, we can make its complementary strand with complementary nucleotides [4].

We need to encode the binary data and convert it to a sequence of nucleotides to store binary data on a DNA strand. Then using a device called a synthesizer, this sequence is converted to actual DNA molecules. Then data are retrieved from DNA molecules using another device called a sequencer.

The details of synthesizing and sequencing operations are out of the scope of this paper, but we can see how DNA storages generally work. Usually, a key-value method is used in digital DNA storage to random-access store and retrieve data. This method converts a file with digital contents to a key-value pair. The key part is unique for each file, and the value is the data within that file. The synthesizer converts this key and value to primer and payload, where the primer is a unique sequence of nucleotides for each file and the payload is a sequence of nucleotides representing the value of the data. In current technologies, synthesizing long strands (e.g., more than 200 nucleotides) faces a significant error rate. Therefore, payload data should be split into many strands. The synthesizer puts the primer part of a file to the beginning and ending head of each strand belonging to that specific file together with an index so it would be easy to retrieve all strands belonging to one specific file in the correct order.

To retrieve the data from DNA strands, the sequencer reads strands with similar primers, sorts them according to each strand's index, and then converts it back to binary data after recovering the primary payload DNA strand. The processof reading and writing data from/to DNA strands is shown in Figure 2.

According to Figure 2-A, a sample file named "foo.txt" for a write process in DNA storage gets converted to a key-value combination. Then the key part gets encoded into a primer, and the value part gets encoded into fragments to keep the number of nucleotides of one strand lower than the limit on strand length. Finally, the synthesizer converts the encoded nucleotides to actual DNA strands with the same primer but different payloads.

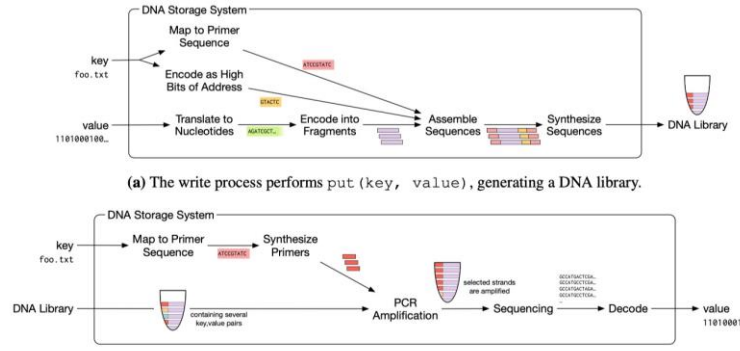


Figure 2: Internal operations of a digital DNA storage system [4]

To simplify recovering original binary data from these strands, an address part is also included inside each strand that helps sort out different fragments to recover original data. The output DNA strands will be stored in a pool that may also contain strands from other files. Figure 2-b shows a read process in which the synthesizer produces the required primer based on the given key (from the "foo.txt" file). Then using PCR, the contents of the pool that contains similar primers to the "foo.txt" file will get amplified and then sent to the sequencer to retrieve the original "foo.txt" file data [4].

2.3. DNA Encoding

There are many encoding methods to convert binary data to DNA nucleotides. Here, we address some parameters that are used to compare different encoding methods to see better improvements made using new methods [10]. The main goals of encoding in digital DNA storage systems are as follows:

- **Better Fault Tolerance:** generating more copies of the same DNA strand is called physical redundancy. When we have more than one copy of a strand, there are more backups in case of an incident or fault that can help recover original data. Having a proper physical redundancy helps recover data in DNA storage when it gets damaged for any reason. It also makes DNA storage more fault-tolerant. Moreover, some nucleotides can be added to a strand responsible for adding fault tolerance in logical redundancy despite physical redundancy. Depending on the encoding method, these extra nucleotides will help recover original data in case of any fault. It is good to keep the number of added nucleotides as low as possible so the ratio of data bits to present nucleotides in a strand (also called logical density) would be high. Also, while using logical redundancy increases the total number of strands the synthesizer needs to generate, the number of final strands is much lower than physical redundancy.
- **Strand length:** Due to some technical limitations that synthesizers have, it is critical to keep the number of nucleotides on each strand below 200 counts. Thus the way we encode data is very important to make a proper logical redundancy while not over-passing strand length limitations.

Encoding methods have been widely used to increase fault tolerance and reduce DNA synthesis costs.

3. Previous Work

DNA-based storage was introduced to store digital information more than ten years ago, and many contributions have been proposed to push this technology forward. In 2016, Seelig et al. [4] proposed a novel and practical method that could store different digital data files in DNA. He showed that it is possible to convert any binary file into a DNA-based key-value pair, so each file has its unique key and related DNA strands. Therefore, it could be possible to differentiate between DNA strands from multiple files. They stored 42 KB of binary data in DNA successfully. In 2017, Seelig et al.

[11] raised their storage capacity to 200 MB. They used their key-value system to store DNA strands in different pools called DNA libraries. In this way, they organized their storage to locate different files in a unified DNA library that simplified access to different strands and made random access possible.

The liquid nature of DNA strands needs the use of devices that can work with liquid and get automated. Micro-array devices that enable automatic liquid transport operations have been available for nearly a decade, and digital microfluidic biochips (DMFB) are the better choice. They are an excellent platform to control and automate the movement of liquid droplets on a glass or chip surface.

In 2001 Chakrabarty et al. [12] used the electro-wetting feature of liquids to move the droplets between different electrodes on two-dimensional surface electrodes. This method provided the ability to perform biological tests on sample droplets by scheduling voltage applications on electrodes. In 2006, authors of [4] described synthesizing and reconfiguring electrodes to shape different paths for droplet movements. They have shown how to schedule basic operations on a chip surface to implement a complete chemical or biological test. In 2016 a new architecture was proposed [13] to overcome the problems of the previous DMFB generations. They use a micro-electrode-dot-array instead of the old two-dimensional array to move the droplets in almost any direction and with different volumes. They have added a sensor underneath each micro-electrode to help detect the droplet's position. Although this method made some improvements, it increased the costs so much. Douglas et al. [14] in 2018 mentioned for the first time that DMFBs could be a good option for manipulating the droplets containing DNA molecules, but they did not propose a practical method.

In 2019, Newman et al. [15] used a DMFB to move droplets to a dried spot on its surface and rotate the droplets. Therefore, they could solve DNA molecules from that spot into the droplet. This approach is attractive, but DNA molecules are generally available in a solution in an actual application, and the proposed method in [15] did not help implement the DNA-based storage.

Along with the computation media of DNA operations, DNA encoding is a critical factor in improving DNA-based storage. In 2012, Church et al. [16] proposed an innovative method for encoding binary data into DNA strands. They used "A" or "C" nucleotide for 0 and "G" or "T" for 1. This way, they always had two nucleotides for each binary digit to make an efficient way for multiple strands of arbitrary binary data to improve the fault-tolerance.

In 2013, Goldman et al. [17] showed that it is possible to convert initial binary data into a string of digits in base three and then convert them into a string of DNA nucleotides. Afterward, DNA strands could get generated with an overlapping pattern, as shown in Figure 3.

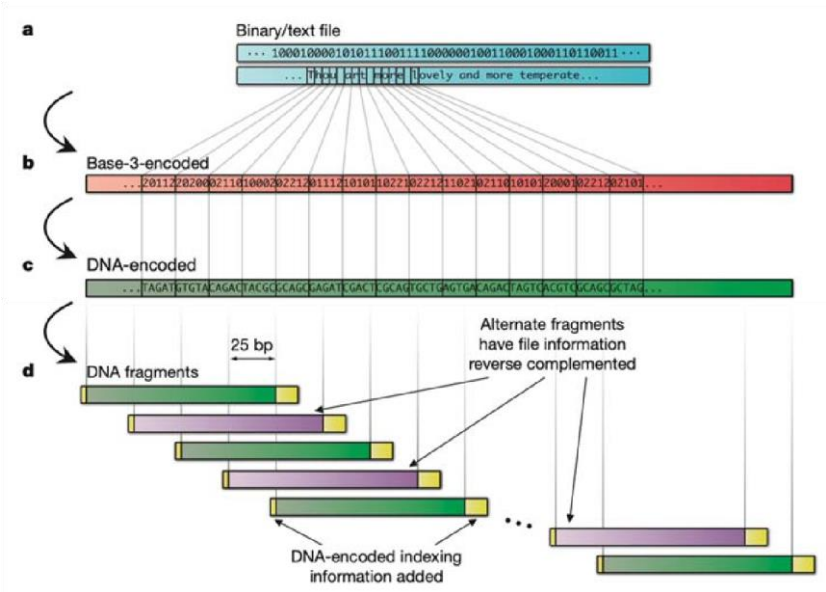


Figure 3: Encoding the digital information into DNA nucleotides using overlapping strands [17]

In this method, the original DNA strand (that is converted from the binary string) is divided into fragments with equal length so that any fragment has 75% nucleotides in common with its previous and next fragment. Moreover, indexing information is added to each fragment so it could be possible to sort them. It is worth noting that this method increases the number of total strands (system cost) in case of finding a faulty or damaged strand.

Another method proposed in 2015 by Grass et al. [18] could encode and protect DNA strands by encapsulating them. Their approach for encoding data first converts every two bytes of primary binary data into three elements of a Galois field of size 47 (Figure 4-A). Then they sorted this data into a block with M rows and N columns as shown in Figure 4-B.

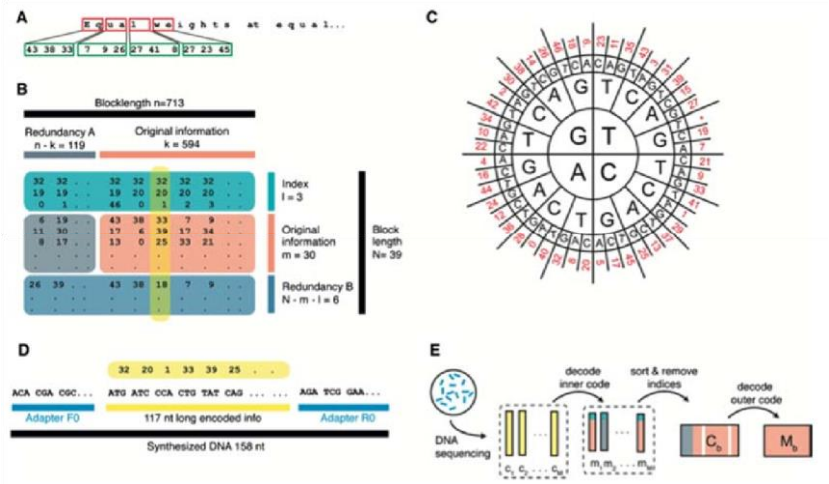
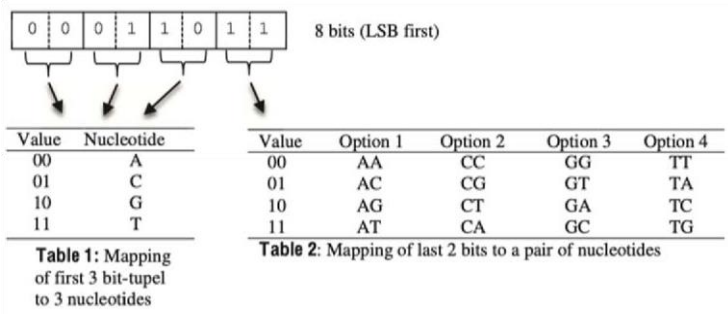


Figure 4: Encoding text into DNA strands using Galois field [18]

In the next step, the authors of [18] used the Reed-Solomon code to add logical redundancy to each element row and the generated elements near each row as inner coding. Then, they added another logical redundancy for each column using RS code and added the generated elements underneath each column (Figure 4-B) as outer coding. Finally, an index is added for every column, and every element gets replaced with a corresponding nucleotide after generating the final block picked up from the Galois wheel (Figure 4-D). The Galois wheel is a mapping table for converting the 47 elements into a three-nucleotide string designed to prevent building homo-polymers (Figure 4-C). Then, the synthesizer converts every column to an actual DNA strand. After sequencing the related strands and generating the primary data block, reversing process results in the primary binary data in order to convert DNA strands back to binary data (Figure 4-E).

In 2016, Blawat et al. [19] offered a new encoding method based on converting each byte of binary data to 5 nucleotides. To do this, they have built two tables, as shown in Figure 5. They used Table 1 for the first six bits of higher value, and Table 2 is used for the 6th and 7th bits (from left). In this way, bits 0 and 1 are mapped to the First nucleotide, 2 and 3 to the second, 4 and 5 to the fourth, and 6 and 7 to the third and fifth.



Value	Nucleotide
00	A
01	C
10	G
11	T

Table 1: Mapping of first 3 bit-tupel to 3 nucleotides

Value	Option 1	Option 2	Option 3	Option 4
00	AA	CC	GG	TT
01	AC	CG	GT	TA
10	AG	CT	GA	TC
11	AT	CA	GC	TG

Table 2: Mapping of last 2 bits to a pair of nucleotides

Figure 5: Mapping tables used for converting binary data into nucleotides [19]

It is worth noting that using the four options in Table 2 prevents some common errors in DNA strand synthesizing. Therefore, there are two rules for choosing the best option from Table 2. Firstly, in a byte conversion to 5 nucleotides, the first three nucleotides (from left) should not be similar, and secondly, the last two nucleotides should not be similar. Using this method, they finally showed that there would be at least 2 DNA strands option for a byte of the binary value. Thus, they have called these clusters A and B to have two clusters to choose the final DNA strands for each byte. They showed that there might be another option available called cluster C, but it is not always available. The ability to choose strands from either cluster A or B (and sometimes C) gives enough freedom to generate better strands and efficiently manage the process. They have also added a parity bit to every strand so that in case of any fault, they can discard it and choose alternatives.

4. The Proposed DMFB Architecture

As described before, two main processes in a DNA-based storage system are reading and writing. In a WRITE process, synthesized DNA strands are moved into the DNA pools, and in the READ process, the liquid from the synthesizer should get mixed with the contents of one or more pools and then sent to sequencers. These processes are done manually using tools in a laboratory, like a pipette in a conventional scenario.

This paper shows that the READ/WRITE process in DNA-based storage can be done using a DMFB efficiently. We propose a novel DMFB architecture that controls the scheduling process to activate signals for different electrodes in a DMFB. We suppose that the synthesizer and sequencer modules are out of our DMFB. Therefore, we need to facilitate the moving of DNA-consisted droplets between the storage pools and these devices with maximum parallelism and controllability.

Figure 6 shows the overall view of the proposed architecture. In this architecture, droplets enter from the bottom part of the chip and exit from the top in the WRITE process or exit from the right in the reading process [20]. As shown in Figure 6, pools are placed at the top of DBMF to minimize the droplet's path, and outputs to the sequencers are at the right part of the chip. Moreover, mixer modules that are activated in a READ process are diagonally arranged so there would be no conflict between droplets to perform parallel readings. The green arrow shows the travel path for droplets in the WRITE process, while the blue arrows show how the reading process is done in two steps. In the WRITE process, the mixer module in the path gets deactivated, as shown in the gray in the picture.
Considering R as the number of rows of the chip and C as the number of columns, and N as the number of droplets, we define S, E, CM, CW, and CR as chip area, the minimum number of electrodes, number of cycles for MIX process, cycles for the WRITE process, and cycles for the READ process respectively. It can be quickly approved that for N number of droplets, there should be at least R=2N rows and C=2N columns to provide enough space for droplets to move across the chip without sticking together. Therefore, the minimum value of chip area (S) can be calculated as Equation 1.

$$S = R \times C = 4 \times N^2 \quad (1)$$

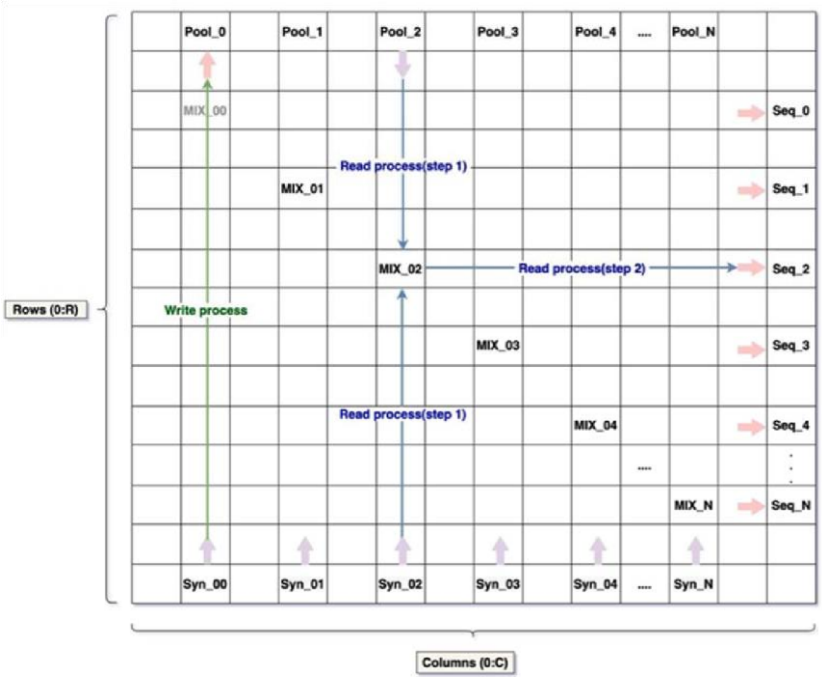


Figure 6: Overview of the proposed architecture

To calculate the number of required electrodes for a complete READ/WRITE process using N droplets, the number of electrodes (E) can be calculated as:

$$E = N \times R + N \times \frac{(C-1+1)}{2} = N \times (R + \frac{C}{2}) \quad (2)$$

By replacing the R and C variables in the above equation, we have:

$$E = N.(2 \times N + 2 \times fracN2) = 3 \times N^2 \quad (3)$$

Therefore, the number of electrodes needed to build this DMFB is in the order of N2, which is an accepted order. Considering BR and BC as blank rows between sequencers and blank columns between synthesizers, the CW and CR will be as Equation 4.

$$C_W = N \times (B_R + 1) \quad (4)$$

Furthermore, CR can be computed as:

$$C_R = N.(B_R + 1) + N \times (B_C + 1) + C_M = N.(B_R + B_C + 2) + C_M \quad (5)$$

It is now proved that the CW and CR have a direct linear relationship with the number of droplets.
The above formulas demonstrate that parallel READ and WRITE processes can be performed efficiently, and the system's complexity is increased linearly when the size of the system grows. It is worth noting that the proposed architecture, with its dynamic size, can be adapted to the size of any parallel process needed to perform DNA storage. The cost overhead will be in a linear relationship to the size of the chip. Here CAD algorithm of droplet routing is described to show how this architecture works. When a WRITE process is supposed to happen, droplets containing DNA molecules get placed at the bottom row of the DMFB. The droplets are in the same column as the corresponding pools at the top of the DMFB. As in a WRITE process, it is only needed to put the droplets into appropriate pools (considering droplets are initially placed in their appropriate place in the same column of the corresponding pool); the control signals will activate the electrodes on the upper row (and same column) of the droplets. Therefore, droplets move one row up. The same process gets repeated until all droplets parallelly reach the pools at the top of the DMFB. In a READ process, droplets from the bottom row (containing required primers) move toward the pools on top. At the same time, sample droplets from pools at the top move toward the bottom row. As stated before, in every column, there

is a Mixer module. Whenever a droplet reaches the Mixer module in the same column, it waits for another droplet to arrive. When both droplets reach the mixer module, it starts to mix these two droplets (for specified time cycles), and then the mixed droplet moves in the right direction toward the appropriate sequencer output. This way, the WRITE, and READ processes are done using our architecture on a DMFB.

5. The Proposed DNA Encoding

As mentioned before, the fault rate of DMFBs is considerable, and this challenge must get noticed in a practical scenario. Many fault tolerance improvement methods in DNA-based computation systems are generally based on encoding the binary data before converting them to DNA strands. Usually, the binary data gets compressed, and then some bits are added to them to increase the fault tolerance of the data in these methods, making it more straightforward to synthesize and recover the data with fewer errors.

The proposed method in this paper differs from this aspect as it is done independently of the type of compression or encoding used for binary data. It can

be added to any data before the synthesis process. In other words, this method works in both binary-encoded data and DNA-encoded data scope.

The base of this method is mapping the logical XOR operations from binary-coded data to nucleotide-coded data. We use a logical operator called nt-XOR, the same logical operations on nucleotide operands. The nt-XOR here is entirely similar to [16] which uses two bits for any nucleotide as Table 1.

Table 1: A possible binary to DNA encoding method

Nucleotide	A	C	G	T
Binary equivalent	00	01	10	11

The truth table for the equivalent binary values of the above table will be as Table 2.

Table 2: The truth table of two-bits XOR in nucleotide-encoded data

	A	**C**	**G**	**T**
A	00	01	10	11
C	01	00	11	10
G	10	11	00	01
T	11	10	01	00

Thus, the truth table for an nt-XOR on nucleotide values will be as Table 3. The important point about the new nt-XOR is that it is directly used for a parity check

Table 3: The truth table of nt-XOR operation on nucleotide values

nt XOR	**A**	**C**	**G**	**T**
A	A	C	G	T
C	C	A	G	T
G	G	T	A	C
T	T	G	C	A

operation for nucleotides. It can detect any change (e.g., fault or error) in the original strands when this parity is violated. However, the parity operations can be done more advanced, like using a parity block that detects the errors and can correct some too. In a parity block, the string of binary data is organized as a two-dimensional array in which a separate parity bit is generated for each row and column. As a result, any change in the value of an element of this array will violate the parity bits in the corresponding row and column, as shown in Table 4. The changed values are highlighted.

Table 4: A parity block on binary values

Row parity = 0	0	1	0	1
Row parity = 1	0	1	1	1
Row parity = 0	1	1	1	1
Row parity = 1	1	0	1	1
	Column parity = 0	**Column parity = 1**	**Column parity = 1**	**Column parity = 0**

Using the above concept, we will implement the parity block on a string of nucleotides. Consider having a string of 64 nucleotides organized as a two-dimensional array as shown in Table 5. In this table, 64 nucleotides are shown that are divided into eight strings with a length of 8 and put each string in each row.

We calculated the nucleotide-based parity for each row and column by applying the nt-XOR on all rows and columns. Therefore, the corresponding nucleotide-based parity for each row and column will be changed with a change

Table 5: A parity block on a string of 64 nucleotides

	Parity Column	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
Byte 0	A	T	G	C	G	A	T	C	A
Byte 1	C	A	G	C	T	G	C	A	G
Byte 2	T	C	T	G	T	T	G	C	A
Byte 3	A	G	C	T	A	A	C	C	A
Byte 4	A	G	T	A	C	C	A	T	G
Byte 5	T	C	G	A	C	G	A	C	G
Byte 6	A	T	T	C	C	G	G	A	A
Byte 7	G	T	A	G	A	C	G	T	T
Parity Row		T	A	G	T	C	C	A	C

on any element of the table which detects the data corruption. In this scheme, data corruption will be detectable and corrected by saving the primary calculated values for each row, column, and parity row using Equation 6.

$$CorrectedV\ alue = CV \oplus NP \oplus OP \quad (6)$$

where $\oplus$ is nt-XOR as described in Table 3.

Where CV is the changed value after data corruption, NP is the newly calculated parity, and OP is the old calculated parity. It can be proved that this method can detect and correct up to two errors.

We build our block using eight rows and 20 columns to implement the above parity block for a DNA strand (w) due to the limitation of a strand length (about 200 nucleotides, as already said [10]). Therefore, it consists of 160 nucleotides for

data and 28 nucleotides for parity (e.g., eight parity nucleotides for eight rows and 20 nucleotides for 20 columns). We can have another nucleotide parity for checking the 28 parity nucleotides. Thus, the total number of nucleotides for the payload part of the strand will be 189, as shown in Table 6.

Table 6: The DNA strand structure of the proposed encoding

Primer(5nt)	Parity 1nt	Parity (28 nt)	Data (160 nt)	Primer(5nt)

Table 6 shows that every strand in our DNA data storage can have this

structure. The synthesizer can produce the parity nucleotides according to the nt-XOR formula. Therefore, we can have a parity block for any binary data.

The above method is entirely independent of the primary encoding of the binary data and is done at the nucleotide level.

5. Simulation Results

The encoding method proposed in this paper improves the fault-tolerant in the cost of logical redundancy due to the extra nucleotides. We calculated its logical density to compare it with other methods. Logical density is the number of bits that can be described with each nucleotide. We assumed that the payload length of the final strand would be 189 nucleotides (e.g., 160 nucleotides for data and 29 nucleotides for parity). Moreover, we have taken that each nucleotide is equal to 2 bits. Therefore, the logical density of the proposed method can be calculated as follows:

$$\frac{(160 \times 2)bits}{189nt} \simeq 1.7bit/nt \quad (7)$$

The calculated value of 1.7 bit/nt shows a great density compared to other encoding methods. Table 7, compares this paper's achievements in comparison with some other methods:

As shown in Table 7, the proposed encoding has a higher logical density than other methods. This improvement decreases the cost and complexity of the storage system.

6. Conclusion

DNA-based data storage is a new technology to overcome the challenges of current storage systems. A major limitation of this technology is working with the

Table 7: The truth table of nt-XOR operation on nucleotide values

Study	Strand length (nucleotides)	Logical density (bits/nt)	Logical density (Payload only)
Church et al. [16]	115	0.60	0.83
Goldman et al. [17]	117	0.19	0.29
Grass et al. [18]	158	0.86	1.16
Bornholt et al. [4]	117	0.57	0.85
Erlich and Zeilinsky [21]	152	1.18	1.55
Blawat et al. [19]	230	0.89	1.08
Organick et al. [22]	150-200	0.81	1.10
This Paper	199	1.60	$1.69 \simeq 1.7$

liquid environment of DNA operations. This paper's proposed digital microfluidic architecture removes the human manual operations with DNA-consisted solutions and provides a parallel and controllable platform for this storage. Aside from a platform for DNA-based storage, this paper proposed a novel method to encode binary data to nucleotides to improve the fault tolerance of the DNA-based system.

In the proposed encoding, adding extra information to the original data to make it fault-tolerant is done at the nucleotide level instead of the binary level. The synthesizers can learn how to calculate the parity nucleotides and help with error detection and correction. The proposed nucleotide-based encoding is independent of the binary data encoding that enables both techniques to be feasible concurrently. Our analyses show that the overhead of this method is also lower than other methods.

It is worth noting that all the resources related to this research are available by sending a message to jahanian@sbu.ac.ir.

7. Conflict of interest

On behalf of all authors, the corresponding author states that the authors have no conflicts to disclose.

References

[1] Y. Dong, F. Sun, Z. Ping, Q. Quyang and L. Qian, DNA storage: research landscape and future prospects, In National Science Review, Vol.7, No.6, June 2020.

[2] Joao Henrique Diniz, Brandao Gervasio, Henrique da Costa Oliveira, Andre Guilherme da Costa Martins, Joao Bosco Pesquero, Bruno Marinaro Verona, Natalia Neto Pereira Cerize, How close are we to storing data in DNA?, Trends in Biotechnology, DOI: 10.1016/j.tibtech.2023.08.001, 2024.

[3] Meng Yu, Xiaohui Tang, Zhenhua Li, Weidong Wang, Shaopeng Wang d, Min Li d, Qiuliyang Yu, Sijia Xie, Xiaolei Zuoand Chang Chen, High-throughput DNA synthesis for data storage, In Chem. Soc. Rev., 2024, 53, 4463-4489.

[4] J. Bornholt, R. Lopez, D. M. Carmean, L. Ceze, G. Seelig, and K. Strauss, "A DNA-based archival storage system," in Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems, 2016, pp. 637-649.

[5] Nirmala Natarajan and Gracia Nirmala Rani Duraisamy , Optimization techniques in digital microfluidic biochips: a survey of sample preparation algorithmic solutions and challenges, In Springer Microfluidics and Nanofluidics, V.29, No. 52, 2025.

[6] Ritwika Majumdar, Piyali Datta, Sagarika Chowdhury and Rajat Kumar Pal, Advancement with digital microfluidic biochips towards sustainability and secured outcome: a comprehensive survey on specific design metrics, Discover Electronics, Vol.1, No.14, 2024.

[7] Y. Cevallos et. al, A brief review on DNA storage, compression, and digitalization, In Nano Communication Networks, Vol.31, March 2022.

[8] K. Chakrabarty and F. Su, Digital microfluidic biochips: synthesis, testing, and reconfiguration techniques. CRC press, 2006.

[9] K. Chakrabarty, "Automated design of microfluidics-based biochips: connecting biochemistry to electronics CAD," in 2006 International Conference on Computer Design, 2006, pp. 93-100.

[10] L. Ceze, J. Nivala, and K. Strauss, "Molecular digital data storage using DNA," Nature Reviews Genetics, vol. 20, no. 8, pp. 456-466, 2019.

[11] J. Bornholt, R. Lopez, D. M. Carmean, L. Ceze, G. Seelig, and K. Strauss, "Toward a DNA-based archival storage system," Ieee Micro, vol. 37, no. 3, pp. 98-104, 2017.

[12] J. Ding, K. Chakrabarty, and R. B. Fair, "Scheduling of microfluidic operations for reconfigurable two-dimensional electro-wetting arrays," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 20, no. 12, pp. 1463-1468, 2001.

[13] Z. Li, K. Y.-T. Lai, P.-H. Yu, T.-Y. Ho, K. Chakrabarty, and C.-Y. Lee, "High-level synthesis for micro-electrode-dot-array digital microfluidic biochips," in Proceedings of the 53rd Annual Design Automation Conference, 2016, pp. 1-6.

[14] D. Carmean, L. Ceze, G. Seelig, K. Stewart, K. Strauss, and M. Willsey, ”DNA data storage and hybrid molecular-electronic computing,” Proceedings of the IEEE, vol. 107, no. 1, pp. 63-72, 2018.
[15] S. Newman et al., ” High density DNA data storage library via dehydration with digital microfluidic retrieval,” Nature communications, vol. 10, no. 1, pp. 1-6, 2019.
[16] G. M. Church, Y. Gao, and S. Kosuri, ”Next-generation digital information storage in DNA,” Science, vol. 337, no. 6102, pp. 1628-1628, 2012.
[17] N. Goldman et al., ”Towards practical, high-capacity, low-maintenance information storage in synthesized DNA,” nature, vol. 494, no. 7435, pp. 77-80, 2013.
[18] R. N. Grass, R. Heckel, M. Puddu, D. Paunescu, and W. J. Stark, ”Robust chemical preservation of digital information on DNA in silica with errorcorrecting codes,” Angewandte Chemie International Edition, vol. 54, no. 8, pp. 2552-2555, 2015.
[19] M. Blawat et al., ”Forward error correction for DNA data storage,” Proce dia Computer Science, vol. 80, pp. 1011-1022, 2016.
[20] M. Pourasadollah, M. Taajobian, and A. Jahanian, ”Flexible and Automatable Microfluidic-based Architecture and CAD Algorithm for Implementation of Large DNA Digital Storage,” in 2022 27th International Computer Conference, Computer Society of Iran (CSICC), Feb. 2022, pp. 1-7. doi: 10.1109/C SICC55295.2022.9780499.
[21] Y. Erlich and D. Zielinski, ”DNA Fountain enables a robust and efficient storage architecture,” Science, vol. 355, no. 6328, pp. 950-954, 2017.
[22] L. Organick et al., ”Random access in large-scale DNA data storage,” Nature biotechnology, vol. 36, no. 3, pp. 242-248, 2018.

Mostafa Pourasadollah received his B.S. degree in computer engineering from Babol Noshirbvani University of technology, Babol, Iran 2011 and the M.S. degrees in computer architecture from Shahid Beheshti University, Tehran, Iran in 2019. He is a PhD. Student at Shahid Beheshti University currently and his current research interests are hardware security and biochip design.

Ali Jahanian received the B.Sc. degree in Computer Engineering from University of Tehran, Tehran, Iran in 1996 and the M.Sc. and Ph.D. degrees in Computer Engineering at Amirkabir University of Technology, Tehran, Iran in 1998 and 2008, respectively. He joined Shahid Beheshti University, Tehran, Iran in 2008. His current research interests are focused on Hardware security and Biochip design.

Dr. Maryam Taajobian completed her B.Sc. in computer engineering at Tehran University and her Master degree in computer architecture at Amirkabir University of Technology. He completed his PhD. in the field of computer systems architecture at the Science and Research Unit of Azad University. He is currently a member of the faculty of Mahdishahr branch of Islamic Azad University and works in the field of digital microfluidic chips and computer networks.

Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

June 2025, Volume 3, Issue 1

A Taxonomy of Blockchain Key Performance Indicators and Measurement Features: A Systematic Review

Kimiya Karimi Dehkordi, CODE ORCID: 0009-0002-7120-4548
Master Student, Department of Computer Engineering, Faculty of Engineering, Shahrekord University, Shahrekord, Iran, kimiya.karimi@stu.sku.ac.ir.
Leila Samimi-Dehkordi✉, CODE ORCID: 0000-0002-2842-0256
Assistant Professor, Department of Computer Engineering, Faculty of Engineering, Shahrekord University, Shahrekord, Iran, samimi@sku.ac.ir.
Abbas Horri, CODE ORCID: 0000-0001-7933-9266
Assistant Professor, Department of Computer Engineering, Faculty of Engineering, Shahrekord University, Shahrekord, Iran, horri@sku.ac.ir.

ABSTRACT
Blockchain technology has revolutionized various industries by enabling decentralized, transparent, and secure systems for applications such as supply chain management, Internet of Things (IoT), and smart contracts. However, evaluating the performance of diverse blockchain systems remains challenging due to the absence of standardized metrics. This systematic review synthesizes findings from 55 peer-reviewed studies to develop a comprehensive taxonomy of Key Performance Indicators (KPIs). Our analysis reveals a critical trade-off: while Proof-of-Work systems (e.g., Bitcoin) offer superior security with hash rates exceeding 500 EH/s, they consume approximately 700 kWh per transaction—a sustainability gap of nearly two orders of magnitude compared to more efficient alternatives. Furthermore, we identify the IoT domain as the most vulnerable to the lack of standardization, where inconsistent KPIs for latency and throughput currently hinder real-time device coordination. We categorize 30 general KPIs into four dimensions—technical, security, economic, and user-centric—while identifying domain-specific KPIs tailored to applications like supply chain logistic, IoT, and smart contracts. Additionally, we map blockchain features required for measuring these KPIs, ensuring traceability to system properties. Derived from a rigorous systematic review of blockchain performance literature, our findings provide a holistic framework for assessing blockchain performance. This framework bridges technical and application- specific needs, offering actionable insights for researchers, developers, and industry practitioners to optimize blockchain systems across public, private, and hybrid architectures. The taxonomy and measurement features support standardized evaluation, paving the way for scalable and sustainable blockchain adoption in emerging domains like decentralized finance.

KEYWORDS
Blockchain, Key Performance Indicators, Systematic Review, Performance Evaluation, Domain-Specific Metrics.

21. INTRODUCTION

Blockchain technology, initially conceptualized by Nakamoto in 2008 [1], has transformed from a foundational element for cryptocurrencies like Bitcoin into a versatile infrastructure that supports decentralized applications across various industries. Its immutable, distributed ledger fosters transparency and trust, facilitating use cases such as supply chain management [2], [3], Internet of Things (IoT) [5], smart contracts [8], and industrial automation [10]. In contrast to traditional centralized systems, blockchain mitigates single points of failure and enhances security through cryptographic mechanisms [5], [12]. However, the diversity of blockchain architectures—public (e.g., Bitcoin, Ethereum [1], [14], private (e.g., Hyperledger Fabric [15]), and hybrid [3]—presents significant challenges for performance evaluation [18], [19].

Performance assessment is essential to ensure that blockchain systems fulfill application-specific requirements, such as high throughput in IoT networks [21], low latency in real-time monitoring [5], and transparency in supply chains [2]. Key Performance Indicators (KPIs) serve as measurable metrics for evaluating efficiency, security, scalability, and user satisfaction [25], [27]. For instance, Transaction Throughput (TPS) quantifies operational capacity [28], while Hash Rate evaluates security in Proof-of-Work (PoW) systems [29]. Economic KPIs, such as Energy Consumption, highlight sustainability concerns [27], and user-centric KPIs, such as User Adoption Rate, indicate system acceptance [25]. Despite their significance, the blockchain literature lacks a unified taxonomy that integrates both general and domain-specific KPIs, along with the system features required for their measurement [18], [27].

Prior studies frequently concentrate on specific aspects, such as the performance of consensus algorithm [27], or the efficiency of smart contract [8], without offering a holistic framework. For example, general KPIs like TPS and Latency are well-documented for public blockchains [18], [28], but domain-specific metrics, such as Delivery Time in supply chains [2], [24] or Access Latency in IoT [5], [34], are less standardized. Furthermore, measuring these KPIs requires specific blockchain features,

Submit Date: 2025-09-08
Revise Date: 2026-05-19
Accept Date: 2026-06-01
✉ Corresponding author

such as transaction timestamps or audit trails [15], [35], which are seldom mapped systematically. This fragmentation hinders cross-domain comparisons and the development of standardized evaluation tools [3], [19].

Beyond traditional domains such as supply chains, IoT, and industrial systems, emerging blockchain applications—including Decentralized Finance (DeFi), Non-Fungible Tokens (NFTs), Web3 platforms, and the metaverse—have gained significant momentum in recent years. Despite the proliferation of blockchain-based Decentralized Applications (DApps) across gaming, finance, and social media, existing performance evaluation frameworks exhibit a pronounced blind spot. Metrics central to DApp ecosystems—such as Daily Active Users (DAU), Transaction Volume (in USD), Retention Rate, and Total Value Locked (TVL)—are conspicuously absent from prior systematic reviews [26]. Similarly, user-centric indicators like Session Duration or Smart Contract Invocation Frequency, which directly inform tokenomics and user engagement models, remain largely overlooked. This omission raises a critical question: who is the intended audience for blockchain KPI frameworks, and why have researchers and practitioners consistently ignored comprehensive metric sets in favor of one or two simplistic indicators? The answer lies in a fundamental disconnect between academic taxonomy and practical application. Infrastructure-focused metrics (e.g., TPS, Hash Rate) dominate the literature because they are easily instrumented at the node level [28], whereas DApp-level metrics often require off-chain analytics or cross-platform data aggregation, which many blockchain systems do not natively support [23], [36]. Consequently, practitioners default to easily accessible metrics, neglecting holistic evaluation. Our work directly addresses this gap by (1) explicitly defining the intended audience for each KPI category (infrastructure engineers, DApp developers, business analysts, and regulators) and (2) explaining why certain metrics are systematically underutilized—thereby transforming a perceived weakness into a structured research agenda.

These domains introduce unique operational and economic requirements, making existing KPI frameworks insufficient. For instance, DeFi platforms demand indicators for Liquidity and Impermanent Loss, while NFTs emphasize Transaction Uniqueness and Market Volatility. Highlighting these domains underscores the need for extending KPI taxonomies to capture performance in rapidly evolving blockchain ecosystems.

To address these gaps, this study conducts a systematic literature review to answer two research questions (RQs):

- RQ1: What KPIs exist in the blockchain literature for evaluating system performance across general and domain-specific contexts?
- RQ2: What blockchain features are required to measure these KPIs effectively?

We analyzed 58 peer-reviewed studies, citing 55 representative ones (25 for general KPIs and 30 for domain-specific KPIs) to develop a comprehensive taxonomy. The general KPIs encompass technical, security, economic, and user-centric dimensions [18], while domain-specific KPIs pertain to applications like supply chain logistics [2], IoT [5], smart contracts [8], and industrial systems [10]. Additionally, we map the blockchain features required for KPI measurement, facilitating practical evaluation across public, private, and hybrid systems [21], [36]. Our contributions include:

- A taxonomy of 30 general KPIs, providing a standardized framework for blockchain evaluation.
- A set of domain-specific KPIs tailored to critical applications, derived from 30 specialist studies.
- A detailed mapping of blockchain features for KPI measurement, ensuring traceability and reproducibility.
- A holistic framework that bridges technical and application-specific performance needs, supporting researchers and practitioners in optimizing blockchain systems.

As shown in Figure 1, the research process begins with a comprehensive review of the literature, proceeds through the screening of relevant sources, and advances to the development of a taxonomy. The identified categories are then used to map key features, culminating in a discussion of the findings and their broader implications.

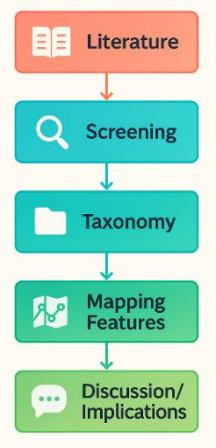


Figure 8. Flowchart of Research Process Stages

This paper is structured as follows: Section 2 describes the review methodology, Section 3 presents the taxonomy and findings, Section 4 analyzes implications and limitations, and Section 5 summarizes contributions and future directions.

22. METHOLOGY

To ensure a rigorous and reproducible review, we adopted the Kitchenham (2007) framework for systematic reviews in software engineering [20]. This framework is particularly well-suited for synthesizing evidence in interdisciplinary fields like blockchain performance evaluation due to its structured approach. Our methodology comprised three key stages: planning the review, conducting the review, and analysis and reporting. This systematic process enabled us to develop a comprehensive taxonomy of blockchain KPIs and their measurement features, addressing both general and domain-specific contexts [18], [19].

22.1. Planning the Review

The planning phase involved defining research objectives, formulating research questions, and establishing the review protocol. Our objective was to identify and categorize blockchain KPIs and their measurement features across general and domain-specific applications. Two research questions (RQs) guided the review:

- RQ1: What KPIs exist in the blockchain literature for evaluating system performance across general and domain-specific contexts?
- RQ2: What blockchain features are required to measure these KPIs effectively?

The review protocol outlined the search strategy, inclusion/exclusion criteria, and data extraction methods. We selected three major databases—Scopus, IEEE Xplore, and Web of Science—for their extensive coverage of computer science, engineering, and blockchain research [3], [20]. The search covered peer-reviewed literature published between 2008 (the inception of blockchain [1]) and April 2024.

To ensure reproducibility, the search strings combined general blockchain terms ("blockchain," "key performance indicators," "performance metrics," "evaluation frameworks") with domain-specific keywords ("supply chain," "Internet of Things (IoT)," "smart contracts," "industrial systems," "consensus algorithms") [10], [27]. Table 1 summarizes the exact keyword combinations and database-specific syntax.

Inclusion criteria: (1) peer-reviewed articles or conference papers, (2) studies explicitly reporting blockchain KPIs or performance metrics, and (3) relevance to either general or domain-specific applications (e.g., supply chain [2], [3], IoT [4], [5], smart contracts [7], [8]).

Exclusion criteria: (1) non-English publications, (2) studies without measurable KPIs, (3) non-peer-reviewed sources (e.g., whitepapers lacking empirical validation), and (4) works unrelated to blockchain performance evaluation.

Table 6. Keyword Combinations and Database-Specific Syntax

Database	Search String (example syntax)
Scopus	TITLE-ABS-KEY("blockchain" AND ("key performance indicator*" OR "performance metric*" OR "evaluation framework*") AND ("supply chain" OR "Internet of Things" OR "IoT" OR "smart contract*" OR "industrial system*" OR "consensus algorithm*"))
IEEE Xplore	("All Metadata": blockchain) AND ("All Metadata": "key performance indicator*" OR "All Metadata": "performance metric*" OR "All Metadata": "evaluation framework*") AND ("All Metadata": "supply chain" OR "All Metadata": "Internet of Things" OR "All Metadata": "IoT" OR "All Metadata": "smart contract*" OR "All Metadata": "industrial system*" OR "All Metadata": "consensus algorithm*")
Web of Science	TS=(blockchain AND ("key performance indicator*" OR "performance metric*" OR "evaluation framework*") AND ("supply chain" OR "Internet of Things" OR "IoT" OR "smart contract*" OR "industrial system*" OR "consensus algorithm*"))

22.2. Conducting the Review

The execution phase involved searching, screening, and selecting relevant studies. After applying the database-specific queries, we initially identified 342 unique records (after duplicate removal). Screening followed a two-phase approach:

1. ***Title and abstract screening:*** applied inclusion/exclusion criteria, reducing the pool to 126 papers.
2. ***Full-text review:*** assessed methodological quality and clarity of KPI definitions, yielding 58 final studies [27].

Of these, 55 were directly cited in the results (25 for general KPIs and 30 for domain-specific KPIs). The remaining three were used only for contextual analysis and KPI validation without direct citation [3].

To enhance transparency, we documented the selection process using a PRISMA flow diagram, which illustrates the number of studies at each stage: identification (n = 342), screening (n = 126), and final inclusion (n = 58). This visual representation supports reproducibility and aligns with best practices for systematic reviews. The PRISMA flow diagram for this systematic review is shown in Figure 2.

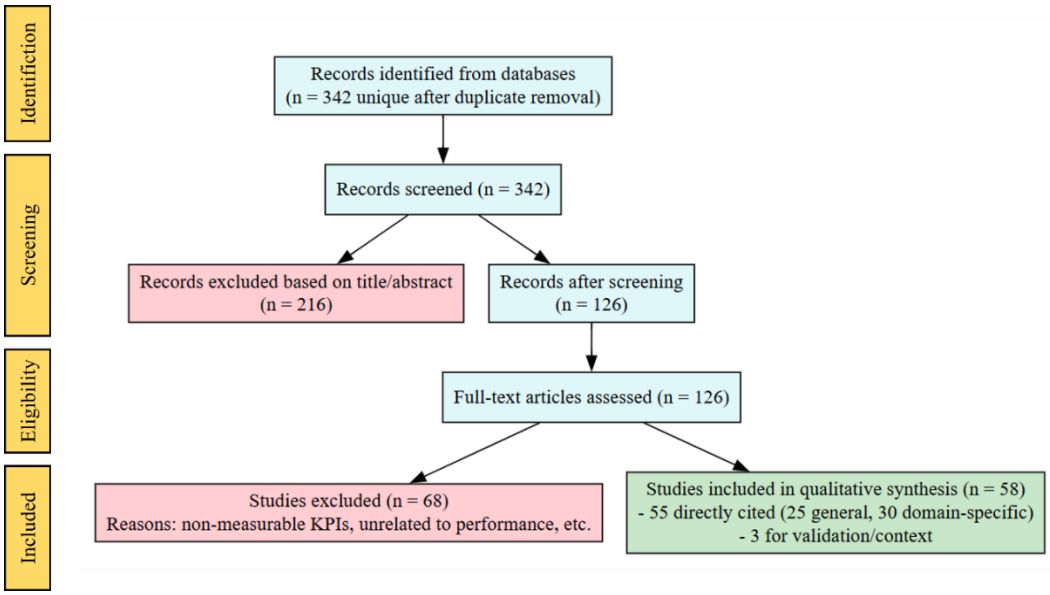


Figure 9. PRISMA Flow Diagram of Systematic Review

22.3. Analysis and Reporting

Data extraction focused on identifying KPIs, their definitions, applicable formulas, and required blockchain features. For general KPIs, we categorized metrics into four dimensions—technical, security, economic, and user- centric—following established frameworks [19], [25]. Examples of these KPIs include Transaction Throughput (TPS) [28], Hash Rate [29], Energy Consumption [30], and User Adoption Rate [25]. For domain-specific KPIs, we extracted metrics from 30 specialist studies, covering applications such as supply chain logistics [3], IoT [5], smart contracts [8], and industrial systems [10]. These metrics included Delivery Time [2], Access Latency [5], and Gas Consumption [41].

We employed both qualitative and quantitative synthesis. Qualitatively, we grouped KPIs by application and blockchain type (public, private, hybrid) to identify patterns and overlaps [15]. Quantitatively, we compiled formulas and measurement approaches to ensure consistency across studies [27]. The extracted data were organized into tables to present general KPIs, domain-specific KPIs, and their measurement features, ensuring traceability and reproducibility [21], [36]. This process resulted in a comprehensive taxonomy addressing RQ1 (identifying KPIs) and RQ2 (mapping measurement features), with findings reported in a structured format suitable for academic and practitioner audiences [20].

23. RESULT

This section presents the findings of our systematic review, addressing RQ1 (identifying blockchain KPIs) and RQ2 (determining measurement features). From an initial pool of 342 papers, 58 met our inclusion criteria. We cited 55 representative studies—25 for general KPIs [25], [36] and 30 for domain-specific KPIs [21], [27]—to ensure comprehensive coverage of unique metrics. The remaining three studies contributed to contextual analysis and KPI validation, enhancing the robustness of our taxonomy [3]. The findings are organized into three subsections: general KPIs, their measurement features, and domain-specific KPIs with corresponding features.

23.1. RQ1: Existing KPIs in Blockchain

Our analysis identified 30 general KPIs, applicable to blockchain systems such as Bitcoin, Ethereum, and Hyperledger Fabric. These KPIs are categorized into four dimensions: technical, security, economic, and user-centric, as shown in Table 2. Each KPI is accompanied by its definition and formula (where applicable), supported by representative studies.

1) Detailed Explanation of General KPIs: The 30 general KPIs in Table 2 provide a robust framework for evaluating blockchain performance across diverse systems. We highlight the most diagnostic KPIs per dimension as below:

- Technical KPIs, such as Transaction Throughput (TPS) and Latency, measure blockchain operational efficiency. TPS quantifies the number of transactions processed per second, while Latency captures the delay between transaction submission and confirmation [1], [28]. Scalability reflects the ability of a blockchain to maintain performance under increasing workload and node participation [18]. Additional indicators, including Block Time, Fork Rate, and Block Propagation Delay, influence responsiveness and network consistency. Shorter block times may reduce latency but increase the probability of temporary forks due to propagation delays [42]. Other KPIs, such as Node Synchronization Time, Chain Growth Rate, and Network Bandwidth Usage, characterize storage, communication, and onboarding overheads in blockchain networks [36], [44], [45]. Collectively, these metrics provide a technical foundation for evaluating blockchain efficiency across public and private architectures.
- Security KPIs evaluate the resilience and integrity of blockchain networks against adversarial behavior. Metrics such as Hash Rate, Attack Cost, and Double-Spending Success Rate quantify the difficulty of compromising consensus mechanisms in public blockchains [12], [42]. Additional indicators, including Orphaned Block Rate, Stale Block Rate, and Consensus Failure Rate, assess network stability and synchronization reliability [42], [47]. Cryptographic Strength and Network Partition Tolerance further characterize the robustness of blockchain protocols under cryptographic or network-level attacks [13], [15]. Together, these KPIs support comparative evaluation of blockchain security across different architectures and consensus mechanisms.
- Economic KPIs assess the financial and sustainability aspects of blockchain systems. Metrics such as Cost per Transaction, Energy Consumption, and Network Operational Cost capture the resource requirements associated with blockchain deployment and maintenance [19], [30]. Additional indicators, including Mining Reward Efficiency and Gas Fee Variability, reflect the economic dynamics of PoW-based and smart-contract platforms [40], [49]. Transaction Value further measures the economic activity supported by blockchain ecosystems [50]. These KPIs enable assessment of operational efficiency and long-term sustainability across blockchain platforms.
- User-Centric KPIs, User-centric KPIs evaluate blockchain adoption and user engagement. Metrics such as User Adoption Rate, Active Address Count, and User Transaction Frequency measure participation and activity levels within blockchain ecosystems [25], [37]. The NVT Ratio relates market capitalization to transaction volume, supporting analysis of economic utilization and network activity [50], [51]. Unlike technical KPIs, user-centric indicators are highly dependent on application context and measurement assumptions. Collectively, these metrics support evaluation of blockchain acceptance and ecosystem growth, particularly in DApp and decentralized finance environments.

By systematically applying the categories and formulas in Table 2, stakeholders can perform holistic evaluations, supporting comparisons across public (e.g., Bitcoin, Ethereum) and private (e.g., Hyperledger Fabric) blockchains [18], [15], [36]. To address the context-dependency concern raised in prior studies, we additionally classify each KPI according to its applicability across blockchain architectures and consensus mechanisms. This distinction clarifies that certain KPIs (e.g., Hash Rate, Mining Reward Efficiency) are specific to PoW-based systems, whereas others remain universally applicable.

Table 7. Comprehensive Overview of Blockchain KPIs

	KPI	Definition	Formula	Applicable To
Technical	Transaction Throughput (TPS)	Transactions processed per second	$TPS = \frac{Total\ Transactions}{Elapsed\ Time\ (seconds)}$ [1], [28]	All Blockchain Types
	Latency	Time from submission to confirmation	$Latency = T_{confirm} - T_{submit}$ [28]	All Blockchain Types
	Block Time	Average time between consecutive blocks	$Block\ Time = \frac{\sum_{i=0}^{n}(T_i - T_{i-1})}{Number\ of\ Blocks}$ [14]	All Blockchain Types
	Scalability	Capacity to handle increased load	$Scalability\ Index = \frac{\Delta TPS}{\Delta Nodes}$ [18]	All Blockchain Types
	Block Propagation Delay	Time for a block to reach all nodes	$Delay = T_{receive} - T_{broadcast}$ [28]	All Blockchain Types
	Fork Rate	Frequency of temporary chain splits	$Fork\ Rate = \frac{Temporary\ Forks}{Total\ Blocks}$ [42]	PoW-based Systems, Some PoS Systems
	Transaction Finality	Time until a transaction is irreversible	$Finality = T_{irreversible} - T_{submit}$ [43]	All Blockchain Types
	Network Bandwidth Usage	Data rate of network traffic	$Bandwidth = \frac{Total\ Data\ Transferred\ (bytes)}{Time\ (seconds)}$ [36]	All Blockchain Types
	Node Synchronization Time	Time to fully sync a new node	$Sync\ Time = T_{full\ sync} - T_{start}$ [44]	All Blockchain Types
	Chain Growth Rate	Rate of blockchain size increase	$Growth\ Rate = \frac{\Delta Size\ (bytes)}{\Delta Time\ (seconds)}$ [45]	All Blockchain Types
Security	Hash Rate	Computational power securing the network	$Hash\ Rate = \Sigma\ Hashes\ per\ Second$ [12]	PoW-based Systems
	51% Attack Cost	Cost to control majority power	$Attack\ Cost = (Hardware\ Cost + Energy\ Cost) \times Attack\ Duration$ [12]	All Blockchain Types (context-dependent)
	Orphaned Block Rate	Percentage of blocks not in main chain	$Orphan\ Rate = \frac{Orphaned\ Blocks}{Total\ Blocks}$ [46]	PoW-based Systems
	Double-Spending Success Rate	Probability of successful double-spend	$P_{DS} = \left(\frac{q}{p}\right)^k, q: attacker\ power, p: network\ power, k: confirmaions$ [42]	Public Blockchains
	Sybil Attack Resistance	Cost to create fake identities	$Cost_{Sybil} = \Sigma\ Stake\ per\ Fake\ ID$, for pos-based systems [29] $Cost_{Sybil} = Hash\ Rate \times Energy\ Cost\ per\ hash \times Attack\ Duration,$ for pow − based systems [29]	All Blockchain Types (context-dependent)
	Consensus Failure Rate	Frequency of consensus breakdowns	$Failure\ Rate = \frac{Failed\ Rounds}{Total\ Rounds}$ [47]	All Blockchain Types
	Cryptographic Strength	Resistance to cryptographic attacks	$Qualitative: Key\ length, Algorithm\ strength$ [13]	All Blockchain Types
	Network Partition Tolerance	Ability to withstand network splits	$Qualitative: \%\ of\ nodes\ isolated\ tolerated$ [15]	All Blockchain Types
	Stale Block Rate	Superseded blocks due to competition	$Stale\ Rate = \frac{Stale\ Blocks}{Total\ Blocks}$ [46]	PoW-based Systems
	Network Disruption Cost	Cost to disrupt network	$Attack\ Cost = Resource\ Cost \times Attack\ Duration$ [48]	All Blockchain Types
Economic	Cost per Transaction	Operational cost per transaction	$Cost/Tx = \frac{Total\ Operational\ Cost}{Total\ Transactions}$ [19]	All Blockchain Types
	Energy Consumption	Power usage per transaction	$Energy/Tx = \frac{Total\ Energy\ (kWh)}{Total\ Transactions}$ [30]	All Blockchain Types
	Mining Reward Efficiency	Reward vs. total cost	$Efficiency = \frac{Reward}{(Energy\ Cost + Hardware\ Cost))}$ [40]	PoW-based Systems
	Gas Fee Variability	Fluctuation in transaction fees	$Variability = \sigma(Gas\ Fees)$ [49]	Smart-Contract Platforms
	Transaction Value	Monetary value of transactions	$Value = \Sigma\ Transaction\ Amounts\ s\ (e.g., BTC)$ [50]	Public Blockchains, DApp Ecosystems
	Network Operational Cost	Network maintenance cost	$Op\ Cost = \Sigma(Node\ Costs + Infrastructure\ Costs)$ [20]	All Blockchain Types
User-Centric	User Adoption Rate	Growth in active users	$Adoption\ Rate = \frac{\Delta Users}{\Delta Time}$ [25]	Public Blockchains, DApp Ecosystems
	Active Address Count	Active addresses per period	$Active\ Count = \Sigma\ Distinct\ Addresses\ in\ period_t$ [25]	Public Blockchains
	NVT Ratio	Market cap vs. transaction volume	$NVT = \frac{Market\ Capitalization}{Transaction\ Volume}$ [50]	Cryptocurrency-based Public Blockchains
	User Transaction Frequency	Avg. transactions per user	$Frequency = \frac{Total\ Transactions}{Active\ Users}$ [37]	Public Blockchains, DApp Ecosystems

23.2. RQ2: Blockchain Features for General KPI Measurement

Three main observations emerge from the mapping between KPIs and blockchain features in Table 3. First, some KPIs rely on directly observable blockchain data, such as timestamps and transaction counts, whereas others require derived metrics or external estimations [42]. Second, several blockchain features are reused across multiple KPIs; for example, timestamps contribute to Latency, Block Time, and Chain Growth Rate measurements. Third, feature requirements vary according to the consensus mechanism. PoW systems depend on metrics such as Hash Rate and mining-related resource usage, whereas PoS systems rely more heavily on validator participation and stake distribution [15], [36]. These observations demonstrate that KPI measurement is strongly influenced by both blockchain architecture and monitoring capabilities.

The features in Table 3 are derived from blockchain protocols and system logs. For example, TPS measurement requires block size and transaction counts [1], [28], while Hash Rate relies on miner computational data [12]. These features ensure that KPIs are measurable across different blockchain types, including public (e.g., Ethereum [14]) and private (e.g., Hyperledger Fabric [15]) systems [36]. The mapping facilitates practical implementation, enabling developers to integrate performance monitoring into blockchain platforms [18], [20]. Figure 3 illustrates the mapping of KPIs to blockchain features using a bipartite graph representation.

Table 8 Blockchain Features Required For General KPI Measurement

Values are approximate and depend on network status; e.g., Bitcoin's hash rate exceeded 500 EH/s in 2023 [12].

KPI	Required Blockchain Features	Applicable To
Transaction Throughput	Block size, transaction count, time interval [1], [28]	All Blockchain Types
Latency	Transaction timestamps (submit, confirm) [28]	All Blockchain Types
Block Time	Block timestamps [14]	All Blockchain Types
Scalability	Node count, TPS under load [18]	All Blockchain Types
Block Propagation Delay	Broadcast and receive timestamps [28]	All Blockchain Types
Fork Rate	Block metadata (temporary forks vs. main chain) [46]	PoW-based Systems, Some PoS Systems
Transaction Finality	Confirmation depth (PoW) or finality time (PoS) [43]	All Blockchain Types
Network Bandwidth Usage	Data transfer logs [36]	All Blockchain Types
Node Synchronization Time	Sync start/end timestamps [44]	All Blockchain Types
Chain Growth Rate	Blockchain size, time [45]	All Blockchain Types
Hash Rate	Miner computational power [12]	PoW-based Systems
Resistance to 51% Attack	Total hash rate, hardware cost, energy cost per hash [12]	All Blockchain Types
Orphaned Block Rate	Block status (orphaned vs. accepted) [46]	PoW-based Systems
Double-Spending Success Rate	Transaction history, chain analysis, attacker's hash power share (q), network power (p), and confirmation count (k). [42]	Public Blockchains
Sybil Attack Resistance	For pos-based systems: Node identity verification data, minimum stake requirement per identity; for pow-based systems: total hash rate, energy cost per hash, attack duration [29]	All Blockchain Types
Consensus Failure Rate	Consensus round logs [47]	All Blockchain Types
Cryptographic Strength	Encryption algorithm, key size [13]	All Blockchain Types
Network Partition Tolerance	Network topology, split logs [15]	All Blockchain Types
Stale Block Rate	Block status (stale vs. accepted) [46]	PoW-based Systems
Attack Cost	Attack type specification (e.g., DoS, 51%), resource usage (computational, financial), time logs [48]	All Blockchain Types
Cost per Transaction	Fee data, operational costs [19]	All Blockchain Types
Energy Consumption	Hash rate (PoW) or node count (PoS), hardware specs [30]	All Blockchain Types
Mining Reward Efficiency	Reward data, energy cost, hardware cost [40]	PoW-based Systems
Gas Fee Variability	Fee history [49]	Smart-Contract Platforms
Transaction Value	Transaction amounts (in native currency) [50]	Public Blockchains, DApp Ecosystems
Network Operational Cost	Node maintenance costs, infrastructure costs [20]	All Blockchain Types
User Adoption Rate	User registration logs [25]	DApp Ecosystems, Public Blockchains
Active Address Count	Address activity logs over time [25]	Public Blockchains
NVT Ratio	Market cap, transaction volume [50]	Cryptocurrency-based Public Blockchains
User Transaction Frequency	Transaction logs, user count [37]	DApp Ecosystems, Public Blockchains

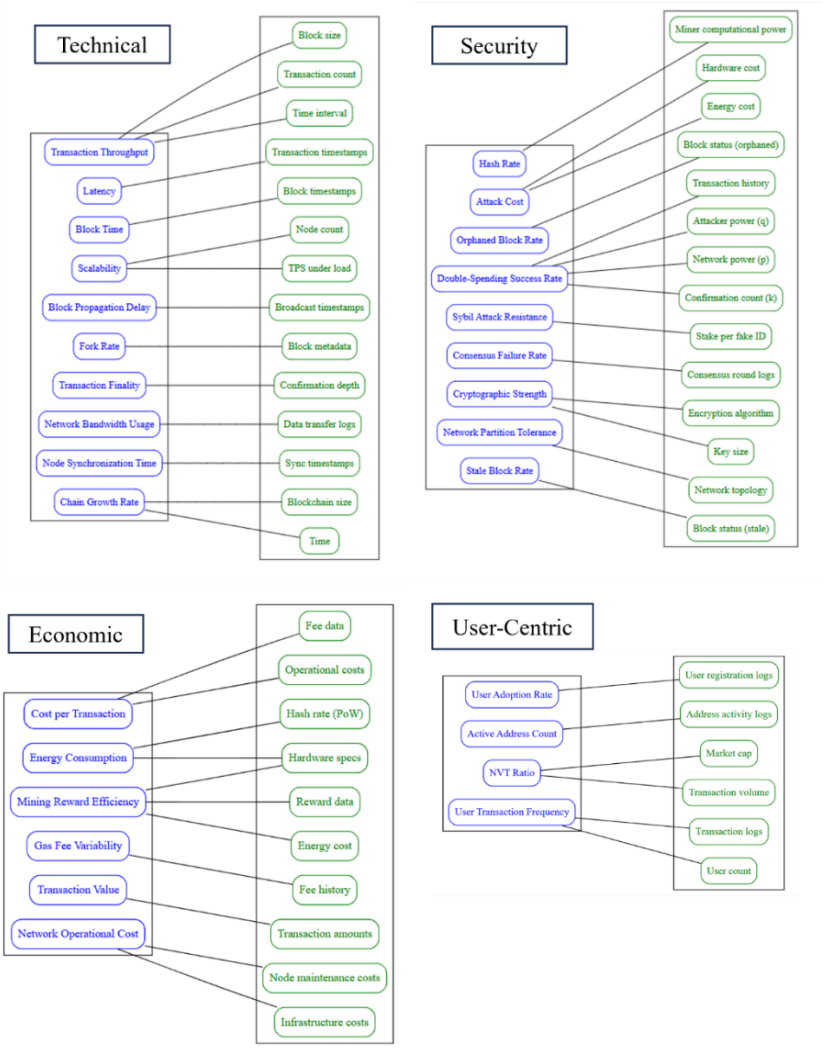


Figure 10 Bipartite Graph for Mapping of KPIs to Blockchain Features

23.3. Domain-Specific KPIs and Features

We analyzed 30 specialist studies to derive domain-specific KPIs tailored to applications such as supply chain management, IoT, smart contracts, and industrial systems. Table 4 lists these KPIs, their formulas or measurement approaches, and contextual details (application, time horizon, blockchain type), addressing RQ1. Table 5 details the blockchain features required for their measurement, addressing RQ2.

We highlight four interpretive patterns that help readers navigate this domain-specific taxonomy.

Pattern 1 – Technical vs. non-technical split: Some KPIs (e.g., Throughput, Gas Consumption) are direct analogs of general KPIs from Table 2; others (e.g., Transparency, User Satisfaction) require qualitative or composite measures. Look for the "Formula / Measurement Approach" column in Table 4 to distinguish quantitative formulas from qualitative checklists.

Pattern 2 – Overlapping KPIs across domains: Throughput appears in IoT, industrial systems, and protocol efficiency studies [35], [51]. Table 4 consolidates such overlaps to avoid redundancy – when you see a KPI labeled "recurring," it means the same measurement approach applies across multiple domains.

Pattern 3 – Time horizon matters: Short-term KPIs (e.g., Access Latency) capture real-time performance; long-term KPIs (e.g., Sustainability Score) require trend data over weeks or months. Table 4's "Time Horizon" column tells you whether to sample second-by-second or aggregate monthly.

Pattern 4 – Emerging vs. established: The bottom rows of Table 4 introduce conceptual KPIs for DeFi, NFTs, and Web3 that lack standardized formulas. These are flagged as "preliminary" – researchers should treat them as research gaps rather than ready-to-use metrics.

1) Discussion of Domain-Specific KPIs: Table 4 presents domain-specific KPIs tailored to blockchain applications such as supply chain management, IoT, smart contracts, industrial systems, and transaction networks. In supply chain applications, metrics such as Delivery Time, Transparency, and Delivery Accuracy support logistics monitoring and traceability [2], [3]. IoT environments emphasize Throughput, End-to-End Delay, and Access Latency to ensure scalable and responsive device coordination [4]. Smart contract platforms rely on KPIs including Gas Consumption, Contract Execution Time, and Vulnerability Count to evaluate execution efficiency and security [49]. Industrial systems prioritize metrics such as System Reliability and Response Time to support operational stability [11].

Several KPIs recur across domains, particularly Throughput, Latency, and Scalability, although their interpretation varies depending on application context. Technical KPIs are generally associated with measurable protocol behavior, whereas non-technical KPIs, such as Transparency, Sustainability Score, and User Satisfaction, often rely on qualitative or composite assessments [3]. Emerging applications, including mobile gaming and transaction network analysis, introduce additional KPIs reflecting the evolving scope of blockchain ecosystems [21], [52].

Table 5 maps domain-specific KPIs to the blockchain features required for their measurement. Frequently reused features include transaction timestamps, audit trails, smart contract execution logs, and consensus records. Technical KPIs primarily rely on on-chain protocol data, whereas non-technical KPIs often require external or qualitative information, such as user feedback and survey data [21], [32]. This mapping supports traceable KPI measurement across public, private, and hybrid blockchain systems while reducing redundant instrumentation requirements.

Table 9. Domain-Specific Blockchain KPIs
(T: Technical, NT: Non-Technical)

Values are approximate and depend on network status; e.g., Bitcoin's hash rate exceeded 500 EH/s in 2023 [12].

Ref.	Application	Time Horizon	Type	KPI (T/NT)	Formula/Approach
[2], [24]	Supply chain	Short-term	Private	Delivery Time (T)	$T_{confirm} - T_{submit}$
[2], [24]	Supply chain	Short-term	Private	Transaction Success Rate (T)	$(\frac{Successful\ Tx}{Total\ Tx}) \times 100\%$
[2]	Supply chain	Short-term	Private	Confirmation Time (T)	$T_{confirm} - T_{start}$
[26], [27], [31]	Consensus scalability	Long-term	Public	Consensus Speed (T)	$T_{consensus} - T_{proposal}$
[27], [31]	Consensus scalability	Long-term	Public	Fault Tolerance (T)	$(\frac{Tolerated\ Failures}{Total\ Nodes}) \times 100\%$
[27], [31]	Consensus scalability	Long-term	Public	Energy Consumption (T)	$\frac{Total\ Energy}{Total\ Tx}$
[27]	Consensus scalability	Long-term	Public	Scalability (T)	$\frac{\Delta Throughput}{\Delta Nodes}$
[7], [8]	Smart contracts	Short-term	Private	Contract Execution Time (T)	$T_{complete} - T_{start}$
[7]	Smart contracts	Short-term	Private	Code Complexity (T)	$\sum Decision\ Points$
[7], [11], [51]	Smart contracts	Short-term	Private	Transaction Rate (T)	$\frac{Total\ Tx}{Time}$
[41], [8]	Smart contracts	Short-term	Private	CPU Usage (T)	$\frac{CPU\ Cycles}{Tx}$
[41]	Smart contracts	Short-term	Private	Memory Consumption (T)	$\frac{Memory}{Tx}$
[8]	Smart contracts	Short-term	Public	Gas Consumption (T)	$\sum Gas\ Units$
[8]	Smart contracts	Short-term	Public	Vulnerability Count (T)	$\sum Vulnerabilities$
[17], [3], [10]	Industrial systems	Long-term	Public	System Reliability (T)	$(\frac{Uptime}{Total\ Time}) \times 100\%$
[9], [10], [11]	Industrial systems	Long-term	Public	Response Time (T)	$T_{response} - T_{request}$
[35], [51]	Industrial systems	Long-term	Public	Throughput (T)	$\frac{Total\ Tx}{Time}$
[4], [5], [51]	IoT	Short-term	Public	End-to-End Delay (T)	$T_{confirm} - T_{initiate}$
[4], [35]	IoT	Short-term	Public	Throughput (T)	$\frac{Total\ Tx}{Time}$
[34], [5]	IoT	Long-term	Public	Access Latency (T)	$T_{access} - T_{request}$
[34], [5], [21]	IoT	Long-term	Public	Scalability (NT)	$\frac{\Delta Throughput}{\Delta Devices}$
[16]-[3]	Supply chain	Long-term	Hybrid	Transparency (NT)	$(\frac{Auditable\ Tx}{Total\ Tx}) \times 100\%$
[16]	Supply chain	Long-term	Hybrid	Data Integrity (NT)	$(\frac{Unmodified\ Tx}{Total\ Tx}) \times 100\%$
[16], [24]	Supply chain	Long-term	Hybrid	Transaction Success Rate (NT)	$(\frac{Successful\ Tx}{Total\ Tx}) \times 100\%$
[11], [22]	Protocol efficiency	Long-term	Private	Throughput (T)	$\frac{Total\ Tx}{Time}$
[11], [51]	Protocol efficiency	Long-term	Private	Resource Consumption (T)	$\sum CPU, Memory, Bandwidth$
[31], [33]	Consensus performance	Long-term	Private	Consensus Time (T)	$T_{final} - T_{start}$
[31], [33]	Consensus performance	Long-term	Private	Fault Tolerance (T)	$(\frac{Tolerated\ Failures}{Total\ Nodes}) \times 100\%$
[9], [5]	Industrial IoT	Long-term	Private	Energy Efficiency (T)	$\frac{Total\ Tx}{Total\ Energy}$
[9], [5], [11]	Industrial IoT	Long-term	Private	Response Time (T)	$T_{response} - T_{request}$
[22], [51]	Vehicle data	Long-term	Public	Throughput (T)	$\frac{Total\ Tx}{Time}$
[22]	Vehicle data	Long-term	Public	Access Speed (T)	$T_{retrieve} - T_{request}$
[5], [21], [23]	Real-time monitoring	Short-term	Public	Latency (T)	$T_{confirm} - T_{submit}$
[23]	Real-time monitoring	Short-term	Public	System Stability (T)	$\frac{\sigma(Throughput)}{Avg\ Throughput}$
[17], [3]	Supply chain	Long-term	Hybrid	Sustainability Score (NT)	$w_1 \cdot Energy\ Efficiency + w_2 \cdot Reliability$
[17], [3]	Supply chain	Long-term	Hybrid	System Efficiency (NT)	$\frac{Successful\ Operations}{Total\ Operations}$
[24]	Supply chain	Long-term	Hybrid	Delivery Accuracy (NT)	$(\frac{Accurate\ Deliveries}{Total\ Deliveries}) \times 100\%$
[52]	Transaction networks	Long-term	Public	Data Depth (T)	$\sum Unique\ Data\ Points$
[52]	Transaction networks	Long-term	Public	Pattern Detection Accuracy (T)	$(\frac{Correct\ Patterns}{Total\ Patterns}) \times 100\%$
[21], [39]	Platform selection	Long-term	Hybrid	Platform Efficiency (NT)	$\frac{Successful\ Operations}{Resource\ Cost}$
[21]	Mobile gaming	Short-term	Public	User Experience (NT)	$Qualitative: User\ satisfaction\ survey$
[32]	System adoption	Long-term	Public	User Satisfaction (NT)	$Qualitative: User\ feedback\ scores$

Table 10. Blockchain Features Required for Domain-Specific KPI Measurement
(T: Technical, NT: Non-Technical)

Ref.	KPI (T/NT)	Required Blockchain Features
[2], [24]	Delivery Time (T)	Transaction timestamps (submit, confirm)
[2], [24]	Transaction Success Rate (T/NT)	Transaction status logs, confirmation events
[2]	Confirmation Time (T)	Transaction timestamps (start, confirm)
[26], [27], [31]	Consensus Speed (T)	Consensus round timestamps (proposal, consensus)
[27], [31]	Fault Tolerance (T)	Node status logs, consensus parameters
[27], [31]	Energy Consumption (T)	Hash rate (PoW), node count (PoS), hardware specs
[27]	Scalability (T)	Node count, throughput logs under load
[7], [8]	Contract Execution Time (T)	Gas usage logs, smart contract event timestamps
[7]	Code Complexity (T)	Smart contract source code, gas consumption metrics
[7], [11], [51]	Transaction Rate (T)	Transaction count, time interval
[41], [8]	CPU Usage (T)	Node resource logs (CPU cycles), gas metrics
[41]	Memory Consumption (T)	Node resource logs (memory allocation), gas metrics
[8]	Gas Consumption (T)	Gas usage logs, transaction operation details
[8]	Vulnerability Count (T)	Smart contract audit logs, vulnerability scan reports
[17], [3], [10]	System Reliability (T)	Node uptime logs, failure event records
[9], [10], [11]	Response Time (T)	Transaction request/response timestamps
[11], [22], [51]	Throughput (T)	Block size, transaction count, time interval
[4], [5], [51]	End-to-End Delay (T)	Transaction timestamps (initiate, confirm), network latency data
[35], [51]	Throughput (T)	Block size, transaction count, time interval
[34], [5]	Access Latency (T)	Access request/response timestamps, smart contract logs
[34], [5], [21]	Scalability (NT)	Device count, throughput logs under load
[16]-[3]	Transparency (NT)	Audit trails, transaction history, public ledger data
[16]	Data Integrity (NT)	Transaction hashes, tamper-proof logs
[16], [24]	Transaction Success Rate (NT)	Transaction status logs, confirmation events
[35], [51]	Throughput (T)	Block size, transaction count, time interval
[35], [11], [51]	Resource Consumption (T)	Node resource logs (CPU, memory, bandwidth)
[31], [33]	Consensus Time (T)	Consensus round timestamps, protocol logs
[31], [33]	Fault Tolerance (T)	Node status logs, consensus parameters
[9], [5]	Energy Efficiency (T)	Transaction count, energy consumption logs (kWh)
[9], [5], [11]	Response Time (T)	Transaction request/response timestamps
[11], [22], [51]	Throughput (T)	Block size, transaction count, time interval
[22]	Access Speed (T)	Data request/retrieval timestamps
[5], [21], [23]	Latency (T)	Transaction timestamps (submit, confirm)
[23]	System Stability (T)	Throughput logs, load variation data
[17], [3]	Sustainability Score (NT)	Energy consumption logs, reliability logs, transaction success rate
[17], [3]	System Efficiency (NT)	Operation logs, resource usage data
[24]	Delivery Accuracy (NT)	Delivery confirmation logs, transaction records
[52]	Data Depth (T)	Transaction metadata, unique data point logs
[52]	Pattern Detection Accuracy (T)	Transaction pattern logs, verification records
[21], [39]	Platform Efficiency (NT)	Operation logs, resource cost metrics
[21]	User Experience (NT)	User feedback logs, satisfaction survey data
[32]	User Satisfaction (NT)	User feedback logs, satisfaction score data

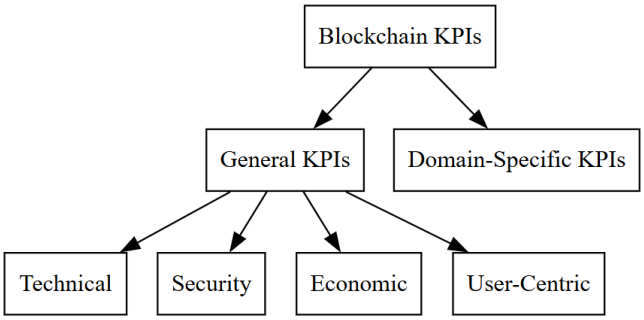


Figure 11. Taxonomy Diagram of Blockchain KPIs

24. DISCUSSION AND RECOMMENDATIONS

24.1. Innovation and Comparison with Prior Work

To our knowledge, this is the first systematic review to integrate both general and domain-specific blockchain KPIs into a unified, traceable taxonomy, addressing a critical gap where prior efforts lacked cross-domain frameworks [19]. The study adopts the Kitchenham (2007) framework with a rigorous two-phase screening and strict inclusion/exclusion criteria, ensuring methodological robustness and reproducibility [20]. Prior surveys have only partially addressed this challenge. For example, Fan et al. [3] provided a systematic survey of blockchain performance evaluation but concentrated mainly on benchmarking methods without offering a structured taxonomy of KPIs. Similarly, Zheng et al. [5] focused on performance and security challenges, yet their KPI coverage remained fragmented and descriptive. Belotti et al. [19] examined a narrow set of consensus-related KPIs, while Tonelli et al. [7] restricted their analysis to smart contract metrics. In contrast, our review synthesizes 30 general KPIs (covering technical, security, economic, and user-centric dimensions) together with domain-specific KPIs from five major application domains (supply chain [3], IoT [5], smart contracts [8], industrial systems [10], and transaction networks [52]).

This dual contribution—a standardized taxonomy of KPIs and their measurable blockchain features—directly addresses RQ1 and RQ2 and provides a more comprehensive and actionable framework than prior reviews. For RQ1, the taxonomy of 30 general KPIs (technical, security, economic, user-centric) and domain-specific KPIs provides a holistic framework for performance assessment [25]. For RQ2, mapping features like transaction timestamps [24], audit trails [3], and consensus logs [27], ensures

practical implementation across public, private, and hybrid systems [21], [36]. This dual contribution establishes a standardized approach for optimizing blockchain design.

24.2. Analysis of Domain-Specific Overlaps and Differences

The interplay between general and domain-specific KPIs reveals context-dependent interpretations rather than redundancy, highlighting the need for tailored prioritization. For instance, supply chain systems studied in [2], adopt Delivery Time as a latency-related KPI to ensure timely logistics, whereas IoT networks emphasize Throughput for large-scale device interaction and scalability [5]. Smart contract platforms like Ethereum [8] instead focus on Gas Consumption, reflecting execution cost efficiency rather than transaction speed.

Although KPIs such as Throughput, Latency, and Scalability appear both in the general taxonomy (Table 1) and in domain-specific contexts (Table 3), their meanings diverge across applications. Transaction Throughput (TPS) generally measures protocol-level performance [28], but in IoT it captures device-driven transaction capacity [5], and in smart contracts it manifests as transaction execution rate [11]. Similarly, Latency is broadly defined as confirmation delay [28], but in supply chains it is expressed as Delivery Time [2], [24], and in consensus mechanisms it corresponds to Consensus Speed [27], [31]. Transparency refers to product traceability in supply chains [16], [24], while in smart contracts it denotes source code availability and auditability [7], [8]. Likewise, Scalability in public blockchains measures node capacity [18], but in IoT it relates to the ability to accommodate growing numbers of connected devices [34]. Finally, Energy Consumption—a critical sustainability KPI in PoW systems (≈700 kWh per transaction [27], [30], with the value varying according to network conditions and time)—is reframed as node-level efficiency in IoT deployments [9].

These variations demonstrate that domain-specific KPIs extend, refine, or reinterpret general metrics rather than duplicating them, underscoring the importance of contextualized KPI frameworks for optimizing blockchain performance in different application domains. Table 6 illustrates this by comparing general blockchain KPIs with their domain-specific adaptations in various sectors. Figure 5 is a comparative visualization of KPI priorities in IoT, Smart Contracts, and Supply Chain domains as a radar graph.

Table 11 Mapping of General KPIs to Domain-Specific Equivalents and Priority Differences

General KPI	Domain-Specific Equivalent	Priority Differences
Transaction Throughput (TPS)	IoT: Throughput [4],[5],[22] Smart Contracts: Transaction Rate [7], [11], [51] Supply Chain: Transaction Success Rate [2], [24]	Critical in IoT for high device transaction volumes; secondary in Smart Contracts, where Gas Consumption is prioritized; moderate in Supply Chain, focusing on reliability.
Latency	IoT: Access Latency [34], [5] Smart Contracts: Contract Execution Time [7], [8] Supply Chain: Delivery Time [2], [24]	Critical in Supply Chain for real-time tracking; moderate in IoT for device responsiveness; lower in Smart Contracts, where execution cost is key.
Transparency	IoT: Not typically emphasized [4], [5] Smart Contracts: Code Transparency [7], [8] Supply Chain: Transaction Transparency [16], [24], [3]	Essential in Supply Chain for traceability; important in Smart Contracts for trust in code; less prioritized in IoT, where scalability dominates.
Scalability	IoT: Device Scalability [34], [5], [21] Smart Contracts: Not typically emphasized [7], [8] Supply Chain: System Efficiency [17], [3]	Critical in IoT for large-scale device networks; moderate in Supply Chain for operational growth; lower in Smart Contracts, where security is prioritized.
Energy Consumption	IoT: Energy Efficiency [9], [5] Smart Contracts: Gas Consumption [41], [8] Supply Chain: Sustainability Score [17], [3]	Critical in IoT for low-power devices; high in Smart Contracts for cost efficiency; moderate in Supply Chain, balancing sustainability with reliability.

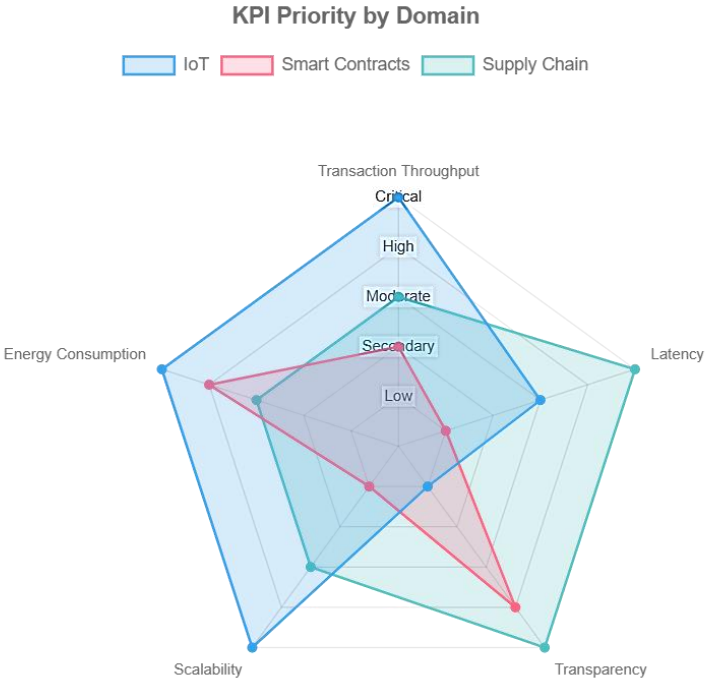


Figure 12. KPI Priorities across IoT, Smart Contracts, and Supply Chain Domains

24.3. Challenges and Needs

Significant challenges remain in applying blockchain KPIs consistently across domains. First, heterogeneous definitions complicate comparability; for instance, Throughput is reported as transactions per second in public blockchains [28], but as operations per second in industrial contexts [10]. Such inconsistencies limit benchmarking reliability and hinder the development of reusable evaluation frameworks. Similarly, Energy Consumption is measured in kWh per transaction for PoW networks [27], yet often approximated at the node level in private systems [10], making cross-system comparisons problematic.

Second, there are measurement limitations. Many KPIs depend on blockchain-internal data (e.g., timestamps, audit trails), but others—such as real-time energy usage—require proprietary logs rarely accessible in private deployments [35]. This restricts reproducibility and calls for collaborative solutions, such as open-source monitoring tools or anonymized data sharing with network operators [21].

Third, some KPIs involve trade-offs that complicate interpretation. For example, maximizing Transaction Throughput in IoT systems [5], can sharply increase Energy Consumption [27], while minimizing Gas Costs in smart contracts [7] [8], may come at the expense of execution latency. Without multi-dimensional evaluation frameworks, such conflicts remain unresolved.

Finally, non-technical KPIs introduce subjectivity and reduce reproducibility. Indicators like Transparency [3], or User Satisfaction [32] often rely on qualitative assessments or surveys, which lack standardized protocols. Similarly, emerging domains such as mobile gaming [21] and transaction networks [52] introduce novel KPIs (e.g., User Experience, Data Depth) without validated measurement models.

Addressing these challenges requires community-driven standardization efforts (e.g., IEEE, ISO) to unify KPI definitions [20], development of automated monitoring tools for real-time measurement, and the design of cross-domain benchmarking frameworks. Moreover, new research should explicitly consider trade-offs among KPIs and expand validated metrics for emerging areas such as DeFi [32], NFTs, and Web3 applications.

Another critical challenge is the lack of standardized KPIs for emerging blockchain domains. Current frameworks predominantly address traditional contexts, leaving DeFi, NFTs, and Web3 applications underexplored. For example, while liquidity and cost efficiency are central to DeFi, no consensus exists on how to measure these indicators uniformly across protocols. NFT performance is similarly fragmented, with studies focusing on market trends rather than reproducible metrics. Without systematic definitions and measurement features, comparisons across emerging platforms remain unreliable. Addressing this gap requires collaborative efforts to establish domain-specific standards and benchmarks.

24.4. Limitations and Scope Constraints

This study has several limitations that should be acknowledged. First, data accessibility posed a challenge, as certain blockchain performance indicators—such as real-time energy consumption or private transaction logs—are rarely available in permissioned systems, limiting reproducibility [15], [35]. Second, KPI definitions across the literature are heterogeneous; for example, *Throughput* is measured either as transactions per second [1], [28] or as operations per second [10], [11], complicating direct comparisons. Third, the scope of our review was restricted to peer-reviewed, English-language publications between 2008 and April 2024, which may introduce selection bias. Moreover, the reliance on three databases (Scopus, IEEE Xplore, and Web of Science) raises the possibility of missing relevant studies outside this scope. Finally, empirical validation remains limited in emerging domains such as DeFi, NFTs, and Web3, where KPIs are still conceptual and lack standardized protocols. These limitations highlight the need for broader datasets, harmonized definitions, and future research dedicated to benchmarking KPIs across both established and emerging blockchain domains.

While our taxonomy addresses key gaps, several limitations remain. First, despite our efforts to include DApp-centric metrics (Section 4.4), the review's search strings prioritized terms like "performance metrics" and "key performance indicators," which may have systematically excluded user experience or business-oriented studies that use alternative terminology (e.g., "engagement," "retention"). Future reviews should employ broader search strategies, including gray literature and industry reports.

Second, the scope of our analysis is necessarily bounded by the peer-reviewed literature. Emerging DApp domains (e.g., Web3 gaming, decentralized social networks) often publish performance data in preprints or technical blogs that do not meet our inclusion criteria. Consequently, metrics such as Social Graph Growth Rate or Gaming Session Length remain preliminary.

Third, and most critically, our study does not empirically validate the utility or feasibility of all 30 general KPIs. The question of whether practitioners *should* use a given KPI—given trade-offs between measurement cost and insight value—is a direction for future research rather than a claim we settle here.

24.5. Filling the Blind Spot and Audience Mapping

Our analysis confirms that existing blockchain KPI taxonomies disproportionately emphasize infrastructure-layer metrics (e.g., TPS, Latency, Block Time) while neglecting application-layer indicators critical to DApp success. Table 7 presents a non-exhaustive list of commonly omitted metrics, their relevance, and the primary audience for each. These metrics are not absent from the literature; rather, they are scattered across domain-specific studies (e.g., mobile gaming [21], DeFi platforms [25]) without integration into a unified framework.

Table 7. DApp-Centric KPIs and Target Audiences

KPI	Definition	Relevance	Primary Audience
Daily Active Users (DAU)	Unique addresses interacting with a DApp per day	User engagement, token velocity	DApp developers, investors
Total Value Locked (TVL)	Aggregate value of assets deposited in DeFi protocols	Financial health, liquidity	DeFi analysts, regulators
Smart Contract Invocation Frequency	Number of times a contract is called per block/interval	Code utility, gas demand	Smart contract auditors
Retention Rate (Day 1, 7, 30)	% of users returning after first interaction	Long-term viability	Product managers
Session Duration	Time spent per user interaction	User experience, stickiness	UX researchers
Transaction Uniqueness	Ratio of distinct senders to total transactions	Sybil resistance, organic activity	Security analysts

The omission of these metrics is not accidental. Unlike node-level KPIs, DApp-level indicators often require:

- **Off-chain indexing**: DAU calculation necessitates tracking addresses across multiple blocks, which many nodes do not store historically [23].
- **Cross-platform correlation**: Retention rate requires linking wallet addresses to session logs, raising privacy concerns [32].
- **Protocol-specific definitions**: TVL calculation varies across DeFi protocols (e.g., Uniswap vs. Aave), hindering standardization [25].

24.6. Barriers to KPI Adoption

A persistent paradox in blockchain performance evaluation is that while taxonomies list dozens of KPIs, empirical studies rarely employ more than two or three [4], [7]. Our review identifies three structural reasons, which also clarify how our framework addresses each:

1. **Instrumentation Overhead**: KPIs such as Attack Cost or Sybil Attack Resistance require adversarial simulation or privileged network data, which are infeasible in live public blockchains [12], [29]. For each KPI in Table 2 and Table 4, we explicitly indicate whether measurement requires basic node logs (accessible to any operator), privileged system access (permissioned blockchains only), or simulation-based estimation (research environments). This allows practitioners to filter KPIs by implementability.
2. **Lack of Standardized Formulas**: As noted in Section 4.3, Throughput is reported inconsistently (TPS vs. operations/second) [10], [11], and Energy Consumption lacks a unified unit (kWh per transaction vs. node-level approximation) [27], [30]. We provide formula templates (Table 2) and recommend units, enabling cross-study replication.
3. **Audience Misalignment**: Infrastructure engineers prioritize TPS and Latency [18], while business stakeholders require Cost per Transaction or User Adoption Rate [25], [37]. Taxonomies that blend these without audience mapping become unusable. We introduce an "Primary Audience" column (expanded from Table 7) and a decision tree (Figure 6) guiding users to relevant KPI subsets based on their role (developer, operator, analyst, regulator).

By systematically diagnosing why most KPIs are ignored—and providing a structured selection mechanism—our framework moves beyond mere enumeration to actionable guidance.

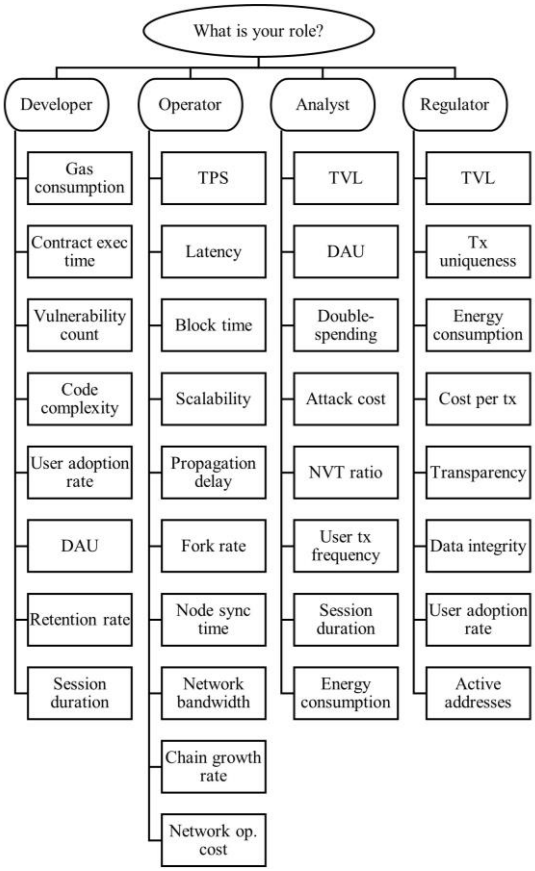


Figure 13. Decision tree for role-based KPI selection

25. CONCLUSION

This systematic review delivers a comprehensive taxonomy of blockchain KPIs, integrating 30 general KPIs and domain- specific KPIs from 55 peer-reviewed studies, addressing RQ1 and RQ2. The general KPIs enable holistic evaluation of systems like Bitcoin [1], Ethereum [14], and Hyperledger Fabric [15], while domain-specific KPIs cater to applications like supply chain [2], [3], IoT [5], [21], smart contracts [8], and industrial systems [10]. The mapped features (e.g., timestamps [2], audit trails [3], [17]) ensure practical measurement across architectures [21], [36]. Our contributions include a standardized framework, tailored domain-specific metrics, and a traceable measurement approach, bridging technical and application-specific needs.

Future research should address several directions, categorized into four areas:

- ***Hybrid Qualitative-Quantitative KPIs:*** Emerging domains such as gaming [21] and healthcare [26] require hybrid metrics that combine quantitative blockchain data (e.g., transaction latency [21], data integrity [16]) with qualitative metrics (e.g., user satisfaction surveys [32], clinical validation). For example, a gaming KPI could integrate latency with player feedback scores, weighted by application goals [21].
- ***Cost-Benefit Analysis Framework:*** Quantifying KPI trade-offs is critical for stakeholders. A framework could weight KPIs (e.g., Throughput vs. Energy Consumption [27]) based on application priorities (e.g., scalability for IoT, reliability for supply chains) and re- source constraints (e.g., hardware costs, energy budgets). For instance, optimizing TPS in IoT may increase energy costs, requiring a balanced cost-benefit model [17].
- ***Automated KPI Monitoring Tools:*** Real-time KPI evaluation requires tools integrated with blockchain nodes, supporting APIs for transaction logs, consensus data, and resource usage [21]. Desired features include scalability, real-time analytics, and cross-platform compatibility. Challenges include data privacy (e.g., anonymizing proprietary logs [35]) and computational overhead. Open-source platforms could accelerate adoption [20].
- ***Cross-Domain Benchmarking Frameworks:*** Standardized benchmarking frameworks should include unified metrics (e.g., aligning TPS [28] with Delivery Time [2]), test scenarios (e.g., stress tests for IoT [5]), and comparison protocols (e.g., public vs. private blockchains [35]). A proposed structure involves a metric repository, standardized testbeds, and open-access datasets, fostering reproducible evaluations [26].
- ***KPIs for Emerging Domains:*** Future work must extend KPI frameworks to fast-growing blockchain applications. In DeFi, this includes formalizing metrics such as *Liquidity Depth*, *Impermanent Loss*, and *Decentralization Index* [32]. For NFTs, indicators like *Transaction Uniqueness*, *Creator Royalties Stability*, and *Market Adoption Rate* are essential. In Web3 and metaverse environments, KPIs such as *Latency*, *Interoperability Index*, and *User Experience* require rigorous definition and empirical validation. Developing and standardizing these metrics will ensure that performance evaluation frameworks remain relevant as blockchain ecosystems evolve into new digital economies.

These directions collectively address technical and non-technical dimensions—ranging from energy-intensive PoW systems [27], [30], and smart contract security [8] to non-technical KPI standardization [21], [32]—driving more robust, scalable, and user-centric blockchain adoption across both established and emerging industries.

26. ACKNOWLEDGMENT

The authors would like to express sincere gratitude to the editors and anonymous reviewers for their invaluable comments and constructive feedback, which significantly contributed to the improvement of this paper.

Declaration of generative AI in the writing processDuring the preparation of this work, the authors used ChatGPT (OpenAI) to improve the readability and language of the manuscript. After using this tool, the authors reviewed and edited the content as needed and takes full responsibility for the final content of this publication.

27. REFERENCES

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[2] Y. Madhwal, Y. Borbon-Galvez, N. Etemadi, Y. Yanovich, and A. Creazza, "Proof of delivery smart contract for performance measurements," *Ieee Access,* vol. 10, pp. 69147–69159, 2022.

[3] C. Fan, S. Ghaemi, H. Khazaei, and P. Musilek, "Performance evaluation of blockchain systems: A systematic survey," *Ieee Access,* vol. 8, pp. 126927–126950, 2020.

[4] M. Alaslani, F. Nawab, and B. Shihada, "Blockchain in IoT systems: End-to-end delay evaluation," *IEEE Internet of Things Journal,* vol. 6, no. 5, pp. 8332–8344, 2019.

[5] X. Zheng, Y. Zhu, and X. Si, "A survey on challenges and progresses in blockchain technologies: A performance and security perspective," *Applied Sciences,* vol. 9, no. 22, p. 4731, 2019.

[6] S. Alrubei, J. Rigelsford, C. Willis, and E. Ball, "Ethereum blockchain for securing the Internet of Things: practical implementation and performance evaluation," in *2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*, 2019: IEEE, pp. 1–5.

[7] R. Tonelli, G. Destefanis, M. Marchesi, and M. Ortu, "Smart contracts software metrics: a first study," *arXiv preprint arXiv:1802.01517,* 2018.

[8] A. Pinna, S. Ibba, G. Baralla, R. Tonelli, and M. Marchesi, "A massive analysis of ethereum smart contracts empirical study and code metrics," *Ieee Access,* vol. 7, pp. 78194–78213, 2019.

[9] Y. Liu, K. Wang, Y. Lin, and W. Xu, "$\mathsf {LightChain} $: a lightweight blockchain system for industrial internet of things," *IEEE Transactions on Industrial Informatics,* vol. 15, no. 6, pp. 3571–3581, 2019.

[10] G. Bovenzi, G. Aceto, V. Persico, and A. Pescapé, "Blockchain Performance in Industry 4.0: Drivers, use cases, and future directions," *Journal of Industrial Information Integration,* vol. 36, p. 100513, 2023.

[11] B. Cao *et al.*, "When Internet of Things meets blockchain: Challenges in distributed consensus," *Ieee Network,* vol. 33, no. 6, pp. 133–139, 2019.

[12] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the security and performance of proof of work blockchains," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 3–16.

[13] D. Mingxiao, M. Xiaofeng, Z. Zhe, W. Xiangwei, and C. Qijun, "A review on consensus algorithm of blockchain," in *2017 IEEE international conference on systems, man, and cybernetics (SMC)*, 2017: IEEE, pp. 2567–2572.

[14] V. Buterin, "Ethereum white paper: a next generation smart contract & decentralized application platform," *First version,* vol. 53, 2014.

[15] E. Androulaki *et al.*, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the thirteenth EuroSys conference*, 2018, pp. 1–15.

[16] K. Kuhi, K. Kaare, and O. Koppel, "Ensuring performance measurement integrity in logistics using blockchain," in *2018 IEEE international conference on service operations and logistics, and informatics (SOLI)*, 2018: IEEE, pp. 256–261.

[17] A. Park and H. Li, "The effect of blockchain technology on supply chain sustainability performances," *Sustainability,* vol. 13, no. 4, p. 1726, 2021.

[18] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "Blockbench: A framework for analyzing private blockchains," in *Proceedings of the 2017 ACM international conference on management of data*, 2017, pp. 1085–1100.

[19] M. Belotti, N. Božić, G. Pujolle, and S. Secci, "A vademecum on blockchain technologies: When, which, and how," *IEEE Communications Surveys & Tutorials,* vol. 21, no. 4, pp. 3796–3838, 2019.

[20] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain technology overview," *arXiv preprint arXiv:1906.11078,* 2019.

[21] G. Yang, K. Lee, K. Lee, Y. Yoo, H. Lee, and C. Yoo, "Resource analysis of blockchain consensus algorithms in hyperledger fabric," *IEEE Access,* vol. 10, pp. 74902–74920, 2022.

[22] T. Jiang, H. Fang, and H. Wang, "Blockchain-based internet of vehicles: Distributed network architecture and performance analysis," *IEEE Internet of Things Journal,* vol. 6, no. 3, pp. 4640–4649, 2018.

[23] P. Zheng, Z. Zheng, X. Luo, X. Chen, and X. Liu, "A detailed and real-time performance monitoring framework for blockchain systems," in *Proceedings of the 40th international conference on software engineering: software engineering in practice*, 2018, pp. 134–143.

[24] M. H. Meng and Y. Qian, "A blockchain aided metric for predictive delivery performance in supply chain management," in *2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, 2018: IEEE, pp. 285–290.

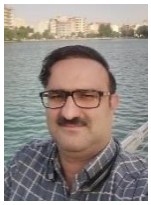
[25] Y. Chen and C. Bellavitis, "Blockchain disruption and decentralized finance: The rise of decentralized business models," *Journal of Business Venturing Insights,* vol. 13, p. e00151, 2020.

[26] I. Pittaras, N. Fotiou, V. A. Siris, and G. C. Polyzos, "Beacons and blockchains in the mobile gaming ecosystem: A feasibility analysis," *Sensors,* vol. 21, no. 3, p. 862, 2021.

[27] Y. Merrad *et al.*, "Blockchain: Consensus algorithm key performance indicators, trade-offs, current trends, common drawbacks, and novel solution proposals," *Mathematics,* vol. 10, no. 15, p. 2754, 2022.
[28] K. Croman *et al.*, "On Scaling Decentralized Blockchains: (A Position Paper)," in *International conference on financial cryptography and data security*, 2016: Springer, pp. 106–125.
[29] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," *Communications of the ACM,* vol. 61, no. 7, pp. 95–102, 2018.
[30] A. De Vries, "Bitcoin's growing energy problem," *Joule,* vol. 2, no. 5, pp. 801–805, 2018.
[31] M. Hayat and H. Winkler, "An analytic hierarchy process for selection of blockchain-based platform for product lifecycle management," *Sustainability,* vol. 14, no. 21, p. 13703, 2022.
[32] M. O. Grida, S. Abd Elrahman, and K. A. Eldrandaly, "Critical success factors evaluation for blockchain's adoption and implementing," *Systems,* vol. 11, no. 1, p. 2, 2022.
[33] Y. Hao, Y. Li, X. Dong, L. Fang, and P. Chen, "Performance analysis of consensus algorithm in private blockchain," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018: IEEE, pp. 280–285.
[34] O. Novo, "Blockchain meets IoT: An architecture for scalable access management in IoT," *IEEE internet of things journal,* vol. 5, no. 2, pp. 1184–1195, 2018.
[35] P. Thakkar, S. Nathan, and B. Viswanathan, "Performance benchmarking and optimizing hyperledger fabric blockchain platform," in *2018 IEEE 26th international symposium on modeling, analysis, and simulation of computer and telecommunication systems (MASCOTS)*, 2018: IEEE, pp. 264–276.
[36] J. Polge, J. Robert, and Y. Le Traon, "Permissioned blockchain frameworks in the industry: A comparison," *Ict Express,* vol. 7, no. 2, pp. 229–233, 2021.
[37] S. Bano, M. Al-Bassam, and G. Danezis, "The road to scalable blockchain designs," *USENIX; login: magazine,* vol. 42, no. 4, pp. 31–36, 2017.
[38] S. Rouhani and R. Deters, "Performance analysis of ethereum transactions in private blockchain," in *2017 8th IEEE international conference on software engineering and service science (ICSESS)*, 2017: IEEE, pp. 70–74.
[39] Z. Zhou, R. Li, Y. Cao, L. Zheng, and H. Xiao, "Dynamic performance evaluation of blockchain technologies," *IEEE Access,* vol. 8, pp. 217762–217772, 2020.
[40] J. Göbel and A. E. Krzesinski, "Increased block size and Bitcoin blockchain dynamics," in *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*, 2017: IEEE, pp. 1–6.
[41] N. Ajienka, P. Vangorp, and A. Capiluppi, "An empirical analysis of source code metrics and smart contract resource consumption," *Journal of Software: Evolution and Process,* vol. 32, no. 10, p. e2267, 2020.
[42] M. Rosenfeld, "Analysis of hashrate-based double spending," *arXiv preprint arXiv:1402.2009,* 2014.
[43] J. Göbel, H. P. Keeler, A. E. Krzesinski, and P. G. Taylor, "Bitcoin blockchain dynamics: The selfish-mine strategy in the presence of propagation delay," *Performance evaluation,* vol. 104, pp. 23–41, 2016.
[44] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper,* vol. 151, no. 2014, pp. 1–32, 2014.
[45] A. Kiayias, E. Koutsoupias, M. Kyropoulou, and Y. Tselekounis, "Blockchain mining games," in *Proceedings of the 2016 ACM Conference on Economics and Computation*, 2016, pp. 365–382.
[46] C. Decker and R. Wattenhofer, "Information propagation in the bitcoin network," in *IEEE P2P 2013 Proceedings*, 2013: IEEE, pp. 1–10.
[47] Y. Sompolinsky and A. Zohar, "Secure high-rate transaction processing in bitcoin," in *Financial Cryptography and Data Security: 19th International Conference, FC 2015, San Juan, Puerto Rico, January 26-30, 2015, Revised Selected Papers 19*, 2015: Springer, pp. 507–527.
[48] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, "Bitcoin and cryptocurrency technologies: a comprehensive introduction," ed: Princeton University Press Princeton, NJ, USA, 2018.
[49] M. Bez, G. Fornari, and T. Vardanega, "The scalability challenge of ethereum: An initial quantitative analysis," in *2019 IEEE International Conference on Service-Oriented System Engineering (SOSE)*, 2019: IEEE, pp. 167–176.
[50] C. Burniske and J. Tatar, "Cryptoassets: The innovative investor's guide to Bitcoin and beyond," 2018.
[51] S. Pongnumkul, C. Siripanpornchana, and S. Thajchayapong, "Performance analysis of private blockchain platforms in varying workloads," in *2017 26th international conference on computer communication and networks (ICCCN)*, 2017: IEEE, pp. 1–6.
[52] J. Zhu, A. Khan, and C. G. Akcora, "Data depth and core-based trend detection on blockchain transaction networks," *Frontiers in Blockchain,* vol. 7, p. 1342956, 2024.

Kimiya Karimi Dehkordi received the B.S. and the M.Sc. degree in Computer Engineering from Shahrekord University. She is currently the Ph.D. student at Isfahan University. Her research interests include blockchain, smart contracts, and model-driven software engineering.
https://orcid.org/0009-0002-7120-4548
kimiya.karimi@stu.sku.ac.ir
Master graduated, Department of Computer Engineering, Faculty of Technology and Engineering, Shahrekord University, Shahrekord, Iran.

Leila Samimi-Dehkordi is an Assistant Professor in the Department of Computer Engineering at Shahrekord University, Iran. She received her B.Sc. degree in Computer Engineering from Iran University of Science and Technology, her M.Sc. degree in Algorithms and Computation from the University of Tehran, and her Ph.D. degree in Software Engineering from the University of Isfahan. Her primary research interests include model-driven software engineering, software language engineering, Blockchain modeling, and gamification. Recently, her work has extended into leveraging large language models (LLMs) and artificial intelligence for automated software modeling and system verification.
https://orcid.org/0000-0002-2842-0256
samimi@sku.ac.ir
Assistant Professor, Department of Computer Engineering, Faculty of Technology and Engineering, Shahrekord University, Shahrekord, Iran.



Abbas Horri is an Assistant Professor in the Department of Computer Engineering at Shahrekord University, Iran. He received both his M.Sc. and Ph.D. degrees in Computer Engineering (Software) from Shiraz University, Iran. His primary research interests include cloud computing, parallel architectures, green computing, and big data. His recent research also extends to resource optimization, distributed systems, and performance evaluation in cloud-edge and blockchain environments.
https://orcid.org/0000-0001-7933-9266
horri@sku.ac.ir
Assistant Professor, Department of Computer Engineering, Faculty of Technology and Engineering, Shahrekord University, Shahrekord, Iran.