Learning Better Intent Representations for Financial Open Intent Classification

Xianzhi Li[1,2]**, Will Aitken**[1,2]**, Xiaodan Zhu**[1,2] **and Stephen W. Thomas**[3]

[1]Department of Electrical and Computer Engineering, Queen's University
[2]Ingenuity Labs Research Institute, Queen's University
[3]Smith School of Business, Queen's University
{21xl17, 16wca, xiaodan.zhu, stephen.thomas}@queensu.ca

Abstract

With the recent surge of NLP technologies in the financial domain, banks and other financial entities have adopted virtual agents (VA) to assist customers. A challenging problem for VAs in this domain is determining a user's reason or intent for contacting the VA, especially when the intent was unseen or *open* during the VA's training. One method for handling open intents is adaptive decision boundary (ADB) post-processing, which learns tight decision boundaries from intent representations to separate known and open intents. We propose incorporating two methods for supervised pre-training of intent representations: prefix-tuning and fine-tuning just the last layer of a large language model (LLM). With this proposal, our accuracy is 1.63% - 2.07% higher than the prior state-of-the-art ADB method for open intent classification on the banking77 benchmark amongst others. Notably, we only supplement the original ADB model with 0.1% additional trainable parameters. Ablation studies also determine that our method yields better results than full fine-tuning the entire model. We hypothesize that our findings could stimulate a new optimal method of downstream tuning that combines parameter efficient tuning modules with fine-tuning a subset of the base model's layers.

1 Introduction

As the popularity of virtual agent (VA) dialogue systems increases and their application in the finance domain is explored, the problem of intent classification demands greater attention. Several recent finance-specific VAs leverage technical advancements to respond to natural language queries (Galitsky and Ilvovsky, 2019; Khan and Rabbani, 2020). Determining the user's intent ensures that the VA can appropriately tailor its responses and/or perform relevant actions. Initial works in intent classification limited the task to classifying utterances as one of N known intents

Utterance	Label
When will I get my card?	Card Arrival
What exchange rates do you offer?	Exchange Rate
My card hasn't arrived yet.	Card Arrival
Is it a good time to exchange?	Exchange Rates
...	...
Is it possible to get a refund?	**Open**
Why has my withdrawal not posted?	**Open**

Table 1: Example user utterances and associated intent labels from banking77 dataset (Casanueva et al., 2020). In this example, only Card Arrival and Exchange Rate intents were known in training and thus refund and withdrawal related requests are Open intents in this context.

and achieved high accuracy (Weld et al., 2021). However, as depicted in Table 1, real-world applications often encounter intents unseen in training data that can be considered as *open* in the current context. Accounting for the open class establishes an $(N + 1)$-class classification task (Shu et al., 2017), where the open class is used as a label for any unidentified intent.

An optimal classifier for this problem must balance correctly labelling known-class utterances while avoiding mistakenly classifying open utterances as one of the known classes. (Zhang et al., 2021a) addresses this problem by proposing a novel loss function to learn an adaptive decision boundary (ADB) for each known intent. At inference, samples that do not fall within any ADB are classified as open. Compact intent representations are required as input for the ADB post-processing learning step and in the case of (Zhang et al., 2021a) the representations are learnt by fine-tuning the last layer of BERT (Devlin et al., 2019). Since most intent classification methods require post-processing on intent representations, our work focuses on deriving richer representations by leveraging large language models (LLM) in an efficacious manner while still minimizing

trainable parameters.

Following the introduction of the transformer in (Vaswani et al., 2017a), an influx of LLM architectures have continually progressed state-of-the-art (SOTA) performance on many natural language processing (NLP) tasks (Otter et al., 2021). Usually these models are pre-trained on a general self-supervised learning task, after which they are fine-tuned for a specific task. Fine-tuning such a model can be computationally prohibitive due to the immense number of trainable parameters. Furthermore, (Kaplan et al., 2020) found that the most important factor for LLM performance is likely model size, indicating that development of even larger models is probable. Inspired by in-context prompting, (Li and Liang, 2021) proposed prefix tuning as a parameter efficient alternative to fine-tuning for natural language generation (NLG). The LLM's parameters are frozen and trainable prefix tokens are prepended to the input sequence. Prefix-tuning has been adapted to natural language understanding (NLU) and performs comparably to full fine-tuning across scales and tasks (Liu et al., 2022).

We achieve SOTA results by augmenting the pre-training architecture of ADB open intent classification (Zhang et al., 2021a) with prefix-tuning. The combination of prefix-tuning with fine-tuning only the last transformer layer was motivated by (Kumar et al., 2022), which discovered that fine-tuning the entire model can distort pre-trained features. We find that alone, both prefix-tuning or fine-tuning the last layer under-performs fine-tuning all of BERT but when trained in tandem, exceeds full fine-tuning.

The rest of this paper is structured as follows: Section 2 summarizes prior works in both intent classification and parameter efficient tuning (PET). Our methodology and model architecture are defined in Section 3. In Sections 4 and 5 respectively, we provide our experimentation structure and corresponding results as well as several ablations. We finish with a conclusion and brief discussion regarding limitations and ethics.

2 Related Works

2.1 Financial Virtual Agents

The effectiveness of VAs has led to their adoption in the financial domain. (Galitsky and Ilvovsky, 2019) demonstrated an exemplary session with a financial VA where the user queried for invest-

ment advice. CalFE leverages commercial chatbot frameworks to train a finance-specific VA (Khan and Rabbani, 2020). (Ng et al., 2020) evaluates the impact of a VA's social presence on usage intention in VAs for finance. All of these works require extracting intent from user utterances.

2.2 Intent Detection

Intent classification is a well-established NLU task but most research limits the problem to known classes (Zhang et al., 2019; E et al., 2019; Qin et al., 2019; Zhang et al., 2021b). While having prior knowledge of all expected intents is ideal, this is rarely possible in a production environment, especially for new dialogue systems. More realistically, a subset of intents are anticipated and new intents are discovered after deployment. (Brychcín and Král, 2017) recognized the challenge of identifying intents prior to training and proposed an unsupervised method to group intents, but by doing so, likely ignored information available in the already identified intents. (Xia et al., 2018) employed zero-shot learning to identify emerging intents but used an LSTM which is hindered by non-parallelized learning and challenges in propagating long-range dependencies. The same issue is present in DeepUnk, a BiLSTM-based intent classification method using margin loss (Lin and Xu, 2019). (Zhan et al., 2021) shared our open intent classification problem formulation but synthetically generated out-of-domain samples for training which may not be as realistic as a fine-grained open class representation.

Our work directly extends the ADB approach to establishing an open class representation (Zhang et al., 2021a). The novelty of our adaptation is in leveraging prefix tuning in combination with partial fine-tuning to improve the pre-training of known intent representations without drastically increasing the number of trainable parameters. In parallel with our work, (Zhang et al., 2022) extended their ADB approach to learn distance-aware intent representations. Doing so resulted in comparable performance to our modification of their original approach. However, our tuning method is model-agnostic and can easily be incorporated with their distance-aware representation learning, likely improving the SOTA further.

2.3 Parameter Efficient Tuning

The desire for PET quickly emerged following the introduction of LLMs. Adapter modules in-

sert task-specific parameters sequentially between transformer layers while the rest of the model remains frozen (Houlsby et al., 2019). (Li and Liang, 2021) and (Lester et al., 2021) simultaneously substantiated the efficacy of prepending tokens to attention mechanisms as a means of efficient tuning. In (Li and Liang, 2021), the prefixes are applied at each layer of the transformer while (Lester et al., 2021) only prepends to the input sequence. (Liu et al., 2022) applied the same method to NLU tasks using deep prefixes with optional reparameterization. Without reparameterization, simple embeddings are learnt for the prefixes. Reparameterization inserts a multilayer perceptron (MLP) between the embeddings and prefix tokens which allows for more complex embeddings.

Recently, (He et al., 2022) determined the theoretical impact of various PET methods and deduced that they are all modifications of a similar function. Allocating additional parameters to other PET modules as suggested by (He et al., 2022) could optimize intent representation beyond what is possible with prefixes alone. For now we limit our work to the most efficient method for low resource settings, prefix-tuning. To the best of our knowledge, this is the first PET work to combine partial fine-tuning with prefix-tuning.

3 Methodology

In this section we explain our procedure for open intent classification. Section 3.1 describes prefix-tuning, the method we supplement partial fine-tuning with. Section 3.2 provides a brief summary of training the original ADB method that we have extended (Zhang et al., 2021a).

3.1 Prefix-Tuning

Prefix-tuning prepends trainable prefix tokens P_k, P_v in front of Key and Value vectors of multi-head attention in each transformer layer. The attention mechanism is applied to the concatenation of prefix and original tokens. Equation 1 details the computation.

$$head = Softmax(Q * Concat(P_k, K)^T) \\ * Concat(P_v, V) \tag{1}$$

Where Q, K, and V are the Query, Key and Value matrices from the original transformer (Vaswani
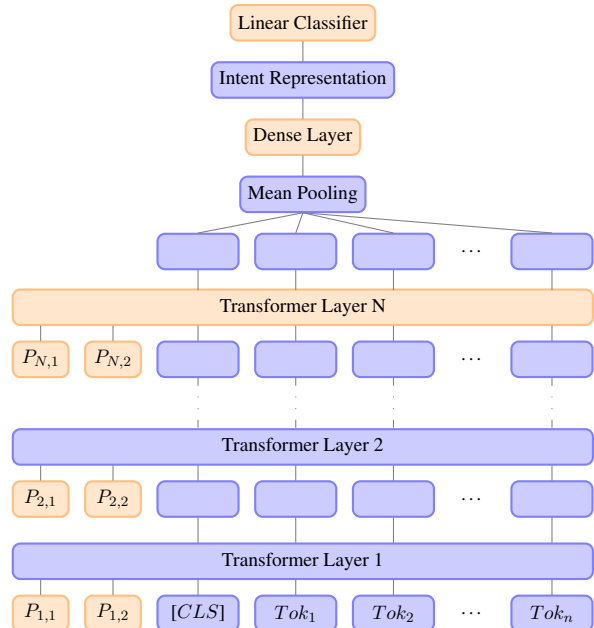

Figure 1: Pre-training architecture for learning intent representations. Orange blocks denote trainable parameters while blue are fixed. For this concrete example, the prefix length has been set to two, but this value is a tunable hyperparameter.

et al., 2017b). P_k and P_v are the additional prefix tokens and are prepended to the Key and Value matrices.

Often, prefix-tuning methods use a MLP to reparameterize the prefix since directly embedding can lead to unstable training and performance decrease (Li and Liang, 2021). However, (Liu et al., 2022) found that for NLU tasks, the efficacy of reparameterization is dependent on the task. From our experiments, we determine that reparameterizing the prefixes is crucial for intent classification. Following training, the MLP weights and biases from reparameterization are dropped and only prefixes are kept.

3.2 Training

Figure 1 illustrates our pre-training architecture of prefix-tuning plus tuning the last transformer layer to extract intent representations. The orange components of the diagram are trainable and the blue are frozen. This example shows a prefix length of two, but the length is a flexible hyperparameter. We detail our entire hyperparameter settings in Section 4.2. The outputs of BERT are first fed into a mean-pooling function to aggregate the sequence into a single vector x_i as described by Equation 2:

KIR	Method	BANKING		OOS		StackOverflow	
		Accuracy	F1-Score	Accuracy	F1-Score	Accuracy	F1-Score
25%	DeepUnk	64.21	61.36	81.43	71.16	47.84	52.05
	$(K+1)$-way	74.11	69.93	/	/	68.74	65.64
	ADB	78.85	71.62	87.59	77.19	86.72	**80.83**
	PFT-ADB	**80.14**	**72.86**	**88.03**	**78.85**	**87.60**	80.78
50%	DeepUnk	72.73	77.53	83.35	82.16	58.98	68.01
	$(K+1)$-way	72.69	79.21	/	/	75.08	78.55
	ADB	78.86	80.90	86.54	85.05	86.40	85.83
	PFT-ADB	**80.40**	**82.44**	**87.60**	**86.87**	**87.06**	**86.22**
75%	DeepUnk	78.52	84.31	83.71	86.23	72.33	78.28
	$(K+1)$-way	81.07	86.98	/	/	81.71	85.85
	ADB	81.08	85.96	86.32	88.53	82.78	85.99
	PFT-ADB	**82.76**	**87.35**	**88.94**	**90.93**	**83.46**	**86.61**

Table 2: Main results for known intent ratios (KIR) 25%, 50%, and 75% on BANKING, OOS, and StackOverflow datasets. Average accuracy and macro F1-Score are reported over all classes.

$$x_i = mp([CLS], Tok_1, Tok_2, ..., Tok_M) \quad (2)$$

where i refers to the current training sample. A dense layer transforms the vector to the intent representation feature space and the resultant vector is finally passed to a linear classifier. We pre-train on known intents and their labels with softmax as the loss function to optimize both the prefix tokens and the last transformer layer. Equation 3 is the softmax loss:

$$Loss = -\frac{1}{n} \sum_{c=1}^{n} \log\left(\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}\right) \quad (3)$$

where n is the batch size and z_j refers to the output logits of j_{th} class.

Following pre-training, the intent representations are extracted from our model for ADB post-processing. ADB learns a tight spherical decision boundary for each known intent. At inference, intent representations that fall outside of all decision boundaries are classified as open. For clarification, the only alteration to the ADB method we employ is the addition of prefix tokens in Figure 1. See (Zhang et al., 2021a) for more information regarding decision boundaries and other training details.

4 Experiments

4.1 Datasets

BANKING: A dataset of 77 banking intents with samples summing to 13,083 banking-specific customer service queries (Casanueva et al., 2020).

It is also commonly referred to as "banking77" but (Zhang et al., 2021a) uses "BANKING" and since we are comparing our results primarily with them, we conform to their choice.

OOS: A subset of CLINC50 specifically designed for out-of-scope intent prediction (Larson et al., 2019) with 22,500 and 1,200 in and out of domain samples respectively over 150 different intents spanning 10 domains.

StackOverflow: The processed version of the StackOverflow dataset (Xu et al., 2015), which has 20 different intents and 1,000 samples for each.

4.2 Experiment Settings

In accordance with previous methods, we sample 25%, 50%, and 75% of intent classes randomly during training as the "known" classes. The remaining are set aside as open classes and removed from training sets. We use BERT (bert-base-uncased) provided by Hugging Face (Wolf et al., 2020) to extract intent representations from utterances. The learning rate for prefixes and transformer parameters is set to 2e-5 since experimenting with setting different learning rates for prefixes and last layer of transformer did not consistently lead to a performance increase. All experiments are conducted on a NVIDIA 2080TI GPU. To fairly compare our method, we keep other hyperparameters the same as (Zhang et al., 2021a). For all results we average performance over ten random seeds.

KIR	Method	BANKING		OOS		StackOverflow	
		Open	Known	Open	Known	Open	Known
25%	DeepUnk	70.44	60.88	87.33	70.73	49.29	52.60
	$(K+1)$-way	80.12	69.39	/	/	74.86	63.80
	ADB	84.56	70.94	91.84	76.80	90.88	**78.82**
	PFT-ADB	**85.65**	**72.19**	**92.11**	**78.50**	**91.58**	78.62
50%	DeepUnk	69.53	77.74	85.85	82.11	43.01	70.51
	$(K+1)$-way	67.26	79.52	/	/	71.88	79.22
	ADB	78.44	80.96	88.65	85.00	87.34	85.68
	PFT-ADB	**80.02**	**82.51**	**89.34**	**86.83**	**88.17**	**86.02**
75%	DeepUnk	58.54	84.75	81.15	86.27	37.59	81.00
	$(K+1)$-way	60.71	87.47	/	/	65.44	87.22
	ADB	66.47	86.29	83.92	88.58	73.86	86.80
	PFT-ADB	**69.18**	**87.66**	**86.80**	**90.96**	**74.78**	**87.40**

Table 3: Open and known comparison of main results for known intent ratios 25%, 50%, and 75% on BANKING, OOS, and StackOverflow datasets. F1-Score and macro F1-Score are reported for open class and known classes respectively.

Regarding prefix-specific settings, we use reparameterization with a hidden size of 512 unless otherwise specified. The overall parameter size is determined by the prefix length. In this task, we found that enlarging the prefix length did not lead to a consistent performance increase due to its low-rank bottleneck. (He et al., 2022) also discusses that allocating additional parameter in self-attention is only worthwhile if they make up less than 0.1% of the parameter budget. Therefore, we choose our default prefix length as 10, which equates to roughly 0.1% of BERT's trainable parameters.

4.3 Baselines

We compare our results to the most competitive open intent classification methods: DeepUnk (Lin and Xu, 2019), $(K+1)$-way (Zhan et al., 2021), and the ADB method we directly extend (Zhang et al., 2021a). The DeepUnk results are taken from (Zhang et al., 2021a) which replaced the BiLSTM with BERT to generate intent representations for fair comparison. (Zhan et al., 2021) also uses BERT as its encoder but keeps just the CLS token's final hidden state instead of pooling the entire sequence. (Zhan et al., 2021) did not test on the same OOS split and cells corresponding to that configuration are left blank for tables in Section 5.

5 Results

Our main results and respective baseline comparisons are presented in Tables 2 and 3. Table 2 is limited to accuracy averaged over all classes,

including the open class and macro F1 over the same set of classes. For a fine-grained analysis of open intent performance, Table 3 contrasts the F1 score of the open class with the macro F1 over the remaining known classes. *PFT-ADB* denotes our method of adding prefix tuning to ADB and the best result for each section is in boldface.

For each dataset we tested, PFT-ADB improves performance on all prior methods with the minor exception of StackOverflow F1-Score and known score. Specifically, as shown in Table 2, we achieve accuracy improvements of (1.63%, 1.95%, 2.07%) on BANKING, (0.50%, 1.22%, 3.03%) on OOS, and (1.01%, 0.76%, 0.82%) on StackOverflow for known intent ratios (25%, 50%, 75%). The consistency of our results across configurations suggests that paying closer attention to pre-training intent representations can enhance the distinction of decision boundaries in the post-processing step. Additionally, we do not add a significant number of trainable parameters to existing methods (only 0.1%), successfully avoiding trading substantial costs for performance increase. Note that our results are comparable to that of the most recently released DA-ADB (Zhang et al., 2022) model. We believe that due to their orthogonal nature, DA-ADB and our approach could be combined together for further performance improvements.

We note that the dataset with the lowest performance gain is StackOverflow. (Zhang et al., 2021a) found that their novel post-processing method, ADB, was most effective on this dataset

Method	Accuracy	F1-score	Open	Known
Emb	64.40	68.56	66.62	68.38
MLP	81.56	85.89	75.98	85.97
Emb+12th L	86.40	88.15	84.44	88.19
MLP+12th L	**90.07**	**91.52**	**88.48**	**91.54**
FFT-NoPT	87.56	89.20	85.71	89.24
ADB	86.32	88.53	83.92	88.58

Table 4: Experiments on the impact of different prefix encoding approaches with and without fine-tuning the last layer of transformer. We use OOS dataset with 75% known Intent Ratio. "Emb" refers to embedding-only method. "MLP" refers to method that uses 2 layers of MLP to encode prefix. "+12th L" means we unfreeze the last layer of transformer. "FFT-NoPT" denotes full fine-tuning without any prefixes.

Method	Length	Accuracy	F1-score	Open	Known
Emb+12th L	10	87.60	89.08	85.82	89.11
	20	**88.25**	**89.49**	**86.80**	**89.52**
	30	87.88	89.44	86.13	89.47
	50	85.70	87.49	83.72	87.53
	80	85.93	87.75	83.87	87.78
	100	87.95	89.45	86.16	89.48
MLP+12th L	10	**90.16**	91.57	**88.57**	91.60
	20	89.67	91.27	87.96	91.30
	30	90.05	**91.59**	88.35	**91.62**
	50	88.65	90.34	86.82	90.37
	80	89.49	91.27	87.51	91.30
	100	89.84	91.51	88.09	91.54

Table 5: Results of tuning with different prefix lengths. We use OOS dataset with 75% known intent ratio.

compared to prior methods. They hypothesized that this was due to being able to form tighter decision boundaries for the technical jargon more prevalent in StackOverflow. Following this reasoning, it could be that for this dataset the post-processing method is paramount and enriching the intent representations alone is not enough to yield a substantial performance improvement.

It is important that an open intent classification method balances the performance on known classes while still identifying open intents. Table 3 verifies that despite changing pre-training tuning methods, ADB post-processing still adequately addresses this issue. The performance increase is consistent between both the open class and known classes for each dataset indicating that prefix-tuning does not interfere with optimizing both aspects of the open intent problem. Again, we anticipate that combining PFT-ADB with the newer DA-ADB could result in even better performance.

The following ablations focus on the OOS dataset since it covers multiple domains and we wanted to generalize beyond just the financial domain.

5.1 Effect of Reparameterization and Tuning Variations

In Table 4 we show that under the same dataset and known intent ratio, performance varies considerably when adopting MLP as prefix encoder. In the first row, the embedding-only method leads to poor results of 64.40% accuracy. Contrarily, introducing a 2 layer MLP to encode prefixes increases the performance by around 15%. More impor-

tantly, the result is stable and reproducible. It indicates that using MLP to reparameterize prefixes is crucial in obtaining a consistent performance.

Results using prefix tuning alone (rows 1 and 2) in this task are slightly worse than ADB's fine-tuning results. In particular, the performance gap in identifying open intent is more salient, revealing prefix-tuning's lower capacity for out-of-scope classification. However, when we incorporate prefix tuning along with tuning the last layer of transformer, we find a surprisingly large performance increase. For embedding and MLP methods, tuning the last layer of transformer gives a performance boost to 86.40% and 90.07%, respectively, with only additional 0.1% of ADB's parameters. Since the latter transformer layer captures high-level feature of utterances, we believe that this small amount of parameter steer the higher layers to learn more task-oriented information as well as fit intents into a better-distributed latent space.

We also try the common method of fully fine-tuning, i.e., unfreezing all of BERT's parameters which was not done in (Zhang et al., 2021a). The performance is still 1% lower than our method while we use only 8.1% of parameters.

5.2 Impact of Prefix Lengths

We experimented with the prefix length to determine its effect on performance. From Table 5, we observe that with the increase of the prefix length from 10 to 100 (parameter size from .1% to 1.6%), the results do not follow the same ascending pattern. We argue that simply adding more prefix tokens would not lead to a consistent performance boost due to its bottleneck. (He et al., 2022) determined that prefix tuning is another form of low-rank update, which cannot make use of more than

x	Just x		x and Rest	
	Accuracy	F1-Score	Accuracy	F1-Score
No-FT	81.56	85.89	81.56	85.89
12	**90.07**	**91.62**	90.07	91.62
11	89.18	91.19	**90.21**	**91.81**
10	89.81	91.80	89.42	91.42
9	88.81	90.89	88.93	91.18
7	87.07	89.79	88.42	90.93
4	85.23	88.21	87.00	89.90
1	84.77	87.73	87.58	90.24

Table 6: Fine-tuning various groupings of transformer layers on OOS with known intent ratio 75%. "No-FT" is prefix-tuning without any fine-tuning. Prefix-tuning configuration was kept constant throughout runs.

Method	Accuracy	F1-Score	Open	Known
Attention	87.49	89.69	85.19	89.73
Feed Forward	86.47	88.92	83.81	88.96
Layer Normalization	86.61	88.81	84.21	88.85
Keys and Values	85.67	88.58	82.39	88.64
Entire Layer	**90.07**	**91.52**	**88.48**	**91.54**

Table 7: Fine-tuning components of final transformer layer on OOS with known intent ratio 75%. Only the parameters of the component(s) are tuned and the rest of the layer is frozen.

0.1% of additional parameters.

5.3 Fine-Tuning Different Groupings of Layers

Combining prefix-tuning with fine-tuning a subset of transformer layers is, to the best of our knowledge, a novel approach. Fine-tuning the last layer alone is ideal for minimizing trainable parameters. We aim to determine whether varying which layer or group of several layers is unfrozen can achieve better results than the last layer alone. Table 6 summarizes our findings. The layer of interest is specified with the variable x. "Just x" is fine-tuning layer x alone and "x and Rest" is fine-tuning the layer x and all subsequent layers. Using this notation, "Layer *1* and Rest" is akin to fine-tuning all of BERT. "No-FT" refers to prefix-tuning without any additional fine-tuning. For this row and when x is 12, the results between the two main columns are of course the same.

Several interesting observations are evident in Table 6. Firstly, the fine-tuning of at least one layer in addition to prefix-tuning is strictly necessary for optimal performance. Under the constraint of tuning just a single layer, the last performs the best. The latter layers of the model are where higher-level details of natural language are processed. We hypothesize that tuning this layer best incorporates the propagation of prior prefixes with the base model. Tuning prior layers may have a similar effect, but if the subsequent layers are frozen, the understanding of prompts is obfuscated since the latter frozen layers have no experience attending to prefixes.

Another notable finding is that if performance is to be prioritized, fine-tuning the final two layers together is better than the last layer alone.

This suggests that the prefixes are complex enough such that their value is maximized when the final two layers tune in tandem. However, the trade off of minor performance increase at the cost of doubling the trainable parameters may not be worth it depending on the application.

Lastly, we note that as layers beyond two are trained in the "x and Rest" column, performance begins to degrade. This supports the observation made by (Kumar et al., 2022) that fine-tuning disturbs pre-trained features in the base model. Training only the final layer(s) avoids perturbing low-level semantics learnt in earlier layers of the base model, but still adds sufficient capacity to attend to the prefixes.

5.4 Fine-Tuning Various Components in Last Layer

While fine-tuning only the last transformer layer reduces the trainable parameter count to 8%, this is still a large value compared to the 0.1% parameter count of the prefixes alone. We isolate various components of the last transformer layer to determine if some could be frozen to further reduce parameter count. The results are presented in Table 7. Tuning the entire layer significantly outperformed any other variation, alluding that there is an important relationship between the prefixes and every component of the final transformer layer. Tuning each of the components in the last layer is essential to procure maximum prefix performance.

6 Conclusion

We have shown that incorporating prefix-tuning with the ADB intent representation pre-training method achieves SOTA results in the financial domain on the banking77 benchmark dataset and others. Furthermore, our tuning method does not sacrifice excessive parameters count for the performance gain. The combination of prefix-tuning

with fine-tuning only the last layer of transformer is simple yet novel to the best of our knowledge and surfaces interesting questions regarding the mechanisms they use to interact. We intend to address the limitations presented hereafter in the near future.

Limitations

Despite achieving SOTA results on open intent classification tasks, our work has several facets that could be furnished further. Firstly, we tune the last layer of transformer along with the prefixes, making our method less parameter efficient than prefixes alone. Other approaches to fine-tuning the last layer of the transformer during pre-training should be investigated. Moreover, this work does not include any other PET method such as adapter tuning (He et al., 2021) or LoRA (Hu et al., 2021). We anticipate that using other PET methods will reveal new observations regarding their interaction with partial fine-tuning. We restrict our study to simple single intent dialogues while industry-deployed models would likely encounter noise as well as multiple intents. Testing the robustness of our method under these conditions could be valuable. Lastly, we plan to research whether our success with prefix-tuning in combination with partial fine-tuning generalizes to other NLU and financial tasks.

Ethics Statement

Recent impressive achievements in NLP thanks to the advent of LLMs do not come without cost. Most relevant to our paper is the environmental impact and inequitable distribution of such technologies (Strubell et al., 2019). The resources required to train a LLM are large which from the environmental perspective increases our contribution to climate change and from an equity perspective limits who can access, research, and use the model.

While the self-supervised pre-training step often has the greatest resource requirements, fine-tuning LLMs is undertaken by many more parties following a model's public release. The numerous task-specific deployments of popular models likely have greater net CO_2 emissions than the initial pre-training. Our work directly combats this concern by promoting parameter efficient tuning as an efficacious alternative to relatively expensive fine-tuning. The fraction of trainable parameters reduces tuning memory requirements, in turn reducing power consumption and environmental impact. Additionally, the reduction of required memory enables the adoption of LLMs by those who do not have access to expensive high-quality hardware or cloud platforms. Finally, storing copies of the model for each task is efficient. Only a single copy of the frozen LLM is needed along with the smaller prefixes and in our case, trained last layer of transformer, resulting in similar benefits as the reduction of memory.

References

Tomáš Brychcín and Pavel Král. 2017. Unsupervised dialogue act induction using Gaussian mixtures. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 485–490, Valencia, Spain. Association for Computational Linguistics.

Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on NLP for ConvAI - ACL 2020*. Data available at https://github.com/PolyAI-LDN/task-specific-datasets.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Haihong E, Peiqing Niu, Zhongfu Chen, and Meina Song. 2019. A novel bi-directional interrelated model for joint intent detection and slot filling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5467–5471, Florence, Italy. Association for Computational Linguistics.

Boris Galitsky and Dmitry Ilvovsky. 2019. On a chatbot conducting a virtual dialogue in financial domain. In *Proceedings of the First Workshop on Financial Technology and Natural Language Processing*, pages 99–101, Macao, China.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022. Towards a unified view of parameter-efficient transfer learning. In *International Conference on Learning Representations*.

Ruidan He, Linlin Liu, Hai Ye, Qingyu Tan, Bosheng Ding, Liying Cheng, Jia-Wei Low, Lidong Bing, and

Luo Si. 2021. On the effectiveness of adapter-based tuning for pretrained language model adaptation.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *ICML*, pages 2790–2799.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models.

Shahnawaz Khan and Mustafa Raza Rabbani. 2020. Chatbot as islamic finance expert (caife): When finance meets artificial intelligence. In *Proceedings of the 2020 4th International Symposium on Computer Science and Intelligent Control*, ISCSIC 2020, New York, NY, USA. Association for Computing Machinery.

Ananya Kumar, Aditi Raghunathan, Robbie Jones, Tengyu Ma, and Percy Liang. 2022. Fine-tuning can distort pretrained features and underperform out-of-distribution.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China. Association for Computational Linguistics.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Ting-En Lin and Hua Xu. 2019. Deep unknown intent detection with margin loss. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5491–5496, Florence, Italy. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Magdalene Ng, Kovila P.L. Coopamootoo, Ehsan Toreini, Mhairi Aitken, Karen Elliot, and Aad van Moorsel. 2020. Simulating the effects of social presence on trust, privacy concerns & usage intentions in automated bots for finance. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 190–199.

Daniel W. Otter, Julian R. Medina, and Jugal K. Kalita. 2021. A survey of the usages of deep learning for natural language processing. *IEEE Transactions on Neural Networks and Learning Systems*, 32(2):604–624.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2078–2087, Hong Kong, China. Association for Computational Linguistics.

Lei Shu, Hu Xu, and Bing Liu. 2017. DOC: Deep open classification of text documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2911–2916, Copenhagen, Denmark. Association for Computational Linguistics.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017a. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017b. Attention is all you need.

H. Weld, X. Huang, S. Long, J. Poon, and S. C. Han. 2021. A survey of joint intent detection and slot-filling models in natural language understanding.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Congying Xia, Chenwei Zhang, Xiaohui Yan, Yi Chang, and Philip Yu. 2018. Zero-shot user intent detection via capsule neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3090–3099, Brussels, Belgium. Association for Computational Linguistics.

Jiaming Xu, Peng Wang, Guanhua Tian, Bo Xu, Jun Zhao, Fangyuan Wang, and Hongwei Hao. 2015. Short text clustering via convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 62–69, Denver, Colorado. Association for Computational Linguistics.

Li-Ming Zhan, Haowen Liang, Bo Liu, Lu Fan, Xiao-Ming Wu, and Albert Y.S. Lam. 2021. Out-of-scope intent detection with self-supervision and discriminative training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3521–3532, Online. Association for Computational Linguistics.

Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip Yu. 2019. Joint slot filling and intent detection via capsule neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5259–5267, Florence, Italy. Association for Computational Linguistics.

Hanlei Zhang, Hua Xu, and Ting-En Lin. 2021a. Deep open intent classification with adaptive decision boundary. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14374–14382.

Hanlei Zhang, Hua Xu, Shaojie Zhao, and Qianrui Zhou. 2022. Learning discriminative representations and decision boundaries for open intent detection.

Jianguo Zhang, Trung Bui, Seunghyun Yoon, Xiang Chen, Zhiwei Liu, Congying Xia, Quan Hung Tran, Walter Chang, and Philip Yu. 2021b. Few-shot intent detection via contrastive pre-training and fine-tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1906–1912, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.