Mitigating Label Mismatches across Datasets using Meta-Learning

Ruben Woortman
10208593
rubenwoortmann@gmail.com

Lodewijk van Keizerswaard
11054115
lodewijk.vankeizerswaard@student.uva.nl

Chris Al Gerges
11727845
chrisalgerges@hotmail.com

Auke Elfrink
13404555
auke.elfrink@student.uva.nl

Abstract

Label mismatches between datasets reduce NLP models' ability to generalize. Finetuning the large-scale transformer models that are commonly used in NLP on new datasets thus requires large amounts of time and computational power. Meta-learning models overcome this disadvantage by training with few-shot learning on several datasets, in order to learn parameters that can quickly adapt to new data. We evaluate the generalization ability of a meta-learning model across datasets in similar domains with different labels. We finetune a pre-trained BERT model using the ProtoMAML framework for meta-learning. We compare to a standard multitask model, also based on pre-trained BERT, by evaluating both models on few-shot classification of unseen datasets. Results show that the meta-learning model significantly outperforms the multitask model using far less data, time and computational power. Moreover, we find that the inner loop optimization of ProtoMAML can be restricted to the task-specific layers, significantly reducing training time. The code can be found at: https://github.com/elfrink1/ATCS.git

1 Introduction

The ability to generalize is the ultimate goal of any NLP model. However, this ability is difficult to acquire in practice, as datasets can differ significantly in content. Even when content is similar, a mismatch in labels can negatively impact performance. Research over the last few years has led to the development of massive transformers models that generalise and perform remarkably well (Tetko et al., 2020; Devlin et al., 2018; Tenney et al., 2019). However, these models are computationally extremely expensive: they have to be pretrained extensively, and to finetune them for a specific task with new labels requires many hours of dedicated gpu-time.

To mitigate this issue, we take a more sophisticated approach to learning: 'meta-learning'. In this learning framework, a model is trained with few-shot learning across several datasets, with the aim of becoming quickly adaptable to new data. In essence, the model learns to learn. Recent studies using this approach have achieved excellent results across a variety of NLP tasks (van der Heijden et al., 2021; Holla et al., 2020a,b; Bansal et al., 2020a,b; Yu et al., 2018; Wu et al., 2019; Dou et al., 2019).

While most of the work on meta-learning investigates cross-domain adaptation, e.g. across languages, this work focuses on the use of meta-learning to mitigate label mismatches between datasets in the same domain.

We use ProtoMAML, a recently developed meta-learning framework which has shown promising results in (multi-lingual) document classification (Triantafillou et al., 2019; van der Heijden et al., 2021). We train on five datasets: HuffPost News Category (HP) (Misra, 2018), AG News (Gulli, 2004), 20 Newsgroups (Lang, 1995), BBC News Classification (BBC) (Competition, 2019), and DBpedia14 (DB) (Zhang et al., 2015).

To show the added value of the few-shot learning approach in mitigating label mismatches, we compare the meta-learning model to a multitask architecture. In this setup, the model learns the different tasks simultaneously, but without the few-shot learning approach of meta-learning. Furthermore, we will evaluate ProtoMAML with multiple setups to investigate the effectiveness of ProtoMAML in learning general features.

Both models are adapted from pretrained BERT, and trained with their respective approach on the HuffPost, DBpedia14 and 20 Newsgroups datasets (Devlin et al., 2018). The models are evaluated on few-shot classification of the two unseen datasets, namely the AG News and the BBC News Classi-

fication datasets. We anticipate the meta-learning model will generalise well, and outperform the multitask model. This hypothesis is supported by the works of (Bansal et al., 2020a) and (Bansal et al., 2020b), who show that meta-learning models can outperform multitask models on the same unseen tasks, even though they are trained on far less data.

Results indicate that our expectations were indeed correct. The ProtoMAML model has significantly better performance in testing than the multitask model, while having seen far less data in training. This clearly shows the value meta-learning has for generalizing across label mismatches.

We contribute the following insights: Firstly, we demonstrate the value of meta-learning to mitigate label mismatches across several datasets in the same domain. This specific issue is (to our knowledge) not addressed in the current literature on meta-learning in NLP. Moreover, while both document classification with meta-learning and ProtoMAML for NLP tasks are studied extensively, the only recorded use of ProtoMAML for document classification we have encountered is the multilingual setup of (van der Heijden et al., 2021). Our evaluation of ProtoMAML's performance on document classification is thus a valuable contribution to the understanding of this method's ability. Also, by our comparison to multitask learning we show that the added value of meta-learning in this case lies not just in learning several tasks simultaneously, but also in the few-shot learning approach. Finally we discover that the inner loop adaptation in ProtoMAML can be eliminated altogether, resulting in a great reduction of training time.

2 Related Work

2.1 Meta-Learning

Meta-learning has seen many successes in computer vision, such as in rare object detection (Wang et al., 2019) and in human motion prediction (Gui et al., 2018) and has recently gained attention in NLP (Yin, 2020). Unlike most Deep Learning techniques, meta-learning makes use of a unique learning paradigm called episodic learning, where the model is trained in collections of episodes, which are essentially tasks with only a few examples. Each episode/task has its own training set and test set, called the support set and query set respectively. At training time, the model performs few-shot learning on episodes that are sampled from a distribution of tasks. At testing time, the model is evaluated on new unseen episodes in a few-shot fashion.

One well-known meta-learning approach is prototypical networks. This approach first computes a mean vector of the embeddings of examples from the support set for each class k , which is *the prototype vector* of that class. Then the distance between each prototype vector and each of the embeddings from the query set are calculated with a distance metric d, such as Euclidean distance. After taking the softmax of the distances, this distance is then minimized with negative log-likelihood loss (Snell et al., 2017).

Another popular meta-learning approach is the Model-Agnostic Meta-Learning (MAML) algorithm (Finn et al., 2017). This is an optimization-based approach that, as the name suggests, can be applied to any architecture and any optimization problem, including classification, regression and reinforcement learning.

Due to the model-agnostic and task-agnostic properties, numerous extensions of MAML have been developed. For example, (Gupta et al., 2018) build upon MAML to develop structured exploration strategies in Deep Reinforcement Learning (Deep-RL). Furthermore, (Garcia et al., 2021) extended the MAML algorithm to take hierarchical task relationships into account for cross-lingual natural language inference.

The ProtoMAML framework used in this study combines the flexibility of MAML with the simple inductive bias of prototypical networks (Triantafillou et al., 2019). The key difference with MAML is that ProtoMAML initializes meta-initialization with prototype vectors, providing a useful starting point for the outer loop optimization.

2.2 Multitask Learning

Multitask Learning (MTL) (Caruana, 1993) is another approach that, like meta-learning, aims to train a model that generalizes through tasks. The main goal of MTL is to improve the performance on the main task by jointly training it with secondary (auxiliary) tasks, where features learned from auxiliary tasks are communicated with the main task. There exist multiple MTL architectures that implement this communication between auxiliary and main task. The simplest architecture is known as hard-parameter sharing, where part of the model (usually the bottom half) is shared among the main and auxiliary tasks, while other parts of

the model (usually the top layer) are task-specific. This is a direct form of communication, as the data of the main and auxiliary task pass through the same layers.

The main intuition behind MTL is that the auxiliary tasks learn features that could be useful for the main task. For example, (Barrett et al., 2018) improved the performance of several NLP tasks, such as sentiment analysis and grammar error detection, by training gaze prediction as an auxiliary task, allowing the main task to focus more on relevant tokens. In another study, (Dankers et al., 2019) trains emotion prediction as an auxiliary task to improve the performance of metaphor detection and vice versa, showing that emotion detection can learn features useful for metaphor detection more easily and vice versa. With the main intuition of MTL in mind, we trained an MTL model where the main task is considered to be document classification, and the 'auxiliary tasks' the different datasets.

3 Methods

Since transformer based models have been very successful in many NLP tasks, the MTL and meta-learning models are based on BERT (Tenney et al., 2019). We use the pre-trained BERT-base model and the associated BERT-tokenizer.[1] A larger version of BERT is available, but it is far more computationally expensive and has no added value for our experiments.

3.1 Meta Learning

The meta-learning approach used is ProtoMAML (Triantafillou et al., 2019), a variant of Model Agnostic Meta-Learning (MAML) (Finn et al., 2017). MAML aims to learn high-quality features that are general across tasks and can also rapidly be reused to fit to an unseen task or dataset using only a few labeled examples. The intuition behind the method is that a model's training should resemble its evaluation, therefore for few-shot classification we want to train on a large number of different low-example tasks despite our limited dataset collection. To achieve this each task is generated by sampling *n-way* classes from a random dataset. Each task T then contains a support set S with *k-shot* examples for each class, used in training the task-specific model; and a query set Q of < 10 class examples with which to evaluate the task model.

In an inner loop the support set S_i of task T_i updates the initial model parameters θ to task-specific parameters θ_i via stochastic gradient descent:

$$\theta_i = \theta - \alpha \nabla_\theta \mathcal{L}_{T_i}(f_\theta)$$

An outer loop then does the meta-learning of initial parameters θ by accumulating the losses for the task-specific models on their query sets for an entire batch of tasks:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{T_i \sim p(T)} \mathcal{L}_{T_i}(f_{\theta_i})$$

Our *meta*-model will be the same BERT-base-uncased model along with a randomly initialized 192-neuron hidden layer.

The modification of ProtoMAML is that it initializes the final task-specific layer weights $\mathbf{W}_k = 2\mathbf{c}_k$ and biases $b_k = ||\mathbf{c}_k||^2$ from class prototypes: average embeddings $\mathbf{c}_k$ as provided by BERT + hidden for each class k in T_i (Snell et al., 2017). This initialization provides the model with a simple inductive bias and reduces the need for the model to learn class features from scratch.

The loss was calculated as Cross-Entropy over the Log Softmax of the task-specific model. To ensure generalizability, the model needs to be able to handle tasks with different number of classes. To achieve this, the amount of sampled classes per episode is chosen randomly from a minimum up to the amount of classes in the dataset. As class labels will be different for each episode the model should not fit to particular classes, instead learning features that enable it to discriminate between classes in general.

3.2 Multitask Learning

The implementation of the multitask model replaces the training step of ProtoMAML with a hard parameter sharing multitask framework (Caruana, 1993). This means that here the lower layers of the BERT-base model are shared between the different datasets (i.e. the different tasks): the embedding layer, and $1 \leq \alpha \leq 10$ transformer head layers are shared. The remaining $\beta = 11 - \alpha$ transformer heads and the pooling layer are task-specific. Similar to the meta learning model, a single fully connected layer with a $softmax$ is added to perform

[1] available from https://huggingface.co/bert-base-uncased

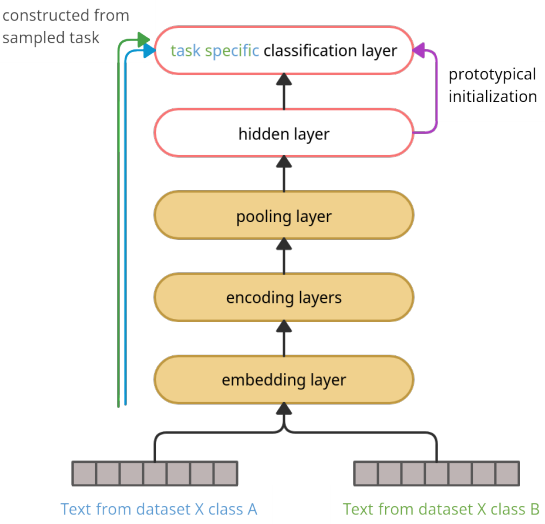

Figure 1: An illustration of the meta-learning model. The (yellow) embedding, encoding and pooling layers are initialized from the pretrained BERT-base model. The task-specific classification layer is recreated for every sampled task (in this case classes A and B from dataset X), and prototypically initialized using the (red) hidden layer.

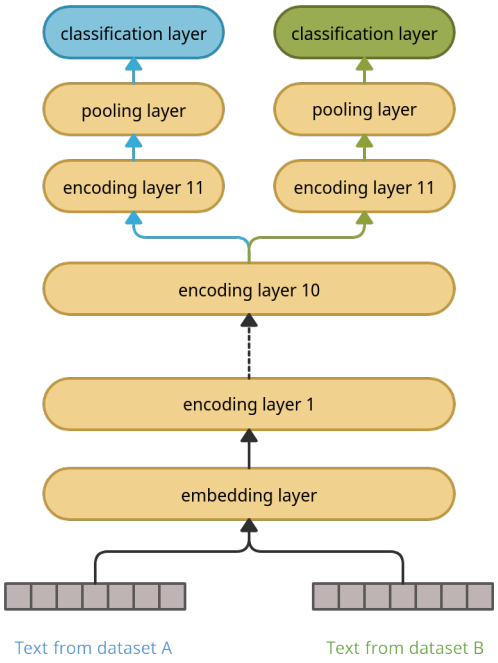

Figure 2: An illustration of the multitask model for $\alpha = 1$ with two datasets. The (yellow) embedding, encoding and pooling layers are initialized from the pretrained BERT-base model. The (blue and green) classification layers are task-specific. The loss is in this case weighted at layer 10.

classification, but in this model the layer is task-specific. Since a loss will be calculated for every task, the loss function for the shared layers has to be weighted. These weights are initialized to be equal between tasks, and then dynamically learned during training. An overview of this model for two datasets can be found in Figure 2.

The evaluation is very similar to the evaluation for the meta learning framework. At evaluation time few-shot evaluation is performed as follows; we append a few-shot-head onto the shared transformer heads. Like the task-specific head, the few-shot-head consists of the β transformer heads and the pooling layer from the pretrained BERT model. This is then followed by a linear layer with hidden size of 192 and a ReLU activation function. This is essentially the original pretrained BERT model with a simple MLP classifier on top, where the α shared transformer heads have been trained by the MTL framework. The transformer heads and the pooling layer in the few-shot-head are loaded with their pre-trained weights from Hugging Face, since we do not want to bias the few-shot head towards a specific dataset. On top of the few-shot head, the final task-specific classification layer from the ProtoMAML algorithm is added. The initialization of this task-specific layer and the few-shot evaluation of the model are exactly the same as for the meta-learning model described in the previous section.

3.3 Dataset Difficulty

Since there is no baseline available for the described setup, some kind of dataset difficulty is required to place the comparison of MTL and meta-learning in context. For this we chose to use the MTL model finetuned separately for each dataset. This means that the entire BERT-base model with a single fully connected layer on top is finetuned for each task. The accuracy of this dataset difficulty model thus represents an upper bound for the performance.

4 Experiments

4.1 Datasets

The datasets used are the HuffPost News Category (HP) (Misra, 2018), AG News (AG) dataset (Gulli, 2004), 20 Newsgroups (NG) (Lang, 1995), the BBC News Classification (BBC) (Competition, 2019), and DBpedia14 (DB) (Zhang et al., 2015)

datasets.[2] They provide a range of different label counts and are very different in size (see Table 1).

For the BBC, AG, DB no validation split was available. These splits were sampled from the training split with ratio's of 0.2, 0.2, and 0.1 respectively. For the NG dataset, no validation or test split was available. The test split was sampled from the whole dataset with a ratio of 0.2. The validation set was then sampled from the remainder with a ratio of 0.1. [3] Since performing cross-validation on each dataset would be too time consuming, we decided to limit our experiments to a single train-val-test split for each dataset.

For training both models, we chose the HP, NG and DB datasets. The HP and DB datasets, news and Wikipedia pages respectively, have the most number of classes and are the largest of the selected datasets. The models are also trained on the NG dataset. This dataset consists of forum posts and is generally less clean than the other two training datasets. The combination of these dataset characteristics intuitively will enable us to train generalizable models.

The AG and BBC datasets were used for testing, since their lower number of classes and high quality will be beneficial to a clear error analysis.

	HP	**DB**	**NG**	**AG**	**BBC**
documents	$200k$	$630k$	$20k$	$130k$	$2k$
classes	41	14	6	4	5

Table 1: Size and number of classes of the used datasets.

4.2 Meta Learning

A hyperparameter grid search for ProtoMAML was performed by using HP, NG, and BBC for training with the results on DB used to evaluate. Considered hyperparameters were: inner learning rates $\alpha = [10^{-02}, 10^{-03}, 10^{-04}]$, outer learning rates $\beta = [10^{-01}, 10^{-02}, 10^{-03}]$, batch sizes = [4, 8, 16], query sizes = [5, 10], and amount of inner loop steps = [5, 10]. Both Adam and Stochastic Gradient Descent (SGD) were considered as optimizers. The random seed was set to 42.

As the BERT gradients can take up a lot of memory, the range of potential classes per episode had to be limited to [3, ..., 12]. During grid search the accuracy was calculated every 5 batches, and because BERT inference is slow this evaluation was also limited to 32 examples per class. To reduce noise caused by potential unrepresentative batches, the task-model accuracy for 4 different DB episodes was averaged, meaning that each evaluation was done on 4 * 32 * way_μ examples.

Due to the variety of tasks in episodic training, the model showed no signs of over-fitting once optimal hyperparameters were found. Therefore we did not implement early stopping, but instead trained the final comparison model on 250 batches, well after convergence at 45 batches. Then final evaluation was done with the average accuracy over task models generated by 10 different support sets. For BBC each task accuracy was calculated on the entire dataset, while for AG this was reduced to 5k random examples per class for each task model.

After testing AG and BBC using the optimal hyperparameters found by our grid search, we used observed trends in the responses of accuracy to various hyperparameter settings to conduct further experiments. Here we aimed to analyse which aspects of the ProtoMAML algorithm affected its performance in document classification. We varied batch sizes and learning rates for 1- and 5-shot learning. We also emulated the work of (Raghu et al., 2019), who found that restricting the inner loop optimization of MAML to the task-specific layer did not significantly change the representations of the lower embedding layers.

4.3 Multitask Learning

The multitask model was trained on 45000 total samples from the training sets, because during preliminary testing, it became clear that training on the entire datasets would not be feasible. The Adam optimizer with a constant learning rate of 0.0001 was used to train the model for a maximum of 40 epochs. The model with the highest performance on the validation set, was used for evaluation.[4] The reported accuracy is the average over 20 runs with randomly sampled support sets, with the same hyperparameters as for the final evaluation of the meta learning model.

[2] Several versions of the NG dataset are available. Here the 18828 version is used.

[3] All sampling was performed with the scikit-learn function *train_test_split*, with a random seed of 42.

[4] Download link for checkpoint of MTL model: https://drive.google.com/file/d/1pjCplwjC4f4yMZMEXGsaoyTcxCkxzPt0/view?usp=sharing

4.4 Dataset Difficulty

The dataset difficulty model was trained on each individual dataset using the Adam optimizer with a constant learning rate of $1.0 * e^{-4}$ and a maximum of 15 epochs. The random seed was set to 20. The model only trained for two epochs on DB, due to its large size. The model with the best performance on the validation split was evaluated on the test split.

5 Results and Analysis

The results of the MTL model and the meta-learning model are shown in table 2.

Model	**HP**	**DB**	**NG**	**AG**	**BBC**
Difficulty	71.3	99.3	94.2	94.2	97.3
MTL-model	-	-	-	68.2	78.4
PM	-	-	-	79.5	87,8
(N) PM	-	-	-	77.7	88.9
(NA)PM	54.3	79.9	72.1	84.8	92.1
(1-shot) PM	41.6	69.8	68.8	78.3	82.1
(10-shot) PM	65.6	85.2	78.3	84.7	93.6

Table 2: The results per model, where for ProtoMAML (PM), N indicates the results were achieved with 1 inner update on two episodes per dataset per batch, while NA shows results for models trained on all but the test set. 1- and 10-shot were both done with NA.

5.1 General Comparison

First of all, we observe that the data difficulty of HP is significantly lower, compared to all other datasets. This is most likely due to the large number of labels in HP. Surprisingly, DB has the highest data difficulty score, even though DB is the largest dataset and has significantly more labels than NG, AG and BBC. However, the high score of DB may be explained by the fact that DB uses the title and the whole content of the document instead of just the title, meaning that more useful features could be extracted by the model.

Next, we analyse the results of the MTL model and protoMAML. The first thing we observe is that, in line with the data difficulty score, for both models the accuracy on BBC is much higher than on AG. This is likely due to the fact that the BBC dataset is significantly smaller than the AG dataset. More importantly, we see that protoMAML outperforms the MTL-model on both datasets, which is inline with our hypothesis. However, the MTL-model still performs moderately well given that it is evaluated on unseen data.

A possible explanation for the lower performance of the MTL-model is that multitask learning only mitigates label mismatches between datasets used for training. If the MTL-model is trained on for example HP, DB and NG datasets, then the weights in the shared transform heads contain general information about HP, DB and NG datasets, including information about the labels. This suggests that the MTL-model still may not generalize well to unseen data, since their label sets could be beyond the scope of the training sets. This in turn also explains the moderately good results from the MTL-model, as some labels from AG and BBC may overlap with labels from the training sets.

Conversely, meta-learning seems to generalize much better than multitask learning. Specifically, protoMAML seems to extract more general features than the MTL-model from each datapoint, since protoMAML learns only from a few samples while the MTL-model learned from 45000 samples. This perfectly illustrates how much more data-hungry multitask learning is compared to meta-learning.

5.2 ProtoMAML Analysis

ProtoMAML was very sensitive to the outer learning rate as 0.01 was the only value where the loss function was monotonically decreasing while still allowing for rapid learning. As optimizing with Adam 5-shot gave inconsistent results, SGD was used. 5-shot classification further showed to be relatively stable across other hyper-parameters, therefore a query set size of 5 was chosen to reduce costs by BERT inference.

The experiments with batch size indicated that larger batch sizes seem to normalize jumps that could occur after each meta-update, possibly due to a reduced chance of a single dataset dominating the entire task batch. Since we only had 3 datasets for training, we found that instead of randomly sampling datasets to fill one batch, batches with one task per dataset resulted in the same regularising effect of a larger batch size, while more than halving training time. The smaller batch size benefitted from a less aggressive inner learning rate of 10^{-4} as this would prevent over-fitting to the now smaller amount of samples in the meta-batch.

Based on these results, we hypothesised that increasing the batch size to contain two samples of each dataset might further improve regularisation for the outer loop meta-learning step. This had

no effect on 5-shot evaluation, except for doubling the convergence rate. It did have a large effect on accuracy and time to convergence for 1-shot evaluation, where the effects of regularization would be more pronounced due to fewer total examples in the meta-update.

Like the work of (Raghu et al., 2019) we emulated, we saw little to no detriment to performance when the inner loop optimization was limited to the task specific layer. However learning speed improved by at least 10%, increasing further with more inner loop updates, while also reducing memory footprint. The method used by the authors was vanilla-MAML, which does not have the benefit of the inductive bias provided by the prototype initialization. Therefore inner loop updates were necessary for the task-specific head to learn class features. Because the prototype initialization of our task-specific head already encodes the Euclidean distances between classes, ProtoMAML might not actually need further task-adaptation. This can be observed by reducing the inner update to 1 iteration, which did not affect 5-shot performance but was in fact necessary for higher accuracy in 1-shot classification. For 1-shot, any further fitting of the model to single class examples simply resulted in over-fitting.

A reduction to 1 iteration also meant our implementation became increasingly similar to Prototypical Networks (Snell et al., 2017) with a difference in when the model is updated; A Prototypical Network updates the model after each task, while ProtoMAML does a meta-optimization through the accumulation of losses for a batch of tasks. To examine the effects of this meta-optimization we allowed the model to optimize after each task. This resulted in a significant drop in accuracy.

The differences between 1-, 5-, and 10-shot classification on different datasets show that tasks which can contain a greater number of classes require more examples to construct representative class prototypes. The benefits of adding additional datasets to the task collection is also clearly visible in the results for AG and BBC.

6 Conclusion

This study has demonstrated the value of meta-learning in mitigating label mismatches between datasets. Our ProtoMAML model outperforms a similar multitask model on few-shot evaluation of unseen datasets with new labels, while using significantly less data, time, and computational power. This confirms our expectations, and contributes valuable knowledge regarding the value of meta-learning for generalization in NLP.

Based on our experiments, we observe that the effectiveness of the ProtoMAML framework specifically stems from 1) its episodic learning, creating many different task episodes from a limited number of classes/datasets; 2) the prototype initialization providing immediate class-discriminating features that do not need further optimization; 3) the meta-optimization regularizing learning over tasks. How well ProtoMAML encourages the learning of general features becomes especially clear when looking at the small amount of examples of an unseen dataset it requires. For 5-shot classification, convergence occurred after having only observed $\sim 8k$ examples, or 15 minutes on a Nvidia Titan RTX.

For future experiments with ProtoMaml, we recommend investigating the effect of query-set size. Increasing query set size did not affect 5-shot classification, but given that 1-shot classification was highly sensitive to changes in other parameters, exploring this setting further seems advisable. Furthermore, we recommend testing other task-specific head initialization methods in combination with MAML's meta-optimization, alongside a comparison with other meta-learning frameworks on the same data, such as LEOPARD (Bansal et al., 2020a), or SMLMT (Bansal et al., 2020b). Finally, we expect that in our text classification setting, further improvement could be had by exposing the model to more datasets.

References

Trapit Bansal, Rishikesh Jha, and Andrew McCallum. 2020a. Learning to few-shot learn across diverse natural language classification tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5108–5123, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Trapit Bansal, Rishikesh Jha, Tsendsuren Munkhdalai, and Andrew McCallum. 2020b. Self-supervised meta-learning for few-shot natural language classification tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 522–534, Online. Association for Computational Linguistics.

Maria Barrett, Joachim Bingel, Nora Hollenstein, Marek Rei, and Anders Søgaard. 2018. Sequence classification with human attention. In *Proceedings*

of the 22nd Conference on Computational Natural Language Learning, pages 302–312.

Richard A. Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *Machine Learning Proceedings 1993*, pages 41–48. Morgan Kaufmann, San Francisco (CA).

Kaggle InClass Competition. 2019. Bbc news classification.

Verna Dankers, Marek Rei, Martha Lewis, and Ekaterina Shutova. 2019. Modelling the interplay of metaphor and emotion through multitask learning. 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 2218–2229.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR.

Jezabel Garcia, Federica Freddi, Jamie McGowan, Tim Nieradzik, Feng-Ting Liao, Ye Tian, Da-shan Shiu, and Alberto Bernacchia. 2021. Cross-lingual transfer with maml on trees. In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 72–79.

Liang-Yan Gui, Yu-Xiong Wang, Deva Ramanan, and José MF Moura. 2018. Few-shot human motion prediction via meta-learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 432–450.

Antonio Gulli. 2004. Ag corpus of news articles.

Abhishek Gupta, Russell Mendonca, YuXuan Liu, Pieter Abbeel, and Sergey Levine. 2018. Meta-reinforcement learning of structured exploration strategies. *arXiv preprint arXiv:1802.07245*.

Niels van der Heijden, Helen Yannakoudakis, Pushkar Mishra, and Ekaterina Shutova. 2021. Multilingual and cross-lingual document classification: A meta-learning approach. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1966–1976, Online. Association for Computational Linguistics.

Nithin Holla, Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. 2020a. Learning to learn to disambiguate: Meta-learning for few-shot word sense disambiguation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4517–4533, Online. Association for Computational Linguistics.

Nithin Holla, Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. 2020b. Meta-learning with sparse experience replay for lifelong language learning. *CoRR*, abs/2009.04891.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339.

Rishabh Misra. 2018. News category dataset.

Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. 2019. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. *CoRR*, abs/1909.09157.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 4080–4090, Red Hook, NY, USA. Curran Associates Inc.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

Igor V Tetko, Pavel Karpov, Ruud Van Deursen, and Guillaume Godin. 2020. State-of-the-art augmented nlp transformer models for direct and single-step retrosynthesis. *Nature communications*, 11(1):1–11.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, 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. *CoRR*, abs/1903.03096.

Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. 2019. Meta-learning to detect rare objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9925–9934.

Jiawei Wu, Wenhan Xiong, and William Yang Wang. 2019. Learning to learn and predict: A meta-learning approach for multi-label classification. 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 4354–4364, Hong Kong, China. Association for Computational Linguistics.

Wenpeng Yin. 2020. Meta-learning for few-shot natural language processing: A survey. *arXiv preprint arXiv:2007.09604*.

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. 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 1206–1215, New Orleans, Louisiana. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *arXiv preprint arXiv:1509.01626*.