Meta-Learned Word Embeddings For Few-Shot Sentiment Classification

William Bakst
Stanford University
wbakst@stanford.edu

Nicholas Bien
Stanford University
nbien@stanford.edu

Oghenetegiri Sido
Stanford University
osido@stanford.edu

Abstract

This paper explores few-shot learning for sentiment classification. There are many current techniques for approaching this task, but recent advancements in Meta Learning have proven to work incredibly well in the few-shot setting. We propose the use of MAML, Model Agnostic Meta-Learning, for quickly learning better cross-domain embeddings and networks for few-shot learning. Our final model achieves state-of-the-art performance.

1 Introduction

Researchers and engineers who build sentiment classification systems need to collect and curate data for each new domain. The effort necessary to annotate these corpora is prohibitive, especially for domains where the features change over time. This paper explores the possibility of training classifiers on a small number of domains and then applying them to other domains. One problem is that classifiers tend to perform poorly when the training and test data come from different distributions. Another problem is that it is not clear how to determine similarity and select domains that are good to train on for testing other domains. We approach this problem by leveraging many, smaller training data sets to train classifiers that generalize and still perform well on test sets that come from different distributions. In order to do so, we propose the use of recent advancements in Meta Learning to learn better cross-domain embeddings such that they can generalize for the few-shot learning setting.

2 Related Work and Hypotheses

We have three hypotheses, which we describe in the following sections. Our first hypothesis leads us to our second, and our second to our third.

2.1 Meta-Learning As An Alternative

We can see many similarities between papers (Artetxe and Schwenk, 2018), (Liu et al., 2018), and (Blitzer et al., 2007). The primary similarity is the desire to use knowledge across many different domains to train more generalized embeddings and classifiers that perform well even on domains with different distributions from the training domains. The positive results of these papers demonstrate to us that incorporating domain knowledge and utilizing different domains for a single task can improve the performance of our overall models. However, each of these papers has a different approach. (Artetxe and Schwenk, 2018) focuses on generalized embeddings across different languages, utilizing the fact that a language x can provide useful information when training classifiers for a different language y. (Liu et al., 2018) focuses on augmenting the input to contain a vector representing a specific domain with the hope that the final model will be able to incorporate this domain vector to better classify examples. (Blitzer et al., 2007) focuses on learning a transformation function θ to extract the salient information from input vectors x. While all of these approaches are different, each one has demonstrated improvements in model performance. We hypothesize that Meta- Learning can act as a better alternative for all of these methods, potentially wrapping them all up into a single system.

2.2 Leveraging BERT to Initialize Meta-Learning

We see from (Devlin et al., 2018) and (Wolf et al., 2018) that learning embeddings that are mindful of long-term dependencies and global context can improve the overall performance of our models. (Devlin et al., 2018) uses transformers, self-attention, and pre-training for a specific task,

whereas (Wolf et al., 2018) focuses on the use of just an RNN. However, there are also other approaches to learning embeddings as seen in (Xu et al., 2018), which focuses on meta-learning for the word2vec scheme. We hypothesize that we can use the Meta Learning with BERT as the initialization to learn how to update BERT embeddings such that they are better suited for target (unseen) few-shot learning domains.

2.3 State-Of-The-Art Performance

Paper (Finn et al., 2017) is an influential work in optimization-based meta-learning and is not specific to natural language processing. Both (Yu et al., 2018) and (Geng et al., 2019) train networks for few-shot text classification. However, these papers are metric-based rather than optimization-based. The earlier paper (Yu et al., 2018) focuses on task clustering to boost classification performance, while the later paper (Geng et al., 2019) directly learns class representations via induction and achieves slightly higher performance on the Amazon Review Sentiment Classification Dataset. We hypothesize that we can bridge the gap between optimization-based and metric-based methods to achieve state-of-the-art performance in few-shot text classification on the same dataset described in section 3.

3 Data

The dataset that we use is the Multi-Domain Sentiment Dataset found at https://www.cs.jhu.edu/~mdredze/datasets/sentiment/. This dataset contains Amazon reviews from 25 different product categories, where we consider each product category a different "domain" to train or test on.

```
i bought two these bags so far . both failed
plastic fastener , one arm just snapped no time .
bag itself ok , but plastic parts belt really
cheap ... i wont buy third one .   2
```

Figure 1: An example review from the Apparel category. It has a 2 star rating.

Each review is represented as a string of text and has a rating out of 5 stars, which we use to label each review example as having either positive or negative sentiment. We use three different labeling schemes as follows:

T5: A review with a rating of 5 stars is labeled positive (+1). All other reviews are labeled negative (-1).

T4: A review with a rating or 4 or more stars is labeled positive (+1). All other reviews are labeled negative (-1).

T2: A review with a rating of 2 or more stars is labeled positive (+1). All other reviews are labeled negative (-1).

Following these schemes, the example in Figure 1 would be labeled positive for T2 and negative for T4 and T5.

The task for this dataset is to train on some number of domains (in this case product categories of reviews) such that the model learns how to learn word embeddings for target domains it has not seen before. For example, the system might train on product categories such as Apparel and Electronics and then test on Books and DVDs.

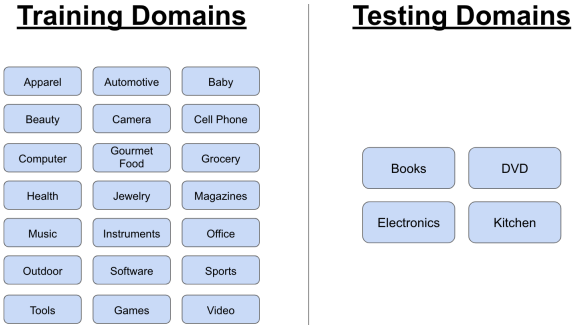

Figure 2: Dataset Split

3.1 Data Split

Accordingly, our data split is as follows in Figure 2. This follows the split used in (Yu et al., 2018). We used the same split as in the paper in order to have a more appropriate benchmark for performance.

3.2 Dataset Exploration

We've conducted an in-depth exploration of our dataset, discovering several factors that will influence the way in which we choose corpora for our meta-learning tasks. First, we note that our vocabulary is expansive at 138,397 unique words. This presents a challenging task in generating useful embeddings for words that occur infrequently. Additionally, with regards to the distribution of reviews, we find that there are 108,317 positive labels and 30,055 negative labels, which we note is a stark imbalance in the distribution of reviews. We will take this into account in choosing how to best

parameterize our meta-learning techniques, as it is easy to be biased towards word embeddings that, when processed, lead to a positive labeling outcome.

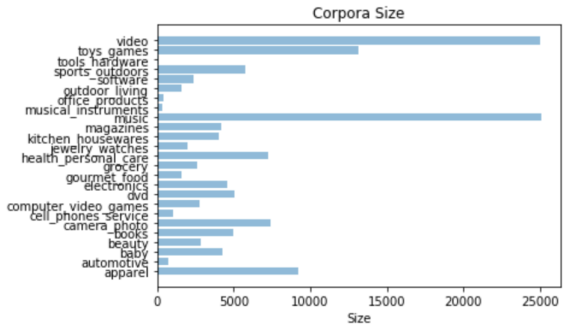

Figure 3: Size of Each Domain

When looking at the sizes of each domain, we see that the *video* and *music* have overwhelmingly more examples than the other domains. In order not to bias our representations towards a certain domain to the detriment of the performance on other domains, we may select parts of larger corpora to maintain balance across all corpora, regardless of size.

As seen in Figure 4, the dataset as a whole is biased towards positive reviews; however, we find that the *cell_phones* and *software* domains have the most balanced distribution. Thus, the *cell_phones* and *software* domains could serve as domains to use for analyzing a systems performance for few-shot learning because of its balanced nature. However, because we will compare our results to those in (Yu et al., 2018), we will follow the same train, dev, test scheme that they use, as described in section 4 as follows.

4 Metrics

In order to compare our results to (Yu et al., 2018), we choose the categories "Books", "DVDs", "Electronics", and "Kitchen/Housewares" for our test categories. Since each category has three binary classification tasks, corresponding to different star thresholds, we have 12 total tasks in the test set. We also chose a development set with the categories "Apparel", "Camera/Photo", "Magazines", and "Office Products". This allows us to monitor metrics on the held-out data without using the test set until we want final results. There are 49 tasks remaining for training across 17 categories (two of the "Tools/Hardware" tasks are omitted due to too few training examples).

The original paper doing few-shot learning with our dataset (Yu et al., 2018) reports the average accuracy across the 12 test tasks. Our model is trained and evaluated in the 5-shot learning setting. A batch of tasks consists of 5 training examples and 5 testing examples from each task. Currently, we are sampling 5 positive and 5 negative examples, then splitting these up randomly into training and test, so that our sets our balanced. The accuracy is then determined by the model's predictions on the test set after one gradient update based on the 5 training examples, for each task.

State-of-the-art performance on this dataset for 5-shot learning is 85.47%, reported in (Geng et al., 2019), using an Induction-Network-Routing model. This outperformed (Yu et al., 2018)'s RobustTTC-FSL task clustering model, which achieved 83.12% accuracy.

5 Models

Below we describe the models that we use to approach our task. We then plan to combine these models to help boost our performance.

5.1 BERT

5.1.1 Masked Language Model

BERT uses a new pre-training objective - **M**asked **L**anguage **M**odel (**MLM**) - to create deep pre-trained bidirectional representations. MLM works as follows: BERT randomly chooses a portion of the input tokens to mask: 80% of the tokens are replaced with [MASK] token, 10% are replaced with a random word, and 10% are left unchanged. The tokens left unchanged aid in guiding the representation towards the semantic meaning of the true word. Accordingly, MLM faciliates the learning of a distributed contextual representation because of the lack of encoder awareness beforehand about which words the encoder must predict.

5.1.2 Sentence Classification Task

As shown in Figure 5, BERT generates three embeddings for each token: the token embedding (a general embedding for the word), the sentence embedding (differentiate between sentence A and B for attention purposes), and the transformer positional embedding (generated via self-attention with a transformer). These are added element-wise to produce embeddings for every word in a tensor. The embeddings generated by BERT are then passed through a linear layer to generate the logits for the labels.

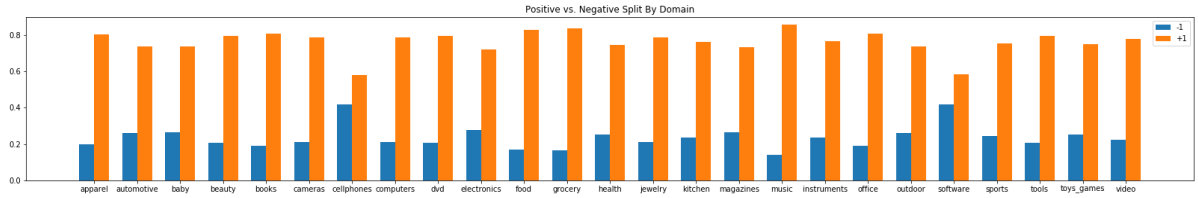

Figure 4: Distribution of Labels for Each domain

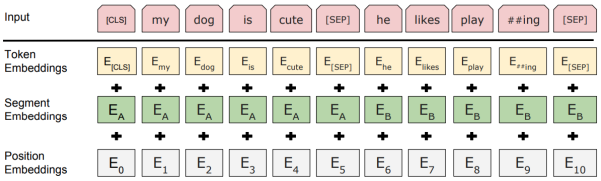

Figure 5: BERT embeddings as function of input tokens

5.1.3 BERT Algorithm

We will now cover the essential aspects of the BERT Algorithm.

Inputs & Outputs

Given a tensor of *input ids*, *token type ids*, and an *attention mask*, BERT will output *encoded layers* and *pooled output*.

Inputs:

input ids are the vocabulary ids for the particular words in our sentence.

token type ids are 0 for words in sentence A and 1 for words in sentence B. In the sentiment classification setting, however, all words are in sentence A.

attention mask is a tensor of the sequence length, and is 0 for padding and 1 for actual words in the sequence.

Outputs:

pooled outputs are our final word embeddings, where every embedding is a 768-vector that corresponds to a single word.

Algorithm Steps

1. Create 3D attention mask based on token type ids
2. Generate initial embeddings from BERT vocabulary (see Figure 5)
3. Encode intial embeddings with the attention mask via self-attention, where output are a series of encoded layers
4. Take last layer as sequence output to be pooled and returned as our final word embeddings

5.2 MAML

Meta-learning refers to training a model to be able to generalize to new tasks with very little data. (Finn et al., 2017) presents Model-Agnostic Meta-Learning (MAML), an approach for training neural networks such that they are easy to fine-tune (i.e. they can perform well on new tasks after only a small number of gradient steps). Unlike prior meta-learning methods, this approach introduces no new parameters and works with a variety of model architectures and loss functions. MAML is designed to perform well at the task of K-shot (few-shot) learning. During meta-training, the model is trained on K samples from a randomly chosen task or batch of tasks. The model is then updated via gradient descent based on the performance of this fine-tuned model on a test set for these same tasks. This meta-learning framework outperforms standard fine-tuning and previous state-of-the-art meta-learning methods in 1-shot and 5-shot learning tasks for image classification and reinforcement learning. We wish to extend MAML to meta-learn word embeddings for few-shot text classification.

5.2.1 MAML Algorithm

The MAML algorithm is parameterized by two learning rates, α and β. The larger one, α, is used to update weights on the meta-training set before evaluation on the meta-testing set. After evaluation and computing the loss on the meta-testing set, the original weights are restored for the next task in the batch. Then, after seeing all tasks, the weights of the model are updated based on the smaller learning rate β, which is multiplied by the sum of the losses across all tasks. See Figure 6 for details. Using this approach, the objective function is:

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
6: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
7: **end for**
8: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
9: **end while**

Figure 6: The MAML algorithm. Reproduced from (Finn et al., 2017).

$$\min_\theta \sum_{\text{task } i} \mathcal{L}^i_{\text{test}}(\theta - \alpha \nabla_\theta \mathcal{L}^i_{\text{train}}(\theta))$$

This loss suits the few-shot learning task because it measures how well the model performs on 5 unseen examples from a category after seeing 5 examples from that category and making one gradient update. By sampling multiple tasks per tasks, we encourage the model to learn general features that apply to many tasks and avoid overfitting to any one particular task. In other words, the algorithm attempts to find a location in the optimization landscape from which task-specific optima are easy to reach with only one gradient update.

6 General Reasoning

Our primary objective is to use meta-learning to boost the performance of one- or few-shot learning models. Our dataset lends itself to our task, as it contains reviews from 25 diverse categories. We are pooling MAML and BERT techniques to leverage a few known domains from our dataset to produce rich word embeddings to help us perform well on other unknown domains in our dataset.

More specifically, BERT's strengths lie in creating context-aware representations of words, where embeddings are not general, as we've seen with GLoVE. BERT uses transformers to perform self-attention across a text-sequence, so each word embeddings is exceedingly related to the context it lies in. MAML, on the other hand, will be leveraged to learn a transformation function θ to extract the salient information from input vectors x. Because BERT is adept at extracting and emphasizing local information within context, we believe that the simultaneous use of MAML will aid in creating more globally aware representations due to its ability to use its growing knowledge base to pick out information from embedding vectors.

7 Experiments

We use the domain split given in Figure 2 for our experiments. We additionally hold out the categories "Apparel", "Camera/Photo", "Magazines", and "Office Products" as a development set. After each iteration during training, we evaluate on the development set and save the model weights whenever the loss on the development set is the lowest we have seen thus far. This way, we can get a sense for how the model will perform on completely unseen categories without evaluating on the test set until the very end.

For training, within each task we have 5 training examples and 5 testing examples. We ensure that there are at least 2 of each class of reviews (positive or negative) in each set and randomly choose the last example. For evaluation, we similarly choose 5 examples for the gradient update, then evaluate on the rest of the examples in the task.

For the MAML model, we used batches of 8 tasks, $\alpha = 0.1$, $\beta = 0.001$ (see 5.2.1), a maximum review length of 100 words, 50-dimensional randomly initialized word embeddings, and 100-dimensional LSTM cells. For the BERT-MAML model, we used batches of 10 tasks, $\alpha = 0.1$, $\beta = 0.001$, a maximum review length of 512 words, 768-dimensional BERT embeddings, and 768-dimensional LSTM cells. Both models were trained for 10000 epochs or until the model was clearly overfitting and we were no longer updating the weights.

8 Results

Model	Test Accuracy
BERT Oracle	77.23
BERT Baseline	22.77
MAML	93.08
BERT + MAML	**99.63**

Table 1: Results for each model on the train, development, and test sets. These tests allow for the models to make a single gradient update on 5 examples from the given domain and then evaluate on all remaining examples in the domain, as described in section 7

Model	Mean Acc.
Match Network (Vinyals et al., 2016)	65.73
Prototypical Network (Snell et al., 2017)	68.17
Relation Network (Sung et al., 2017)	83.74
ROBUSTTC-FSL (Yu et al., 2018)	83.12
Induction-Network-Routing (Geng et al., 2019)	85.47
MAML	93.08
BERT-MAML	**99.63**

Table 2: Comparing the success of each model built for this dataset and task

We can see that our hypothesis was in fact correct: MAML produces state-of-the-art results, and using BERT initializations futher boosts this performance.

9 Analysis

In the following sections, we offer an analysis and discussion of our models and results. We focus our analysis primarily on the reasons why we would expect a particular architecture to have the results that we see.

9.1 BERT Oracle

The BERT oracle is a system that shows us how well BERT could perform on this task, even if it is cheating. We trained a BERT model on the data as though it were a regular supervised learning task. Rather than giving the model just five examples from each domain, we provide the model with a large training set containing examples from each domain and then test it on a held out set of examples.

However, we can see from our results that BERT does not perform particularly well even when it is the cheating oracle. We also note that, when evaluated on each scheme, the oracle does the best on the T2 scheme (88.98) and the worst on the T5 scheme (60.02). This is likely because there are many more positive reviews than negative reviews in the overall dataset, and the T2 scheme matches this distribution better than the T4 or T5 domains.

This BERT oracle shows us that even though the BERT architecture works well for many tasks, it will have a tough time succeeding in the few-shot learning task when it has even less information to train on.

9.2 BERT Baseline

We use BERT as a baseline because it is known to perform well for nearly all tasks; however, BERT is not designed for the few-shot learning setting. There are several possible reasons for this. One clear reason has to do with BERT's massive architecture and embedding size. While our MAML model utilizes a simple BiLSTM, BERT uses a complex multi-stage network with a multi-layer embedding structure and deep self-attention with a transformer. Accordingly, BERT historically performs well on tasks when trained with several training examples, which allows BERT to be fine-tuned to a particular task. In our few-shot paradigm, the lack of training examples prevents the BERT model from learning adequately to be able to generalize to future, unseen examples. Nevertheless, this experiment provides a useful lower-bound for performance.

9.3 MAML with Random Initialization

We hypothesize that MAML does well even with random word embeddings because it is designed to be able to find optima within a domain. Although random embeddings tend not to work very well, the Meta-Learning framework is good at quickly learning new embeddings for the new setting given just 5 examples. Although initially random, the embeddings and classifier weights are not random at test time because have been optimized to suit the general features of the training set categories, which are reasonably similar to the test set categories. We believe MAML's strength is that it is an optimization-based approach to meta-learning, rather than metric-based. Instead of trying to interpolate predictions for new categories based on similar tasks in the training set, as in (Yu et al., 2018), we train a model which can generalize to any (similar) task using just a gradient update. The

key insight that makes this possible is the use of training and testing mini-sets within a single iteration, which allows us to measure and optimize generalization performance.

9.4 MAML with BERT Initialization

MAML with Random Initialization already performs better than the current state-of-the art system for this dataset and task; however, more often than not a good initialization can help a model perform even better. In this case, using BERT embeddings to initialize the MAML model boosted our performance by more than 6%, which is a substantial increase in performance.

The BERT embeddings that we use are much larger than the random embeddings (768- vs. 50-dimensional). This provides the model with significantly more information for each word, which enables it to better learn how to take gradient steps for each task to update the embeddings. This is likely one of the primary reasons why using BERT initializations gave our model such a boost.

We also note that BERT initializations provide the model with general, context-aware information for each word that the random initialization model is missing. By starting off with these rich embeddings, our model can update the underlying sentiment within each embedding while still maintaining the rich information already present. This enables the model to better learn relationships between words and how to update the embeddings for the sentiment classification task. This is likely another one of the primary reasons why using BERT initializations gave our model such a boost.

10 Future Work

We would like to see if using embeddings of a similar size to that of the BERT initializations would enable the MAML with Random Initializations model to have success more similar to that of MAML with BERT. Note that MAML without BERT used 50-dimensional vectors while our MAML with BERT initilization utilized 768-dimensional vectors, as is customary for BERT tasks. We are curious to see whether the increased dimensionality of the embedding size was a major contributor to the increased performance. We also want to try using Transformer Networks instead of BiLSTMs, since they are the current state-of-the-art for language understanding.

Furthermore, given the success of this model for our dataset and task, we believe it is worth looking into how well this model can perform on other similarly difficult tasks, primarily the one- and zero-shot learning settings. It would also be interesting to attempt other natural language tasks in the few-shot setting besides sentiment classification, such as question answering or neural translation. This would provide a series of other tasks that could evaluate the richness of our embeddings.

Acknowledgements

We would like to thank Jayadev Bhaskaran for his valuable feedback over the course of our work on this project.

Authorship Statement

William Bakst and Nicholas Bien conceived of the idea of using MAML for learning embeddings for the few-shot learning setting. Oghenetegiri Sido conceived of the idea of incorporating BERT. Oghenetegiri Sido implemented the BERT baseline. Nicholas Bien and William Bakst implemented the MAML baseline framework. All three authors worked on combining and testing the MAML with BERT Initialization framework. All three authors ran experiments, analyzed the results, and helped write the final report.

References

Mikel Artetxe and Holger Schwenk. 2018. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *CoRR*, abs/1812.10464.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 440–447, Prague, Czech Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. *CoRR*, abs/1703.03400.

Ruiying Geng, Binhua Li, Yongbin Li, Yuxiao Ye, Ping Jian, and Jian Sun. 2019. Few-shot text

classification with induction network. *CoRR*, abs/1902.10482.

Qi Liu, Yue Zhang, and Jiangming Liu. 2018. Learning domain representation for multi-domain sentiment classification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 541–550, New Orleans, Louisiana. Association for Computational Linguistics.

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

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. 2017. Learning to compare: Relation network for few-shot learning. *CoRR*, abs/1711.06025.

Oriol Vinyals, Charles Blundell, Timothy P. Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. *CoRR*, abs/1606.04080.

Thomas Wolf, Julien Chaumond, and Clement Delangue. 2018. Meta-learning a dynamical language model. *CoRR*, abs/1803.10631.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Lifelong domain word embedding via meta-learning. *CoRR*, abs/1805.09991.

Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse few-shot text classification with multiple metrics. *CoRR*, abs/1805.07513.

A Supplemental Material

The code for this project can be found at:
`https://github.com/wbakst/meta-learned-embeddings`