# Simultaneous Selection and Adaptation of Source Data via Four-Level Optimization

**Pengtao Xie[1]*, Xingchen Zhao[2]†, Xuehai He[3]†**
[1]UC San Diego, USA, [2]Northeastern University, USA, [3]UC Santa Cruz, USA
p1xie@ucsd.edu

## Abstract

In many NLP applications, to mitigate data deficiency in a target task, source data is collected to help with target model training. Existing transfer learning methods either select a subset of source examples that are close to the target domain or try to adapt all source examples into the target domain, then use selected or adapted source examples to train the target model. These methods either incur significant information loss or bear the risk that after adaptation, source examples which are originally already in the target domain may be outside the target domain. To address the limitations of these methods, we propose a four-level optimization based framework which simultaneously selects and adapts source data. Our method can automatically identify in-domain and out-of-domain source examples and apply example-specific processing methods: selection for in-domain examples and adaptation for out-of-domain examples. Experiments on various datasets demonstrate the effectiveness of our proposed method.

## 1 Introduction

Transfer learning (TL) (Zhuang et al., 2020), which aims at improving a model in a target domain by utilizing data from a source domain, has been broadly studied in natural language processing. One paradigm of TL methods (Sun et al., 2011; Song et al., 2012; Wang et al., 2017b; Patel et al., 2018; Qu et al., 2018; Liu et al., 2019b) focuses on selecting a subset of source examples that are close to the target domain and using selected examples as additional training data for the target model. Another paradigm of TL methods (Pan et al., 2010; Ganin et al., 2016; Bousmalis et al., 2017) focuses on adapting the entire set of source examples into the target domain and using adapted source examples as additional training data for the target model. The problem of selection based methods is that unselected examples are discarded. Though having a domain discrepancy with target data, unselected examples still contain useful information that can be leveraged to improve the target model. Discarding them would lead to information loss. The problem of adaptation based methods is that some source examples may already be in the target domain; performing domain adaptation on these source examples is a waste of effort; and, even worse, after adaptation, these source examples may be outside the target domain.

To address the limitations of both paradigms of methods, we propose a new approach which simultaneously performs selection and adaptation of source examples: Our method automatically identifies which source examples are in the same domain as target data (referred to as in-domain source data) and which are not (referred to as out-of-domain source data); for in-domain source data, they are directly used to train the target model; for out-of-domain source data, they are first adapted, then utilized to train the target model. Compared with previous methods, our approach has the following advantage: Instead of using a single way to deal with all source examples (either performing selection or adaptation), our method applies example-specific ways to deal with different examples based on whether they are in-domain or out-of-domain.

Our method is based on four-level optimization, which performs the following four stages end-to-end. At the first stage, a domain distance network is trained based on self-supervised learning. At the second stage, the domain distance network is used to identify out-of-domain source examples and adapt them into the target domain. At the third stage, in-domain source examples selected by the domain distance network and adapted source examples are used to train a target model. At the fourth stage, the trained target model is evaluated on a validation set and data weights

*Corresponding author
†Equal contribution.



Transactions of the Association for Computational Linguistics, vol. 12, pp. 449–466, 2024. https://doi.org/10.1162/tacl_a_00658
Action Editor: Tim Baldwin. Submission batch: 10/2022; Revision batch: 8/2023; Published 5/2024.

of the domain distance network are updated by minimizing the validation losses. Experiments on a variety of datasets demonstrate the effectiveness of our proposed method. To summarize, the major contributions of this work is that we propose a four-level optimization based framework for simultaneous selection and adaptation of source examples in transfer learning and demonstrate the effectiveness of our proposed method on two NLP applications.

## 2 Related Works

**Domain Adaptation (DA).** DA (Pan et al., 2010; Ganin et al., 2016; Bousmalis et al., 2017; Sun and Saenko, 2016; Long et al., 2017b; Ben-David et al., 2010; Hoffman et al., 2018; Long et al., 2017a; Kang et al., 2019; Long et al., 2015; Long et al., 2017b; Hoffman et al., 2018; Mitsuzumi et al., 2021) considers the problem of transferring knowledge from a label-rich source domain to a label-deficient target domain where the two domains have distributional discrepancies. There are mainly two paradigms of approaches. One paradigm (Sun and Saenko, 2016; Long et al., 2017b; Ben-David et al., 2010; Kang et al., 2019) is based on metric learning, where a distance metric is defined to measure the distribution discrepancy between domains and domain-invariant representations are learned by minimizing the distance. The other paradigm is based on adversarial learning (Hoffman et al., 2018; Long et al., 2017a; Tzeng et al., 2017; Sankaranarayanan et al., 2018; Motiian et al., 2017), which learns a domain discriminator and a feature learning network adversarially. The domain discriminator is trained to tell whether an instance is from source domain or target domain, while the feature learning network learns domain-invariant features by fooling the domain discriminator. CDAN (Long et al., 2017a) uses multilinear conditioning to capture the cross-covariance between feature representations and classifier predictions, and leverages entropy conditioning to control the uncertainty of classifier predictions. MME (Saito et al., 2019) performs adaptation by maximizing the conditional entropy of unlabeled target data w.r.t the classifier and minimizing it w.r.t the feature encoder. SRDC (Tang et al., 2020) performs deep discriminative clustering with source regularization for unsupervised domain adaptation. These methods perform adaptation on all source data, which leads to waste of efforts and incurs a risk of moving in-domain source data outside the target domain.

**Data Selection in Transfer Learning.** Many methods (Jiang and Zhai, 2007; Foster et al., 2010; Moore and Lewis, 2010; Axelrod et al., 2011; Ge and Yu, 2017; Ruder and Plank, 2017; Sivasankaran et al., 2017; Zhang et al., 2017; Guo et al., 2019; Liu et al., 2019a; Tang and Jia, 2019; Wang et al., 2019a,b; Bateson et al., 2020) have been developed for selecting source data that is suitable for training target models, based on reinforcement learning (Patel et al., 2018; Qu et al., 2018; Liu et al., 2019b), adversarial learning (Wang et al., 2019a), curriculum learning (Zhang et al., 2017; Wang et al., 2019b), entropy (Song et al., 2012; Wang et al., 2017b), Bayesian optimization (Ruder and Plank, 2017), multi-task learning (Ge and Yu, 2017), and bi-level optimization (BLO) (Ren et al., 2018, 2020; Hu et al., 2019; Shu et al., 2019; Wang et al., 2020a,b). BLO based approaches select data by minimizing a validation loss, where the lower-level optimization problem trains network weights on a training dataset and the upper-level optimization problem learns data selection variables on a validation set. These methods select part of source data and discard the rest, which incurs information loss.

**Transfer Learning (TL).** TL (Pratt, 1993; Mihalkova et al., 2007; Niculescu-Mizil and Caruana, 2007; Pan and Yang, 2009; Luo et al., 2017; Zhuang et al., 2020) aims at training a better target model by utilizing source data. Many TL methods have been developed, including those based on 1) distribution alignment (Huang et al., 2006; Foster et al., 2010; Wang et al., 2017b; Ngiam et al., 2018), 2) regularization (Luo et al., 2008; Tommasi et al., 2010; Duan et al., 2012), 3) adversarial domain-invariant representation learning (Ganin et al., 2016; Long et al., 2017a; Hoffman et al., 2018; Zhang et al., 2019), and 4) latent space projection (Borgwardt et al., 2006; Pan et al., 2010; Long et al., 2013; Wang et al., 2017a). These methods either select part of source data or adapt all source data, which leads to information loss, waste of efforts, and the risk of adapting in-domain source data outside the target domain.

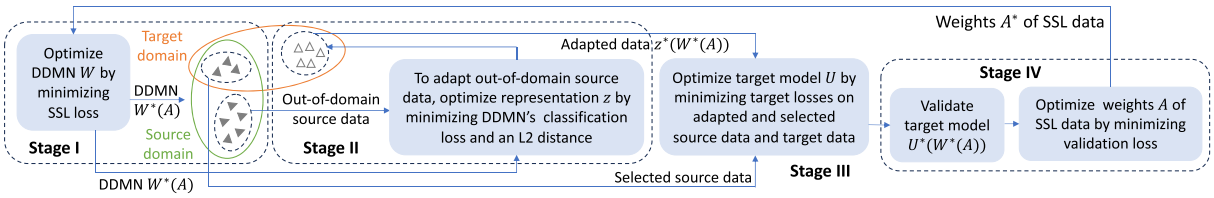


Figure 1: The overall framework.

**Bi-level Optimization (BLO).** BLO finds broad applications in hyperparameter tuning (Feurer et al., 2015), data selection (Shu et al., 2019; Ren et al., 2020; Wang et al., 2020b), training data generation (Such et al., 2019), neural architecture search (Liu et al., 2018), learning rate adaptation (Baydin et al., 2017), meta learning (Finn et al., 2017), etc. In BLO-based methods, model weights are learned by solving an inner optimization problem and meta parameters are learned by solving an outer optimization problem. The two optimization problems are nested. Different from existing BLO-based methods, our method is based on four-level optimization.

## 3 Methods

In this section, we present the method for simultaneous selection and adaptation of source examples based on four-level optimization. We aim to train a target model for a specific target domain using dataset $D_t$. In practical scenarios, the target domain often suffers from a lack of labeled training data. This lack can lead to overfitting on the training data and poor generalization on test data. To address this issue, we utilize a dataset from a source domain, $D_s$, which has an abundance of labeled examples. However, there is a notable discrepancy between the source and target data. We categorize $D_s$ examples into two types: those that belong to the same domain as $D_t$ (in-domain source data) and those that do not (out-of-domain source data). We aim to select in-domain source data and adapt out-of-domain source data into the target domain, and use selected and adapted source data to train the target model. The overall framework is shown in Figure 1. The notations are shown in Table 1.

### 3.1 A Four-Level Optimization Framework

We propose a four-level optimization framework to perform simultaneous selection and adaptation of source data. The framework consists of four learning stages which are performed end-to-end.

| Notation | Meaning |
|---|---|
| $D_t$ | Target dataset. |
| $N_t$ | The number of examples in $D_t$. |
| $D_t^{(tr)}$ | Training dataset of $D_t$. |
| $D_t^{(val)}$ | Validation dataset of $D_t$. |
| $d_{t,i}^{(tr)}$ | The $i$-th example in $D_t^{(tr)}$. |
| $D_s$ | Source dataset. |
| $N_s$ | The number of examples in $D_s$. |
| $d_{s,i}$ | The $i$-th example in $D_s$. |
| $q$ | Query example. |
| $R$ | A set of examples. |
| $o$ | Binary label regarding whether $q$ and $R$ are from the same domain. |
| $M$ | The number of SSL training examples. |
| $W$ | The DDMN's weight parameters. |
| $a$ | The weight of an SSL training example. |
| $A$ | The weights of all SSL training examples. |
| $W^*(A)$ | The optimal solution of $W$, which depends on $A$. |
| $f(q, D_t; W^*(A))$ | A binary label regarding whether $q$ is out of the target domain. |
| $W_1^*(A)$ | The encoder in $W^*(A)$. |
| $W_2^*(A)$ | The rest of layers in $W^*(A)$ besides $W_1^*(A)$. |
| $\hat{z}(q; W_1^*(A))$ | The latent representation of $q$ extracted by $W_1^*(A)$. |
| $z(q)$ | The adapted representation of $q$. |
| $z^*(q, W^*(A))$ | The optimal solution of $z(q)$, which depends on $W^*(A)$. |
| $\gamma$ | A tradeoff parameter in Eq.(2). |
| $\lambda$ | A tradeoff parameter in Eq.(6). |
| $U$ | Target model. |
| $\widehat{U}$ | A sub-network of $U$, which takes $z^*(q, W^*(A))$ as input. |
| $\ell_{tgt}$ | The target model's training loss. |
| $L_t$ | The target model's training loss on target training data. |
| $L_s$ | The target model's training loss on selected source data. |
| $L_a$ | The target model's training loss on adapted source data. |

Table 1: Notations.

**Stage I.** At the first stage, we learn a domain distance metric network via self-supervised learning (SSL) (He et al., 2019; Chen et al., 2020). This network takes a query example $q$ and a set of examples $R$ as inputs and predicts a binary label representing whether $q$ is in the domain of

$R$. The architecture of this network is as follows. For $q$ and each example in $R$, an encoder is used to generate a latent representation for the example. Then self-attention (Vaswani et al., 2017) is performed on these latent representations to generate attentive representations. Finally, the attentive representation of $q$ and the averaged attentive representation of examples in $R$ are concatenated and fed into a feedforward layer to predict the binary label.

To learn this domain distance metric network (DDMN), we construct self-supervised training examples. We randomly sample a subset of examples $R$ from $D_t$, and randomly sample a query example $q_t$ from $D_t$ and a query example $q_s$ from $D_s$. We label $(q_t, R)$ as a positive pair since $q_t$ and $R$ are both from target domain, and label $(q_s, R)$ as a negative pair since $q_s$ is from source domain and $R$ is from the target domain. This procedure repeats $M$ times, yielding $2M$ training examples denoted by $D_{ssl} = \{(q_i, R_i, o_i)\}_{i=1}^{2M}$ where $o_i$ is a binary label representing whether $q_i$ and $R_i$ are from the same domain. Let $W$ denote weight parameters of the DDMN. We learn $W$ by minimizing the binary classification loss: $\sum_{i=1}^{2M} \ell(q_i, R_i, o_i; W)$ where $\ell$ is a cross-entropy loss.

Note that these self-supervised examples may be noisy since the binary labels are given based on heuristics without human scrutiny. It could be the case that $q_s$ happens to be in the same domain as $R$ while $q_t$ is not. It is necessary to automatically identify and remove these noisy examples. To achieve this goal, we associate each example $(q_i, R_i, o_i)$ with a weight $a_i \in [0, 1]$. The smaller $a_i$ is, the more likely that $(q_i, R_i, o_i)$ is noisy. We multiply $a_i$ to the loss $\ell(q_i, R_i, o_i; W)$. If $a_i$ is close to zero, $a_i \ell(q_i, R_i, o_i; W)$ is made close to zero, which effectively removes $(q_i, R_i, o_i)$ from the training set. This stage amounts to solving the following optimization problem:

$$W^*(A) = \text{argmin}_W \sum_{i=1}^{2M} a_i \ell(q_i, R_i, o_i; W), \quad (1)$$

where $A = \{a_i\}_{i=1}^{2M}$. $W^*(A)$ denotes that $W^*$ is a function of $A$. This is because $W^*$ is a function of the loss which is a function of $A$. $A$ is tentatively fixed at this stage and will be updated at a later stage. $A$ cannot be updated at this stage. Otherwise, all the values in $A$ would be zero.

**Stage II.** At the second stage, we use the learned domain distance network to identify out-of-domain source examples and adapt them into the target domain. For each data example $q$ from the source dataset $D_s$, we feed $q$ and the target dataset $D_t$ into $W^*(A)$, which predicts a binary label $f(q, D_t; W^*(A))$. If $f(q, D_t; W^*(A)) = 0$, it means $q$ is out of the target domain, and we use the domain distance network to adapt it into the target domain. The adaptation is performed in the following way. In $W^*(A)$, let $W_1^*(A)$ denote the encoder (containing all layers before self-attention) and $W_2^*(A)$ denote the rest of layers (including self-attention and feedforward layers) used for predicting the binary label. Let $\hat{z}(q; W_1^*(A))$ denote the latent representation of $q$ extracted by $W_1^*(A)$—the layer before self-attention. We learn another representation $z(q)$ of $q$ which falls into the target domain (in the latent space) and is close to $\hat{z}(q; W_1^*(A))$. Closeness is measured using $L_2$ distance. To encourage $z(q)$ to fall into the target domain, we encourage $W_2^*(A)$ to predict that $z(q)$ is in the target domain, i.e., minimizing the binary classification loss $l(z(q), D_t, t_q = 1; W_2^*(A))$. At this stage, we solve the following optimization problem for each source example $q$ that is predicted to be out-of-domain:

$$\begin{aligned} &z^*(q, W^*(A)) \\ &= \text{argmin}_{z(q)} \quad \underbrace{l(z(q), D_t, t_q = 1; W_2^*(A))}_{\text{Encourage } z(q) \text{ to fall into the target domain}} \\ &\qquad + \gamma \quad \underbrace{\|z(q) - \hat{z}(q; W_1^*(A))\|_2^2}_{\text{Encourage } z(q) \text{ to be close to } \hat{z}(q; W_1^*(A))} \end{aligned} \quad (2)$$

where $\gamma$ is a tradeoff parameter.

**Stage III.** At the third stage, we use selected and adapted source examples, together with the target training data $D_t^{(tr)}$, to train the target model $U$. Let $\ell_{tgt}$ denote the loss function of the target task, $N_s$ and $N_t$ denote the number of examples in $D_s$ and $D_t$, and $f(q, D_t; W^*(A))$ denote the binary label predicted by the domain distance network $W^*(A)$ regarding whether a source example $q$ is in the target domain. $d_{t,i}^{(tr)}$ denotes the $i$-th example in $D_t^{(tr)}$, $d_{s,i}$ denotes the $i$-th example in $D_s$. For a source example which is predicted to be in the target domain (where $f(d_{s,i}, D_t; W^*(A)) = 1$), it is directly used to train the target model $U$ by minimizing the loss $\ell_{tgt}(U, d_{s,i})$. For a source

example which is predicted to be out of the target domain (where $f(d_{s,i}, D_t; W^*(A)) = 0$), its adapted representation $z^*(d_{s,i}, W^*(A))$ obtained in the second stage is used to train the target model. Note that since $z^*(d_{s,i}, W^*(A))$ is already a latent representation, only part of weight parameters (denoted by $\widehat{U}$) in the target model $U$ is needed to make prediction on $z^*(d_{s,i}, W^*(A))$. We define the loss on target training data as:

$$L_t(U) = \sum_{i=1}^{N_t} \ell_{tgt}(U, d_{t,i}^{(tr)}), \tag{3}$$

the loss on selected source data as:

$$\begin{aligned} &L_s(U, W^*(A)) \\ &= \textstyle\sum_{i=1}^{N_s} f(d_{s,i}, D_t; W^*(A)) \ell_{tgt}(U, d_{s,i}), \end{aligned} \tag{4}$$

and the loss on adapted data as:

$$\begin{aligned} &L_a(\widehat{U}, W^*(A)) \\ &= \sum_{i=1}^{N_s} (1 - f(d_{s,i}, D_t; W^*(A))) \ell_{tgt}(\widehat{U}, z^*(d_{s,i}, W^*(A))). \end{aligned} \tag{5}$$

At this stage, we solve the following optimization problem:

$$\begin{aligned} &U^*(W^*(A)) \\ &= \text{argmin}_U \; L_t(U) + \lambda \left( L_s(U, W^*(A)) + L_a(\widehat{U}, W^*(A)) \right), \end{aligned} \tag{6}$$

where $\lambda$ is a tradeoff parameter.

**Stage IV.** At the fourth stage, we use the trained target model to make predictions on the validation dataset $D_t^{(val)}$ of the target task. We update the weights $A$ of self-supervised training examples in the first stage by minimizing the validation loss:

$$\min_A \; L(U^*(W^*(A)), D_t^{(val)}). \tag{7}$$

**A Four-level Optimization Based Framework.** Putting all these pieces together, we have the following four-level optimization based framework.

$$\begin{aligned} &\underbrace{\min_A \; L(U^*(W^*(A)), D_t^{(val)})}_{\text{Stage IV}} \\ &s.t. \\ &\underbrace{U^*(W^*(A)) = \text{argmin}_U \; L_t(U) + \lambda \left( L_s(U, W^*(A)) + L_a(\widehat{U}, W^*(A)) \right)}_{\text{Stage III}} \\ &\underbrace{\begin{aligned} &z^*(q, W^*(A)) \\ &= \text{argmin}_{z(q)} \; l(z(q), D_t, t_q = 1; W_2^*(A)) + \gamma \| z(q) - \hat{z}(q; W_1^*(A)) \|_2^2 \end{aligned}}_{\text{Stage II}} \\ &\underbrace{W^*(A) = \text{argmin}_W \sum_{i=1}^{2M} a_i \ell(q_i, R_i, o_i; W)}_{\text{Stage I}} \end{aligned} \tag{8}$$

To make the objective at the third stage differentiable, we approximate $f(d_{s,i}, D_t; W^*(A))$ using the probability calculated by $W^*(A)$ regarding whether $d_{s,i}$ and $D_t$ are in the same domain.

### 3.2 Optimization Algorithm

We leverage a gradient-based method (Liu et al., 2018) to solve the problem in Eq.(8). Convergence of this algorithm has been analyzed (Ghadimi and Wang, 2018; Grazzi et al., 2020; Ji et al., 2021; Liu et al., 2021; Yang et al., 2021). At each level of the optimization problem, the exact value of the optimal solution (on the left-hand side of the equal sign, marked with $*$) is computationally expensive to compute. To address this problem, following Liu et al. (2018), we approximate the optimal solution using a one-step gradient descent update and plug the approximation into the next level of optimization problem. In the sequel, $\frac{\partial \cdot}{\partial \cdot}$ denotes partial derivative. $\frac{d \cdot}{d \cdot}$ denotes an ordinary derivative. Following Liu et al. (2018), we approximate $W^*(A)$ using one-step gradient descent update of $W$:

$$W^*(A) \approx W' = W - \eta_w \nabla_W \sum_{i=1}^{2M} a_i \ell(q_i, R_i, o_i; W). \tag{9}$$

We plug $W^*(A) \approx W'$ into the loss function at the second stage and get an approximated objective. Let $W_2'$ and $W_1'$ denote the approximations of $W_2^*(A)$ and $W_1^*(A)$, respectively. The approximated objective is $O(z(q), W_2', W_1') = l(z(q), D_t, t_q = 1; W_2') + \gamma \| z(q) - \hat{z}(q; W_1') \|_2^2$. We approximate $z^*(q, W^*(A))$ using one-step gradient descent update of $z(q)$ w.r.t the approximated objective:

$$\begin{aligned} &z^*(q, W^*(A)) \approx \\ &z'(q) = z(q) - \eta_z \nabla_{z(q)} O(z(q), W_2', W_1'). \end{aligned} \tag{10}$$

We plug $z^*(q, W^*(A)) \approx z'(q)$ and $W^*(A) \approx W'$ into the objective at the third stage and get an approximated objective. Let $g(d_{s,i}, D_t; W^*(A))$ denote the probability that $d_{s,i}$ and $D_t$ are in the same domain. We approximate $U^*(W^*(A))$ using one-step gradient descent update of $U$ w.r.t the approximated objective:

$$\begin{aligned} &U^*(W^*(A)) \\ &\approx U' = U - \eta_u \nabla_U \left( L_t(U) + \lambda \left( L_s(U, W') + L_a(\widehat{U}, W') \right) \right). \end{aligned} \tag{11}$$

Finally, we plug the approximation $U^*(W^*(A)) \approx U'$ into the validation loss at the fourth stage and update $A$ by minimizing the approximated loss using gradient descent:

$$A \leftarrow A - \eta_a \nabla_A L(U', D_t^{(val)}). \tag{12}$$

**while** *not converged* **do**
  1. Update the approximation $W'$ of $W^*(A)$ using Eq.(9)
  2. Update the approximation $z'(q)$ of $z^*(q, W^*(A))$ using Eq.(10)
  3. Update the approximation $U'$ of $U^*(W^*(A))$ using Eq.(11)
  4. Update $A$ using Eq.(13)

**end**

**Algorithm 1:** Optimization algorithm.

For $\nabla_A L(U', D_t^{(val)})$, it can be computed as:

$$\nabla_A L(U', D_t^{(val)}) = \frac{dW'}{dA}\left(\frac{\partial U'}{\partial W'} + \sum_{i=1}^{N_s} \frac{dz'(d_{s,i})}{dW'} \frac{\partial U'}{\partial z'(d_{s,i})}\right) \nabla_{U'} L(U', D_t^{(val)}), \tag{13}$$

where

$$\frac{\partial U'}{\partial z'(d_{s,i})} = -\eta_u \lambda (1 - g(d_{s,i}, D_t; W')) \nabla^2_{z'(d_{s,i}),U} \ell_{tgt}(\widehat{U}, z'(d_{s,i})), \tag{14}$$

$$\frac{dz'(d_{s,i})}{dW'} = -\eta_z \nabla^2_{W',z(q)} (l(z(q), D_t, t_q = 1; W_2') + \gamma \|z(q) - \hat{z}(q; W_1')\|_2^2), \tag{15}$$

$$\frac{\partial U'}{\partial W'} = -\eta_u \lambda \nabla^2_{W',U} \sum_{i=1}^{N_s} (g(d_{s,i}, D_t; W') \ell_{tgt}(U, d_{s,i}) + (1 - g(d_{s,i}, D_t; W')) \ell_{tgt}(\widehat{U}, z'(d_{s,i}))) \tag{16}$$

$$\frac{dW'}{dA} = -\eta_w \nabla^2_{A,W} \sum_{i=1}^{2M} a_i \ell(q_i, R_i, o_i; W) \tag{17}$$

The gradient descent update of $A$ in Eq.(13) can run one or more steps. After $A$ is updated, the one-step gradient-descent approximations in Eq.(9), (10), and (11), which are functions of $A$, change with $A$ and need to be re-updated. Then, the gradient of $A$, which is a function of one-step gradient-descent approximations, needs to be re-calculated and is used to refresh $A$. In sum, the update of $A$ and the updates of one-step gradient-descent approximations mutually depend on each other. These updates are performed iteratively until convergence. Algorithm 1 shows the algorithm.

In the gradient of $A$ calculated using the chain rule, the number of chains is the same as the number of levels in our proposed four-level optimization formulation. This shows that this optimization algorithm preserves the four-level nested optimization nature of the proposed formulation.

### 3.3 Reduce Computation and Memory Costs

To reduce computation and memory costs, we adopt the following methods.

- We reduced the frequencies of updating (including calculating hypergradients of) the weights $A$ of self-supervised training examples. They were updated every 8 mini-batches (i.e., iterations) instead of on every mini-batch. We empirically found that this greatly reduced computational costs without significantly sacrificing accuracy. The rest of the parameters were updated on every mini-batch.
- We added a decorrelation regularizer (Cogswell et al., 2015) on $W$ and $U$, which significantly speeds up convergence and allows reducing the number of epochs by half without sacrificing convergence quality.
- Parameter tying was performed to reduce the number of weight parameters and computation costs. We let $W$ and $U$ share the same feature learning layers. These layers account for >95% of parameters in each of these models. Sharing them across models significantly reduces the number of total parameters, which reduces the computational costs of updating these parameters.
- We optimized the implementation of our method to speed up computation by leveraging techniques including 1) automatic mixed precision (Micikevicius et al., 2017), 2) using multiple (4, specifically) workers and pinned memory in PyTorch DataLoader, 3) using cudNN autotuner, 4) kernel fusion, and so forth.

### 3.4 Applications

In this section, we apply the proposed four-level optimization framework for two NLP applications.

**Text Classification.** In many text classification problems, training data in a target domain is limited. To address the lack of target training data, one can leverage data from a source domain.

**Visual Question Answering on Pathology Images.** Pathology imaging (Mohan, 2015) is broadly used for identifying the causes and effects of diseases or injuries. Given a pathology image, being able to answer questions about the clinical findings contained in the image is very important

| Domain | Dataset | Label Type | Train | Dev | Test | Classes |
|---|---|---|---|---|---|---|
| Biomedical | ChemProt | relation classification | 4169 | 2427 | 3469 | 13 |
| | RCT | abstract sent. roles | 180040 | 30212 | 30135 | 5 |
| Computer Science | ACL-ARC | citation intent | 1688 | 114 | 139 | 6 |
| | SciERC | relation classification | 3219 | 455 | 974 | 7 |
| News | HyperPartisan | partisanship | 515 | 65 | 65 | 2 |
| | AGNews | topic | 115000 | 5000 | 7600 | 4 |
| Reviews | Helpfulness | review helpfulness | 115251 | 5000 | 25000 | 2 |
| | IMDB | review sentiment | 20000 | 5000 | 25000 | 2 |

Table 2: Statistics of datasets used in Gururangan et al. (2020).

for medical decision-making (He et al., 2020). However, collecting a large-scale visual question answering (VQA) dataset is challenging, due to the lack of doctors for making questions and answers from pathology images. A dataset collected in He et al. (2020) has about 33K question-answer pairs generated from around 5K pathology images. Although largest of its kind, it is still relatively small compared with common VQA datasets. To mitigate the deficiency of training data, we collect an auxiliary source dataset. From the pathology literature, we collect 1792 pathology figures and create 36,471 VQA questions using the method proposed in He et al. (2020).

## 4 Experiments

In this section, we present experimental results on text classification and visual question answering on pathology images. Following the common data assumption in transfer learning, the amount of labeled source data in our experiments is significantly larger than that of target data. Every experiment runs 4 times with different random initializations. For all experiments, we performed significance tests using double-sided t-tests. The p-values of our methods against baselines are all less than 0.001, which shows that our methods are significantly better than baselines.

### 4.1 Text Classification

**Dataset.** Following Gururangan et al. (2020), we experiment with four domains: biomedical, computer science, news, and reviews. For the biomedical domain, we use two target datasets: ChemProt (Kringelum et al., 2016) and RCT (Dernoncourt and Lee, 2017), and one source dataset which contains 2.68 million full-text papers from S2ORC (Lo et al., 2019) with 7.55 billion tokens. For the computer science domain, we use two target datasets: ACL-ARC (Jurgens et al., 2018) and SciERC (Luan et al., 2018), and one source dataset which contains 2.22 millon full-text papers from S2ORC (Lo et al., 2019) with 8.1 billion tokens. For the news domain, we use two target datasets: HyperPartisan (Kiesel et al., 2019) and AGNews (Zhang et al., 2015), and one source dataset which contains 11.9 million articles from RealNews (Zellers et al., 2019) with 6.66 billion tokens. For the reviews domain, we use two target datasets: Helpfulness (McAuley et al., 2015) and IMDB (Maas et al., 2011), and one source dataset which contains 24.75 million articles from Amazon Reviews (He and McAuley, 2016) with 2.11 billion tokens. Statistics of the target datasets are summarized in Table 2. In our method, we split the original target training set into a new training set and a validation set, with a ratio of 1:1. The new training set is used as $D_{cls}^{(tr)}$ and the validation set is used as $D_{cls}^{(val)}$. Note that baseline methods are trained on the combination of $D_{cls}^{(tr)}$ and $D_{cls}^{(val)}$.

**Baselines.** We compare our method with the following baselines. In baseline methods, the target data used for training the target model includes both training and validation sets.

- Domain adaptive pretraining (DAPT) (Gururangan et al., 2020): Given an RoBERTa-base model which has been pretrained on large amounts of corpora (used in Liu et al., 2019c), we continue to pretrain it on source data, then finetune it on target data.
- Task adaptive pretraining (TAPT) (Gururangan et al., 2020): Given an RoBERTa-base model which has been pretrained on large amounts of corpora (used

in Liu et al., 2019c), we continue to pretrain it on input texts of each target dataset.

- Data selection methods for transfer learning, based on Bayesian optimization (BO) (Ruder and Plank, 2017), minimax game (MMG) (Wang et al., 2019a), learning to select instance (LSI) (Huan et al., 2021).
- Domain adaptation methods, including DANN (Ganin et al., 2016), CDAN (Long et al., 2017a), MME (Saito et al., 2019), SRDC (Tang et al., 2020), SSDA (Kim and Kim, 2020), GDA (Mitsuzumi et al., 2021), and ATDOC (Liang et al., 2021).
- SimCSE (Gao et al., 2021): A contrastive learning method. The same input sentence is fed into a pretrained RoBERTa encoder twice by applying different dropout masks, to get two different embeddings. These two embeddings are labeled as being ''similar''. Embeddings of different sentences are labeled as being ''dissimilar''. Contrastive learning is performed on these ''similar'' and ''dissimilar'' pairs.

**Implementation Details.** In the domain distance network, the hyperparameters of self-attention and feed-forward layer are the same as those in Transformer (Vaswani et al., 2017). The subset $R$ has varying cardinality (sampled uniformly). $M$ is set to 10k. The tradeoff parameter $\gamma$ and $\lambda$ is set to 0.1 and 0.5 respectively. Baselines and our method receive a similar amount of tuning time and efforts. F1 is used as the evaluation metric. We use RoBERTa-base as a data encoder. For a fair comparison, most of our hyperparameters are the same as those in Gururangan et al. (2020). The maximum text length was set to 512. For all datasets, we used a batch size of 16 with gradient accumulation. We used the AdamW optimizer (Loshchilov and Hutter, 2017) with a warm-up proportion of 0.06, a weight decay of 0.1, and an epsilon of 1e-6. In AdamW, $\beta_1$ and $\beta_2$ are set to 0.9 and 0.98, respectively. The maximum learning rate was 2e-5. For the reader's convenience, we summarize the hyperparameters as follows.

- The number $M$ of self-supervised training examples: 10k
- Tradeoff parameters $\gamma$ and $\lambda$: 0.1, 0.5
- Maximum text length: 512
- Batch size: 16
- Optimizer: AdamW
- Warm-up proportion, weight decay, and epsilon in AdamW: 0.06, 0.1, and 1e-6
- $\beta_1$ and $\beta_2$ in AdamW: 0.9, 0.98
- Maximum learning rate: 2e-5

To tune the hyperparameters, we randomly split the validation set into two equal-sized subsets, denoted by $A$ and $B$. For each configuration of hyperparameters, we use the validation set $A$ to learn the importance weights of self-supervised training examples. Then we measure the performance of the trained model on validation set $B$. Hyperparameter values yielding the best performance on validation set $B$ are selected. Each baseline method received an equal amount of tuning effort as that for our method.

**Results and Analysis.** Table 3 shows the results. From this table, we make the following observations. **First**, our method outperforms source data selection methods including BO, MMG, and LSI. In these baseline methods, out-of-domain source examples are discarded, which incurs information loss. In contrast, our method adapts out-of-domain source examples into the target domain and uses the adapted examples to train the target model. **Second**, our method works better than domain adaptation methods including DANN, CDAN, MME, SRDC, SSDA, GDA, and ATDOC. The reason is: These methods perform adaptation on all source examples without identifying which ones are already in the target domain; as a result, some in-domain source examples may be adapted out of the target domain. In contrast, our method first identifies which source examples are already in-domain and only performs adaptation on out-of-domain examples. **Third**, our method outperforms vanilla RoBERTa-base which does not leverage source data to learn representations. This demonstrates that leveraging source data is helpful for improving the target model. **Fourth**, our method performs better than DAPT. In DAPT, all source examples are leveraged to pretrain the target encoder without considering the fact that some source examples have large domain discrepancy with the target domain and are not suitable for pretraining the target encoder. **Fifth**, our method outperforms TAPT and SimCSE. These methods do not leverage auxiliary source data or select/adapt source data.

| Method | ChemProt | RCT | ACL-ARC | SciERC | HyperPartisan | AGNews | Helpfulness | IMDB | Average |
|---|---|---|---|---|---|---|---|---|---|
| RoBERTa-base | $81.9_{1.0}$ | $87.2_{0.1}$ | $63.0_{5.8}$ | $77.3_{1.9}$ | $86.6_{0.9}$ | $93.9_{0.2}$ | $65.1_{3.4}$ | $95.0_{0.2}$ | 81.3 |
| TAPT | $82.6_{0.4}$ | $87.7_{0.1}$ | $67.4_{1.8}$ | $79.3_{1.5}$ | $90.4_{5.2}$ | $94.5_{0.1}$ | $68.5_{1.9}$ | $95.5_{0.1}$ | 83.2 |
| DAPT | $84.2_{0.2}$ | $87.6_{0.1}$ | $75.4_{2.5}$ | $80.8_{1.5}$ | $88.2_{5.9}$ | $93.9_{0.2}$ | $66.5_{1.4}$ | $95.4_{0.1}$ | 84.0 |
| SimCSE | $83.2_{0.2}$ | $87.6_{0.1}$ | $69.5_{2.6}$ | $80.5_{0.7}$ | $90.9_{2.7}$ | $94.7_{0.1}$ | $68.7_{1.7}$ | $95.7_{0.1}$ | 83.9 |
| BO | $83.3_{0.2}$ | $87.5_{1.0}$ | $75.4_{3.3}$ | $79.1_{1.3}$ | $88.1_{4.2}$ | $94.0_{0.2}$ | $68.4_{2.1}$ | $95.3_{0.2}$ | 83.9 |
| MMG | $83.0_{0.4}$ | $87.4_{1.0}$ | $75.1_{4.1}$ | $79.6_{0.8}$ | $87.9_{2.1}$ | $94.2_{0.1}$ | $66.7_{1.5}$ | $95.7_{0.1}$ | 83.7 |
| LSI | $83.2_{0.3}$ | $87.5_{1.0}$ | $75.3_{4.5}$ | $80.2_{0.6}$ | $88.7_{1.9}$ | $94.5_{0.1}$ | $68.6_{1.0}$ | $95.4_{0.1}$ | 84.2 |
| DANN | $83.5_{0.3}$ | $87.7_{0.1}$ | $75.5_{3.7}$ | $80.5_{1.1}$ | $90.4_{3.0}$ | $94.3_{0.1}$ | $66.8_{2.8}$ | $95.2_{0.1}$ | 84.2 |
| CDAN | $83.8_{0.2}$ | $87.4_{0.1}$ | $75.9_{2.4}$ | $80.9_{1.4}$ | $88.7_{5.5}$ | $94.1_{0.2}$ | $67.3_{3.4}$ | $95.8_{0.2}$ | 84.2 |
| MME | $84.0_{0.1}$ | $87.4_{0.1}$ | $75.7_{5.1}$ | $80.5_{0.8}$ | $89.2_{2.8}$ | $94.6_{0.2}$ | $68.9_{2.7}$ | $95.4_{0.1}$ | 84.5 |
| SRDC | $84.3_{0.3}$ | $87.3_{0.1}$ | $75.7_{3.4}$ | $80.7_{1.0}$ | $88.9_{3.5}$ | $94.1_{0.1}$ | $67.6_{3.0}$ | $95.1_{0.1}$ | 84.2 |
| SSDA | $83.9_{0.5}$ | $87.9_{0.1}$ | $75.9_{2.7}$ | $81.0_{1.4}$ | $88.5_{2.6}$ | $94.4_{0.1}$ | $67.0_{1.6}$ | $95.7_{0.2}$ | 84.3 |
| GDA | $84.1_{0.3}$ | $87.3_{0.1}$ | $75.4_{2.4}$ | $80.8_{0.9}$ | $90.5_{4.1}$ | $94.2_{0.2}$ | $66.8_{2.4}$ | $95.5_{0.1}$ | 84.3 |
| ATDOC | $84.5_{0.2}$ | $87.5_{0.1}$ | $75.6_{3.6}$ | $81.1_{1.2}$ | $89.0_{5.7}$ | $93.9_{0.2}$ | $67.4_{2.8}$ | $95.1_{0.1}$ | 84.3 |
| No-Adapt | $84.1_{0.2}$ | $87.3_{0.1}$ | $75.9_{2.2}$ | $81.4_{1.0}$ | $90.3_{4.2}$ | $94.1_{0.1}$ | $66.9_{3.3}$ | $95.3_{0.2}$ | 84.4 |
| No-In-Domain | $84.5_{0.1}$ | $87.5_{0.1}$ | $75.5_{4.5}$ | $81.6_{1.7}$ | $88.8_{3.9}$ | $94.6_{0.2}$ | $67.1_{3.1}$ | $95.6_{0.1}$ | 84.4 |
| Separate | $85.3_{0.5}$ | $88.1_{0.1}$ | $75.8_{2.9}$ | $82.7_{0.9}$ | $90.1_{3.6}$ | $94.0_{0.2}$ | $68.3_{1.5}$ | $95.2_{0.1}$ | 84.9 |
| MMD | $85.9_{0.4}$ | $88.3_{0.1}$ | $75.5_{4.9}$ | $82.3_{1.1}$ | $89.5_{1.8}$ | $94.1_{0.1}$ | $68.9_{1.2}$ | $95.4_{0.2}$ | 85.0 |
| AD | $85.6_{0.2}$ | $88.7_{0.1}$ | $75.9_{3.1}$ | $82.6_{0.8}$ | $90.3_{3.0}$ | $94.7_{0.2}$ | $67.4_{1.7}$ | $95.8_{0.1}$ | 85.1 |
| WMVL | $85.5_{0.4}$ | $88.5_{0.1}$ | $75.7_{2.8}$ | $81.9_{1.3}$ | $90.6_{4.6}$ | $94.6_{0.1}$ | $68.1_{2.3}$ | $95.1_{0.1}$ | 85.0 |
| H-Divergence | $85.4_{0.3}$ | $88.7_{0.1}$ | $75.9_{4.6}$ | $82.1_{0.9}$ | $89.2_{1.9}$ | $94.2_{0.1}$ | $68.5_{2.6}$ | $95.3_{0.1}$ | 84.9 |
| Our full method | $\mathbf{87.1}_{0.2}$ | $\mathbf{90.4}_{0.1}$ | $\mathbf{77.6}_{2.3}$ | $\mathbf{84.4}_{0.7}$ | $\mathbf{92.4}_{1.2}$ | $\mathbf{95.7}_{0.1}$ | $\mathbf{70.9}_{0.8}$ | $\mathbf{96.6}_{0.1}$ | **86.9** |

Table 3: Text classification results. Following Gururangan et al. (2020), the results are micro-F1 for ChemProt and RCT, and macro-F1 for other datasets. For each $x_y$ entry, $x$ and $y$ represent the mean and standard deviation of four random runs, respectively.

### 4.2 Visual Question Answering on Pathology Images

**Datasets.** For the target dataset, we use PathVQA (He et al., 2020), which contains 1,670 pathology images and 32,795 question-answer pairs. Of these, 16,466 questions are open-ended, with the following types: what, where, when, whose, how, why, how much/how many. The rest are close-ended ''yes/no'' questions. Based on images, the dataset is split into a training, validation, and test set with a ratio of 3:1:1 approximately. Note that baseline methods are trained on the combination of the training and validation sets. We collected a source dataset containing 1792 pathology figures extracted from papers in medRxiv where each figure has a caption. We create 36,471 VQA questions using the method proposed in He et al. (2020). This source dataset will be made available publicly.

**Implementation Details.** For the target model, we experimented with two state-of-the-art VQA models—LXMERT (Tan and Bansal, 2019) and bilinear attention networks (BAN) (Kim et al., 2018)—each containing an image encoder, a question encoder, and an answer generation head. Hyperparameters mostly follow those in previous work (Tan and Bansal, 2019; Kim et al., 2018; Yang et al., 2016). For LXMERT, the hidden size of the text encoder was set to 768.

The initial learning rate was set to 5e-5 with the Adam (Kingma and Ba, 2014) optimizer used. The batch size was set to 256. The model was trained for 200 epochs. For BAN, words in questions and answers were represented using GloVe (Pennington et al., 2014) vectors. The initial learning rate was set to 0.005 with the Adamax optimizer (Kingma and Ba, 2014) used. The batch size was set to 512. The model was trained for 200 epochs.

From questions and answers in the PathVQA dataset, we create a vocabulary of 4,631 words that have the highest frequencies. Data augmentation is applied to images, including shifting, scaling, and shearing. We compare with baselines similar to those in Section 4.1. The Pretrain baseline works as follows: We pretrain the target encoder using source data, then finetune the target encoder using target data. Three metrics were used for evaluation, including BLEU (Papineni et al., 2002), macro-averaged F1 (Goutte and Gaussier, 2005), and accuracy (Malinowski and Fritz, 2014). We implement the methods using PyTorch and perform training on four GTX 1080Ti GPUs.

**Results and Analysis.** The results are shown in Table 4. From this table, we make similar observations as those in Table 3. The analysis of reasons is similar to that for results in Table 3. The

| | Accuracy(%) | BLEU-1(%) | BLEU-2(%) | BLEU-3(%) | F1(%) | Average | Runtime(h) |
|---|---|---|---|---|---|---|---|
| *LXMERT (Tan and Bansal, 2019) based experiments* | | | | | | | |
| Vanilla LXMERT (Tan and Bansal, 2019) | 57.6 | 57.4 | 3.1 | 1.3 | 9.9 | 25.9 | 29 |
| Pretrain (He et al., 2016) | 59.3 | 59.0 | 4.6 | 2.6 | 11.3 | 27.4 | 35 |
| BO (Ruder and Plank, 2017) | 59.5 | 58.9 | 3.8 | 2.7 | 11.4 | 27.3 | 41 |
| MMG (Wang et al., 2019a) | 59.3 | 58.4 | 3.7 | 2.6 | 10.9 | 27.0 | 38 |
| LSI (Huan et al., 2021) | 59.2 | 58.8 | 3.5 | 2.8 | 10.6 | 27.0 | 32 |
| DANN (Ganin et al., 2016) | 59.9 | 59.4 | 4.3 | 3.2 | 11.2 | 27.6 | 30 |
| CDAN (Long et al., 2017a) | 60.2 | 58.9 | 4.7 | 3.3 | 11.9 | 27.8 | 35 |
| MME (Saito et al., 2019) | 59.7 | 59.0 | 3.6 | 2.9 | 11.7 | 27.4 | 30 |
| SRDC (Tang et al., 2020) | 58.8 | 58.7 | 4.8 | 2.9 | 12.2 | 27.5 | 39 |
| SSDA (Kim and Kim, 2020) | 59.3 | 58.3 | 4.4 | 3.0 | 12.0 | 27.4 | 34 |
| GDA (Mitsuzumi et al., 2021) | 59.8 | 59.1 | 4.6 | 2.8 | 12.3 | 27.7 | 36 |
| ATDOC (Liang et al., 2021) | 59.6 | 59.6 | 4.3 | 3.1 | 12.2 | 27.8 | 30 |
| Ours | **62.5** | **61.1** | **5.2** | **3.7** | **12.9** | **29.1** | 29 |
| *BAN (Kim et al., 2018) based experiments* | | | | | | | |
| Vanilla BAN (Kim et al., 2018) | 55.1 | 56.2 | 3.2 | 1.2 | 8.4 | 24.8 | 25 |
| Pretrain (He et al., 2016) | 58.4 | 58.6 | 4.3 | 1.6 | 10.3 | 26.6 | 28 |
| BO (Ruder and Plank, 2017) | 58.3 | 58.5 | 4.2 | 2.0 | 10.9 | 26.8 | 32 |
| MMG (Wang et al., 2019a) | 58.7 | 58.1 | 4.6 | 2.3 | 11.2 | 27.0 | 29 |
| LSI (Huan et al., 2021) | 58.3 | 58.7 | 4.3 | 2.1 | 11.5 | 27.0 | 30 |
| DANN (Ganin et al., 2016) | 58.8 | 58.4 | 4.3 | 2.3 | 11.0 | 27.0 | 31 |
| CDAN (Long et al., 2017a) | 58.6 | 58.6 | 4.5 | 2.5 | 11.4 | 27.1 | 33 |
| MME (Saito et al., 2019) | 59.1 | 58.9 | 4.9 | 2.3 | 11.1 | 27.3 | 26 |
| SRDC (Tang et al., 2020) | 58.9 | 58.7 | 4.7 | 2.5 | 11.6 | 27.3 | 35 |
| SSDA (Kim and Kim, 2020) | 59.3 | 59.0 | 4.4 | 2.6 | 11.4 | 27.3 | 32 |
| GDA (Mitsuzumi et al., 2021) | 59.5 | 58.8 | 4.9 | 2.4 | 11.7 | 27.5 | 30 |
| ATDOC (Liang et al., 2021) | 59.1 | 59.1 | 5.0 | 2.7 | 11.5 | 27.5 | 29 |
| Ours | **62.4** | **61.7** | **5.6** | **3.2** | **12.5** | **29.1** | 26 |

Table 4: Results on the PathVQA dataset. Runtime (hours) for training is measured on a 1080TI GPU.

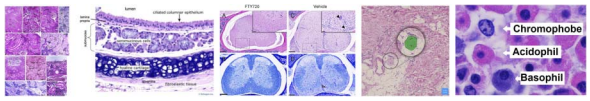


Figure 2: Randomly sampled source pathology figures identified by our method as being out of the target domain.

training time of our method is similar to that of baselines. Figure 2 shows some randomly sampled source pathology figures identified by our method as being out of the target domain. These images contain subfigures and texts, which are different from the target dataset.

### 4.3 Ablation Studies

#### 4.3.1 Ablation by Removing Certain Components

To better understand the contributions of individual components in our framework, we perform the following ablation studies.

- **No-Adapt**. Out-of-domain source examples are discarded instead of being adapted. This is equivalent to removing stage II and the loss term $\mathbb{I}(f(d_{s,i}, D_t; W^*(A)) = 0)\ell_{tgt}(\widehat{U}, z^*(d_{s,i}, W^*(A)))$ in Eq.(6) of stage III.
- **No-In-Domain**. In-domain source examples are discarded instead of being used for training the target model. This is equivalent to removing the loss term $\mathbb{I}(f(d_{s,i}, D_t; W^*(A)) = 1)\ell_{tgt}(U, d_{s,i})$ in Eq.(6) of stage III.
- **Separate**. Different stages are performed separately instead of jointly.

Table 3 shows ablation study results. From this table, we make the following observations. **First**, our full method works better than No-Adapt. This shows that it is beneficial to adapt out-of-domain source examples into the target domain and use adapted examples to train the target model, and our method is effective in achieving this goal. **Second**, our full method outperforms No-In-Domain, which demonstrates that the source examples selected by our method are useful for training the target model. **Third**, our full method achieves better performance than Separate. Our full method performs source data selection, adaptation, and target model training jointly, which enables these different tasks to mutually influence each other

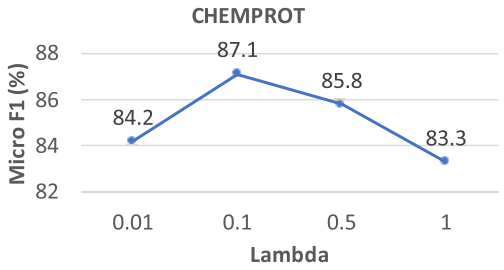


Figure 3: How the micro F1 on CHEMPROT changes with $\lambda$.

to achieve the best overall performance. Such a mechanism is lacking in Separate.

#### 4.3.2 Ablation on the Adaptation Component

We perform an ablation study of the adaptation component in stage II of our framework by replacing it with the following baselines.

- Maximum mean discrepancy (MMD) (Kang et al., 2019), which is a broadly used metric for measuring discrepancies of two distributions. MMD-based adaptation method learns latent representations so that the selected out-of-domain source examples and the target examples have small MMD in the latent space.
- Adversarial adaptation (AD) (Ganin et al., 2016), which learns latent representations so that a domain discriminator cannot tell whether an example is from the source or from the target.

Table 3 shows the results. As can be seen, our adaptation method works better than the baselines. The reason is: The domain distance metric in our method is learned using many self-supervised training examples; it can better measure domain discrepancy and facilitate domain adaptation.

#### 4.3.3 Ablation on the Selection Mechanism

We perform an ablation study of the selection mechanism in our framework by replacing it with the following baselines.

- Each source example is associated with a weight in $[0, 1]$. A larger weight indicates the example is more likely to be in the target domain. We learn these data weights by minimizing the validation loss (WMVL) of the target model.
- We use an $\mathcal{H}$-divergence (Elsahar and Gallé, 2019) based metric to measure domain similarity between a source example and the target dataset.

Table 3 shows the results. As can be seen, our selection mechanism works better than the baselines. The reason is: In our framework, the domain distance network (DSN) is learned in a discriminative way by performing classification on self-supervised training examples. Discriminative training can enable the DSN to better distinguish in-domain source examples from out-of-domain ones. In contrast, the selection mechanisms in the two baselines lack discriminability.

#### 4.3.4 Ablation on the Tradeoff Parameter $\lambda$

We investigate how the performance of our method is affected by the tradeoff parameter $\lambda$. Figure 3 shows the test accuracy on ImageCLEF. As can be seen, when $\lambda$ increases from 0.01 to 0.1, the accuracy improves. This is because the selected and adapted source data is used as additional training resources for the target model. However, as $\lambda$ further increases, the accuracy drops. This is because the source data is not as reliable as the target data. An excessively large $\lambda$ renders too much emphasis to be put on less-reliable source data.

#### 4.3.5 Ablation on the Train-Validate Ratio

In the next ablation study, we investigate how the performance of our method varies under different split ratios between target training and validation datasets. The study was performed on the text classification task. Table 5 shows the results. As can be seen, a more balanced ratio (e.g., 1:1) yields better performance. When the ratio is largely imbalanced (e.g., 1:9, 1:4, 1:0.1, 1:0.25), the performance is worse. If the target training dataset $D_t^{(tr)}$ is much smaller than the target validation dataset $D_t^{(val)}$, the target model's weight parameters $U$, which are trained on $D_t^{(tr)}$, will not be sufficiently trained due to the lack of training data and thereby yield poorer performance. On the other hand, if $D_t^{(val)}$ is much smaller than $D_t^{(tr)}$, the weights $A$ of SSL training examples, which are optimized by minimizing the loss on $D_t^{(val)}$, will not be sufficiently optimized due to the lack of data and thereby yield worse performance as well. Note that since $D_t^{(val)}$ and $D_t^{(tr)}$ are obtained

| Train-val ratio | CHEMPROT | RCT | ACL-ARC | SCIERC | HYPERPARTISAN | AGNEWS | HELPFULNESS | IMDB | Average |
|---|---|---|---|---|---|---|---|---|---|
| 1:9 | $79.4_{0.4}$ | $86.9_{0.1}$ | $73.7_{2.7}$ | $78.5_{1.3}$ | $87.2_{2.3}$ | $93.0_{0.2}$ | $65.5_{3.0}$ | $93.3_{0.1}$ | 82.2 |
| 1:4 | $82.0_{0.4}$ | $87.8_{0.1}$ | $75.1_{3.9}$ | $81.0_{1.5}$ | $89.6_{3.0}$ | $94.1_{0.1}$ | $68.2_{2.7}$ | $94.1_{0.2}$ | 84.0 |
| 1:2 | $84.2_{0.1}$ | $89.1_{0.1}$ | $76.4_{5.0}$ | $83.2_{0.9}$ | $91.7_{4.2}$ | $94.5_{0.1}$ | $70.0_{0.3}$ | $95.9_{0.2}$ | 85.6 |
| 1:1 | $\mathbf{87.1}_{0.2}$ | $90.4_{0.1}$ | $\mathbf{77.6}_{2.3}$ | $\mathbf{84.4}_{0.7}$ | $92.4_{1.2}$ | $\mathbf{95.7}_{0.1}$ | $\mathbf{70.9}_{0.8}$ | $\mathbf{96.6}_{0.1}$ | **86.9** |
| 1:0.5 | $86.8_{0.4}$ | $\mathbf{90.6}_{0.1}$ | $77.1_{2.9}$ | $84.0_{1.2}$ | $\mathbf{92.8}_{1.7}$ | $95.6_{0.1}$ | $70.4_{1.1}$ | $96.1_{0.1}$ | 86.7 |
| 1:0.25 | $86.4_{0.2}$ | $89.9_{0.1}$ | $76.9_{4.2}$ | $83.8_{1.6}$ | $91.9_{1.9}$ | $95.2_{0.2}$ | $69.9_{1.6}$ | $95.3_{0.2}$ | 86.2 |
| 1:0.1 | $85.3_{0.5}$ | $88.5_{0.1}$ | $76.3_{2.8}$ | $82.6_{0.9}$ | $91.7_{2.2}$ | $94.9_{0.1}$ | $69.4_{0.8}$ | $95.0_{0.1}$ | 85.5 |

Table 5: Text classification results of our method under different split ratios between target training and validation sets.

| Method | CHEMPROT | RCT | ACL-ARC | SCIERC | HYPERPARTISAN | AGNEWS | HELPFULNESS | IMDB | Average |
|---|---|---|---|---|---|---|---|---|---|
| BLO | $85.3_{0.4}$ | $88.7_{0.1}$ | $75.4_{2.9}$ | $83.1_{1.1}$ | $90.7_{4.5}$ | $94.9_{0.2}$ | $68.3_{2.2}$ | $95.0_{0.1}$ | 85.2 |
| Ours | $\mathbf{87.1}_{0.2}$ | $\mathbf{90.4}_{0.1}$ | $\mathbf{77.6}_{2.3}$ | $\mathbf{84.4}_{0.7}$ | $\mathbf{92.4}_{1.2}$ | $\mathbf{95.7}_{0.1}$ | $\mathbf{70.9}_{0.8}$ | $\mathbf{96.6}_{0.1}$ | **86.9** |

Table 6: Ablation study results of the BLO method on text classification.

by splitting the original target training dataset, it is always possible to obtain a balanced split.

#### 4.3.6 A Bi-level Optimization Based Ablation Setting

We also perform an ablation study which reduces the proposed four-level optimization problem to a BLO problem. The study was conducted on the text classification task. For each source example $d_{s,i}$, we learn a weight $b_i \in [0, 1]$. A larger weight indicates that $d_{s,i}$ is more likely to be in the target domain. Let $B = \{b_i\}_{i=1}^{N_s}$ where $N_s$ is the number of source examples. We use the Transformer (Vaswani et al., 2017) $T$ to perform domain adaptation. It takes a source text $t$ as input and generates an adapted text $f(t, T)$ which is expected to be in the target domain. The Gumbel-Softmax (Jang et al., 2017) trick is used to deal with the non-differentiability of generated texts. At the lower level in the BLO formulation, we train the target model $U$. We define the training loss on selected source data as:

$$L_s(U, B) = \sum_{i=1}^{N_s} b_i \ell_{tgt}(U, d_{s,i}), \qquad (18)$$

and the training loss on adapted source data as:

$$L_a(U, B, T) = \sum_{i=1}^{N_s} (1 - b_i) \ell_{tgt}(U, f(d_{s,i}, T)). \qquad (19)$$

The optimization problem at this level is:

$$U^*(B, T) = \text{argmin}_U \; L_t(U) + \lambda \left(L_s(U, B) + L_a(U, B, T)\right), \qquad (20)$$

where $L_t(U)$ is the training loss on target training data, as defined in Eq.(3). At the upper level, we evaluate $U^*(B, T)$ on the target validation set $D_t^{(val)}$ and update $B$ and $T$ by minimizing the validation loss:

$$\min_{B,T} \; L(U^*(B, T), D_t^{(val)}). \qquad (21)$$

The overall BLO formulation is:

$$\begin{aligned} &\min_{B,T} \; L(U^*(B, T), D_t^{(val)}) \\ &s.t. \; U^*(B, T) = \\ &\text{argmin}_U \; L_t(U) + \lambda \left(L_s(U, B) + L_a(U, B, T)\right). \end{aligned} \qquad (22)$$

Table 6 shows the results. As can be seen, this BLO method performs worse than our method. The reason is: Our method uses self-supervised learning to learn domain discrepancy (in stage I) and leverages the learned domain discrepancy metric to perform domain adaptation (in stage II). Such mechanisms are lacking in the BLO method. This further demonstrates the necessity of stage I and II in our method.

| Method | Accuracy |
|---|---|
| BO (Ruder and Plank, 2017) | 79.9 |
| MMG (Wang et al., 2019a) | 82.1 |
| LSI (Huan et al., 2021) | 83.9 |
| Separate | 83.5 |
| WMVL | 81.7 |
| H-Divergence | 84.0 |
| Our full method | **88.2** |

Table 7: Human evaluation results.

### 4.4 Human Evaluation

We perform a human evaluation on whether the identified in-domain and out-of-domain source examples are indeed in or out of the target domain.

The study is performed on CHEMPROT, RCT, and ACL-ARC. For each dataset, we randomly sample 200 source texts. Three undergraduates were asked to label whether these source texts are in-domain or out-of-domain. Majority vote is leveraged to decide the final label. The Kappa score among the annotations is 0.75, which indicates a strong level of agreement among the annotators. Different methods are applied to predict whether each source text is in-domain or not. Table 7 shows the results. Our method achieves the best accuracy in identifying in-domain and out-of-domain source examples, due to its mechanism of learning the domain distance network in a discriminative way (as analyzed in Section 4.3).

## 5 Conclusions and Discussion

We propose a framework for simultaneous selection and adaptation of source examples in transfer learning. Our method automatically identifies which source examples are in or out of the target domain, and performs example-specific operations (either selection or adaptation). This is different from previous methods, which 1) discard out-of-domain source examples, leading to information loss; or 2) try to adapt all source examples into the target domain, incurring a risk of moving in-domain source examples outside the target domain. Our framework is based on four-level optimization. Experiments on text classification and visual question answering demonstrate the effectiveness of our proposed method.

## References


Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362.

Mathilde Bateson, Hoel Kervadec, Jose Dolz, Herve Lombaert, and Ismail Ben Ayed. 2020. Source-relaxed domain adaptation for image segmentation. *CoRR*, abs/2005.03697. https://doi.org/10.1007/978-3-030-59710-8_48

Atilim Gunes Baydin, Robert Cornish, David Martínez-Rubio, Mark Schmidt, and Frank D. Wood. 2017. Online learning rate adaptation with hypergradient descent. *CoRR*, abs/1703.04782.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine Learning*, 79(1):151–175. https://doi.org/10.1007/s10994-009-5152-4

Karsten M. Borgwardt, Arthur Gretton, Malte J. Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J. Smola. 2006. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57. https://doi.org/10.1093/bioinformatics/btl242, PubMed: 16873512

Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. 2017. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3722–3731. https://doi.org/10.1109/CVPR.2017.18

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*.

Michael Cogswell, Faruk Ahmed, Ross Girshick, Larry Zitnick, and Dhruv Batra. 2015. Reducing overfitting in deep networks by decorrelating representations. *arXiv preprint arXiv:1511.06068*.

Franck Dernoncourt and Ji Young Lee. 2017. PubMed 200k RCT: A dataset for sequential sentence classification in medical abstracts. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 308–313, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Lixin Duan, Dong Xu, and Ivor Wai-Hung Tsang. 2012. Domain adaptation from multiple sources: A domain-dependent regularization approach. *IEEE Transactions on Neural Networks and Learning Systems*, 23(3):504–518. https://doi.org/10.1109/TNNLS.2011.2178556, PubMed: 24808555

Hady Elsahar and Matthias Gallé. 2019. To annotate or not? Predicting performance drop under domain shift. 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 2163–2173, Hong Kong, China. Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1222

Matthias Feurer, Jost Springenberg, and Frank Hutter. 2015. Initializing Bayesian hyperparameter optimization via meta-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29. https://doi.org/10.1609/aaai.v29i1.9354

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR.org.

George Foster, Cyril Goutte, and Roland Kuhn. 2010. Discriminative instance weighting for domain adaptation in statistical machine translation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 451–459.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.emnlp-main.552

Weifeng Ge and Yizhou Yu. 2017. Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning. *CoRR*, abs/1702.08690. https://doi.org/10.1109/CVPR.2017.9

Saeed Ghadimi and Mengdi Wang. 2018. Approximation methods for bilevel programming. *arXiv preprint arXiv:1802.02246*.

Cyril Goutte and Eric Gaussier. 2005. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European Conference on Information Retrieval*. https://doi.org/10.1007/978-3-540-31865-1_25

Riccardo Grazzi, Luca Franceschi, Massimiliano Pontil, and Saverio Salzo. 2020. On the iteration complexity of hypergradient computation. In *International Conference on Machine Learning*, pages 3748–3758. PMLR.

Han Guo, Ramakanth Pasunuru, and Mohit Bansal. 2019. Autosem: Automatic task selection and mixing in multi-task learning. *CoRR*, abs/1904.04153.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of ACL*. https://doi.org/10.18653/v1/2020.acl-main.740

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2019. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. https://doi.org/10.1109/CVPR.2016.90

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*. https://doi.org/10.1145/2872427.2883037

Xuehai He, Yichen Zhang, Luntian Mou, Eric Xing, and Pengtao Xie. 2020. Pathvqa: 30000+ questions for medical visual question answering. *arXiv preprint arXiv:2003.10286*.

Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. 2018. Cycada: Cycle-consistent adversarial domain adaptation. In *International Conference on Machine Learning*, pages 1989–1998. PMLR.

Zhiting Hu, Bowen Tan, Ruslan Salakhutdinov, Tom M. Mitchell, and Eric P. Xing. 2019. Learning data manipulation for augmentation and weighting. *CoRR*, abs/1910.12795.

Zhaoxin Huan, Yulong Wang, Yong He, Xiaolu Zhang, Chilin Fu, Weichang Wu, Jun Zhou,

Ke Ding, Liang Zhang, and Linjian Mo. 2021. Learning to select instance: Simultaneous transfer learning and clustering. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1950–1954. https://doi.org/10.1145/3404835.3462992

Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex Smola. 2006. Correcting sample selection bias by unlabeled data. *Advances in Neural Information Processing Systems*, 19:601–608. https://doi.org/10.7551/mitpress/7503.003.0080

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparametrization with gumble-softmax. In *International Conference on Learning Representations (ICLR 2017)*. OpenReview. net.

Kaiyi Ji, Junjie Yang, and Yingbin Liang. 2021. Bilevel optimization: Convergence analysis and enhanced design. In *International Conference on Machine Learning*, pages 4882–4892. PMLR.

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in nlp. ACL.

David Jurgens, Srijan Kumar, Raine Hoover, Daniel A. McFarland, and Dan Jurafsky. 2018. Measuring the evolution of a scientific field through citation frames. *TACL*. https://doi.org/10.1162/tacl_a_00028

Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G. Hauptmann. 2019. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4893–4902. https://doi.org/10.1109/CVPR.2019.00503

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 Task 4: Hyperpartisan news detection. In *SemEval*. https://doi.org/10.18653/v1/S19-2145

Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. 2018. Bilinear attention networks. In *NIPS*.

Taekyung Kim and Changick Kim. 2020. Attract, perturb, and explore: Learning a feature alignment network for semi-supervised domain adaptation.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jens Kringelum, Sonny Kim Kjærulff, Søren Brunak, Ole Lund, Tudor I. Oprea, and Olivier Taboureau. 2016. ChemProt-3.0: A global chemical biology diseases mapping. In *Database*. https://doi.org/10.1093/database/bav123, PubMed: 26876982

Jian Liang, Dapeng Hu, and Jiashi Feng. 2021. Domain adaptation with auxiliary target domain-oriented classifier. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16632–16642. https://doi.org/10.1109/CVPR46437.2021.01636

Hanxiao Liu, Karen Simonyan, and Yiming Yang. 2018. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*.

Hong Liu, Mingsheng Long, Jianmin Wang, and Michael Jordan. 2019a. Transferable adversarial training: A general approach to adapting deep classifiers. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 4013–4022. PMLR.

Miaofeng Liu, Yan Song, Hongbin Zou, and Tong Zhang. 2019b. Reinforced training data selection for domain adaptation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics.

Risheng Liu, Yaohua Liu, Shangzhi Zeng, and Jin Zhang. 2021. Towards gradient-based bilevel optimization with non-convex followers and beyond. In *Advances in Neural Information Processing Systems*, 34.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019c. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Dan S. Weld. 2019. S2orc: The semantic scholar open research corpus. *arXiv preprint arXiv:1911.02782*.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. 2015. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105.

Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. 2017a. Conditional adversarial domain adaptation. *arXiv preprint arXiv:1705.10667*.

Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S. Yu. 2013. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2200–2207. https://doi.org/10.1109/ICCV.2013.274

Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. 2017b. Deep transfer learning with joint adaptation networks. In *International conference on machine learning*, pages 2208–2217. PMLR.

I. Loshchilov and F. Hutter. 2017. Fixing weight decay regularization in Adam. *ArXiv*, abs/1711.05101.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium, Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1360

Ping Luo, Fuzhen Zhuang, Hui Xiong, Yuhong Xiong, and Qing He. 2008. Transfer learning from multiple source domains via consensus regularization. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pages 103–112. https://doi.org/10.1145/1458082.1458099

Zelun Luo, Yuliang Zou, Judy Hoffman, and Li Fei-Fei. 2017. Label efficient learning of transferable representations across domains and tasks. *arXiv preprint arXiv:1712.00123*.

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*.

Mateusz Malinowski and Mario Fritz. 2014. A multi-world approach to question answering about real-world scenes based on uncertain input. In *NIPS*.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *ACM SIGIR*. https://doi.org/10.1145/2766462.2767755

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2017. Mixed precision training. *arXiv preprint arXiv:1710.03740*.

Lilyana Mihalkova, Tuyen Huynh, and Raymond J. Mooney. 2007. Mapping and revising markov logic networks for transfer learning. In *Proceedings of AAAI*. volume 7, pages 608–614.

Yu Mitsuzumi, Go Irie, Daiki Ikami, and Takashi Shibata. 2021. Generalized domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1084–1093. https://doi.org/10.1109/CVPR46437.2021.00114

Harsh Mohan. 2015. *Textbook of Pathology*. Jaypee Brothers Medical Publishers. https://doi.org/10.5005/jp/books/12412

Robert C. Moore and Will Lewis. 2010. Intelligent selection of language model training data.

Saeid Motiian, Quinn Jones, Seyed Iranmanesh, and Gianfranco Doretto. 2017. Few-shot adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, 30.

Jiquan Ngiam, Daiyi Peng, Vijay Vasudevan, Simon Kornblith, Quoc V. Le, and Ruoming Pang. 2018. Domain adaptive transfer learning with specialist models. *arXiv preprint arXiv:1811.07056*.

Alexandru Niculescu-Mizil and Rich Caruana. 2007. Inductive transfer for Bayesian network structure learning. In *Artificial Intelligence and Statistics*, pages 339–346. PMLR.

Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. 2010. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*,

22(2):199–210. https://doi.org/10.1109/TNN.2010.2091281, PubMed: 21095864

Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359. https://doi.org/10.1109/TKDE.2009.191

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *ACL*. https://doi.org/10.3115/1073083.1073135

Yash Patel, Kashyap Chitta, and Bhavan Jasani. 2018. Learning sampling policies for domain adaptation. *CoRR*, abs/1805.07641.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*. https://doi.org/10.3115/v1/D14-1162

Lorien Y. Pratt. 1993. Discriminability-based transfer between neural networks. In *Advances in Neural Information Processing Systems*, pages 204–204.

Chen Qu, Feng Ji, Minghui Qiu, Liu Yang, Zhiyu Min, Haiqing Chen, Jun Huang, and W. Bruce Croft. 2018. Learning to selectively transfer: Reinforced transfer learning for deep text matching. *CoRR*, abs/1812.11561.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. *arXiv preprint arXiv:1803.09050*.

Zhongzheng Ren, Raymond Yeh, and Alexander Schwing. 2020. Not all unlabeled data are equal: Learning to weight data in semi-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21786–21797. Curran Associates, Inc.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with bayesian optimization. *CoRR*, abs/1707.05246. https://doi.org/10.18653/v1/D17-1038

Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. 2019. Semi-supervised domain adaptation via minimax entropy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8050–8058. https://doi.org/10.1109/ICCV.2019.00814

Swami Sankaranarayanan, Yogesh Balaji, Carlos D. Castillo, and Rama Chellappa. 2018. Generate to adapt: Aligning domains using generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8503–8512. https://doi.org/10.1109/CVPR.2018.00887

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems*, pages 1919–1930.

Sunit Sivasankaran, Emmanuel Vincent, and Irina Illina. 2017. Discriminative importance weighting of augmented training data for acoustic model training. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4885–4889. IEEE. https://doi.org/10.1109/ICASSP.2017.7953085

Yan Song, Prescott Klassen, Fei Xia, and Chunyu Kit. 2012. Entropy-based training data selection for domain adaptation. In *Proceedings of COLING 2012: Posters*, pages 1191–1200.

Felipe Petroski Such, Aditya Rawal, Joel Lehman, Kenneth O. Stanley, and Jeff Clune. 2019. Generative teaching networks: Accelerating neural architecture search by learning to generate synthetic training data. *CoRR*, abs/1912.07768.

Baochen Sun and Kate Saenko. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision*, pages 443–450. Springer. https://doi.org/10.1007/978-3-319-49409-8_35

Qian Sun, Rita Chattopadhyay, Sethuraman Panchanathan, and Jieping Ye. 2011. A two-stage weighting framework for multi-source domain adaptation. *Advances in Neural Information Processing Systems*, 24:505–513.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*. https://doi.org/10.18653/v1/D19-1514

Hui Tang, Ke Chen, and Kui Jia. 2020. Unsupervised domain adaptation via structurally regularized deep clustering. https://doi.org/10.1109/CVPR42600.2020.00875

Hui Tang and Kui Jia. 2019. Discriminative adversarial domain adaptation. *CoRR*, abs/1911.12036.

Tatiana Tommasi, Francesco Orabona, and Barbara Caputo. 2010. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3081–3088. IEEE. https://doi.org/10.1109/CVPR.2010.5540064

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. 2017. Adversarial discriminative domain adaptation. *arXiv preprint arXiv:1702.05464*. https://doi.org/10.1109/CVPR.2017.316

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

B. Wang, Minghui Qiu, Xisen Wang, Y. Li, Y. Gong, X. Zeng, J. Huang, Bo Zheng, Deng Cai, and Jingren Zhou. 2019a. A minimax game for instance based selective transfer learning. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. https://doi.org/10.1145/3292500.3330841

Jindong Wang, Yiqiang Chen, Shuji Hao, Wenjie Feng, and Zhiqi Shen. 2017a. Balanced distribution adaptation for transfer learning. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 1129–1134. IEEE. https://doi.org/10.1109/ICDM.2017.150

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2017b. Instance weighting for neural machine translation domain adaptation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1482–1488, Copenhagen, Denmark. Association for Computational Linguistics. https://doi.org/10.18653/v1/D17-1155

Xinyi Wang, Hieu Pham, Paul Michel, Antonios Anastasopoulos, Jaime Carbonell, and Graham Neubig. 2020a. Optimizing data usage via differentiable rewards. In *International Conference on Machine Learning*, pages 9983–9995. PMLR.

Yulin Wang, Jiayi Guo, Shiji Song, and Gao Huang. 2020b. Meta-semi: A meta-learning approach for semi-supervised learning. *CoRR*, abs/2007.02394. https://arxiv.org/abs/2007.02394

Yunyun Wang, Dan Zhao, Yun Li, Ke-Jia Chen, and H. Xue. 2019b. The most related knowledge first: A progressive domain adaptation method. In *PAKDD*. https://doi.org/10.1007/978-3-030-26142-9_9

Junjie Yang, Kaiyi Ji, and Yingbin Liang. 2021. Provably faster algorithms for bilevel optimization. *Advances in Neural Information Processing Systems*, 34.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. 2016. Stacked attention networks for image question answering. In *CVPR*. https://doi.org/10.1109/CVPR.2016.10

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *Advances in Neural Information Processing Systems*, 32.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *NeurIPS*.

Yabin Zhang, Hui Tang, Kui Jia, and Mingkui Tan. 2019. Domain-symmetric networks for adversarial domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5031–5040. https://doi.org/10.1109/CVPR.2019.00517

Yang Zhang, Philip David, and Boqing Gong. 2017. Curriculum domain adaptation for semantic segmentation of urban scenes. *CoRR*, abs/1707.09465. https://doi.org/10.1109/ICCV.2017.223

Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. 2020. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76. https://doi.org/10.1109/JPROC.2020.3004555