Meta Distant Transfer Learning for Pre-trained Language Models

Chengyu Wang[1,2], **Haojie Pan**[2], **Minghui Qiu**[2]*, **Fei Yang**[1], **Jun Huang**[2], **Yin Zhang**[3]
[1] Zhejiang Lab, [2] Alibaba Group
[3] College of Computer Science and Technology, Zhejiang University
{chengyu.wcy,haojie.phj,minghui.qmh}@alibaba-inc.com,
huangjun.hj@alibaba-inc.com,
yangf@zhejianglab.com, zhangyin98@zju.edu.cn

Abstract

With the wide availability of Pre-trained Language Models (PLMs), multi-task fine-tuning across domains has been extensively applied. For tasks related to distant domains with different class label sets, PLMs may memorize non-transferable knowledge for the target domain and suffer from negative transfer. Inspired by meta-learning, we propose the *Meta Distant Transfer Learning* (Meta-DTL) framework to learn the cross-task knowledge for PLM-based methods. Meta-DTL first employs task representation learning to mine implicit relations among multiple tasks and classes. Based on the results, it trains a PLM-based meta-learner to capture the transferable knowledge across tasks. The weighted maximum entropy regularizers are proposed to make meta-learner more task-agnostic and unbiased. Finally, the meta-learner can be fine-tuned to fit each task with better parameter initialization. We evaluate Meta-DTL using both BERT and ALBERT on seven public datasets. Experiment results confirm the superiority of Meta-DTL as it consistently outperforms strong baselines. We find that Meta-DTL is highly effective when very few data is available for the target task.

1 Introduction

Owning to the availability of *Pre-trained Language Models* (PLMs), the performance of various text classification tasks has been significantly improved. Notable PLMs include BERT (Devlin et al., 2019), ALBERT (Lan et al., 2020), XLNet (Yang et al., 2019), T5 (Raffel et al., 2020), GPT-3 (Brown et al., 2020) and many others. It can be safely concluded that PLM-based approaches achieve state-of-the-art results for a majority of text classification tasks.

Among these methods, a key procedure of PLMs is *fine-tuning*, which enables parameters of PLMs to fit specific datasets. Hence, the performance of PLMs on a downstream task may be limited by the availability of the training set. As reported by several benchmarks such as GLUE (Wang et al., 2019b) and SuperGLUE (Wang et al., 2019a), some PLMs may not perform well in low-resource tasks. A popular solution in NLP is *transfer learning* (Zhuang et al., 2019; Alyafeai et al., 2020). For PLMs, these models can be fine-tuned over both source-domain and target-domain datasets by various multi-task training strategies (Li et al., 2019; Arase and Tsujii, 2019). Unfortunately, several studies reveal that multi-task training of PLMs across domains does not always guarantee satisfactory results (Sun et al., 2019; Wang et al., 2020). As PLMs usually have large parameter space and strong memorization power, learning from source-domain datasets may force PLMs to *memorize* non-transferable knowledge of source domains, leading to the *negative transfer* effect (Wang et al., 2019d).

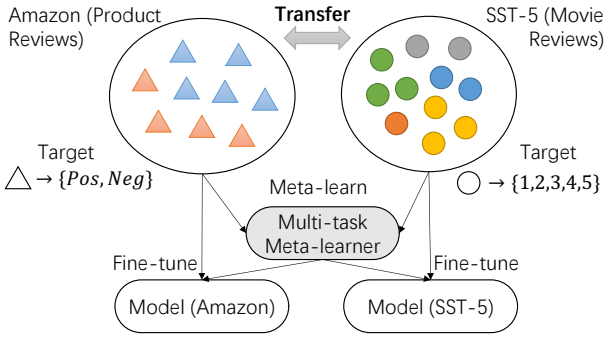

Figure 1: A simple example of *Meta Distant Transfer Learning* for review analysis. (Best viewed in color.)

Besides, a large number of transfer learning algorithms address tasks across similar sub-domains, with the same set of class labels. [1] When there exist large domain gaps and class label differences, these transfer learning solutions are likely to fail. Consider a simple, motivation example in Figure 1. De-

* Corresponding author.

[1] For example, the Amazon Reviews (Blitzer et al., 2007) dataset is often used for evaluating transfer learning algorithms, which consists e-commerce product reviews divided into four sub-domains: book, DVD, electronics and kitchen.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 9742–9752
November 7–11, 2021.

spite the fact that the two datasets (SST-5 (Socher et al., 2013) and Amazon Reviews (Blitzer et al., 2007)) are diverse in domains and classification targets, they aim to solve similar review analysis tasks. It would be beneficial for the two task-specific models to learn from each other. A few methods address the *distant domain* issue in transfer learning (Tan et al., 2017; Xiao and Zhang, 2020), but are not designed for PLMs. A natural question arises: *how can we transfer knowledge across distant domains with different classification targets for PLM-based text classification*?

Recently, meta-learning has been studied extensively, which learns parameters that can be adapted to a group of similar tasks (Finn et al., 2017, 2018). For PLMs, Wang et al. (2020) suggest that training a *meta-learner* for PLMs is highly effective to capture transferable knowledge across sub-domains. However, this method is not designed for tasks across diverse domains and class label sets. Additionally, it lacks the mechanism to learn *task-agnostic* representations and may fit too much to specific targets in certain datasets.

To this end, the *Meta Distant Transfer Learning* (Meta-DTL) framework is proposed[2]. Specially, Meta-DTL employs a *task representation learning* procedure to obtain a collection of *prototype vectors* for each task. To understand how to transfer across these tasks and classes, we construct a *Meta Knowledge Graph* (Meta-KG) to characterize the implicit relations among tasks and classes, based on the representations of multiple tasks. The *meta-learner* in Meta-DTL can be initialized by any PLMs and trained by multi-task learning with rich *meta-knowledge* injected from Meta-KG. Additionally, we design the *Weighted Maximum Entropy Regularizers* to make the model more *task-agnostic* and *unbiased*. Finally, the meta-learner can be fine-tuned to fit each task using its own training set. In this way, the model is able to digest the cross-task, transferable knowledge and alleviates *negative transfer*.

We apply Meta-DTL to BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2020) for three sets of NLP tasks (seven public datasets in total): i) coarse and fine-grained review analysis across domains; ii) natural language inference (across sentence relation prediction and scientific question answering); and iii) lexical semantics (across hypernymy detection and lexical relation classification). Experiments show that Meta-DTL consistently outperforms strong baselines. We also show that Meta-DTL is highly useful for text classification when very few training samples of the target task are available.

[2]We name our algorithm to be *Meta Distant Transfer Learning* because it is inspired by the idea of meta-learning to capture the cross-task knowledge for task adaptation. We would like to clarify that Meta-DTL is used in a *distant transfer learning* setting for PLMs, instead of the traditional *K-way N-shot* setting in meta-learning (Finn et al., 2017, 2018).

2 Related Work

In this section, we summarize the related work on PLMs, transfer learning and meta-learning.

2.1 Pre-trained Language Models

PLMs have brought NLP to a new era, pushing the performance of various NLP tasks to new heights (Qiu et al., 2020). Among these models, ELMo (Peters et al., 2018) employs BiLSTM to learn context-sensitive embeddings from both directions. BERT (Devlin et al., 2019) is one of the most popular PLMs that learns language representations by transformer encoders. ALBERT (Lan et al., 2020) employs several parameter sharing and factorization techniques to reduce the sizes of BERT-style models. Other transformer encoder-based architectures include Transformer-XL (Dai et al., 2019), XLNet (Yang et al., 2019), Big Bird (Zaheer et al., 2020), etc. The encoder-decoder architectures are used in T5 (Raffel et al., 2020) and GPT-3 (Brown et al., 2020), which are ultra-large PLMs with 11 billion and 175 billion parameters, respectively. PLMs can also be pre-trained by supervised tasks, such as MT-DNN (Liu et al., 2019). Apart from pre-training PLMs, a few works focus on fine-tuning, such as Sun et al. (2019); Cui et al. (2019); Zhao and Bethard (2020); Wang et al. (2020). Different from these works, we pay attention to transferring knowledge across distant domains for PLMs.

2.2 Transfer Learning

Transfer learning is a widely used paradigm for transferring resources from source domains to target domains (Pan and Yang, 2010; Lu et al., 2015; Zhuang et al., 2019; Wang et al., 2019c). For deep neural networks, it is common practice to learn similar tasks by multi-task learning. Among these methods, the "shared-private" architectures are frequently applied (Liu et al., 2017; Chen et al., 2018; Qiu et al., 2019), consisting of task-specific sub-

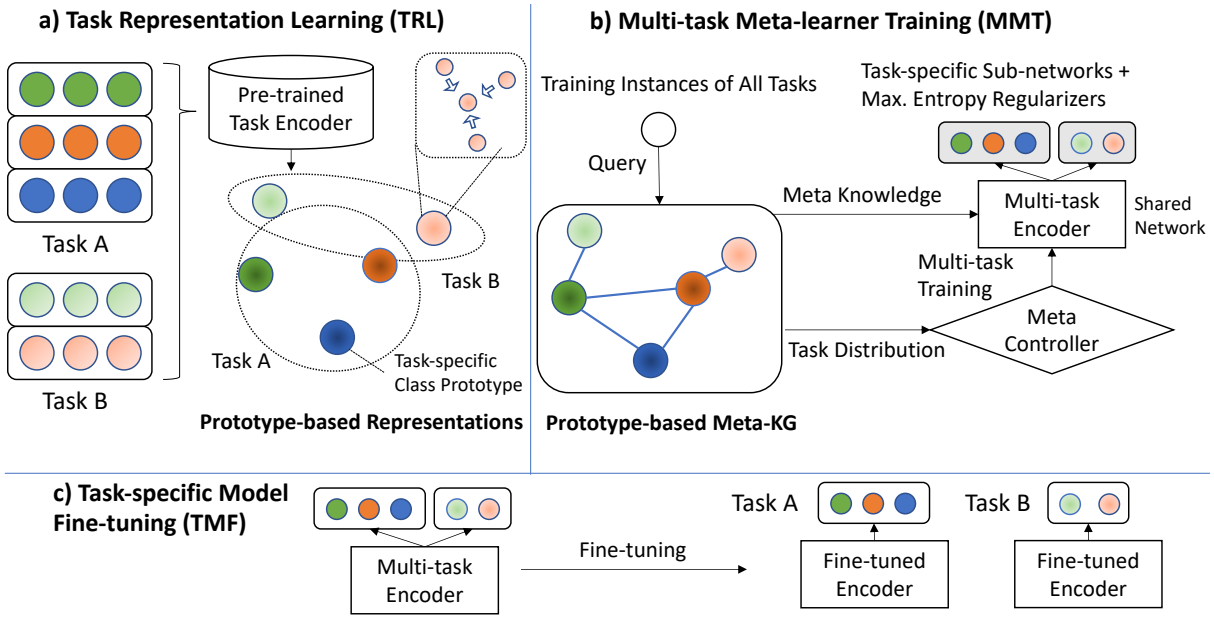

Figure 2: The high-level architecture of Meta-DTL. (Best viewed in color.)

networks and a shared sub-network. Meta-DTL transfers knowledge across tasks from a different perspective, which captures transferable knowledge by the meta-learner and passes the knowledge to task-specific models by fine-tuning.

2.3 Meta-learning

Meta-learning aims to train meta-learners that can quickly adapt to different tasks with little training data (Vanschoren, 2018). Typically, meta-learning is applied in few-shot learning as a *K-way N-shot* problem, such as few-shot image classification in computer vision (Zhang et al., 2020; Afrasiyabi et al., 2020). In NLP, applications that employ meta-learning include few-shot link prediction in knowledge graphs (Chen et al., 2019), relation classification (Ye and Ling, 2019; Wang et al., 2021b), natural language generation (Chen et al., 2020), question answering (Hua et al., 2020), named entity recognition (Yang and Katiyar, 2020), text classification (Bao et al., 2020; Wang et al., 2021a) etc. In contrast, Meta-DTL is not a typical *K-way N-shot* algorithm. Similar to Wang et al. (2020); Pan et al. (2021), it leverages the techniques of meta-learning to obtain the meta-learner, which is better at learning knowledge across tasks.

3 Meta-DTL: The Proposed Framework

In this section, we first present our task. After that, the technical details of the Meta-DTL framework are elaborated.

3.1 A Brief Overview of Meta-DTL

3.1.1 Task Overview

Let $\mathcal{T}_1, \cdots, \mathcal{T}_K$ be K text classification tasks, with the corresponding training sets denoted as $\mathcal{D}_1, \cdots, \mathcal{D}_K$. In the single-task setting, the goal of the task $\mathcal{T}_i$ is to learn a model from $\mathcal{D}_i$ to map its input instance to one of the class labels in $\mathcal{C}_i$, where $\mathcal{C}_i$ is the class label set of $\mathcal{T}_i$.

Apart from the domain differences, we consider the situation where the class label sets may also be different. Formally, among the K tasks, there exists at least one task pair ($\mathcal{T}_i$ and $\mathcal{T}_j$) such that the lassociated abel sets $\mathcal{C}_i \neq \mathcal{C}_j$. [3] Re-consider the example in Figure 1. The goal is to classify reviews into a 5-point rating scale and a positive/negative rating scale, respectively, in two distant domains (i.e., movies and e-commerce products). We can conclude it is crucial for the model to learn *what to transfer* and *how to transfer* across these tasks, in order to improve the performance of both models.

3.1.2 Solution Overview

An overview of Meta-DTL is shown in Figure 2. It consists of three modules: i) *Task Representation Learning* (TRL), ii) *Multi-task Meta-learner Training* (MMT), and iii) *Task-specific Model Fine-tuning* (TMF).

Specially, for each task $\mathcal{T}_i$, TRL employs a *pre-*

[3] We add such constraint in our work to show that Meta-DTL works well in more challenging scenarios. Nonetheless, Meta-DTL also works without such constraint.

trained task encoder to do a one-pass scan over the training set $\mathcal{D}_i$. It represents each task $\mathcal{T}_i$ as a collection of *prototypical vectors*, denoted as $\mathcal{P}_i = \{\vec{p}_{i,j}\}$ where $\vec{p}_{i,j}$ is the j-th prototypical embedding vector of $\mathcal{T}_i$, corresponding to the j-th class in $\mathcal{D}_i$. Here, $\mathcal{P}_i$ gives us a panoramic picture of the task $\mathcal{T}_i$ in the embedding space. As we aim to address *distant transfer learning*, simple multi-task training inevitably suffers from *negative transfer*. In MMT, we obtain a meta-learner $\mathcal{M}$ that only digests *transferable knowledge* across all the K tasks. We first construct a prototype-based *Meta Knowledge Graph* (Meta-KG, denoted as G) from $\mathcal{P}_i, \cdots, \mathcal{P}_K$, implicitly describing the relations among tasks and classes. For each training instance of all tasks $x_{i,j}$, we query $x_{i,j}$ in G to generate the meta-knowledge score $m_{i,j}$, which represents the degree of the *knowledge transferability* of the input $x_{i,j}$. [4] Additionally, the *Weighted Maximum Entropy Regularizers* (WMERs) are proposed and integrated into the model to make the meta-learner $\mathcal{M}$ more *task-agnostic* and *unbiased*. Finally, in TMF, we fine-tune the meta-learner $\mathcal{M}$ to generate the K classifiers for the K tasks, based on their own training sets $\mathcal{D}_1, \cdots, \mathcal{D}_K$.

3.2 Task Representation Learning

The first step of TRL is to learn the implicit relations among classes across K tasks. In meta-learning, *prototypes* are frequently employed to characterize the class information by concrete representations (Snell et al., 2017). We notice that in NLP tasks, a lot of class labels have rich meanings that are useful to model the class semantics. For example, the label *positive* in review analysis can be directly associated with positive terms in reviews (e.g., "price-worthy", "good value", "enjoyable"). Let $\mathcal{D}_{i,j}$ be the subset of $\mathcal{D}_i$ with instances assigned to the j-th class label, i.e., $\mathcal{D}_{i,j} = \{x_{i,j} \in \mathcal{D}_i | c_{i,j} = c_j, c_j \in \mathcal{C}_i\}$. The j-th prototypical vector $\vec{p}_{i,j}$ of task $\mathcal{T}_i$ is defined as follows:

$$\vec{p}_{i,j} = \frac{1}{|\mathcal{D}_{i,j}|} \sum_{x_{i,j} \in \mathcal{D}_{i,j}} \mathcal{E}(x_{i,j}, c_{i,j})$$

where $\mathcal{E}(\cdot, \cdot)$ is an embedding function that encodes both the textual input $x_{i,j}$ and its class label $c_{i,j}$ by PLMs. By combining all such vectors, we obtain the representation of the task $\mathcal{T}_i$ as: $\mathcal{P}_i = \{\vec{p}_{i,j}\}$. [5]

The *self-attention mechanism* (Vaswani et al., 2017) frequently used in PLMs such as BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2020) makes it quite straight-forward to implement the function $\mathcal{E}(\cdot, \cdot)$. For single-text classification, the input to the pre-trained encoder is formatted as "[CLS]$x_{i,j}$[SEP]$c_{i,j}$[SEP]". For text pairs, the input is "[CLS]$x^{(1)}_{i,j}$[SEP]$x^{(2)}_{i,j}$[SEP]$c_{i,j}$[SEP]" where $x^{(1)}_{i,j}$ and $x^{(2)}_{i,j}$ represent the pair of $x_{i,j}$. During the forward pass of the hidden layers, the text inputs and the class label can attend to each other. Hence, the label information is fused into the input text representations. Finally, we take the average pooled output of the last encoder layer as $\mathcal{E}(x_{i,j}, c_{i,j})$. [6]

Algorithm 1 Meta-learner Training Algorithm

1: Construct the Meta-KG $G(V, L)$;
2: **for** each training instance $x_{i,j} \in \bigcup_{i=1,\cdots,K} \mathcal{D}_i$ **do**
3: Compute $\alpha_{i,j}$, $\beta_{i,j}$ and $m_{i,j}$ based on G;
4: **end for**
5: Restore the underlying PLM's parameters from the pre-trained model, with others randomly initialized;
6: **while** number of training steps does not reach a limit **do**
7: Sample a batch $\mathcal{B} = \{x_{i,j}\}$ from $\bigcup_{i=1,\cdots,K} \mathcal{D}_i$, each selected with the probability $p(x_{i,j}) = \frac{1}{K|\mathcal{D}_i|}$;
8: Update all parameters by minimizing the loss function $\sum_{x_{i,j} \in \mathcal{B}} \mathcal{L}(x_{i,j})$;
9: **end while**
10: **return** the meta-learner $\mathcal{M}$ (i.e., the collection of updated parameters of the PLM).

3.3 Multi-task Meta-learner Training

The training algorithm of the meta-learner is summarized in Algorithm 1.

3.3.1 Obtaining the Meta-knowledge

After TRL, we represent all the acquired knowledge $\mathcal{P}_i, \cdots, \mathcal{P}_K$ as the Meta-KG $G(V, L)$. In G, each prototypical vector $\vec{p}_{i,j}$ is treated as a node

[4] Utilizing the meta-knowledge score $m_{i,j}$ can be also viewed as selecting the *most common* training instances in the *distant transfer learning* setting.

[5] Note that we have also tested using multiple prototypical vectors (generated by K-means) to represent the semantics of a class and achieved similar performance. For simplicity, we use the "one-vector-per-class" setting in this paper.

[6] In some cases where the class labels have no semantic meanings (for example, IDs only), the input embeddings of the class labels to the task encoder can be randomly initialized.

in V. Each edge $l_{i,j,m,n} \in L$ denotes the similarity between two prototypical vectors $\vec{p}_{i,j}$ and $\vec{p}_{m,n}$. For simplicity, we have assume edge weight $w(l_{i,j,m,n}) = \cos(\vec{p}_{i,j}, \vec{p}_{m,n})$, with $\cos(\cdot,\cdot)$ being the cosine similarity function. Hence, G is a highly condensed representation of all the K tasks, considering the class semantics.

During MMT, for each input $x_{i,j}$, we query $x_{i,j}$ in G to generate the meta-knowledge $m_{i,j}$. Here, we treat the meta-knowledge as a scalar to represent the *degree of transferablity*. Firstly, we define the *instance-level meta-knowledge* $\alpha_{i,j}$ as follows:

$$\alpha_{i,j} = \max_{\vec{p}_{m,n} \in \tilde{\mathcal{P}}_i} \cos(\mathcal{E}(x_{i,j}, c_{i,j}), \vec{p}_{m,n})$$

where $\tilde{\mathcal{P}}_i$ is the collection of all prototypical vectors not associated with the task $\mathcal{T}_i$. Hence, if $(x_{i,j}, c_{i,j})$ is similar to the instances in any class of the $K-1$ tasks, it should be more *transferable*, thus is more useful when we train the meta-learner.

However, using the weight $\alpha_{i,j}$ alone is not sufficiently robust when the input instance is an *abnormal* sample (i.e., an outlier). We further consider the *class-level meta-knowledge* $\beta_{i,j}$ as follows:

$$\beta_{i,j} = \max_{\vec{p}_{m,n} \in \tilde{\mathcal{P}}_i} \cos(\vec{p}_{i,j}, \vec{p}_{m,n})$$

where we replace $\mathcal{E}(x_{i,j}, c_{i,j})$ with its class prototypical vector $\vec{p}_{i,j}$. We can see that the computation is highly efficient as all such weights have been pre-computed and stored in G. Finally, the meta-knowledge $m_{i,j}$ is computed as: $m_{i,j} = \frac{\alpha_{i,j}+\beta_{i,j}}{2}$.

3.3.2 Training the Meta-learner

As discussed earlier, the properties of a *good* meta-learner should be twofold: i) *capturing transferable knowledge* and ii) *being task-agnostic and unbiased*.

The architecture of the meta-learner in Meta-DTL is similar to MT-DNN (Liu et al., 2019) where each task has its own task-specific output layer, plus a shared PLM-based encoder. The parameters of the underlying PLM encoder are initialized by its pre-training results. A *meta controller* is designed to control the training process. It constrains that each data instance $x_{i,j}$ is selected with the probability $p(x_{i,j}) = \frac{1}{K|\mathcal{D}_i|}$. Hence, each task $\mathcal{T}_i$ is selected with the probability $p(\mathcal{T}_i) = \sum_{x_{i,j} \in \mathcal{D}_i} p(x_{i,j}) = \frac{1}{K}$. Here, we employ the uniform distribution, i.e., $p(\mathcal{T}_i) = \frac{1}{K}$, which ensures that each task has *equal* opportunity to be learned. During training, the first loss is the *weighted* cross-entropy loss $\mathcal{L}_{CE}(x_{i,j})$: [7]

$$\mathcal{L}_{CE}(x_{i,j}) = -\sum_{c \in \mathcal{C}_i} \mathbf{1}_{(c_{i,j}=c)} m_{i,j} \log \tau_c(x_{i,j})$$

where $\mathbf{1}_{(\cdot)}$ is the indicator function that returns 1 if the input function is true and 0 otherwise. $\tau_c(x_{i,j})$ is the predicted probability of $x_{i,j}$ associated with the class $c \in \mathcal{C}_i$. $\mathcal{L}_{CE}(x_{i,j})$ ensures that each sample $x_{i,j}$ is weighted by $m_{i,j}$. Hence, *transferable* instances gain larger weights during training.

However, minimizing $\mathcal{L}_{CE}(x_{i,j})$ may result in a *biased* meta-learner. Consider a simple example where $\mathcal{T}_1$ and $\mathcal{T}_2$ are highly similar to each other; $\mathcal{T}_3$ is more dis-similar. Based on the previous procedure, training instances of $\mathcal{T}_1$ and $\mathcal{T}_2$ have larger weights in general. Hence, the meta-learner is biased towards $\mathcal{T}_1$ and $\mathcal{T}_2$, which gives poor initialization values when learning the final model for $\mathcal{T}_3$. To make the model more *task-agnostic*, inspired by Jamal and Qi (2019), we integrate the meta-knowledge into the maximum entropy regularization and propose the *Weighted Maximum Entropy Regularizers* (WMERs) as an auxiliary loss, denoted as $\mathcal{L}_{ME}(x_{i,j})$:

$$\mathcal{L}_{ME}(x_{i,j}) = -\sum_{c \in \mathcal{C}_i} \frac{m_{i,j}}{|\mathcal{C}_i|} \log \tau_c(x_{i,j})$$

where the predicted probability of each sample $x_{i,j}$ is compared against the $|\mathcal{C}_i|$-dimensional uniform distribution $(\frac{1}{|\mathcal{C}_i|}, \cdots, \frac{1}{|\mathcal{C}_i|})$ by cross-entropy, weighted by $m_{i,j}$. $\mathcal{L}_{ME}(x_{i,j})$ penalizes the meta-learner for fitting too much to specific tasks, avoiding the generation of *biased* models. Finally, the total sample-wise loss $\mathcal{L}(x_{i,j})$ is derived as:

$$\mathcal{L}(x_{i,j}) = -\sum_{c \in \mathcal{C}_i} m_{i,j}(\mathbf{1}_{(c_{i,j}=c)} + \frac{\lambda}{|\mathcal{C}_i|}) \log \tau_c(x_{i,j})$$

where $\lambda \in (0, 1)$ is a pre-defined balancing factor between the two losses.

Discussion. Based on the derivation of $\mathcal{L}(x_{i,j})$, we can see that each sample $x_{i,j}$ is only associated with a $|\mathcal{C}_i|$-length constant vector where each element is $m_{i,j}(\mathbf{1}_{(c_{i,j}=c)} + \frac{\lambda}{|\mathcal{C}_i|})$, which can be pre-computed based on G. Re-consider the process in Algorithm 1. We do not use second-order update steps similar to MAML (Finn et al., 2017). This is because i) such training process of large-scale PLMs would be computationally expensive;

[7] For simplicity, we omit all regularization terms in the loss function throughout this paper. The three losses $\mathcal{L}_{CE}(x_{i,j})$, $\mathcal{L}_{ME}(x_{i,j})$ and $\mathcal{L}(x_{i,j})$ refer to the *sample-wise loss* only.

ii) our algorithm does not have any meta-testing steps. Hence, our algorithm is highly efficient for learning across multiple NLP tasks in a large scale.

3.4 Task-specific Model Fine-tuning

After obtaining the meta-learner $\mathcal{M}$, in TMF, we fine-tune $\mathcal{M}$ to generate K classifiers for the K underlying tasks separately, based on their own task-specific training sets $\mathcal{D}_1, \cdots, \mathcal{D}_K$. The meta-knowledge and WMERs are removed from the loss function. Hence, the total *dataset-level* loss function $\mathcal{L}^*(\mathcal{T}_i)$ of the task $\mathcal{T}_i$ is defined as follows:

$$\mathcal{L}^*(\mathcal{T}_i) = -\sum_{x_{i,j} \in \mathcal{D}_i} \sum_{c \in \mathcal{C}_i} \mathbf{1}_{(c_{i,j}=c)} \log \tau_c^*(x_{i,j})$$

where $\tau_c^*(x_{i,j})$ is the task-specific prediction function of the input $x_{i,j}$ w.r.t. the class $c \in \mathcal{C}_i$.

4 Experiments

In this section, we conduct extensive experiments to evaluate the performance of Meta-DTL and compare it against strong baselines.

4.1 Datasets and Experimental Settings

We employ both BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2020) as our PLMs to evaluate Meta-DTL[8]. Three sets of NLP tasks are used for evaluation, with the statistics of all the seven public datasets reported in Table 1:

- **Review Analysis**: It transfers knowledge across three datasets for coarse and fine-grained review sentiment classification, namely Amazon (Blitzer et al., 2007), IMDb (Maas et al., 2011) and SST-5 (Socher et al., 2013). Note that the domains of SST-5 and IMDb are different from Amazon.

- **Natural Language Inference**: Two different sentence pair classification tasks related to Natural Language Inference (NLI) are considered. MNLI (Williams et al., 2018) is a large-scale benckmark dataset, with the task of predicting the relation between a sentence pair as "entailment", "neutral" or "contradiction". SciTail (Khot et al., 2018) is a scientific question answering task, with only two labels: "entailment" and "neutral".

- **Lexical Semantics**: We further consider two term pair classification tasks extensively studied in lexical semantics. Shwartz (Shwartz et al., 2016) is a popular dataset for hypernymy detection, which aims at classifying term pairs into "hypernymy" ("is-a") or "non-hypernymy" based on their semantic meanings. BLESS (Baroni and Lenci, 2011) is a dataset derived from WordNet, which is used to evaluate lexical relation classification models. The BLESS task involves a wider spectrum of lexical relation types, such as hypernymy, co-hyconymy and meronymy.

In each set of the experiments, we transfer knowledge from all the other tasks in the same set to the target one. For example, the model for SST-5 is trained by transferring the knowledge from both Amazon and IMDb, together with its own training set (SST-5). The training/development/testing splits of Amazon and MNLI are the same as in Wang et al. (2020). As IMDb does not contain a separate development set, we randomly sample a proportion of the training set for parameter tuning. We use the lexical split of the Shwartz dataset in the experiments to prevent lexical memorization and make the results more robust. For data splits of other datasets, refer to their original papers. [9]

We implement Meta-DTL and all the baselines on two popular PLMs: BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2020). All the algorithms are implemented with TensorFlow and trained with NVIDIA Tesla V100 GPU (32GB). We use *Accuracy* as the evaluation metric for all the tasks. For better reproductivity, we uniformly set the sequence length as 128 for the first two sets of experiments and 32 for the third, and set the batch size as 32. The learning rate is tuned from $\{5e-5, 1e-5\}$. The numbers of epochs for MMT and TMF are tuned from $1 \sim 3$ and $3 \sim 5$, respectively. λ is set to 0.1 in default. The parameter regularization and the optimizer settings are the same as in Devlin et al. (2019). Since we do not modify the architecture of our final models, models trained by Meta-DTL should have the same size and in-

[8] We use Google's official base models. See: https://github.com/google-research/bert and https://github.com/google-research/albert.

[9] Note that the experimental settings in our work may look similar to Wang et al. (2020). However, the task of the previous work is to transfer knowledge among different sub-domains within the same task, such as different sub-domain datasets in Amazon and MNLI, respectively. Our experimental settings are significantly more challenging as we aim to transfer knowledge across datasets with distant domains, language styles and class labels. The learning gaps between tasks in our work are much larger than Wang et al. (2020).

Name	Task Description	Classification Label Set	#Train	#Dev.	#Test
SST-5	Fine-grained movie review analysis	{1, 2, 3, 4, 5}	8,544	1,101	2,210
Amazon	Coarse-grained product review analysis	{positive, negative}	7,000	500	500
IMDb	Coarse-grained movie review analysis	{positive, negative}	23,785	1,215	25,000
MNLI	NLI across multiple genres	{entailment, neutral, contradiction}	382,702	10,000	9,815
SciTail	Scientific question answering	{entailment, neutral}	23,596	1,304	2,126
Shwartz	Hypernymy detection	{hypernymy, non-hypernymy}	20,335	1,350	6,610
BLESS	Lexical relation classification	{event, meronymy, random, co-hyponymy, attribute, hypernymy}	18,582	1,327	6,637

Table 1: Statistics and data summarization of all the seven public datasets used in the experiments.

PLM	Method	Review Analysis Tasks				NLI Tasks			Lexical Semantic Tasks		
		SST-5	Amazon	IMDb	Avg.	MNLI	SciTail	Avg.	Shwartz	BLESS	Avg.
Bert	Single-task	53.4	89.3	95.2	79.3	83.0	92.4	87.7	92.6	93.2	92.9
	Multi-task	53.2	89.8	95.6	79.5	83.8	92.0	87.9	92.8	93.0	92.9
	Task Comb.	53.2	89.5	94.1	78.9	83.7	92.2	87.9	91.3	91.7	91.5
	Meta-FT*	53.6	91.0	95.8	80.1	83.9	93.4	88.6	92.8	93.5	93.1
	Meta-DTL	**54.6**††	**91.8**††	**98.2**††	**81.5**	**84.2**†	**93.6**††	**88.9**	**93.2**††	**94.8**††	**94.0**
Albert	Single-task	51.0	87.6	93.6	77.4	80.7	88.2	84.4	92.0	90.7	91.3
	Multi-task	50.3	88.1	94.2	77.5	81.0	88.3	84.6	92.4	91.0	91.7
	Task Comb.	49.8	88.0	93.6	77.1	80.8	85.2	83.0	91.4	90.6	91.0
	Meta-FT*	50.8	88.4	95.0	78.0	81.2	88.7	84.9	92.4	91.9	92.1
	Meta-DTL	**51.2**††	**88.8**††	**97.6**††	**79.2**	**82.4**††	**89.2**††	**85.8**	**92.8**†	**93.4**††	**93.1**

Table 2: General performance of Meta-DTL and all the baselines over all the datasets in terms of accuracy. The p-values of the paired t-tests for each dataset are marked as follows: †† : $p < 0.05$ and † : $0.05 < p < 0.1$.

ference speed as BERT (Devlin et al., 2019) or ALBERT (Lan et al., 2020). In the following experiments, we reproduce results for all baselines, and report the accuracy scores of both baselines and our method averaged from three random runs (with different seeds). Hence, the impact of random seeds is minimized.

4.2 General Performance Comparison

In this section, we compare Meta-DTL against previous approaches. The following four methods are considered as strong baselines:

- **Single-task**: Fine-tuning BERT (Devlin et al., 2019) or ALBERT (Lan et al., 2020) on the single-task training set only.
- **Multi-task**: Fine-tuning the PLM on all the tasks by multi-task learning. Each task has its own prediction heads, with the architecture similar to MT-DNN (Liu et al., 2019).
- **Task Combination**: Combining all the training sets and treating them as one task. The label set of this method is: $\bigcup_{i=1,\cdots,K} \mathcal{C}_i$.
- **Meta-FT***: To our knowledge, Meta-FT (Wang et al., 2020) achieves the highest performance on cross-domain transfer learning for PLMs. However, it can not handle tasks with different class label sets. We implement a variant named Meta-FT*, which has a separate prediction head for each task.

The results of Meta-DTL and the baselines on all seven testing sets are shown in Table 2. Generally speaking, the performance gains of Meta-DTL over all three sets of tasks and seven datasets are consistent. With the integration of Meta-DTL, the accuracy of fine-tuned BERT boosts 2.2% for review analysis, 1.2% for NLI and 1.1% for two lexical semantic tasks. A similar conclusion holds for ALBERT. It shows that even the tasks are different in domains and class label sets, Meta-DTL can make PLMs learn from these distant tasks effectively. We also conclude that simple multi-task training does not have significant improvement. Hence, trivial learners may easily suffer from *negative transfer*. For example, the performance on SST-5 drops on both BERT and ALBERT when the model is jointly trained with Amazon and IMDb, since the learning objective of SST-5 is different from those of Amazon and IMDb. Meta-FT* (Wang et al., 2020) is the most competitive approach but is inferior to Meta-DTL due to the lack of modeling the learning process across distant tasks and task-agnostic designs. In summary, Meta-DTL's distant transfer learning ability is hence clearly confirmed.

Task	w/o.IMK	w/o.WMER	Full
SST-5	**54.0**	53.8	54.6
Amazon	90.6	**90.8**	91.8
IMDb	97.0	**97.6**	98.2
MNLI	84.0	**84.1**	84.2
SciTail	**92.9**	92.7	93.6
Shwartz	91.8	**92.2**	93.2
BLESS	93.5	**93.8**	94.8
Avg.	86.4	**86.6**	87.2

Table 3: The ablation study results of Meta-DTL on seven testing sets in terms of accuracy. IMK stands for "injecting meta-knowledge".

Task	$\alpha_{i,j}$	$\beta_{i,j}$	$m_{i,j}$ (w/o. class label)
SST-5	54.0	54.2	54.2
Amazon	91.1	90.6	91.5
IMDb	97.9	97.2	98.1
MNLI	83.8	83.2	84.0
SciTail	92.2	91.0	93.2
Shwartz	90.4	92.2	93.1
BLESS	93.1	93.2	94.8
Avg.	86.0	85.9	86.9

Table 4: Meta-DTL performance with different meta-knowledge on seven testing sets in terms of accuracy.

4.3 Detailed Model Analysis

In this section, we analyze the algorithmic performance of Meta-DTL in various aspects.

Ablation Study. In Meta-DTL, we employ two important techniques to capture transferable knowledge, namely injecting meta-knowledge and applying WMERs. In the ablation study, we disable one technique from our full model each time. We report the results on the testing sets of the seven tasks, with BERT as the underlying PLM, shown in Table 3. The results show that injecting meta-knowledge is slightly more effective than applying WMERs in five out of seven tasks. However, there is no large difference between the two techniques. Therefore, both techniques are proved important for acquiring a good meta-learner.

Analysis of Meta-knowledge. We further analyze how different parts of the meta-knowledge contribute to the overall performance, with results shown in Table 4. Columns entitled $\alpha_{i,j}$ and $\beta_{i,j}$ refer to the adoption of one type of the scores only. $m_{i,j}$ (w/o. class label) is a variant of Meta-DTL without using the class label information for task representation learning. From the results, we can see that both $\alpha_{i,j}$ and $\beta_{i,j}$ are effective for learning the meta-knowledge. Comparing $m_{i,j}$ (w/o. class label) to the full implementation, we can see that injecting class label information is also useful.

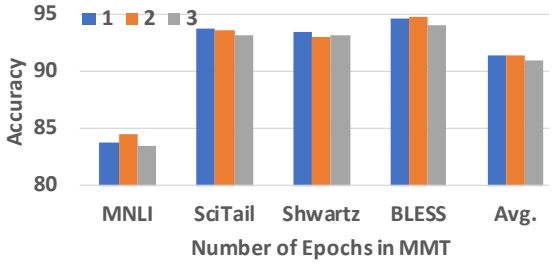

Figure 3: Tuning the number of epochs in MMT (%).

Parameter Analysis. During MMT, the training data in each batch is sampled from different tasks based on $p(\mathcal{T}_i)$. The setting $p(\mathcal{T}_i) = \frac{1}{K}$ leads to the under-sampling of large datasets and over-sampling of small ones. Hence, it is infeasible to compute the number of epochs. In this work, we say MMT finishes one epoch when it runs $\frac{\sum_{i=1,\cdots,K}|\mathcal{D}_i|}{|\mathcal{B}|}$ training steps. We vary the number of epochs in MMT, keep other parameters as default and report the performance of downstream tasks over the development sets. The results on NLI and lexical semantics tasks are shown in Figure 3. The experiments show that too many epochs may hurt the overall performance, forcing the PLM to memorize too much information from non-target tasks. We suggest that one or two epochs of MMT are sufficient for most cases. We also fix the number of epochs in MMT as 2 and tune the hyper-parameter λ from 0 to 0.5, with the results illustrated in Figure 4. We can see that a suitable choice for λ ranges from 0.1 to 0.2. A larger value of λ can inject too much class label noise in the training process, harming the performance. We also tune the hyper-parameters during TMF (i.e, the learning rate and epoch). We find that when we apply Meta-DTL, we generally do not need to change hyper-parameter settings compared to the original fine-tuning approach (Devlin et al., 2019). Due to space limitation, we do not elaborate.

4.4 Learning with Small Data

One advantage of transfer learning across tasks is to reduce the amount of labeled training data for target tasks. As MNLI is the largest dataset among all, we randomly sample only 1%, 2%, 5%, 10% and 20% of the original training data to train

Score	Task	Review Text	Label
High	Amazon	...it is just one big failure and is to be avoided...	negative
	IMDb	I can't believe I waste my time watching this garbage!...	negative
	SST-5	The more you think about the movie, the more you will probably like it.	4 (weakly positive)
	SST-5	No, I hate it.	1 (strongly negative)
Low	Amazon	Racism is not the problem with this book - sure...5 Chinese brothers...	negative
	SST-5	... plays like a badly edited , 91-minute trailer (and) the director can't...	1 (strongly negative)

Table 5: Cases of review texts in Amazon, SST-5 and IMDb with high and low meta-knowledge scores.

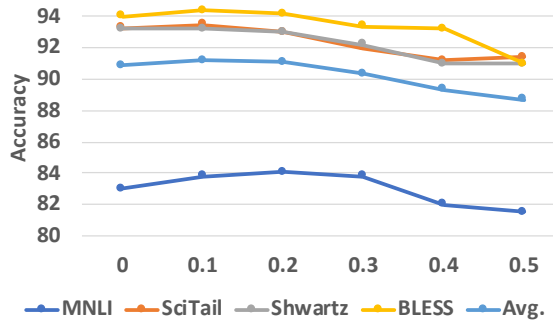

Figure 4: Tuning the hyper-parameter λ in MMT (%).

PCT	Single	Meta-FT*	Meta-DTL
1%	62.5	64.1	**66.5 (+4.0%)**
2%	67.5	68.2	**69.8 (+2.3%)**
5%	72.8	73.8	**74.2 (+1.4%)**
10%	75.8	76.2	**77.6 (+1.8%)**
20%	80.4	80.8	**81.4 (+1.0%)**

Table 6: Prediction accuracy on the testing set when only part of the MNLI training set is employed (%).

the model. The full SciTail training set is used for knowledge transfer. We list the results on the MNLI testing set with and without Meta-DTL training in Table 6. The results produced by Meta-FT* are also compared. As seen, Meta-DTL improves the performance regardless of the percentages of the training sets. It has a larger increase in accuracy on smaller training sets (a 4.0% increase on 1% of the training set vs. a 1.0% increase on 20%).

4.5 Case Studies

We further present some cases for a better understanding of what the meta-knowledge is across tasks. In Table 5, review texts from Amazon, SST-5 and IMDb with high and low $m_{i,j}$ scores are illustrated, together with their class labels. As seen, although there exist some domain and class label differences, our algorithm is able to find review texts that express general polarities and should be transferable across the three tasks. For instance, the expressions with high scores such as "one big failure" and "garbage" give strong indications of their polarities, no matter what the class label set is concerned. In contrast, low-score texts "5 Chinese brothers" and "91-minute trailer" describe specific details about certain subjects, and are not much useful for knowledge transfer. Hence, the learned meta-knowledge is truly insightful.

5 Conclusion and Future Work

In this paper, we propose the Meta-DTL framework for PLMs, to capture knowledge from tasks with distant domains and class labels. Extensive experiments confirm the effectiveness of Meta-DTL from various aspects. Future work includes: i) applying Meta-DTL to other PLMs and NLP tasks, and ii) exploring how it can benefit other NLP models.

Acknowledgement

This work is supported by Open Research Projects of Zhejiang Lab (No. 2019KD0AD01/004), China Postdoctoral Science Foundation (No. 2021M692956), the NSFC projects (No. U19B2042, No. 62072399), MoE Engineering Research Center of Digital Library, and the Fundamental Research Funds for the Central Universities. We thank the anonymous reviewers for their helpful comments.

References

Arman Afrasiyabi, Jean-François Lalonde, and Christian Gagné. 2020. Associative alignment for few-shot image classification. In *ECCV*, pages 18–35.

Zaid Alyafeai, Maged Saeed AlShaibani, and Irfan Ahmad. 2020. A survey on transfer learning in natural language processing. *CoRR*, abs/2007.04239.

Yuki Arase and Jun'ichi Tsujii. 2019. Transfer fine-tuning: A BERT case study. In *EMNLP-IJCNLP*, pages 5392–5403.

Yujia Bao, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2020. Few-shot text classification with distributional signatures. In *ICLR*.

Marco Baroni and Alessandro Lenci. 2011. How we BLESSed distributional semantic evaluation. In *GEMS*, pages 1–10.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, pages 440–447.

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. In *NeurIPS*.

Cen Chen, Yinfei Yang, Jun Zhou, Xiaolong Li, and Forrest Sheng Bao. 2018. Cross-domain review helpfulness prediction based on convolutional neural networks with auxiliary domain discriminators. In *NAACL-HLT*, pages 602–607.

Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. 2019. Meta relational learning for few-shot link prediction in knowledge graphs. In *EMNLP-IJCNLP*, pages 4216–4225.

Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang. 2020. Few-shot NLG with pre-trained language model. In *ACL*, pages 183–190.

Baiyun Cui, Yingming Li, Ming Chen, and Zhongfei Zhang. 2019. Fine-tune BERT with sparse self-attention mechanism. In *EMNLP-IJCNLP*, pages 3546–3551.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL*, pages 2978–2988.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pages 1126–1135.

Chelsea Finn, Kelvin Xu, and Sergey Levine. 2018. Probabilistic model-agnostic meta-learning. In *NeurIPS*, pages 9537–9548.

Yuncheng Hua, Yuan-Fang Li, Gholamreza Haffari, Guilin Qi, and Tongtong Wu. 2020. Few-shot complex knowledge base question answering via meta reinforcement learning. In *EMNLP*, pages 5827–5837.

Muhammad Abdullah Jamal and Guo-Jun Qi. 2019. Task agnostic meta-learning for few-shot learning. In *CVPR*, pages 11719–11727.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *AAAI*, pages 5189–5197.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *ICLR*.

Zhongyang Li, Xiao Ding, and Ting Liu. 2019. Story ending prediction by transferable BERT. In *IJCAI*, pages 1800–1806.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. In *ACL*, pages 1–10.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. In *ACL*, pages 4487–4496.

Jie Lu, Vahid Behbood, Peng Hao, Hua Zuo, Shan Xue, and Guangquan Zhang. 2015. Transfer learning using computational intelligence: A survey. *Knowl. Based Syst.*, 80:14–23.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *ACL*, pages 142–150.

Haojie Pan, Chengyu Wang, Minghui Qiu, Yichang Zhang, Yaliang Li, and Jun Huang. 2021. Meta-kd: A meta knowledge distillation framework for language model compression across domains. In *ACL/IJCNLP*, pages 3026–3036.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*, pages 2227–2237.

Minghui Qiu, Bo Wang, Cen Chen, Xiaoyi Zeng, Jun Huang, Deng Cai, Jingren Zhou, and Forrest Sheng Bao. 2019. Cross-domain attention network with wasserstein regularizers for e-commerce search. In *CIKM*, pages 2509–2515.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained models for natural language processing: A survey. *CoRR*, abs/2003.08271.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *ACL*.

Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical networks for few-shot learning. In *NIPS*, pages 4077–4087.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune BERT for text classification? In *CCL*, pages 194–206.

Ben Tan, Yu Zhang, Sinno Jialin Pan, and Qiang Yang. 2017. Distant domain transfer learning. In *AAAI*, pages 2604–2610.

Joaquin Vanschoren. 2018. Meta-learning: A survey. *CoRR*, abs/1810.03548.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *NeurIPS*, pages 3261–3275.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

Bo Wang, Minghui Qiu, Xisen Wang, Yaliang Li, Yu Gong, Xiaoyi Zeng, Jun Huang, Bo Zheng, Deng Cai, and Jingren Zhou. 2019c. A minimax game for instance based selective transfer learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019*, pages 34–43. ACM.

Chengyu Wang, Haojie Pan, Yuan Liu, Kehan Chen, Minghui Qiu, Wei Zhou, Jun Huang, Haiqing Chen, Wei Lin, and Deng Cai. 2021a. Mell: Large-scale extensible user intent classification for dialogue systems with meta lifelong learning. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3649–3659.

Chengyu Wang, Minghui Qiu, Jun Huang, and Xiaofeng He. 2020. Meta fine-tuning neural language models for multi-domain text mining. In *EMNLP*, pages 3094–3104.

Chengyu Wang, Minghui Qiu, Jun Huang, and Xiaofeng He. 2021b. KEML: A knowledge-enriched meta-learning framework for lexical relation classification. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, pages 13924–13932. AAAI Press.

Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime G. Carbonell. 2019d. Characterizing and avoiding negative transfer. In *CVPR*, pages 11293–11302.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL-HLT*, pages 1112–1122.

Qiao Xiao and Yu Zhang. 2020. Distant transfer learning via deep random walk. *CoRR*, abs/2006.07622.

Yi Yang and Arzoo Katiyar. 2020. Simple and effective few-shot named entity recognition with structured nearest neighbor learning. In *EMNLP*, pages 6365–6375.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, pages 5754–5764.

Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Multi-level matching and aggregation network for few-shot relation classification. In *ACL*, pages 2872–2881.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *NeurIPS*.

Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. 2020. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *CVPR*, pages 12200–12210.

Yiyun Zhao and Steven Bethard. 2020. How does bert's attention change when you fine-tune? an analysis methodology and a case study in negation scope. In *ACL*, pages 4729–4747.

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2019. A comprehensive survey on transfer learning. *CoRR*, abs/1911.02685.