Meta-Learning for Few-shot Domain Adaptation

Aman Hussain 12667447
Ivan Bardarov 12579572
Omar Elbaghdadi 12660256
Bálint Hompot 12746452

Abstract

Domain shift occurs when the test distribution is different from the training distribution on the same task, usually degrading predictive performance. We study meta-learning approaches to few-shot domain adaptation for the sentiment classification task. We use two representative meta-learning methods: Prototypical networks and MAML, and a multitask learning baseline. We find that the multitask baseline proves to be quite strong, outperforming both meta-learning methods. However, MAML achieves performance close to multitask learning when domain shift is high. We also find that smart support set selection increases model performance slightly for all models studied.

1 Introduction

Machine learning models have shown great capabilities of being able to learn from labeled data (He et al., 2015; Dozat and Manning, 2017). However, most of these models rely on the assumption that train and test data are drawn from the same, or very similar, distributions (domains). When the test data distribution is significantly different from the training data distribution, model performance usually drops drastically (Blitzer et al., 2007). This is very common in practice, and is also called domain shift. The larger the domain shift, the larger its effect on performance.

Gathering labeled data for every possible domain is too expensive, and for many domains infeasible. Instead, we want the model to be able to *adapt* to a new domain using no or as few additional labels as possible. The goal of *domain adaptation* is to construct a learning method that facilitates this. This has been heavily studied, and proposed solutions use unsupervised (Ganin et al., 2017; Blitzer et al., 2006) and semi-supervised methods (Steedman et al., 2003; McClosky et al., 2006; Jiang and Zhai, 2007), mainly by aligning feature representations between source and target domains.

We adopt a novel approach to domain adaptation for language data, using meta-learning (Schmidhuber, Jürgen, 1987; Thrun and Pratt, 1998). The meta-learning paradigm has been shown to work well on few-shot classification tasks (Snell et al., 2017; Santoro et al., 2016; Finn et al., 2017). With its ability to adapt to new tasks quickly, we explore whether it proves beneficial for the domain adaptation problem.

While Li and Hospedales (2020) also use meta-learning for domain adaptation, they use it as a starting point for other domain adaptation tasks and for image data. We only consider meta-learning as-is, and consider a language task instead: sentiment classification (Pang et al., 2002). Sentiment classification has been used for domain adaptation before (Blitzer et al., 2007). In the context of sentiment classification, for example, the word "fast" would be associated with a positive sentiment for a review of a battery charger, while it would be negative for a battery review.

We experiment with a dataset of 14 Amazon product review domains, and 2 movie review domains. We compare the performance of meta-learning domain adaptation approaches with a multitask (Caruana, 1997) baseline. We find that:

- The meta-learning methods are outperformed by the multitask baseline for the task of sentiment classification. This is surprising, as meta-learnig methods are explicitly optimized for fast adaptation.
- MAML achieves performance close to multitask learning when the domain shift is high, but performs a lot worse when the domain shift is smaller.
- Selection of high quality support sets, us-

ing pointwise mutual information, slightly improves performance on all testing setups, opening up further lines of investigation.

2 Related Work

Domain Adaptation A large amount of research has been done on learning under domain shift. Early work on unsupervised and semi-supervised domain adaptation include *bootstrapping* (Steedman et al., 2003; McClosky et al., 2006), learning a *shared feature space over domains* (Blitzer et al., 2006), and *instance weighting* (Jiang and Zhai, 2007). More recent approaches make use of adversarial learning (Ganin et al., 2017; Goodfellow et al., 2014) and fine-tuning (Sennrich et al., 2016). Ruder and Plank (2018) find that the earlier approaches are strong baselines in the context of neural networks. In conjunction with our research, meta-learning for domain adaptation is starting to be explored as well. Li and Hospedales (2020) use meta-learning to find a good parameter initialization for other domain adaptation methods, and obtain improvements on several image domain adaptation benchmarks. This is different from our work in that we use the meta-learned model as-is, and not as a base for another domain adaptation method, and that our experiments concern language data.

Meta-Learning *Meta-learning*, or learning to learn (Schmidhuber, Jürgen, 1987; Thrun and Pratt, 1998), aims to improve its future learning efficiency during training, allowing it to quickly adapt to a new task. The paradigm has been proposed as a way to overcome fundamental challenges of traditional supervised learning methods: their disability to generalize well to unseen data, and the large amount of data needed to train such a model. In contrast, humans are usually able to generalize after just one or a few examples of a given object. Meta-learning approaches have been shown to work well for few-shot classification tasks (Ravi and Larochelle, 2017; Vinyals et al., 2016; Snell et al., 2017), in which a model is asked to make predictions on a new task after seeing only a few examples.

We explore two meta-learning approaches for domain adaptation. One is the metric-based *prototypical network (Protonet)* (Snell et al., 2017), which can be seen as an end-to-end trainable nearest neighbor approach. Another is the optimization-based *MAML* (Finn et al., 2017), which attempts to find a parameter initialization from which a network can adapt quickly to a new task with a limited amount of new data. Like Blitzer et al. (2007), we investigate domain adaptation for sentiment analysis. They similarly use Amazon product reviews, but we investigate more product categories, as well as movie reviews.

Multitask Learning We use multitask learning as a baseline. Multitask learning (Caruana, 1997) trains a single model using multiple tasks at the same time. This allows the method to share information it learns from multiple tasks, and even learn domain-agnostic features, which might help it transfer knowledge to different domains as well. Liu et al. (2017) use a private-shared multitask architecture to explicitly learn domain-agnostic features, and performs experiments on the same datasets that we do. Their results are, however, not comparable to ours. We use a hard-sharing architecture, in which parameters are shared across all tasks, and we use BERT (Devlin et al., 2018) as model encoder, while they use an LSTM (Hochreiter and Schmidhuber, 1997).

3 Methods & Task Formulation

In this section, we first describe the specific models we use for the domain adaptation task, drawing from multitask and meta-learning. We also describe a method quantifying *similarity* between domains, which we use as a guide for designing our experiments.

3.1 Meta-Learning for Domain Adaptation

Meta-learning aims to improve its learning algorithm during training. It optimizes for this explicitly by using an inner learning algorithm that learns a task, and an outer (meta) learning algorithm that optimizes an outer objective, such as generalization performance or learning speed of the inner algorithm (Ravi and Larochelle, 2017). In the context of domain adaptation, the meta objective is to quickly perform well on a task in a new domain.

Meta-learning usually employs an *episodic learning* strategy, in which learning happens over *episodes*. An episode consists of a *support* set, used to train the inner algorithm, and a *query* set, which is used to optimize the meta objective given the performance on the support set. When the support set contains k samples from each class in a classification task with N classes, it is called *k-shot N-class* classification. In our sentiment classification task,

the number of labels is always fixed to 2, positive and negative.

In this work, we investigate two specific meta-learning algorithms: Model-Agnostic Meta-Learning and Prototypical Networks.

3.2 Model-Agnostic Meta-Learning (MAML)

MAML is an optimization-based meta-learning framework that can be applied to any task using gradient descent to update its parameters. Consider a set of tasks $\mathcal{T}$, in our case sentiment classification on different domains, along with a distribution over these tasks $p(\mathcal{T})$. MAML updates a model's weights such that different tasks can be optimized quickly starting from that set of weights.

Consider a model f_θ, parameterized by θ. For a single task $\mathcal{T}_i$, the model performs one or more within-episode gradient steps on the support set in order to get parameters specialized for that task, θ_i'. For one step, the update is given by:

$$\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta),$$

where $\mathcal{L}_{\mathcal{T}_i}$ is a loss function evaluated on the support set, and α is a hyperparameter. The meta-objective optimizes not a single task, but multiple tasks simultaneously:

$$\min_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'}),$$

where $\mathcal{L}_{\mathcal{T}_i}$ is now evaluated on the query set. This objective can then be optimized using stochastic gradient descent.

This, however, requires calculating and storing second-order derivatives, which makes the training computationally expensive. First-order MAML (*FOMAML*) approximates meta-objective gradients by making the assumption that they only depend on the latest θ_i' from the inner gradient updates. It has been found that FOMAML performs nearly as well as regular MAML (Nichol et al., 2018). Hence, we use FOMAML in all our experiments.

A downside of MAML is that each meta-update step requires storing weights and gradients of multiple model instances simultaneously. This heavily increases memory requirements, especially so for modern architectures with many parameters.

3.3 Prototypical networks (Protonet)

Prototypical networks (Snell et al., 2017) are usually categorized as a metric-based meta-learning method. These methods are inspired by nearest neighbor and kernel density estimation techniques.

Given the support set of every class, the idea is to form corresponding class prototypes, and then use these class prototypes to classify the query set. To do this, the support and query set samples are first encoded into feature vectors. A class prototype is then obtained as the mean of all feature vectors from that class's support set. Each sample in the query set is then classified based on its feature vector's proximity to the class prototypes, and is assigned to its nearest prototype.

Formally, the predicted distribution of class probabilities for any given sample $\mathbf{x}$ from the query set is defined as follows:

$$P(y = c \mid \mathbf{x}) = \frac{\exp(-d_\varphi(f_\theta(\mathbf{x}),\ \mathbf{v}_c))}{\sum_{c' \in C} \exp(-d_\varphi(f_\theta(\mathbf{x}),\ \mathbf{v}_{c'}))},$$

where f_θ is the Prototypical network, $\mathbf{v}_c$ is the prototype for class c and d_φ is some distance measure. We use Euclidean distance in our experiments. The loss function is the negative log-likelihood of the above expression, which is equivalent to a cross-entropy loss.

In contrast to MAML, Protonet doesn't update its parameters based on new data, but uses it instead to generate prototype vectors. There is also no update done in the inner algorithm during training, and there is no need to store multiple sets of weights. This makes Protonet much more lightweight and compute efficient compared to MAML.

3.4 Multitask Learning

In multitask learning, multiple tasks are performed by the same model, such that parameters are shared over the tasks. This allows the learning procedure to gain from sharing knowledge obtained from beneficial tasks. Although many schemes for multitask learning have been proposed (Liu et al., 2017), we use one of its simplest variants: a hard-sharing model (Caruana, 1997). In the hard-sharing model, different tasks share *all* model parameters. In the context of domain adaptation, we consider each domain a different task, even though they are all still sentiment classification tasks. We expect multitask to be a very strong baseline, as we already expect it to learn domain-agnostic representations, which will be useful for domain transfer.

In contrast to the meta-learning methods, the multitask approach does not explicitly optimize for

performing well on few-shot classification. However, it does require the model to learn domain-agnostic features, which should help it perform better on an unseen domain.

3.5 Measuring Domain Similarity

Domain similarity measures quantify how similar a pair of domains are. One subclass of such measures compares word distributions, relying on the assumption that similar domains share more common terms than dissimilar ones. For example, reviews about "tablets" and "smartphones" have a large number of terms in common, such as "screen", "battery", "charger" and so on, than "smartphones" and "kitchen" reviews would have.

It has been found that domain similarity correlates well with the performance of predictive models, where in general performance decreases for domains that are less similar (Van Asch and Daelemans, 2010). In our experiments, we use the α-*Renyi* divergence[1] (Rényi, 1961) as distance measure, which uses the term frequencies as a way to describe the difference between two domains. It is computed as follows:

$$\text{Renyi}(P, Q, \alpha) \coloneqq \frac{1}{\alpha - 1} \log_2 \left(\sum_k p_k^{1-\alpha} q_k^{\alpha} \right),$$

where p_k is the relative frequency of a token k in the first corpus P, and q_k is the relative frequency of token k in the second corpus Q. We use a value of $\alpha = 1/2$, since it is symmetric in this case, and arguably more interpretable.

We further extend the idea and calculate an aggregated Renyi (aR) which can be used to compare a set of *multiple* domains with a different domain:

$$aR(T, v) \coloneqq \frac{1}{N} \sum_{t \in T} \text{Renyi}(t, v),$$

where T is a set of domains, and v is another domain.

3.6 Pointwise Mutual Information (PMI)

PMI quantifies the likelihood of the co-occurence of two random events. It considers the joint probability of them occurring and downscales it by the marginals to account for the individual contributions of each event. We use it to find characteristic words for some domains, in an experiment where we consider different support set sampling strategies.

[1]The Renyi divergence has a parameter α and Kullback-Leibler is a special case of the Rényi divergence

4 Experiments

We evaluate how models perform on domain adaptation in the context of sentiment classification. To this end, a model is trained on a set of training domains, after which its performance is evaluated on an unseen test domain.

Data We use the Fudan review dataset [2] (Liu et al., 2017). which consists of 16 individual sentiment classification datasets. They can broadly be categorized into Amazon product reviews and movie reviews. The product reviews are of different domains including books, DVDs, electronics, and so on. The goal is to classify the reviews into one of two categories: positive or negative. Additional details and statistics about the data used can be found in Table 3.

We use 14 domains for training, 1 for validation and 1 for testing. There are many options for selecting the validation and the test domains, but not all are equally informative. We hypothesize that we should find better performance on domains that are closer to the average training distribution and worse results for the ones with a larger domain shift. We therefore choose to test using two setups: one where the test domain is similar to the train domains, and one where the test domain is very different from the training domains. To this end, we measure the domain shift of each possible test domain with respect to all the other domains using the average Renyi (aR) metric described in Section 3.5. The resuling similarities are visualized in Figure 1. We find that the domain "MR" significantly differs from all other training domain, which also makes sense as it is not about product reviews. Using this knowledge, we pick two of the possible validation-test pairs:

- sports_outdoors - dvd, $aR = 0.46$ (least shift),
- imdb - MR, $aR = 0.74$ (largest shift).

Models The Protonet, multitask and MAML approaches share the same model architecture. Text representations are obtained using a pre-trained BERT[3] (Devlin et al., 2018) encoder network, with a multi-layer perceptron (*head*) stacked on top. BERT has been shown to extract useful features,

[2]https://github.com/FrankWork/fudan_mtl_reviews

[3]We use a BERT base uncased implementation from HuggingFace.

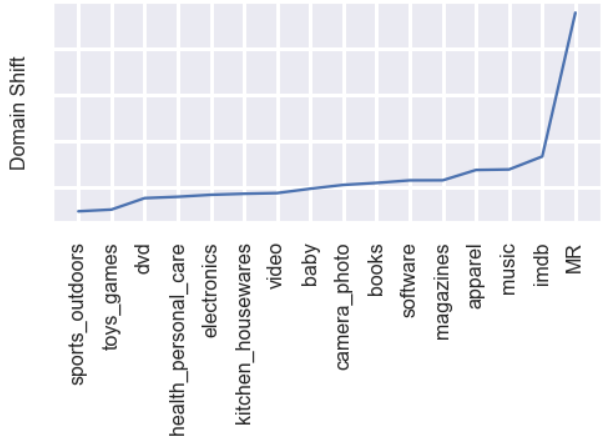

Figure 1: Domain shift across domains.

efficiently capturing context and improving performance in many downstream tasks. In our experiments, we fine-tune its 2 final layers. BERT's CLS token's representation is used as sentence representation, with 768 dimensions. We truncate all sequences longer than 512 tokens, since BERT cannot handle those. Qualitative inspection however shows that the first 512 tokens should be enough to infer the general sentiment of the text. The feed-forward network has 2 layers with hidden sizes 512 and 256, and ReLU activations[4]. To make predictions, multitask and MAML use a single-layer softmax classifier that maps the activations from the head to 2 classes. In contrast, Protonet computes the Euclidean distance between representations with each of the class prototype representations and normalizes them using a softmax function.

Training and Evaluation During training, we minimize the cross-entropy loss between the predictions and labels. The model weights are split into 2 groups updated by separate optimizers: Adam (Kingma and Ba, 2017) for the head and softmax layers, and Adam with a weight decay of 0.01 and warm-up (Loshchilov and Hutter, 2019; Howard and Ruder, 2018) for BERT. For each model, we use a linear learning rate warm-up and annealing schedule. We use 100 warm up steps and a learning rate of 5×10^{-5}. For all models, we clip gradient norms to 1. We use a dropout (Srivastava et al., 2014) of 0.1 in the BERT encoder.

Since we are interested in the practical case where few data is available, we also differentiate models based on how much data they have seen. We consider two schemes: the full and limited data schemes. In the full data scheme, all the methods see all of the samples in the training domains once. In the limited data scheme, each model is only allowed to see 15% of examples (3500 examples) in the training domains. In both cases, results are reported on the model performing best on the validation set during training.

Models are evaluated on a hold-out test domain using test accuracy. We consider a few-shot evaluation scheme, in which a model is fine-tuned on a support set containing 5 positive and 5 negative samples. Thereafter, the fine-tuned model is evaluated on the remainder of the test domain. We find that multitask performance actually degrades when it is fine-tuned on the test domain. For this reason, we report both zero-shot and few-shot multitask performance. The multitask model fine-tunes for 1 step, MAML fine-tunes for 3 steps, while Protonet is not fine-tuned, but creates prototype representations instead. This procedure is repeated for 5 different support sets randomly sampled from the test domain, to get more reliable test scores.

Support Set Selection with PMI To investigate the effect of support set selection on test domain performance, we use a heuristic to find "good" support sets. To this end, we use PMI (Section 3.6) to score samples based on how characteristic they are for a domain and sentiment class. The hypothesis here is that characteristic samples make better support sets, and allow the fine-tuning model to get a more informed perspective of the unseen domain.

The PMI score for a review is the average PMI score of all the words in it. The PMI scores for the individual words in a domain are calculated based on all the data available for that domain. Since many terms appear only once, they get assigned a large PMI score for the domain they are from. To overcome this, the term PMIs are only calculated for the first 3000 terms which appear more than than 5 times in *all* domains. A support set containing the sentences with the largest PMI score is selected.

Multitask The multitask model shares all parameters between domains. Since it is not a meta-learning approach, it does not use episodic learning, but standard batching. During training, one batch contains samples from multiple domains. We use a batch size of 16.

[4]Since Protonet does not need a classifying head, we add an extra neuron in the last layer (=257) to have an equal number of parameters.

	Low Domain Shift (DVD)		High Domain Shift (MR)	
	Limited data	Full data	Limited data	Full data
Multitask Zero-Shot	**88.28** ±**0.54**	**91.30** ±**0.19**	80.20 ±0.54	**83.02** ±**0.24**
Multitask	87.55 ±0.42	90.55 ±0.45	77.94 ±0.10	82.16 ±0.45
Protonet	85.73 ±3.21	89.12 ±0.40	78.98 ±1.84	80.91 ±1.48
MAML	83.94 ±1.74	87.32 ±0.58	**80.30** ±**0.54**	82.31 ±0.65

Table 1: Model performance on four different train-test data setup. Means and standard deviations across three different random seeds are given. All results except for Multitask Zero-Shot are given on 5-shot evaluation.

Protonet Each episode has a support and a query set containing examples of positive and negative sentiment from a domain. For accurate prototype creation, each support set contains an equal number of positive and negative samples ($k = 5$). The samples from the support set are passed through the model and the averaged representations serve as class prototypes. The final loss is then calculated using these prototypes and averaged across $2k$ query samples.

MAML Each episode for MAML has 5 domains, sampled uniformly. From each domain, 10 support samples are used to perform 3 inner gradient descent steps on separate copies of the network. In the meta-update, the last set of parameters from the inner loop is used to compute the loss over 10 query samples.

5 Results and Analysis

We first show our main result, which is the performance of the multitask and meta-learning models on domain adaptation for the sentiment classification task. Table 1 shows model test accuracy on the four different train-test data setup described in Section 4. We include the zero-shot multitask model, which does not do any fine-tuning on the test dataset, since it has a higher performance than its fine-tuned counterpart, and thus represents our baseline at its strongest.

Suprisingly, the zero-shot multitask model performs best on all testing setups except limited data high domain shift, in which MAML slightly outperforms it. This shows that the multitask approach is an extremely good baseline for the domain adaptation task, as it outperforms meta-learning methods which explicitly optimize for fast adaptation. It is also quite remarkable that the multitask performance degrades when fine-tuning on the test domain in all setups. It might be that the fine-tuning interferes with domain-agnostic representations of the model.

MAML performs quite a lot worse than the other methods on the low domain shift setup. In the high domain shift setup, it actually closes this gap, even outperforming the zero-shot multitask model, albeit not significantly. This result suggests that the power of MAML for domain adaptation might only be visible when evaluated on domains with larger domain shift. Whether this is true, and why this happens, could be investigated in further work. In addition, we find that Protonet outperforms MAML in the low domain shift case, but suffers more in performance when domain shift is larger.

We try to give plausible reasons for the observed model performance, especially the high performance of the multitask approach. First, the encoder model that we use, BERT, is a high-capacity model with over a hundred million parameters. Recent work shows that such high-capacity language models are excellent few shot learners in their own right (Brown et al., 2020). The multitask approach takes full advantage of the encoder modeling capacity, and may be induced to learn domain-agnostic features by training on different domains. It might be that the added complexity introduced by the meta-learning training regime is instead a detriment to performance.

Another possible explanation lays with the specific task that we're examining. It might be that for the sentiment classification task, domain shift is not as large of a problem as it would be for other tasks, such that the power of meta-learning approaches over multitask approaches cannot be observed as well as they could be. We hypothesize that sentiment might be inferred from more general words than ones that change meaning under domain shift. However, further experiments would need to be performed to test this hypothesis.

To investigate the effect of domain shift during learning, we look at the training process closely.

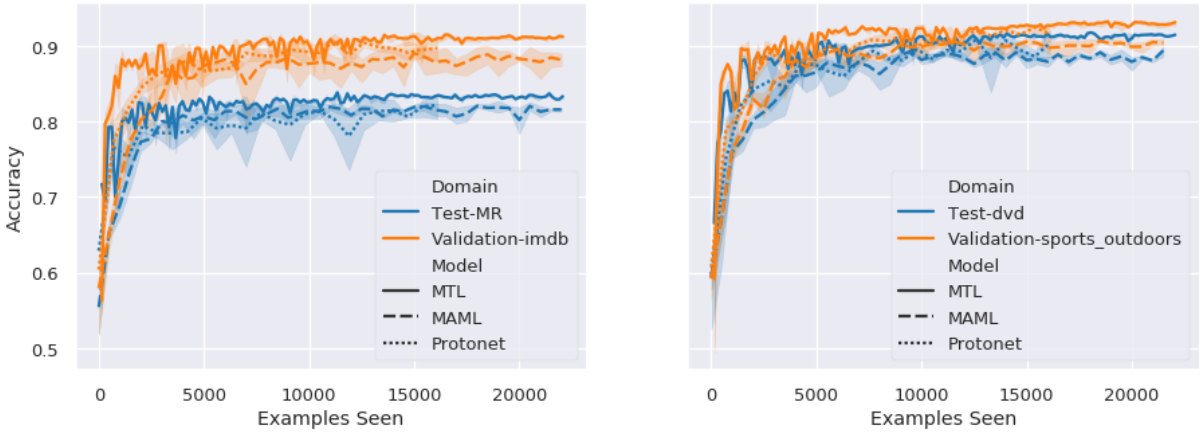

Figure 2: Trends in accuracy of different model types on out-of-domain data for the high domain shift train-test split (left) and low domain shift train-test split (right). Shown are means and standard deviations across three run.

	Low Domain Shift (DVD)	High Domain Shift (MR)
Multitask	90.76 ±0.39	82.45 ±0.52
Protonet	89.84 ±0.42	81.60 ±0.24
MAML	89.23 ±0.63	82.73 ±0.23

Table 2: Test accuracy using the support set with the highest PMI scores. Means and standard deviations across three different random seeds are given.

Figure 2 records the accuracy on the validation and test domains throughout the training. Here, we make two observations: first, neither the validation nor the test curves decline after saturation in any of the experiments. This shows that the models don't overfit on the data or the domains. Second, none of the methods are able to generalize across the domain shift. There is a clear gap in the accuracy of all three models when they are evaluated on a very different domain (MR), whereas this gap is non-existent when they are evaluated on similar domains.

To explore the effect of support set selection on performance, we test the trained models on support sets that are most characteristic of their domain according to PMI (Section 3.6). Table 2 shows that the highly representative support set improves performance for all models slightly. MAML performance on the low domain shift full data setup improves quite significantly, implying that the choice of support set does make quite a large difference for it.

6 Conclusion

We propose to use a meta-learning approach for sentiment classification under domain shift. We find that meta-learning approaches MAML and Protypical Networks don't improve domain adaptation performance on the sentiment classification task over a multitask approach, which proves to be a very strong baseline instead. This is surprising, as meta-learning approaches explicitly optimize for quick adaptation. MAML underperforms the multitask approach severely when domain shift is not very high, but closes this gap as the domain shift increases. This suggests a direction for future work into investigating meta-learning performance for even larger domain shifts, or harder adaptation problems in general. We further find that selecting support sets with characteristic words in unseen domains slightly increases performance for all methods. Future work could investigate which support set selection methods work well for domain adaptation, and what properties good support sets should have in this context.

References

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.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language*

Processing, EMNLP '06, pages 120–128, Sydney, Australia. Association for Computational Linguistics.

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. *arXiv:2005.14165 [cs]*. ArXiv: 2005.14165.

Rich Caruana. 1997. Multitask Learning. *Machine Learning*, 28(1):41–75.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*. ArXiv: 1810.04805.

Timothy Dozat and Christopher D. Manning. 2017. Deep Biaffine Attention for Neural Dependency Parsing. *arXiv:1611.01734 [cs]*. ArXiv: 1611.01734.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *arXiv:1703.03400 [cs]*. ArXiv: 1703.03400.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2017. Domain-Adversarial Training of Neural Networks. In *Domain Adaptation in Computer Vision Applications*, pages 189–209. Springer International Publishing, Cham. Series Title: Advances in Computer Vision and Pattern Recognition.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*. ArXiv: 1512.03385.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, pages 1735–1780.

Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. *arXiv:1801.06146 [cs, stat]*. ArXiv: 1801.06146.

Jing Jiang and ChengXiang Zhai. 2007. Instance Weighting for Domain Adaptation in NLP. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 264–271, Prague, Czech Republic. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*. ArXiv: 1412.6980.

Da Li and Timothy Hospedales. 2020. Online Meta-Learning for Multi-Source and Semi-Supervised Domain Adaptation. *arXiv:2004.04398 [cs]*. ArXiv: 2004.04398.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial Multi-task Learning for Text Classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1–10, Vancouver, Canada. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. *arXiv:1711.05101 [cs, math]*. ArXiv: 1711.05101.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective Self-Training for Parsing. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 152–159, New York City, USA. Association for Computational Linguistics.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On First-Order Meta-Learning Algorithms. *arXiv:1803.02999 [cs]*. ArXiv: 1803.02999.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - EMNLP '02*, volume 10, pages 79–86, Not Known. Association for Computational Linguistics.

Sachin Ravi and Hugo Larochelle. 2017. Optimization as a Model for Few-Shot Learning. *International Confer- ence on Learning Representations (ICLR)*.

Sebastian Ruder and Barbara Plank. 2018. Strong Baselines for Neural Semi-Supervised Learning under Domain Shift. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1044–1054, Melbourne, Australia. Association for Computational Linguistics.

Alfréd Rényi. 1961. On Measures of Entropy and Information. The Regents of the University of California. ISSN: 0097-0433.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. Meta-Learning with Memory-Augmented Neural Networks. *International Conference on Learning Representations (ICLR)*, 48:9.

Schmidhuber, Jürgen. 1987. *Evolutionary principles in self-referential learning, or on learning how to learn: The meta-meta-... hook*. Ph.D. thesis, Technische Universität München, München.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical Networks for Few-shot Learning. *arXiv:1703.05175 [cs, stat]*. ArXiv: 1703.05175.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Mark Steedman, Rebecca Hwa, Stephen Clark, Miles Osborne, Anoop Sarkar, Julia Hockenmaier, Paul Ruhlen, Steven Baker, and Jeremiah Crim. 2003. Example Selection for Bootstrapping Statistical Parsers. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 236–243.

Sebastian Thrun and Lorien Pratt. 1998. Learning to Learn: Introduction and Overview. In Sebastian Thrun and Lorien Pratt, editors, *Learning to Learn*, pages 3–17. Springer US, Boston, MA.

Vincent Van Asch and Walter Daelemans. 2010. Using Domain Similarity for Performance Estimation. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, pages 31–36, Uppsala, Sweden. Association for Computational Linguistics.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. 2016. Matching Networks for One Shot Learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3630–3638. Curran Associates, Inc.

Appendix A

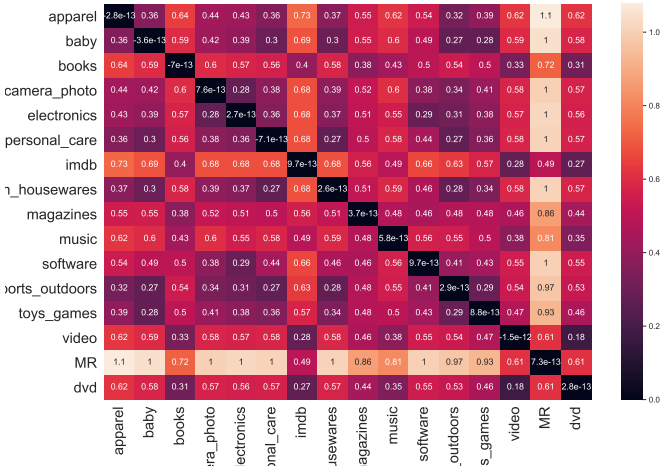

Figure 3: Heatmap of the distribution distances between each of the 16 datasets

	Train	Validation	Test	Average Review Length
apparel	1280 (49%)	320 (49%)	400 (54%)	67
baby	1200 (53%)	300 (54%)	400 (50%)	119
books	1280 (50%)	320 (50%)	400 (50%)	183
camera_photo	1277 (49%)	320 (55%)	400 (48%)	138
electronics	1278 (50%)	320 (51%)	400 (50%)	117
health_personal_care	1280 (52%)	320 (45%)	400 (47%)	94
kitchen_housewares	1280 (51%)	320 (48%)	400 (48%)	98
magazines	1256 (50%)	314 (51%)	400 (54%)	127
music	1280 (50%)	320 (48%)	400 (50%)	149
software	1212 (53%)	303 (50%)	400 (53%)	149
sports_outdoors	1279 (51%)	320 (49%)	400 (48%)	106
toys_games	1280 (51%)	320 (46%)	400 (52%)	102
video	1280 (50%)	320 (48%)	400 (53%)	162
dvd	1280 (51%)	320 (47%)	400 (50%)	196
MR	1280 (48%)	320 (50%)	400 (52%)	21
imdb	1280 (50%)	320 (49%)	400 (50%)	270

Table 3: Statistics for each of the domain datasets. The percentage of positive sentiment samples are given in parentheses. The last two datasets are movie revews instead of product reviews.