Semi-supervised Meta-learning for Cross-domain Few-shot Intent Classification

Judith Yue Li
Salesforce Research / Palo Alto, CA, USA
yuel@alumni.stanford.edu

Jiong Zhang
LinkedIn AI / Sunnyvale, CA, USA
jiozhang@linkedin.com

Abstract

Meta-learning aims to optimize the model's capability to generalize to new tasks and domains. Lacking a data-efficient way to create meta training tasks has prevented the application of meta-learning to the real-world few shot learning scenarios. Recent studies have proposed unsupervised approaches to create meta-training tasks from unlabeled data for free, e.g., the SMLMT method (Bansal et al., 2020a) constructs unsupervised multi-class classification tasks from the unlabeled text by randomly masking words in the sentence and let the meta learner choose which word to fill in the blank. This study proposes a semi-supervised meta-learning approach that incorporates both the representation power of large pre-trained language models and the generalization capability of prototypical networks enhanced by SMLMT. The semi-supervised meta training approach avoids overfitting prototypical networks on a small number of labeled training examples and quickly learns cross-domain task-specific representation only from a few supporting examples. By incorporating SMLMT with prototypical networks, the meta learner generalizes better to unseen domains and gains higher accuracy on out-of-scope examples without the heavy lifting of pre-training. We observe significant improvement in few-shot generalization after training only a few epochs on the intent classification tasks evaluated in a multi-domain setting.

1 Introduction

Recent developments of large scale pre-trained models, such as BERT (Devlin et al., 2019), GPT (Brown et al., 2020) and XLNet (Yang et al., 2020), have significantly advanced the natural language processing (NLP) techniques. However, these models still rely on fine-tuning on a relatively large number of labeled samples (> 1000) to achieve high accuracy even for tasks seen during training (Howard and Ruder, 2018). Recent studies (Brown et al., 2020; Bansal et al., 2019; Dou et al., 2019) have demonstrated that these large language models have the potential to be few shot learners, i.e., capable of adapting to a new task or a new domain by training only on a few examples with the aid of meta-learning. Meta-learning tackles the few-shot learning problem through learning a robust yet flexible representation from a variety of tasks in a so-called meta training stage, so that the model can quickly adapt to new tasks with only a few examples. In addition, random sampling is introduced in the design of meta training tasks to avoid memorization, a phenomenon in which the meta learner memorizes a function that directly associates an input with the label when no real learning occurs (Yin et al., 2019).

Meta-learning approaches such as the optimization-based MAML (Finn et al., 2017), the metric-based Prototypical Networks (ProtoNet) (Snell et al., 2017) and etc., have been successfully applied in NLP domain (Yin, 2020). Dou et al. (2019) successfully applied MAML and its variants to low-resource text classification tasks on the GLUE dataset (Wang et al., 2018). It showed models trained with MAML, first-order MAML and REPTILE (Nichol and Schulman, 2018) outperform strong baseline models such as BERT and MT-DNN (Liu et al., 2015). Bansal et al. (2019) developed a method LEOPARD that generalizes MAML to handle diverse NLP tasks. They used pre-trained BERT (Devlin et al., 2019) as the underlying task-agnostic base model, coupled with a task-dependent softmax classification parameter generator. The meta trained BERT learns better initial parameters, which helped to reach high accuracy across 17 down steam NLP tasks with very few examples per class.

However, successful implementations of meta-learning depend on the availability of a diverse set of tasks with plenty of labeled data during

Proceedings of the 1st Workshop on Meta Learning and Its Applications to Natural Language Processing (MetaNLP 2021), pages 67–75
Bangkok, Thailand (online), August 5, 2021.

meta training. To create meta-learning tasks in a data-efficient manner, a number of papers have tried to explore the idea of unsupervised meta-learning. These methods explore to learn representations through automatically constructing tasks from unlabeled dataset and utilize learned representation functions for specific task prediction. Hsu et al. (2018) proposed to leverage clustering embeddings to construct tasks from unlabeled data and then apply meta-learning method for explicitly optimizing for adapting to new tasks. Khodadadeh et al. (2020) proposed to sample objects with synthetic labels from the latent space and generate meta-tasks using generative models. In the domain of natural language processing, Bansal et al. (2020b) proposed Subset Masked Language Modeling Tasks (SMLMT), which automatically construct self-supervised tasks by masking out certain tokens from sentences as labels to create few shots classification tasks from unlabeled data. The study showed that meta training with these diverse unsupervised tasks can prevent over-fitting to specific supervision tasks, leading to better generalization than pre-training language-model followed by fine-tuning.

In this study, we focus on cross-domain few shot classification with the goal to investigate whether we can meta train a large pre-trained language model (e.g., BERT) in a semi-supervised fashion without access to a large number of labeled data or meta training tasks. The resulting representation should generalize and adapt well to a new domain, and provide clear separations between in-domain and out-of-scope (OOS) examples (Zhang et al., 2020). Our base meta learner consists of an embedding function (e.g., BERT) and ProtoNet (Snell et al., 2017) as the general supervised classifier, which can be fine-tuned either using the supervised N-way K-shot classification tasks (supervised meta training) or together with the self-supervised SMLMT tasks (semi-supervised meta training). We compares classifiers with supervised meta-training against classifiers trained without the diverse meta training tasks. We then compare the semi-supervised meta-learner with the supervised approach without adding additional labeled data. The resulting text representations will be evaluated in terms of their few-shot generalization accuracy, their capability to detect OOS examples, and their ability to adapt when more training examples are included.

While Bansal et al. (2020b) focuses on the cross-problem transfer capability of SMLMT trained with a general-purpose corpus like Wikipedia, our study further investigates the cross-domain transfer capability of SMLMT within a problem, i.e., whether additional self-supervised training on the unlabeled data from the domain of interest (e.g., dialogues) can help generalize a seen problem to a new unseen domain. Moreover, SMLMT as a classification task combines well with metric-based meta learners like ProtoNet (Snell et al., 2017). Compared to optimization-based meta learners like MAML (Finn et al., 2017), ProtoNet is easier to optimize and scale, has a simpler inductive bias therefore works well for very-few-shot classification problems. These properties are complementary to MAML and can provide good initialization for the latter (Triantafillou et al., 2019).

2 Methods

2.1 Model architecture of ProtoNet with BERT

Prototypical networks (ProtoNet) (Snell et al., 2017) is a metric-based meta-learning approach for the problem of few-shot classification, where an encoder model learns to project samples to an embedding space. In stead of training on batches of training data, meta learners are trained on episodes that contain support set D^{tr} for training and query set D^{ts} for evaluation. The support set will be projected to the embedding space to formulate class prototypes c_n, and then classification of the query example is done by computing the softmax of the negative distances between the embedded query and each class prototype.

$$y^{ts} = g(D^{tr}, x^{ts}) = softmax(-d(f_\theta(x^{ts}), c_n)) \quad (1)$$

Compared to optimization-based MAML, ProtoNet is more memory efficient and easy to optimize. Similar to Nearest Neighbor, ProtoNet is a non-parametric method that can be integrated with any embedding function f_θ, where θ is the learnable meta parameters. This method reflects simpler inductive bias and so far it is limited to classification problems.

The design of the embedding function f_θ can vary depending on the NLP applications. For intent classification, we find the best performance can be achieved by integrating the metric-based meta-learning approach ProtoNet with the popular pre-trained model (e.g., BERT (Devlin et al.,

	N=5, K=5		N=5, K=10	
Dataset	Meta-Test Accuracy	Meta-Test Std	Meta-Test Accuracy	Meta-Test Std
1.unseen examples (banking)	0.935	0.044	0.940	0.042
2.unseen examples	0.914	0.056	0.948	0.040
3.unseen classes	0.883	0.060	0.917	0.049
4.unseen domains	0.870	0.066	0.908	0.055

Table 1: Meta test accuracy and standard deviations for ProtoNet on CLINC150 few shot intent classification dataset.

2019), RoBERTa (Liu et al., 2019)). These large pre-trained language models are quite effective for learning task-agnostic features as well as the task-specific representations with proper fine-tuning. We take advantage of this transfer learning feature of these pre-trained models and use it as the embedding function f_θ. Here the meta parameters θ are the weights of the pre-trained model which will be fine-tuned during meta training to learn a task-agnostic representation that should also generalize well to a new domain during meta testing.

2.2 Subset Masked Language Modeling Tasks

With the hope of further improving classification accuracy, we would like to leverage the unlabeled data set through self-supervision during meta training stage. The key for self-supervised meta-learning is how to construct self-supervised tasks and how it can be combined with the supervised tasks. Following the Subset Masked Language Modeling Tasks (SMLMT) approach (Bansal et al., 2020a), we first construct a vocabulary from tokens in all the sentences except those labeled sentences used as hold-out test set and calculate their frequency. To balance the number of tokens and the number of sentences associated to each token, we select tokens appeared from 30 times to 100 times to be labels and then masked these tokens in associated sentences as training samples for SMLMT, with the token as labels. Since SMLMT is also a classification task, the meta-learner introduced in the last section can be used to solve both the self-supervised and the supervised classification tasks, yielding a new semi-supervised meta training approach to tackle the few shot intent classification problem.

2.3 Out-of-Scope Evaluation

In addition to the standard few shot learning evaluation where the model is only evaluated on samples from in-scope class distribution, a more realistic evaluation setting involves the Out-of-Scope (OOS) class, in which samples come from a different distribution, e.g., random utterances not related to any registered intent class in a dialogue.

We adopt the OOS evaluation strategy (Zhang et al., 2020; Larson et al., 2019) which adds an additional OOS class in the meta testing stage, while the meta training stage remains to be the same. A sample is assigned to the OOS class if the probabilistic prediction for the best class is under a specified threshold T with value between 0 and 1. The threshold values is chosen to maximize J_{in_oos} (Equation 4), the sum of In-Domain-Accuracy (A_{in}, Equation 2) and OOS-Recall (R_{oos}, Equation 3).

$$A_{in} = C_{in}/N_{in} \tag{2}$$

where C_{in} is the number of correctly predicted in-domain intent examples and N_{in} is the total number of in-domain intent examples.

$$R_{oos} = C_{oos}/N_{oos}, P_{oos} = C_{oos}/(N'_{oos}) \tag{3}$$

where C_{oos} is the number of correctly predicted OOS intent examples, N_{oos} is the number of OOS intent examples and N'_{oos} is the number of predicted OOS examples.

$$J_{in\_oos} = A_{in} + R_{oos} \tag{4}$$

We also report the OOS precision P_{oos} and OOS F1 score $F1_{oos}$ for an optimized threshold T.

3 Experiments, Results and Discussion

There have been a number of papers that have explored the idea of unsupervised meta-learning,

where tasks are constructed automatically from an unlabeled dataset and a meta-learner is pre-trained on these tasks without using any labeled dataset. Can we extend these ideas to the case where we have a small number of supervised meta-training tasks rather than zero meta-training tasks, to construct a semi-supervised meta-learner? We hope to explore answers to the following questions through experiments: (a) Whether meta training effectively improve domain adaptation? and (b) Will the semi-supervised approach outperform the supervised meta-learning given the same number of labeled data?

3.1 CLINC150 Few Shot Intent Classification

The CLINC150 (Larson et al., 2019) intent classification dataset consists of 150 different intent classes across 10 different domains, i.e., Banking, Credit Cards, Work and Travel. Each domain has 15 tasks, each comes with 150 labeled examples. The data is split in the following ways to evaluate meta-learning for different few shot learning settings:

1. **single domain unseen examples:** Pick only one domain Banking. The training data is sampled from 15 classes from the banking domain, where each class has 100 examples. The validation and testing data is sampled from the same class distributions with 20 and 30 examples per class respectively.

2. **multi-domain unseen examples:** Distribute the 150 classes uniformly among training, validation and testing splits, with a ratio of 100:20:30. The training data is sampled from 150 classes with 100 examples each class from all domains. The validation and testing data consist of 20 and 30 examples per class respectively.

3. **multi-domain unseen classes:** In order to test the model's generalization capability to unseen new classes, the 15 classes under each domain are separated according to 10:2:3 ratio, so that no tasks in testing set or validation set will appear in the training time. The training data, validation set and testing set is sampled from 100, 20, 30 classes among 10 different domains respectively, where each class contains 150 examples.

4. **multi-domain unseen domain:** The problem is made more difficult by creating a data splits in which the training, validation and testing data all come from different domains, which will test whether the model will efficiently adapt to domains unseen. The training data is sampled from 75 classes among 5 different domains (banking, kitchen, home, auto commute and small talk), where each class contains 150 examples. The validation and testing data is sampled from the 2 (utility, credit cards) and 3 (travel, work, meta) domains respectively, where each domain has 15 classes and each class has 150 examples.

We run ProtoNet with BERT on each few shot setting and the results are shown in Table 1. The few shot test accuracy decreases when examples in meta testing time come from a class or a domain that is unseen during meta training time. Increase k or the number of support samples per task improves the test accuracy. For $k = 5$ the best results are achieved by training with learning rate $4e-6$, 6 ways for 300 episodes during meta training. Note the learning rate is reduced by half for every 50 episodes.

3.2 Cross Domain Intent Classification with Limited Labeled Data

A more challenging but realistic few shot learning setting is that during meta training we don't have enough labeled data available per class, and labeled data in the same domain is not available. Yet we have large amount of unlabeled data from the same domain. How will the result change if we reduce the available labeled examples per class during meta training from 150 to 50 or less for the unseen domain set up?

Following the set up of unseen domains, the problem is made more challenging by sampling training tasks from only 25 classes among 5 different domains (banking, kitchen, home, auto commute and small talk), where each class only contains 50 labeled examples. The validation and testing data is sampled from the 2 (utility, credit cards) and 3 (travel, work, meta) other domains respectively, where each domain has 15 classes and each class has 50 examples. The rest of the examples is aggregated into a pool of unlabeled data for unsupervised training. Details about the data splits is shown in Table 2. To evaluate model performance on OOS examples, we also randomly sample OOS intents from the 1200 Out-of-Scope examples that are not belonged to the 150-intent classes provided

	unlabeled	train	valid	test
# domains	10	5	2	3
# classes	223	25	30	45
# examples	11900	1250	1500	2250

Table 2: The data splits for meta training, meta validation and meta testing

	ProtoNet with meta training	
# labeled data per class	Meta Test Acc	Meta Test Std
20	0.832	0.077
50	0.851	0.068
100	0.846	0.073
150	0.864	0.073

Table 3: Meta test accuracy changes with the number of available labeled data per class

by Larson et al. (2019) during meta testing time.

By varying the number of available labeled examples during meta training, we observe how meta test accuracy and standard deviation changes in respond to more labeled training data. As shown in Table 3, increasing the number of labeled samples per class from 20 to 150 improves the test accuracy from 0.832 to 0.864 for 5-way 5-shot learning.

3.3 CLINC150 with ProtoNet + SMLMT

The next research question is whether we can leverage the unlabeled data to improve the meta test accuracy. Here we create the unsupervised tasks following the SMLMT (Bansal et al., 2020b), where additional meta training tasks are created by masking a randomly picked token (here we use [MASK] from BERT's vocabulary) and let the model classifies which token has been replaced. The token is selected to appear at least 30 times in the examples, but no more than 100 times, which filters out common words and leaves enough examples for the model to learn the representations of important words that differentiate different intents. The most important hyperparameters to tune are learning rate, sampling ratio and number of ways during training. Sampling ratio controls when to train with the SMLMT tasks and when to train with the supervised tasks. The best validation accuracy is reached at learning rate $8e-6$, sampling ratio 0.7 and 9 ways during meta training. Here the training "ways" is typically selected to be larger than the testing "ways" to gain good performance (Snell et al., 2017). The details on the hyper parameters chosen for this experiment can be found in Table 4. After running for 800 episodes, the test accuracy is shown in Table 5. Note the learning rate is reduced by half for every 100 episodes.

	ProtoNet	ProtoNet with SMLMT
learning rate	4e-6	8e-6
N way (meta training)	6	9
K shots (smlmt task)	NA	15
# of episodes per epoch	50	100
# of epoch	6	8
smlmt sample ratio	NA	0.6

Table 4: Hyperparameters used for ProtoNet

As suggested by Table 5 supervised meta training on diverse tasks from different domains (5^{th} and 8^{th} row) improves generalization to tasks in unseen domains. Figure 1 highlights the meta test accuracy and meta test standard deviation of three different approaches for 5 shots and 10 shots scenarios with BERT as the embedding function. ProtoNet with meta training consistently outperforms the baseline results from the nearest neighbor and ProtoNet (meta test only), in which no meta training involved. Even though Nearest Neighbor with the BERT encoder is a strong baseline, which achieves 80% for 5 shots and 85% for 10 shots, the ProtoNet improves the baseline by 5 points through meta training on different tasks in different domains.

The results also suggest that additional self-supervised training through SMLMT further improve few-shot generalization if we compare the ProtoNet results to the ProtoNet + SMLMT results. The blue bar in Figure 1 shows the ProtoNet results, which trained using only supervised task, and the orange bar shows the semi-supervised ProtoNet results using both labeled and unlabeled data. Semi-supervised ProtoNet improves the ProtoNet results further by an additional 5 points, which achieves 90.6% for $K = 5$ and 93.9% for $K = 10$. Note that the the semi-supervised ProtoNet with 50 labeled data outperforms the supervised ProtoNet with 150 labeled data (86.4% in Table 3). These results (details see Table 5) show that meta training on diverse tasks, especially the SMLMT tasks generated from unlabeled data yield better general-

	N=5, K=5		N=5, K=10	
Approach	Meta-Test Accuracy	Meta-Test Std	Meta-Test Accuracy	Meta-Test Std
Nearest Neighbour + BERT	0.795	0.079	0.852	0.066
ProtoNet + BERT (no meta training)	0.795	0.083	0.842	0.073
ProtoNet + BERT (supervised meta training)	0.851	0.068	0.891	0.061
ProtoNet + BERT + SMLMT (semi-supervised meta training)	0.906	0.066	0.939	0.047
ProtoNet + RoBERTa (no meta training)	0.887	0.078	0.924	0.066
ProtoNet + RoBERTa (supervised meta training)	0.975	0.037	0.981	0.033
ProtoNet + RoBERTa + SMLMT (semi-supervised meta training)	0.980	0.038	0.986	0.025

Table 5: Meta test accuracy and standard deviations of applying different meta-learning approaches with K=5 and K=10 respectively for CLINC150 dataset.

Approach	Threshold	OOS F1	OOS Recall	OOS Precision	In-domain Accuracy
ProtoNet + BERT (no meta training)	0.4	0.471	0.772	0.339	0.723
ProtoNet + BERT	0.9	0.494	0.703	0.381	0.796
ProtoNet + BERT + SMLMT	0.9	0.601	0.787	0.487	0.856
ProtoNet + RoBERTa (no meta training)	0.3	0.286	1.000	0.167	0.200
ProtoNet + RoBERTa	0.9	0.632	0.562	0.722	0.958
ProtoNet + RoBERTa + SMLMT	0.9	0.766	0.771	0.761	0.959

Table 6: Evaluation statistics on OOS examples with N=5, K=10 respectively for protoNet with BERT and RoBERTa

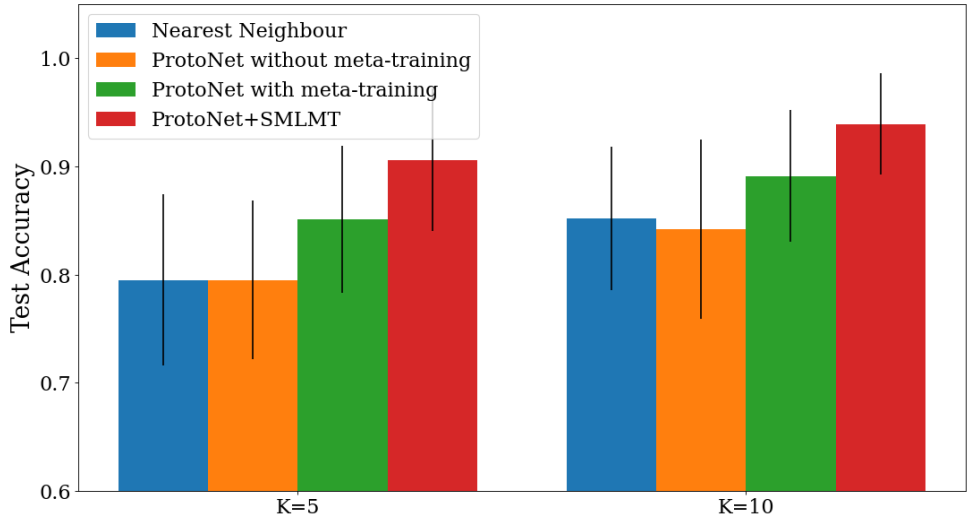

Figure 1: Meta test accuracy of applying four different meta-learning approaches with K=5 and K=10 respectively.

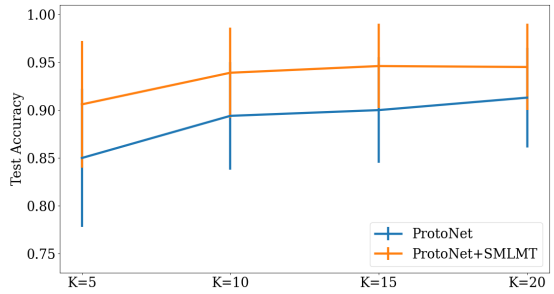

Figure 2: Meta test accuracy of ProtoNet increases with K shots, while performance plateaus around $K = 20$

ization capability.

Varying number of supporting examples K used per task during meta testing also has an effect on the meta testing accuracy. As shown in Figure 2, increase K from 5 to 15 improves test accuracy of ProtoNet with SMLMT from 85% to 94.5%, and ProtoNet from 85.0% to 91.3% while the performance plateaued for $K = 20$ (details see Table 7). Changing embedding function from BERT to RoBERTa ($7^{th}, 8^{th}, 9^{th}$ rows in Table 5) significantly improves the meta test accuracy, suggesting that RoBERTa is a better pre-trained model for intent classification.

3.4 Results on Out-of-Scope Examples

We also evaluate the performance of our meta learners on OOS examples by including an extra OOS class during meta testing. Two embedding functions, i.e., BERT and RoBERTa are evaluated in three settings: no meta training, with supervised meta training and with semi-supervised meta training (with SMLMT). The meta training procedure remains the same as previous setup. During meta testing, we first pick the threshold T (see section 2.3) and then report the F1, precision, recall for OOS and In-domain Accuracy with the selected threshold. While OOS precision and recall usually fluctuates a lot with thresholds, OOS F1 score is a better indicator of OOS accuracy.

As shown in Table 6 meta training improves OOS F1 score significantly and semi-supervised meta training gives the best OOS F1 score, 0.601 with BERT and 0.766 with RoBERTa. RoBERTa as embedding function performs better than BERT after meta training, with a nearly 10-point improvement for in-domain accuracy (0.959 vs 0.856) and OOS F1 score (0.766 vs 0.601). RoBERTa without fine tuning performs worse than BERT when OOS examples are included, probably due to the selecting criterion for the threshold. The inclusion of OOS examples clearly reduce the in-domain accuracy. For example, the in-domain accuracy for ProtoNet + BERT + SMLMT changes from 0.939 without OOS examples (Table 5) to 0.856 with OOS examples (Table 6). However, the accuracy gain compared to no meta training (0.723) and supervised-only meta training (0.796) is quite significant.

3.5 Visualization of Word Importance

To have a better understanding of why meta training on semi-supervised tasks yield better generalization capability, we analyze the token importance by plotting the gradients of the prediction with respect to the token embedding for each token as shown in Figure 3. The token with larger gradient indicates it's more important for the prediction result. Meta training changes the distribution of word importance. For example, for the sentence "I want to schedule a pto request on march 1 - 2". We see that the meta learner shifts its attention from "on march" before training, to the most important word "schedule pto request" after training, which helps it to effectively identify this sentence as a `pto_request` intent. The same observation is true for the sentence "tell me where my flight is scheduled to start boarding", where the top 3 important tokens has changed from "me, where, is" to "my, flight, is" after training, leading to the prediction of intent "flight_status". Therefore, the better generalization is powered by effective representation learning (a pre-trained BERT already yields good representation for intent classification) and also learning to attend to the right words.

4 Conclusion

We proposed a semi-supervised meta-learning approach for cross-domain few-shot intent classification by incorporating the representation power of pre-trained language model with the fast adaptation capability of ProtoNet enhanced through self-supervision. This methodology tackles the realistic few shot learning setting where not enough meta training tasks exist and meta learner trained only on supervised tasks suffers from over-fitting on a small number of labeled data. The experiments have shown that meta learner generalizes better to new domains and predicts more accurately on out-of-scope examples if trained with additional meta training tasks created through self-supervision

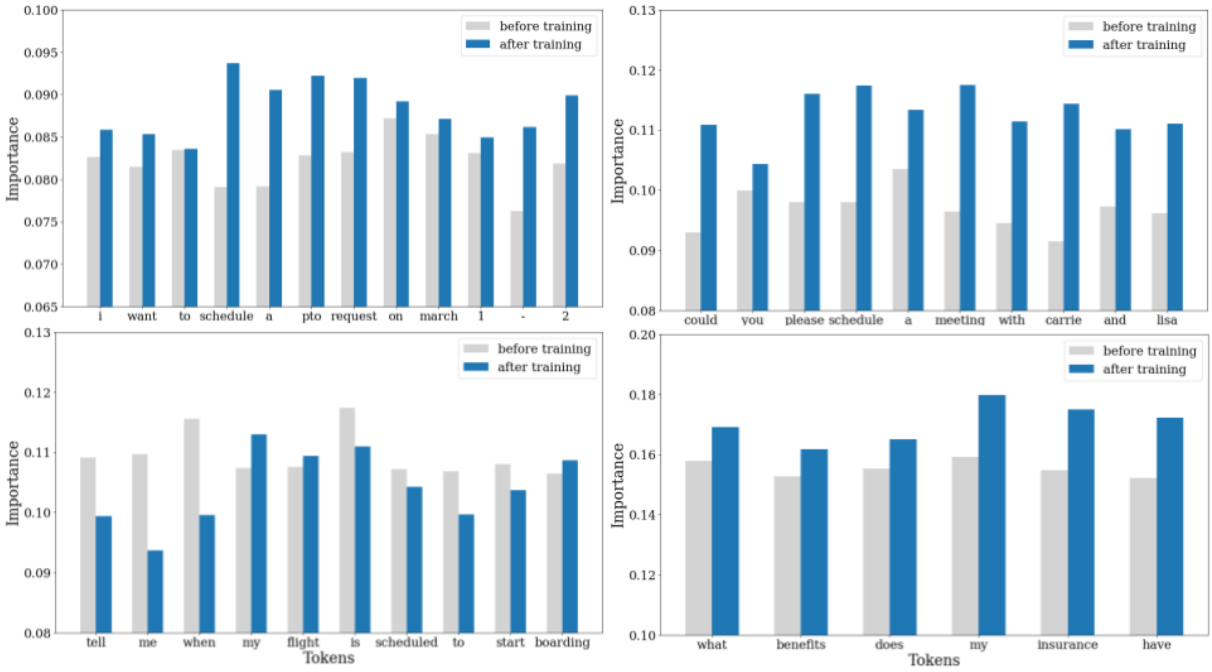

Figure 3: The importance of tokens of each sentence before and after training.

from unlabeled data. Compared to pre-training language models using self-supervision, the volume of unlabeled data required for our semi-supervised meta training is rather small and the optimization is much easier. However, it effectively improve the few-shot generalization and out-of-scope accuracy by learning a better cross-domain representation and learning to quickly attend to the right word in new domains. While ProtoNet has its limitations due to simpler inductive bias, the resulting presentation can be used to initialize more sophisticated meta learner and extend beyond the classification problems. Future directions include exploring different ways to combine various types of meta learners, different designs of self-supervised tasks as well as validating our algorithms on other datasets.

5 Acknowledgement

This work is inspired by our CS330 course project at Stanford. We would like to thank Professor Chelsea Finn for the discussion of research directions and her wonderful lectures on meta-learning.

References

Trapit Bansal, Rishikesh Jha, and Andrew McCallum. 2019. Learning to few-shot learn across diverse natural language classification tasks. *CoRR*, abs/1911.03863.

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020a. Self-Supervised Meta-Learning for Few-Shot Natural Language Classification Tasks. *arXiv*.

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020b. Self-Supervised Meta-Learning for Few-Shot Natural Language Classification Tasks. *arXiv*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. 2019. Investigating meta-learning algorithms for low-resource natural language understanding tasks. 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 1192–1197, Hong Kong, China. Association for Computational Linguistics.

			ProtoNet with meta training		ProtoNet with SMLMT	
# labeled data per class	N way	K shot	Meta Test Acc	Meta Test Std	Meta Test Acc	Meta Test Std
50	5	5	0.850	0.072	0.906	0.066
50	5	10	0.894	0.056	0.939	0.047
50	5	15	0.900	0.055	0.946	0.044
50	5	20	0.913	0.052	0.945	0.045

Table 7: Meta test accuracy of ProtoNet increases with K shots, while performance plateaus around $K = 20$

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks.

Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. *arXiv*.

Kyle Hsu, Sergey Levine, and Chelsea Finn. 2018. Unsupervised learning via meta-learning. *CoRR*, abs/1810.02334.

Siavash Khodadadeh, Sharare Zehtabian, Saeed Vahidian, Weijia Wang, Bill Lin, and Ladislau Bölöni. 2020. Unsupervised meta-learning through latent-space interpolation in generative models.

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.

X. Liu, Jianfeng Gao, X. He, L. Deng, Kevin Duh, and Ye-Yi Wang. 2015. Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In *HLT-NAACL*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Alex Nichol and John Schulman. 2018. Reptile: a scalable metalearning algorithm. *arXiv: Learning*.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. *CoRR*, abs/1703.05175.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. 2019. Meta-Dataset: A Dataset of Datasets for Learning to Learn from Few Examples. *arXiv*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2020. Xlnet: Generalized autoregressive pretraining for language understanding.

Mingzhang Yin, George Tucker, Mingyuan Zhou, Sergey Levine, and Chelsea Finn. 2019. Meta-Learning without Memorization. *arXiv*.

Wenpeng Yin. 2020. Meta-learning for few-shot natural language processing: A survey. *ArXiv*, abs/2007.09604.

Jian-Guo Zhang, Kazuma Hashimoto, Wenhao Liu, Chien-Sheng Wu, Yao Wan, Philip S Yu, Richard Socher, and Caiming Xiong. 2020. Discriminative Nearest Neighbor Few-Shot Intent Detection by Transferring Natural Language Inference. *arXiv*.