Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

January 2025, Special Volume 2, Issue 1

Improvement in intent detection and slot filling by model enhancement and different data augmentation strategies

Mohammad Mahdi HajiRamezanAli✉, Code ORCID: 0009-0007-4226-5604
ShahabDanesh University, Qom, Pardisan, Iran, hajiramezanali@shdu.ac.ir
Hasan Deldar, Code ORCID: 0000-0002-2379-4427
ShahabDanesh University, Qom, Pardisan, Iran, hdeldar@shdu.ac.ir
Mohammad Mehdi Homayounpour, Code ORCID: 0009-0003-1562-932X
ShahabDanesh University, Qom, Pardisan, Iran, homayoun@shdu.ac.ir

Abstract— Intent detection and slot filling are crucial for understanding human language and are essential for creating intelligent virtual assistants, chatbots, and other interactive systems that interpret user queries accurately. Recent advancements, especially in transformer-based architectures and large language models (LLMs), have significantly improved the effectiveness of intent detection and slot filling. This paper, proposes a method for effectively utilizing low volume fine-tuning data samples to enhance the natural language comprehension of lightweight language models, yielding a nimble and efficient approach. Our approach involves augmenting new data while increasing model layers to enhance understanding of desired intents and slots. We explored various synonym replacement methods and prompt-generated data samples created by large language models. To prevent semantic meaning disturbance, we established a lexical retention list containing non-$\boldsymbol{O}$ slots to preserve the sentence's core meaning. This strategy enhances the model's slot precision, recall, F1-score, and exact match metrics by 1.41%, 1.8%, 1.61%, and 3.81%, respectively, compared to not using it. The impact of increasing model layers was studied under different layer arrangement scenarios. Our results show that our proposed solution outperforms the baseline by 10.95% and 4.89% in exact match and slot F1-score evaluation metrics.

Keywords— intent detection, slot filling, joint model, BERT, language model, data augmentation

I. Problem definition

Intent detection and slot filling can be considered as a sequence-labeling problem. Given an input sentence $W = \{w_1, w_2, \dots, w_t\}$ with t words, the model must assign a predefined label from the slot set $S = \{s_1, s_2, \dots, s_c\}$ consisting of c slots. This task can be approached as a single-label or multi-label classification task; this paper focuses on single-label classification, where each word w_x is assigned only one corresponding slot s_y. Slot filling is done in BIO format, categorizing words into three classes: 1) a single slot where a word has all necessary information and is labeled as B; 2) the beginning of an information slot labeled as B too. (e.g. the first word in an address); 3) continuation of that slot is labeled as I (rest of the words in an address). Any word that holds no information to the identified intent is marked as O. Intent detection mirrors slot filling but uses a distinct set of e intents, denoted as $I = \{i_1, i_2, \dots, i_e\}$. The selected intent i_e applies to the entire input sequence W, to depict the overall purpose of that utterance.

II. Transformer-based Language models and LLMs

The introduction of transformer-based language models first proposed by [1], has benefitted various tasks including intent detection and slot filling. Models like BERT [2], a bidirectional auto encoder transformer, pretrained on masked language modeling and next sentence prediction tasks exemplify this advancements. The emergent of larger models along with research [3] have demonstrated promising results to make use of bigger models with numerous amounts of training parameters. Models like GPT-3 [4], FLAN-T5-XL [5] and PaLM [6] containing 175 billion, 11 billion and 540 billion parameters, respectively are few examples of recently developed large language models. These models benefit in-context learning, a phenomenon which helps the model learn by observing one or a few examples of desired task at inference time. This provided a whole new pathway to study both existing and new problems.

III. Related works

IV. Intent detection and slot filling with primitive language models

Intent detection and slot filling have been explored from multiple perspectives. Among previous studies that treated intent detection and slot filling as a joint model, [7] have proposed a way to decrease the inference time by nearly 11 times, using a BERT model and a two-pass refinement mechanism to prevent order mismatch in the BIO tagging format. In an another

Submit Date: 2025-04-30
Accept Date: 2025-05-05
✉ Corresponding author

approach, [8] presented a multi-lingual multi-task framework for these tasks. Other works including [9], [10] and [11] put effort to use the predicted intents to better capture the related slots and vice versa in a joint learning manner. Notably [12], showed that training models for Farsi as a low-resource language with data from resource-rich languages demonstrated the benefits of multilingual pretrained language models.

V. Large language models in intent detection and slot filling

The development of LLMs has advanced intent detection and slot filling. [13] utilized generative LLMs like Mistral 7B [14] and Claude v3 Sonnet [15] with adaptive in-context learning and chain of thought prompting to adapt these models for intent detection task. This approach integrated these models with resource efficient language models in a hybrid type to address the high hardware resources demands. Another study, [16] examined zero-shot capabilities for three LLMs: Mistral, Flan-T5-XL, and a proprietary LLM known as granite.13b.v2. In line with the previous approach, they utilized prompts based on fine-tuning data, while concentrating on developing a zero-shot system to label desired tokens in dialogue turns. The ILLUMINER [17] method approached intent detection and slot filling as a language generation task, using instruction-tuned LLMs such as Falcon-7B-Instruct[18] and BLOOMZ [19] (Fine-tuned on BLOOM [20]). For intent detection, they listed possible intent labels to choose from in a given instruction, expecting an overall intent output for the input utterance. In slot filling, a single-prompt information extraction was employed differing from other approaches [21].

VI. Proposed Method

VII. Overview

By using a Bidirectional transformer encoder Like BERT or its derivatives (e.g.: DistilBERT[22], RoBERTa [23], ALBERT [24] and mBERT), our method first takes in a sequence of input utterances $W = \{w_1, w_2, \dots, w_t\}$ and adds two special tokens $[CLS]$ and $[SEP]$ to the beginning and the end of sequence, respectively. After that, the BERT-based model produces the contextualized latent representations for each input token, including $[CLS]$ and $[SEP]$ tokens. Following [25], the key idea in detecting the intents is to use the contextualized latent representation of $[CLS]$ token to choose the correct intent from the predefined set of intents. While implementing the demonstrated method, optimizing certain aspects of the problem becomes crucial, including managing low-volume data regimes and ensuring resource efficiency. In the following sections, we address these challenges by: a) adding additional layers to models, b) testing various architectures of intent and slot classifiers, and c) augmenting on low-volume data to enhance model's performance. Overview of our method is illustrated in Figure 1.

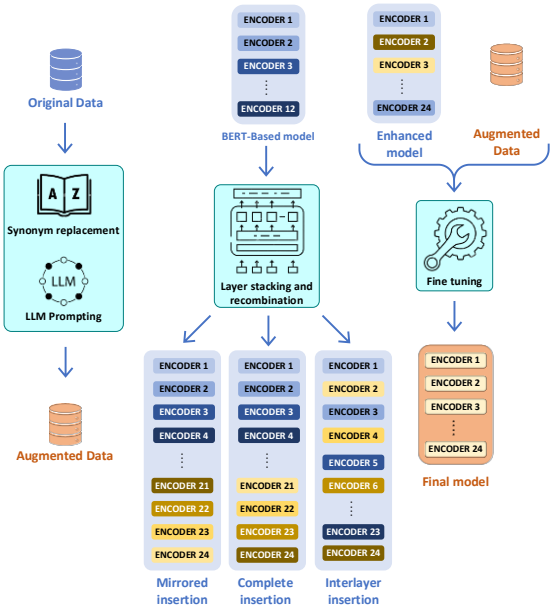

Figure 1 Our proposed method includes two major parts: A) data augmentation phase(left) and B) Model enhancement techniques (middle). The enhanced model is then fine-tuned on the augmented data (right).

VIII. Enhancing the model

Adding extra layers to BERT-based models has been explored by [26] and [27] previously. In first approach, the authors aimed to differentiate the impact of various layers of BERT-based models on learning dependencies among input utterances and subsequently sought to incorporate additional layers into these models. In contrast, the second method focused on achieving improved results on evaluation metrics while scaling the BERT model from 12 layers to 1000 layers. Following the described works, we adopted for a similar approach, specifically to double the number of layers in utilized model. We experimented three different strategies in our enhancement technique: a) interlayer insertion, b) complete insertion and c) mirrored insertion. Each technique is detailed in Figure 2.

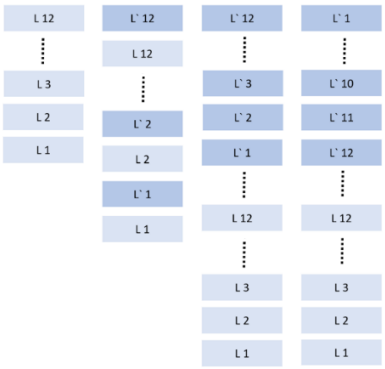

Figure 2 Illustration of layer duplication methods presented in separate columns from left to the right: 1) normal mode without layer addition. 2) interlayer insertion 3) complete insertion 4) mirrored insertion.

IX. Different Architectures for intent and slot classifier

Similar to [12], our approach features a dropout layer followed by a linear layer with c neurons to identify potential slots. The model's output excluding the CLS token representation serves as input for this classifier. The same intent classifier operates similarly, differing only in the number of neurons (e many neurons) and input which uses the latent representation of the CLS token.

X. Data augmentation

Unlike languages such as English and Chinese, Farsi can be considered as a low-resource language. This is practically significant when fine-tuning models for intent detection and slot filling tasks, as there is less high-quality data available in this language. As a result, exploring methods to augment new fine-tuning data has led us to various data augmentation solutions, namely: a) synonym replacement, and b) data augmentation with large language models through prompting. Each of the aforementioned options is explained in the following sections.

XI. Synonym replacement

The concept behind this method is to identify synonyms in input utterances and substitute them with the original words. The rationale for this approach is that synonyms share similar semantic characteristics, making them viable alternative candidates for conveying in a message in a given input. However, it is evident that the overall divergence in meaning of an input utterance after synonym replacement means that not all synonyms are well-suited for this task. Nevertheless, we chose to evaluate the impact of this data augmentation approach on models' performance. In exploring this idea, two implementations have emerged as: a) word-wise replacement. b) sentence-wise replacement.

a) Word-wise replacement

In word-wise replacement, we search for each training word and retrieve n number of its synonyms. Then for every retrieved synonym, we build a new training example, consisting of the replaced synonym and all other input words. It is worth mentioning that the general structure of the training sample remains unchanged and at each instance, only one word is altered. An example of this method is shown in Figure 3.

Figure 3 An example of word-wise replacement data augmentation. The first line is the primary sample with it's slots oriented vertically. The second and third lines depict the augmented samples, with replaced synonyms appeared in red color, underlined text.

b) Sentence-wise replacement

Unlike previous method where for each word synonym replacement a new sample gets created, here we gather all available synonyms for every word in a training sample and by replacing all of them in the original sample, a new so called ***warped sample*** is made. This process gets repeated until we run out of synonyms for that given sample. Since all words do not have the same number of synonyms, we look for the word with the highest number of synonyms and take number of synonyms as our scale and build many augmented samples. For the rest of the words in an input utterance that have a smaller number of synonyms, first we put all synonyms one by one in the subsequent augmented samples and when each synonym list for a given word reaches its end, we randomly select previously chosen synonyms for the rest of augmented samples. It's worth noting that due to the nature of this technique, the number of generated examples in this approach is far less than the word-wise replacement method. It is evident that produced training samples in this method are more prone to change the overall meaning of the original sample. Figure 4 shows this method in detail.

Figure 4 An example of sentence-wise augmentation: Synonyms re replaced with their primary words at each step, resulting in a distinctly different augmented sample.

c) **Lexical retention list**

Upon examining the two mentioned synonym replacement approaches, it is evident that replacement is not always beneficial. Thus, we propose extracting the non-O slot tokens and excluding them from replacement. The key concept behind this idea is to preserve as much information as possible in the sample by only altering the O words and leaving all other words intact.

d) **Number of synonyms to retrieve and maximum number of samples**

As controllable parameters, number of synonyms and maximum number of samples can be specified to reflect the efficiency of augmented data volume. This aspect will be examined in Ablation study.

XII. Data augmentation through prompting

Querying language models to achieve new samples has been already experimented by [28] and [29]. In the first study, a BERT language model was prompted to identify named entities in unlabeled examples. The second research utilized GPT J 6B [30] on DAILY DIALOG [31] dataset to augment data through weakly-supervised filters. Our approach involved using Llama 3.1 70B [32], Chat GPT 4 o mini [33] and Qwen2.5-72B-Instruct [34] to generate desired data samples. For each intent, we provided a separate prompt with a few training samples to the models. Although the specific prompts varied by model, they maintained a consistent overall structure. To avoid repetitive samples, we specifically instructed the models to create structurally diverse samples for each intent.

XIII. Evaluation

XIV. Datasets

To evaluate our approach for detecting intents and slots, we tested several related datasets described below.

XV. MASSIVE 1M

MASSIVE dataset [35] is a labeled multilingual dataset for intent detection and slot filling, covering 51 languages including Farsi. It consists 12664 training samples and 2974 samples for both development and testing in Farsi, encompassing 60 intents and 108 slots.

XVI. MODERN ISC

This proprietary dataset, produced by "Modern isc" company[1], aims to develop natural language understanding capabilities for a virtual assistant in e-banking domain. This dataset contains data under 21 different intents and 110 slots, with 1943 training samples and 210 validations samples. Since the purpose of making this dataset is to sponsor a finance related chatbot challenge, therefore the test set data is retained by the company for final evaluations, so we used the validation set as test set to measure our methods performance.

XVII. Evaluation metrics

For evaluation, we employed precision, recall and F1 measure scores for slot filling and utilized accuracy score for intent detection. To assess overall model performance, we also utilized an exact match metric which is considered true when all predicted slots and intent for a sample are predicted correctly, otherwise it is false.

XVIII. Experimented models

Following [25], we chose the joint intent detection and slot filling approach, executing our experiments with various models of BERT family. Given that the input data is in Farsi, we searched for a BERT-based model Pretrained in that language. By exploring online resources such as GitHub[2] and huggingface[3], we identified several suitable candidate models, listed in the first and second columns of Table 1.

XIX. Preliminary experiment

We first trained candidate models on MASSIVE dataset to evaluate their overall capability. As our final goal was to maximize intent detection and slot filling performance on MODERN ISC dataset, we put our efforts to apply enhancement techniques on this dataset and employed MASSIVE dataset exclusively to filter out the unsuitable candidates, saving both time and resource. For trainings, we used the AdamW optimizer [36] with the learning rate of $5e-5$ and an epsilon value of $1e-8$, and trained the models for 10 epochs. The results are shown in Table 1.

Among these models, the two models "**sharif-dal/dal-bert**" and "**HooshvareLab/bert-fa-base-uncased**" yielded the best results. We further analyzed the capability of these two selected models. An analysis was done on wrong inferences of MASSIVE dataset test set to better understand the weaknesses and strengths of these models. The analyzed errors are: A) the number of incorrect intent predictions (intent mismatch). B) actual O slots incorrectly marked as non-O prediction (true O false slot). C)

[1] www.modernisc.com
[2] www.github.com
[3] www.huggingface.co

actual non-*O* slots predicted as *O* (true slot false *O*). D) Actual non-*O* slots predicted as other wrong non-*O* slots (slot mismatch). The MASSIVE test-set consists of 2974 samples and a total of 21834 slots. We also explored how different architectures effect intent detection and slot filling neural networks classifiers, by adding two linear layers and two GELU [37] activation layers. The results are summarized in Table 2.

Table 1 Results of candidate models on MASSIVE dataset.

Model Family	Model name	Slot precision	Slot recall	Slot F1 score	Intent accuracy	Exact match
BERT	sharif-dal/dal-bert	70.81	**78.08**	74.27	86.78	65.09
DistilBERT	HooshvareLab/distilbert-fa-zwnj-base	68.92	72.04	70.44	85.58	60.64
RoBERTa	HooshvareLab/roberta-fa-zwnj-base	49.47	57.16	53.04	83.69	47.67
DistilBERT	HooshvareLab/distilbert-fa-zwnj-base-ner	68.93	71.19	70.04	86.08	60.40
RoBERTa	HooshvareLab/roberta-fa-zwnj-base-ner	56.06	63.79	59.67	84.16	51.00
ALBERT	HooshvareLab/albert-fa-zwnj-base-v2	62.05	66.71	64.30	83.37	53.71
BERT	HooshvareLab/bert-fa-base-uncased	**73.75**	77.51	**75.58**	**86.85**	**65.97**
BERT	HooshvareLab/bert-fa-zwnj-base-ner	71.30	74.91	73.06	86.11	62.60
BERT	HooshvareLab/bert-fa-zwnj-base	71.13	75.51	73.25	86.55	63.71
ALBERT	m3hrdadfi/albert-fa-base-v2-clf-persiannews	60.07	63.97	61.96	83.69	53.53
MBERT	persiannlp/mbert-base-parsinlu-entailment	66.30	73.43	69.68	85.73	59.07

Table 2 Comparison of different potential errors in incorrect inferences bwtween two top models. Model names with 2L_2G, indicate the addition of two linear and GELU activation layers. Lower values are preferable.

Model name	Intent mismatch	True O false slot	True slot false O	Slot mismatch
sharif-dal/dal-bert	**375**/2974	541/21834	**561**/21834	**387**/21834
sharif-dal/dal-bert_2L_2G	400/2974	510/21834	650/21834	578/21834
HooshvareLab/bert-fa-base-uncased	391/2974	**506**/21834	633/21834	392/21834
HooshvareLab/bert-fa-base-uncased_2L_2G	393/2974	534/21834	700/21834	705/21834

The results indicate that adding two extra linear and GELU activation layers does not provide an advantage over the simpler architecture with a dropout layer and a linear layer. Although based on Table 2 the "**sharif-dal/dal-bert**" showed better performance, the overall differences are minimal, and initial results in Table 1 suggested that "**HooshvareLab/bert-fa-base-uncased**" model has best overall performance. Therefore, we selected "**HooshvareLab/bert-fa-base-uncased**" as our main model.

XX. Main results

After identifying best model, two separate tests were conducted to evaluate the impact of data augmentation and model enhancement on MODERN ISC dataset which has fewer samples than MASSIVE dataset, making it a suitable benchmark under that set of conditions.

XXI. Data augmentation results

Table 3 compares various data augmentation scenarios with the results obtained without augmentation. Initially, it appears that word-wise synonym replacement without lexical retention list yields the best results; however this method generates the most samples, skewing the comparison. To accurately assess the impact of other states, we equalized the sample sizes to align with the number of prompt-generated samples as detailed in Table 4.

By equalizing the number of training samples, the differences between word-wise synonym replacement methods (with and without using lexical retention list) become clear. Replacing only one word at a time better preserves meaning compared altering the entire input. While avoiding lexical retention list improves slot prediction, it results in 1% drop in exact match accuracy

compared to using them. To further investigate the impact of lexical retention list, we tested different model enhancement techniques under both conditions.

XXII. Model enhancement results

Using the best results from Table 4 as a reference, we tested various enhancing strategies under different data augmentation methods, with the results shown in Table 5.

Experiments detailed in Table 5 demonstrate performance improvements by increasing model layers. Notably, the mirrored insertion layer enhancement method, which employs lexical retention list significantly, outperforms the model without layer insertion across several metrics: slot precision, slot recall, slot F1 score, intent accuracy and exact match improved by 2.29%, 2.22%, 2.26% 0.48% and 8.1% respectively. This indicates that increasing model depth enhances its ability to understand semantic meaning relationships among input utterance tokens.

Table 3 The effect of different data expansion methods on evaluation metrics with no sample limitation.

Data expansion type	**No. of synonyms**	**No. of samples**	**lexical retention list usage**	**Slot precision**	**Slot recall**	**Slot F1 score**	**Intent accuracy**	**Exact match**
No data expansion	0	1943	No	84.22	87.5	85.83	92.85	64.76
Word-wise synonym replacement	1	16062	Yes	89.78	91.52	90.64	92.85	74.76
	1	33770	No	**92.21**	**93.75**	**92.97**	**94.28**	**82.38**
Sentence-wise synonym replacement	1	1942	Yes	72.84	76.38	74.57	88.57	45.23
	1	1942	No	72.00	75.00	73.46	81.90	45.23
Data augmentation via prompting	0	3000	No	80.64	83.88	82.23	92.85	58.57

Table 4 Different data expansion methods with nearly equal amounts of training samples.

Data expansion type	**No. of samples**	**lexical retention list usage**	**Slot precision**	**Slot recall**	**Slot F1 score**	**Intent accuracy**	**Exact match**
No data expansion	1943	No	84.22	87.5	85.83	**92.85**	64.76
Word-wise synonym replacement	3000	Yes	86.49	88.05	87.26	91.42	**68.57**
	3000	No	**87.5**	**89.44**	**88.46**	91.42	67.61
Sentence-wise synonym replacement	3000	Yes	77.48	82.22	79.78	87.61	54.76
	3000	No	83.26	85.69	84.46	90.00	57.61
Data augmentation via prompting	3000	No	80.64	83.88	82.23	**92.85**	58.57

Table 5 Comparision of model enhancing techniques with best data augmentation methods.

Architecture	**lexical retention list usage**	**Slot precision**	**Slot recall**	**Slot F1 score**	**Intent accuracy**	**Exact match**
Best method with Normal architecture (12 layers)	No	87.5	89.44	88.46	91.42	67.61
	Yes	86.49	88.05	87.26	91.42	68.57
Interlayer insertion	No	88.31	90.27	89.25	91.90	73.33
	Yes	88.32	89.30	88.81	90.47	67.14
Complete insertion	No	87.38	89.44	88.40	91.42	68.57
	Yes	88.34	90.55	89.43	**92.38**	67.61
Mirrored insertion	No	88.38	89.86	89.11	**92.38**	71.90
	Yes	**89.79**	**91.66**	**90.72**	91.90	**75.71**

XXIII. Conclusion

We demonstrated that intent detection and slot filling tasks can be effectively addressed via a joint training approach. We selected the best publicly available pretrained BERT-based language model for FARSI language. Additionally, we explored dataset expansion and showed that dataset augmentation is both a viable and an effective strategy for improving results. Along with data augmentation, we demonstrated making use of lexical retention list is benefitial for keeping the augmented data from semantic meaning divergence. We also found that enhancing models can better capture contextual representations. Together, these strategies offer a reliable solution that does not require the extensive processing resources associated with large language models. Future works could focus on multiple intent detection and utilizing intents to predict slots and vice versa.

XXIV. Ablation study

XXV. Effect of augmented data volume on model's performance

To minimize resource use while fine-tuning candidate models with various data augmentation and model enhancement strategies, we conducted a test comparing different volumes of augmented data on the optimal solution. The results are presented in Figure 5. Comparing the results of 3K samples with larger data sest indicates that increasing the volume of data leads to improved values in slot precision, slot recall and slot F1-score. However the rise in these evaluation metrics is not a consistent upward trend, as it partially levels off at 6K, 9K and 12K sample sizes. Notabaly for 15K sample size dataset the differences become significant, yielding better results at higher values. In contrast, for intent detection, adding more data samples does not enhance performance remaining around 92% on average with nearly 1% variance across all tested sample sizes. In the exact match evaluation metric, results improve as sample size increases, except for 12K sample size. Comparing 15K sample size with 3K shows a 5.71% improvement simply by providing more augmented data.

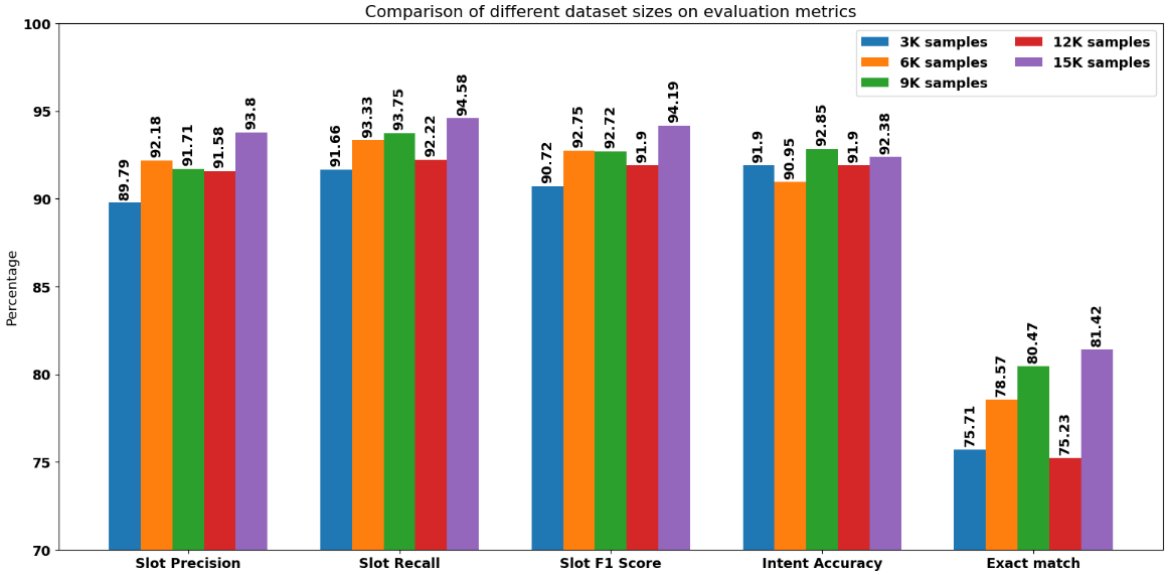

Figure 5 Comparison on performance of different dataset sizes for the optimal solution.

XXVI. BIO tagging order correction & Irrelevant slot elimination

Observing the generated tags at inference time, we discerned that in some cases the trained model fails to preserve true BIO tagging order, causing it to have dropped performance while filling the slots. To overcome this, we implemented a correction method in which it keeps track of produced slots by the model and then corrects the slots that are not tagged in the correct order by using a pre-written code. The reason behind using this strategy lies behind resource efficiency and problem simplicity. We also observed that in some cases, unrelated slots are selected with respect to the identified intent. To prevent unrelated slots to be seen at inefernce time, first we acquired the list of each intent slots and then tried to look for predicted slots among valid slot values for that intent. Invalid slots then got replaced with O token. The result of using these enhancement techniques are shown in Figure 6. The yielded results resemble no difference in intent accuracy and exact match evaluation metrics. On the other hand, in slot precision and slot recall, there are subtle differences. Empowering BIO tagging order correction seems to increase slot precision but does not help with slot recall, while using Irrelevant slot elimination tries to refine slot recall but fails in achieving better slot precision.

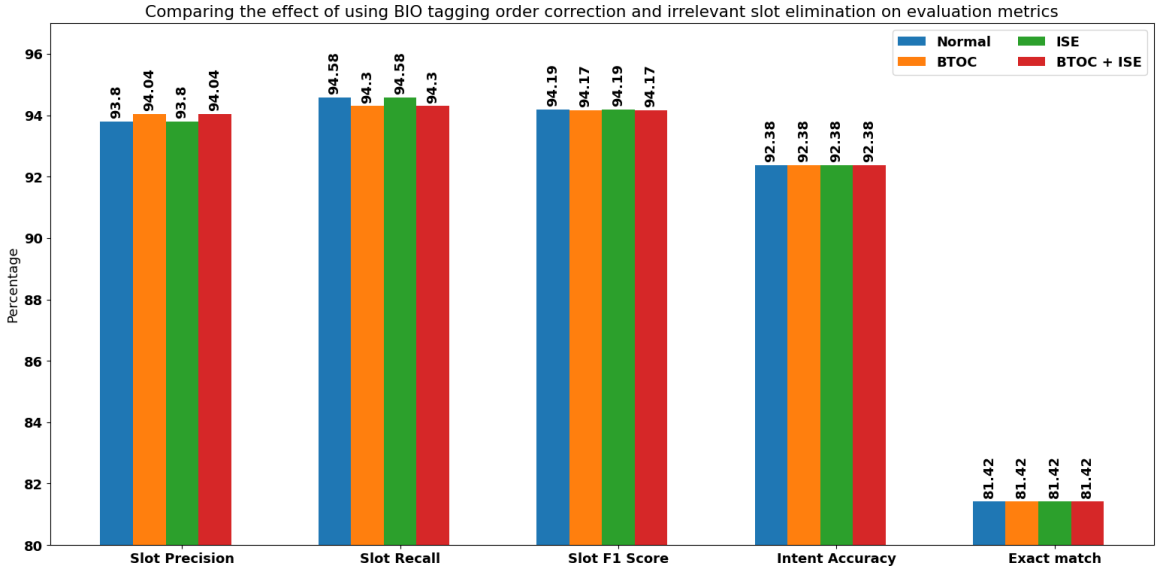

Figure 6 Comparing the effect of using BIO tagging order correction and Irrelevant slot elimination on evaluation metrics in optimal solution trained on 15K samples.'BTOC' and 'ISE' stancd for BIO tagging order correction and Irrelevant slot elimination, respectively

1.1.1.1.1 Acknowledgment

We are thankful to Amirkabir University[4] by establishing free educational courses on natural language processing in finance to give us valuable insights. We would also like to thank MODERN ISC company, both for dataset preparation and granting access to it.

1.1.1.1.2 Acknowledgment of AI Assistance

The literature review, ideas, experiments, analysis of the results, and writing of this article were all carried out by its authors. ChatGPT was used solely to improve the writing in some sentences.

1.1.1.1.3 References

[1] A. Vaswani, "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, 2017.

[2] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *NAACL HLT 2019 - 2019 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 1, no. Mlm, pp. 4171–4186, 2019.

[3] J. Kaplan *et al.*, "Scaling Laws for Neural Language Models," 2020, [Online]. Available: http://arxiv.org/abs/2001.08361

[4] T. B. Brown *et al.*, "Language models are few-shot learners," *Adv. Neural Inf. Process. Syst.*, vol. 2020-Decem, 2020.

[5] H. W. Chung *et al.*, "Scaling Instruction-Finetuned Language Models," pp. 1–54, 2022, [Online]. Available: http://arxiv.org/abs/2210.11416

[6] A. Chowdhery *et al.*, "PaLM: Scaling Language Modeling with Pathways," pp. 1–87, 2022, [Online]. Available: http://arxiv.org/abs/2204.02311

[7] D. Wu, L. Ding, F. Lu, and J. Xie, "SlotRefine: A fast non-autoregressive model for joint intent detection and slot filling," *EMNLP 2020 - 2020 Conf. Empir. Methods Nat. Lang. Process. Proc. Conf.*, pp. 1932–1937, 2020, doi: 10.18653/v1/2020.emnlp-main.152.

[8] M. Firdaus, A. Ekbal, and E. Cambria, "Multitask learning for multilingual intent detection and slot filling in dialogue systems," *Inf. Fusion*, vol. 91, no. September 2022, pp. 299–315, 2023, doi: 10.1016/j.inffus.2022.09.029.

[9] L. Qin, W. Che, Y. Li, H. Wen, and T. Liu, "A stack-propagation framework with token-level intent detection for spoken language understanding," *EMNLP-IJCNLP 2019 - 2019 Conf. Empir. Methods Nat. Lang. Process. 9th Int. Jt. Conf. Nat. Lang. Process. Proc. Conf.*, no. 2, pp. 2078–2087, 2019, doi: 10.18653/v1/d19-1214.

[10] N. A. Tu, H. T. T. Uyen, T. M. Phuong, and N. X. Bach, "Joint Multiple Intent Detection and Slot Filling with Supervised Contrastive Learning and Self-Distillation," *Front. Artif. Intell. Appl.*, vol. 372, pp. 2370–2377, 2023.

[11] D. Tavares, P. Azevedo, D. Semedo, and R. Sousa, "Task Conditioned BERT for Joint Intent Detection and Slot-filling".

[12] R. Zadkamali, S. Momtazi, and H. Zeinali, "Intent detection and slot filling for Persian: Cross-lingual training for low-resource languages," *Nat. Lang. Process.*, pp. 1–16, 2024, doi: 10.1017/nlp.2024.17.

[13] G. Arora, S. Jain, Merugu, and Srujana, "Intent Detection in the Age of LLMs".

[14] "Mistral Model." [Online]. Available: https://docs.mistral.ai/getting-started/models/

[15] "Anthropic Claude." [Online]. Available: https://docs.anthropic.com/claude/docs/intro-to-claude

[16] G. P. S. Bhargav *et al.*, "An Approach to Build Zero-Shot Slot-Filling System for Industry-Grade Conversational Assistants," no. Llm, 2024, [Online]. Available: http://arxiv.org/abs/2406.08848

[17] P. Mirza, V. Sudhi, S. R. Sahoo, and S. R. Bhat, "ILLUMINER: Instruction-tuned Large Language Models as Few-shot Intent Classifier and Slot Filler," *2024 Jt. Int. Conf. Comput. Linguist. Lang. Resour. Eval. Lr. 2024 - Main Conf. Proc.*, pp. 8639–8651, 2024.

[4] https://aut.ac.ir/

[18] "Falcon-7B-Instruct." [Online]. Available: https://huggingface.co/tiiuae/falcon-7b-instruct
[19] "BLOOMZ", [Online]. Available: https://huggingface.co/bigscience/bloomz-7b1
[20] B. Workshop *et al.*, "BLOOM: A 176B-Parameter Open-Access Multilingual Language Model," 2022, [Online]. Available: http://arxiv.org/abs/2211.05100
[21] Y. Hou, C. Chen, X. Luo, B. Li, and W. Che, "Inverse is Better! Fast and Accurate Prompt for Few-shot Slot Tagging," *Proc. Annu. Meet. Assoc. Comput. Linguist.*, pp. 637–647, 2022, doi: 10.18653/v1/2022.findings-acl.53.
[22] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," pp. 2–6, 2019, [Online]. Available: http://arxiv.org/abs/1910.01108
[23] Y. Liu *et al.*, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," no. 1, 2019.
[24] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: a Lite Bert for Self-Supervised Learning of Language Representations," in *8th International Conference on Learning Representations, ICLR 2020*, 2020, pp. 1–17.
[25] Q. Chen, Z. Zhuo, and W. Wang, "BERT for Joint Intent Classification and Slot Filling," 2019, [Online]. Available: http://arxiv.org/abs/1902.10909
[26] W.-T. Kao, T.-H. Wu, P.-H. Chi, C.-C. Hsieh, and H.-Y. Lee, "BERT's output layer recognizes all hidden layers? Some Intriguing Phenomena and a simple way to boost BERT," 2020, [Online]. Available: http://arxiv.org/abs/2001.09309
[27] D. Shen, "FoundationLayerNorm: Scaling BERT and GPT to 1,000 Layers," pp. 1–7.
[28] J. Liu, Y. Chen, and J. Xu, "Low-Resource NER by Data Augmentation With Prompting," *IJCAI Int. Jt. Conf. Artif. Intell.*, pp. 4252–4258, 2022, doi: 10.24963/ijcai.2022/590.
[29] M. Chen *et al.*, "Weakly Supervised Data Augmentation Through Prompting for Dialogue Understanding," pp. 1–17, 2022, [Online]. Available: http://arxiv.org/abs/2210.14169
[30] A. Wang, Ben Komatsuzaki, "GPT J 6B." [Online]. Available: https://github.com/kingoflolz/mesh-transformer-jax
[31] Y. Li, H. Su, X. Shen, W. Li, Z. Cao, and S. Niu, "DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset," pp. 986–995, 2017, [Online]. Available: http://arxiv.org/abs/1710.03957
[32] "Llama 3.1", [Online]. Available: https://huggingface.co/meta-llama/Llama-3.1-70B
[33] O. AI, "Chat GPT 4o mini", [Online]. Available: https://chatgpt.com/?model=gpt-4o-mini
[34] A. Yang *et al.*, "Qwen2 Technical Report," pp. 1–26, 2024, [Online]. Available: http://arxiv.org/abs/2407.10671
[35] J. Fitzgerald, M. Britan, A. Sanchez, and L. Crist, "Understanding Dataset with 51 Typologically-Diverse Languages," 2022.
[36] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *7th Int. Conf. Learn. Represent. ICLR 2019*, 2019.
[37] D. Hendrycks and K. Gimpel, "Gaussian Error Linear Units (GELUs)," *arXiv Prepr. arXiv1606.08415*, 2016, [Online]. Available: http://arxiv.org/abs/1606.08415

Mohammad mahdi HajiRamezan Ali was born in 1995 in Iran. He received his BSc degree in software engineering from Shahab Danesh University 2017, and has started his MSc in Artificial intelligence and robotics from Shahab Danesh University since 2023. His research interests in NLP field includes question answering, task-oriented dialogue systems, and joint intent detection and slot filling.
https://orcid.org/0009-0007-4226-5604

M. Mehdi Homayounpour was born in 1960 in Iran. He received his BSc degree in Electronics from Amirkabir University of Technology in 1986, MSc in Telecommunications from Khaje Nasireddin Toosi in 1989, and Ph.D. in Electrical Engineering from Paris-11 University, Paris, France. He has been a faculty member of Computer Engineering Department at Amirkabir University of Technology (Tehran Polytechnics), Tehran, Iran, since 1995 and a is now faculty member of Shahab Danesh University. His research interests include signal and speech processing, machine learning, natural language processing, and hardware design.
https://orcid.org/0009-0003-1562-932X