Reinforced Training Data Selection for Domain Adaptation

Miaofeng Liu[♣∗†], **Yan Song**[♠†], **Hongbin Zou**[♦∗], **and Tong Zhang**[♥]
[♣]MILA & DIRO, Université de Montréal water3er@gmail.com
[♠]Tencent AI Lab clksong@gmail.com
[♦]School of Electronic Engineering, Xidian University hongbinzou@gmail.com
[♥]The Hong Kong University of Science and Technology tongzhang@ust.hk

Abstract

Supervised models suffer from the problem of domain shifting where distribution mismatch in the data across domains greatly affect model performance. To solve the problem, training data selection (TDS) has been proven to be a prospective solution for domain adaptation in leveraging appropriate data. However, conventional TDS methods normally requires a predefined threshold which is neither easy to set nor can be applied across tasks, and models are trained separately with the TDS process. To make TDS self-adapted to data and task, and to combine it with model training, in this paper, we propose a reinforcement learning (RL) framework that synchronously searches for training instances relevant to the target domain and learns better representations for them. A selection distribution generator (SDG) is designed to perform the selection and is updated according to the rewards computed from the selected data, where a predictor is included in the framework to ensure a task-specific model can be trained on the selected data and provides feedback to rewards. Experimental results from part-of-speech tagging, dependency parsing, and sentiment analysis, as well as ablation studies, illustrate that the proposed framework is not only effective in data selection and representation, but also generalized to accommodate different NLP tasks.

1 Introduction

Learning with massive data suffers from "Pyrrhic victory" where huge amounts of resource, e.g., computation, annotation, storage, etc., are consumed with many issues, one of which is that data quality considerably affects the performance of learned models. Especially in natural language processing (NLP), such phenomenon is incredibly significant where noise and inaccurate annotations are demolishing models' robustness when applying them across domains (Bollegala et al., 2011; Plank and Van Noord, 2011; Song and Xia, 2013; Ruder and Plank, 2018; Liu et al., 2018). Statistically, distribution mismatch is often observed between training and test data in such case. As a straightforward solution to reduce the impact of the mismatch, TDS is effective for learning across domains (Ruder and Plank, 2017) by preventing negative transfer from irrelevant samples and noisy labels (Rosenstein et al., 2005) while achieving equivalent performance with less computational efforts (Fan et al., 2017; Feng et al., 2018), especially when compared with learning-intensive domain adaptation methods such as sample reweighing (Borgwardt et al., 2006), feature distribution matching (Tzeng et al., 2014) and representation learning (Csurka, 2017).

Although various TDS-based domain adaptation approaches were proposed for NLP tasks (Daumé III, 2007; Blitzer et al., 2007a; Søgaard, 2011), most of them only consider scoring or ranking training data under a certain metric over the entire dataset, and then select the top n (or a proportion, which is a predefined hyper-parameter) items to learn. However, such pre-designed metrics are, always, neither able to cover effective characteristics for transferring domain knowledge nor can be applied in different data nature. Even though there exists a versatile metric, its hyper-parameter setting still demands further explorations. Moreover, conventional TDS is separate from model training, which requires more steps before an adapted model can be used, and restricts selecting appropriate instances when there is no feedback from the task. In doing so, the features or data representations of the selected instances are not adaptively learned and optimized, especially for neural models. Smarter TDS approaches are thus expected for domain adap-

∗ This work was done during the internship of Miaofeng Liu and Hongbin Zou at Tencent AI Lab.
† Corresponding authors.

tation to accommodate different data and tasks.

Consider that TDS is, in general, a combinatorial optimization problem with exponential complexity, it is impossible to try all possible combinations of training instances. An efficient solution to this problem is to transform it into a sequence of decision-making on whether select a (or a group of) training instance at each step, where previous decision should influence later ones. In this case, RL can be an appropriate vechile. To this end, one has to tackle two missions: to properly measure the correlation between a training sample and the target domain, and to guide the selection process with the feedback from the selected samples according to a specific task. For these missions, in this paper, we propose an RL framework for TDS that jointly learns the representation of the training data with respect to the target domain and selects them according to a learned distribution of selection probabilities. In detail, there are two major components in our framework: a selection distribution generator (SDG) for producing the selection probabilities, and a task-specific predictor including a feature extractor for learning data representations and a classifier[1] for measuring the performance of the selected data. The SDG and the predictor are pipelined by taking each others' output as their inputs and optimized accordingly via RL. With this framework, RL ensures the TDS process being conducted without requiring a predefined threshold and can automatically select the best instances in the training data as well as learn task- and domain-specific representations for them according to the target domain. As a result, useful information from the source domain is properly organized and represented and the redundant or noisy data are avoided in training the target domain specific models. Experimental results from three NLP tasks, namely, part-of-speech (POS) tagging, dependency parsing and sentiment analysis, illustrate that our approach achieves competitive performance, which confirm the validity and effectiveness of our approach. The code of this work is available at https://github.com/timerstime/SDG4DA

2 The Approach

We follow the common TDS setting in domain adaptation, i.e., for a task $\mathcal{T}$, one taking labeled instances from a source domain $\mathcal{D}_S$ as the pool, and some unlabeled data from a target domain $\mathcal{D}_T$ as the guidance. The routine of expected approaches for TDS is then to generate an optimal subset of data from the pool and train a model on it for $\mathcal{T}$.

Based on such routine, we design our approach with an architecture illustrated in Figure 1, with two major components, namely, the SDG and the predictor. The key component for TDS is the SDG, which produces a distribution vector based on the representation of the selected source data from the last selection step, then data instances are selected according to the vector and new reward is generated for next round of data selection. To update the SDG, different measurements can be used to assess the discrepancy between the representations of the selected source data and the guidance set and then approximates the value function for updating. The predictor takes the selected data and generates their representations in the feature extractor and trains a task-specific model by the classifier. The details of our framework is unfolded in the following subsections, in which we give the details of the two components and how they are jointly learned.

2.1 The Predictor

The predictor is the main component to train a particular model for $\mathcal{T}$. In our approach we decompose the predictor into two parts, the feature extractor and the classifier, and use them separately. The feature extractor serves as the representation learning module that transform selected data to vectors, while the classifier trains on the vector for $\mathcal{T}$. In this study, the predictor is a neural model so that the aforementioned separation are conducted by splitting neural layers. Normally, the feature extractor is the first n-1 layers of the predictor with n layers in total; the classifier is then the last layer.

The Feature Extractor Data in its original form, especially natural language, is usually difficult to be directly used in computation. The feature extractor thus serves as a critical component in our approach to transform the data into distributed representations for their efficient use. There are two-way inputs for the feature extractor. One is the guidance set $X_g^T = \{x_1^T, x_2^T, ..., x_m^T\}$, a collection of unlabeled data drawn from the target domain, serving as the reference for TDS. The other input is the selected data from the source domain in a "data bag", which is a batch of a certain amount of instances to facilitate TDS in this study. In detail, let $X_S = \{x_1^S, x_2^S, ..., x_n^S\}$, $\forall x_i^S \in \mathcal{X}_S$ denote the

[1]It is not necessarily a classifier, e.g., such as a tagger. However we use the term classifier for simplicity.

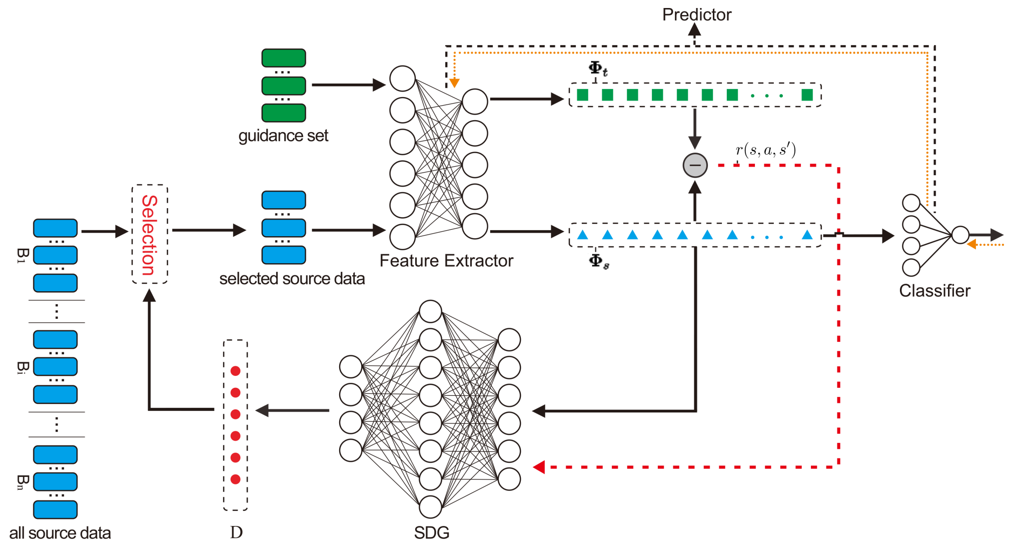


Figure 1: The architecture of our TDS framework, with a predictor (including a feature extractor and a classifier) and a selection distribution generator. All black solid arrows refer to data flow, while the red dashed arrow denotes reward with the orange dotted arrows indicating back-propagation of gradients from training the predictor.

data from the source domain, we uniformly and randomly partitions the entire data set into N disjoint data bags marked as $\{B_1, B_2, ..., B_N\}$, with $B_j = \{x^S_{(j-1)n/N+1}, x^S_{(j-1)n/N+2}, ..., x^S_{jn/N}\}$ and $j \in \{1, 2, ..., N\}$. Through the feature extractor, the guidance set and the selected data are transformed into two collections of distribution vectors.

The Classifier When each TDS round is done, the classifier is trained on the representations of the selected data for $\mathcal{T}$. During the training, the classifier passes the gradients to the feature extractor according to the labels of the selected data. The parameters of the classifier and the feature extractor are updated accordingly (with a learning rate β).

2.2 The Selection Distribution Generator

A multi-layer perceptron (MLP) model is used as the SDG, which learns the selection policy optimized by the reward from the representations of the guidance set and the selected data by RL. In doing so, at each step, the SDG is fed by a collection of representations for a data bag from the feature extractor. We denote the collection $\mathbf{\Phi}_{B_j} = \{r^j_1, r^j_2, ..., r^j_{|B_j|}\}$, where r^j_l $(l = 1, 2, ..., |B_j|)$ is the vector of the l-th sample[2] in B_j.[3] Then SDG maps $\mathbf{\Phi}_{B_j}$ into a vector $\mathbf{D}_{B_j} = (p^j_1, p^j_2, ..., p^j_{|B_j|})$, $p^j_l(l = 1, 2, ..., |B_j|)$, which represents the probability for each instance on the confidence of selecting it. To learn the SDG, each $\mathbf{\Phi}_{B_j}$ is measured with $\mathbf{\Phi_t}$ to give a reward in our framework, which is described in the following subsection.

2.3 The Reinforcement Learning Framework

We jointly train the SDG and the predictor with policy gradient method (Sutton et al., 1999), which favors actions with high rewards from better selected instances. The entire learning process is described in Algorithm 1, in which the notations are described in the following texts.

RL Components in Learning the SDG

- **State** $(s_1, s_2, ...s_j, ...s_N)$ includes a collection of states for all j with respect to N data bags, where each s_j indicates a state including selected instances $\hat{B}_j$ sampled from B_j according to the distribution vector $\mathbf{D}_{B_j}$, and parameters of the feature extractor for the $\hat{B}_j$. For simplicity we use $\mathbf{\Phi}_{\hat{B}_j}$ and $\mathbf{\Phi_t}$ to represent state s_j.
- **Action** For each state, the action space A is a 0-1 judgment to decide if selecting an instance (1) or not (0). An action $a = \{a_k\}^{|B_j|}_{k=1} \in \{0, 1\}^{|B_j|}$, which is obtained from $\mathbf{D}_{\hat{B}_j}$.[4] After each action, the framework gives new $\mathbf{\Phi}_{\hat{B}_j}$, then transforms state s into s'. The policy is defined as $P_{\mathbf{W}}(a|s)$.
- **Reward** The mathematical goal of TDS is to ensure that the selected data fit the distribution of the target domain. Hence we set a reward

[2]Representations in the collection follow the same order of their corresponding data instances in the bag.

[3]Similarly, the collection of representations for the guidance set is denoted as $\mathbf{\Phi_t}$.

[4]The process of assigning the value, i.e., 1 or 0, to k-th element of a can be formulated by sampling from a Bernoulli distribution parameterized by p^j_k of $\mathbf{D}_{B_j}$ w.r.t. B_j.

Algorithm 1: Joint training algorithm in our approach

```
Input: Training data in bags $\mathbf{B} = \{B_1, B_2, ..., B_N\}$;
       epochs $L$; $\mathbf{W}$ (SDG), $\mathbf{\Psi}$ (predictor, including
       feature extractor $\mathbf{\Theta}$); Loss function of the
       predictor $F(\mathbf{\Psi}, \hat{B}_j)$; $n_J$; $d(\cdot,\cdot)$; $\gamma$.
Output: Updated $\mathbf{W}$ and $\mathbf{\Psi}$ ($\mathbf{\Theta}$).
Initialize $\mathbf{W}$, $\mathbf{\Psi}(\mathbf{\Theta})$ with standard Gaussian distribution;
for epoch l = 1 to L do
    $\Sigma = 0$;
    for $k$ = 1 to $n_J$ do
        $\Sigma_r = 0$;
        Shuffle $\{B_1, B_2, ..., B_N\}$;
        for each $B_j \in \mathbf{B}$ do
            $\mathbf{\Phi}^{s_j}_{B_j} \leftarrow \mathbf{\Theta}_{j-1}(B_j)$; $\mathbf{\Phi}^{s_j}_t \leftarrow \mathbf{\Theta}_{j-1}(X^T_g)$;
             On current bag state $s_j$,
             $\mathbf{D}_j \leftarrow \mathbf{W}(\mathbf{\Phi}^{s_j}_t)$; select $\mathbf{\Phi}^{s_j}_{\hat{B}_j}$ from $\mathbf{\Phi}^{s_j}_{B_j}$
             via $\mathbf{D}_j$ (take action $a_j$);
            $r(s_{j-1}, a_j, s_j) \leftarrow$
              $d(\mathbf{\Phi}^{s_{j-1}}_{\hat{B}_{j-1}}, \mathbf{\Phi}^{s_{j-1}}_t) - \gamma d(\mathbf{\Phi}^{s_j}_{B_j}, \mathbf{\Phi}^{s_j}_t)$
            $\Sigma_r \leftarrow \Sigma_r + \gamma^{j-1} r(s_{j-1}, a_j, s_j)$
            $\mathbf{\Psi} \leftarrow \mathbf{\Psi} - \beta \nabla_{\mathbf{\Psi}} F(\mathbf{\Psi}, \hat{B}_j)$;
             ( $\mathbf{\Theta}_j \leftarrow \mathbf{\Theta}_{j-1} - \beta \nabla_{\mathbf{\Theta}} F(\mathbf{\Psi}, \hat{B}_j)$ ) ;
        end
        $\Sigma \leftarrow \Sigma + \sum_{j=1}^{N} \nabla_{\mathbf{W}} \log \pi_{\mathbf{W}}(a^k_j | s^k_j) \Sigma_r$;
    end
    $\nabla_{\mathbf{W}} \widetilde{J}(\mathbf{W}) \leftarrow \frac{1}{n_J} \Sigma$;
    $\mathbf{W} \leftarrow \mathbf{W} + \tau \nabla_{\mathbf{W}} \widetilde{J}(\mathbf{W})$;
end
```

$r(s, a, s')$ to assess the distance between $\mathbf{\Phi}_{\hat{B}_j}$ and $\mathbf{\Phi}_t$ in the current state (s') and its previous state (s):

$$r(s, a, s') = d(\mathbf{\Phi}^s_{\hat{B}_{j-1}}, \mathbf{\Phi}^s_t) - \gamma d(\mathbf{\Phi}^{s'}_{\hat{B}_j}, \mathbf{\Phi}^{s'}_t) \tag{1}$$

where $d(\cdot, \cdot)$ is a distribution discrepancy measurement, which can be implemented by different information-bearing functions. $\gamma \in (0, 1)$ is a discounting constant that decreases the impact from future distribution differences. Note that Eq. (1) is conducted in a sequential manner based on two adjacent data bags B_{j-1} and B_j, of which $\mathbf{\Phi}^{s'}_{\hat{B}_j}$ is impacted by $\mathbf{\Phi}^s_{\hat{B}_{j-1}}$ via parameters Ψ of the feature extractor updated by $\hat{B}_{j-1}$. Consequently, the state transition probability $\mathcal{P}(s'|s, a)$ is determined by stochastic optimization and other randomness in training, e.g., dropout (Srivastava et al., 2014). When better instances are selected, the reward is then expected to produce a higher value because the measurement for the previous state $d(\mathbf{\Phi}^s_{\hat{B}_{j-1}}, \mathbf{\Phi}^s_t)$ is supposed to give a larger distance between $\mathbf{\Phi}_{\hat{B}_{j-1}}$ and $\mathbf{\Phi}_t$ than that for the current state.

Distribution Discrepancy Measurements

To measure each $\hat{B}_j$ and the X^T_g, let $P = (p_1, \cdots, p_n)$ be the normalized element-wise average of $\Phi_{\hat{B}_j}$ and Q the average of $\mathbf{\Phi}_t$ similarly, we use the following measurements for $d(\cdot, \cdot)$:

- **JS**: The Jensen-Shannon divergence (Lin, 1991), $d(P, Q) = \frac{1}{2}[D_{KL}(P||M) + D_{KL}(Q||M)]$ where $D_{KL}(P||Q) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i}$, with $M = \frac{1}{2}(P + Q)$.
- **MMD**: The maximum mean discrepancy (Borgwardt et al., 2006), $d(P, Q) = \|P - Q\|$.
- **RÉNYI**: The symmetric Rényi divergence (Rényi, 1961), $d(P, Q) = \frac{1}{2}[Ry(P, M) + Ry(Q, M)]$, $Ry(P, Q) = \frac{1}{\alpha - 1} \log(\sum_{i=1}^{n} \frac{p_i^\alpha}{q_i^{\alpha-1}})$. We set $\alpha = 0.99$ following Van Asch and Daelemans (2010).
- **LOSS**: The guidance loss, defined as $d = -\frac{1}{m} \sum_{i=1}^{m} \sum_{y_t \in \mathcal{Y}_t} y_t \log p_\Phi(y_t | x^T_i)$, where y_t is the label of instance t from the guidance set, and p_ϕ the learned conditional probability of the predictor. Note that, different from aforementioned measurements, LOSS requires labels from the target domain, thus is only set as a comparison to other measurements used in our approach.

Optimization The following object is optimized to obtain the optimal distribution generation policy:

$$J(\mathbf{W}) = \mathbb{E}_{P_{\mathbf{W}(a|s)}}[\sum_{j=1}^{N} \gamma^{j-1} r(s_j, a_j)] \tag{2}$$

Then the parameters of the SDG, i.e., $\mathbf{W}$, is updated via policy gradient (Sutton et al., 1999) by

$$\mathbf{W} \leftarrow \mathbf{W} + \tau \nabla_{\mathbf{W}} \widetilde{J}(\mathbf{W}) \tag{3}$$

where τ is the discounting learning rate[5], the gradient $\nabla_{\mathbf{W}} J(\mathbf{W})$ is approximated by

$$\nabla_{\mathbf{W}} \widetilde{J}(\mathbf{W}) = \frac{1}{n_J} \sum_{k=1}^{n_J} \sum_{j=1}^{N} \nabla_{\mathbf{W}} \log \pi_{\mathbf{W}}(a^k_j | s^k_j) \sum_{j=1}^{N} \gamma^{j-1} r(s^k_j, a^k_j),$$

with j referring to the j-th step (corresponding to the j-th data bag) in RL, and k the k-th selection process to estimate $\nabla_{\mathbf{W}} J(W)$, which is updated after every n_J times of selection over all N data bags, where n_J is a predefined hyper-parameter.

[5]τ and the aforementioned β can be self-adapted by the optimizer, such as Adam (Kingma and Ba, 2014).

TASK	POS TAGGING/DEPENDENCY PARSING						SENTIMENT ANALYSIS			
DOMAIN	**A**	**EM**	**N**	**R**	**WB**	**WSJ**	**B**	**D**	**K**	**E**
LABELED	3.5K	4.9K	2.4K	3.8K	2.0K	3.0K	2K	2K	2K	2K
UNLABELED	27K	1,194K	1,000K	1,965K	525K	30K	4.5K	3.6K	5.7K	5.9K

Table 1: Statistics of all datasets used in our experiments, with the number presenting labeled or unlabeled samples in each domain. The domain abbreviations in different tasks are explained as follows. A:Answer, EM:Email, N:News, R:Reviews, WB:Weblogs, WSJ:Wall Street Journal, and B:Book, D:DVD, K:Kitchen, E:Electronics.

3 Experiment

To evaluate our approach, we conduct experiments on three representative NLP tasks: POS tagging, dependency parsing, and sentiment analysis. Details about the experiments are described as follows.

3.1 Datasets

Two popular datasets are used in our experiments. For POS tagging and dependency parsing, we use the dataset from the SANCL 2012 shared task (Petrov and McDonald, 2012), with six different domains. For sentiment analysis, we use the product review dataset from (Blitzer et al., 2007b), with four domains. Note that for all datasets, there exists both labeled and unlabeled samples in each domain. The statistics and the domains for the aforementioned datasets are reported in Table 1.

3.2 Settings

A major difference between our approach and other data selection methods is that the threshold (number of instances to be selected), n, is not fixed in our approach. Instead, it chooses the most effective ones automatically. For fair comparison, we record the resulted n from our approach in different tasks and use it in other methods to guide their selection. In all experiments, we use a multi-source domain setting where the source domain includes all labeled data from the dataset except that for the target domain, i.e., we take turns selecting a domain as the target domain, and use the union of the rest as the source domain. The number of bags, N, is set separately for each dataset to ensure a uniform bag size of 1K samples. For the guidance set, we follow Ruder and Plank (2017) and randomly select half of the instances from all the test data in the target domain discarding their labels.

Consider that the starting reward needs to be calculated from a reliable feature extractor, we adopt a "soft starting" before the regular training, were we pre-train the predictor on all source data for 2 epochs, then initialize parameters of SDG with Gaussian variables. Afterwards the predictor and SDG follow ordinary learning paradigm in each training epoch. In all experiments, we use Adam (Kingma and Ba, 2014) as the optimizer, and set γ to 0.99 following Fan et al. (2017) and n_J to 3.

	A	**EM**	**N**	**R**	**WB**	**WSJ**
JS-E	93.16	93.77	94.29	93.32	94.92	94.08
JS-D	92.25	93.43	93.54	92.84	94.45	93.32
T-S	93.59	94.65	94.76	93.92	95.32	94.44
TO-S	93.36	94.65	94.43	94.65	94.03	94.22
T+TO-S	94.33	92.55	93.96	93.94	94.51	94.98
T-S+D	93.64	94.21	93.57	93.86	95.33	93.84
TO-S+D	94.02	94.33	94.62	94.19	94.93	94.67
RANDOM	92.76	93.43	93.75	92.62	93.53	92.68
ALL	95.16	95.90	95.90	95.03	95.79	95.64
SDG (JS)	95.37	95.45	96.23	95.64	96.19	95.74
SDG (MMD)	**95.75**	96.23	96.40	95.51	**96.95**	96.12
SDG (RÉNYI)	95.52	**96.31**	**96.62**	**95.97**	96.75	**96.35**
SDG (LOSS)	95.46	95.77	95.92	95.50	96.03	95.82

Table 2: POS tagging results (accuracy %).

3.3 POS tagging

The Predictor We use the Bi-LSTM tagger proposed in Plank et al. (2016) as the predictor.

Baselines Following Ruder and Plank (2017), we compare our approach to five baselines: 1) **JS-E**: top instances selected according to Jensen-Shannon divergence. 2) **JS-D**: top instances selected from the most similar source domain, where the similarity between domains are determined by Jensen-Shannon divergence. 3) Bayesian optimization (Brochu et al., 2010) with the following settings: **T-S**, term distribution similarity; **TO-S**, topic distribution similarity; **T+TO-S**, joint term and topic distribution similarity; **T-S+D**, term distribution similarity and diversity; **TO-S+D**, topic distribution similarity and diversity. 5) **RANDOM**: a random selection model that selects the same number of instances with the n given by our approach. 6) **ALL**: The predictor is trained on all source data.

Results POS tagging results are reported in Table 2. Overall, our approach with different distri-

	A	EM	N	R	WB	WSJ
JS-E	81.02	80.53	83.25	84.66	85.36	82.43
JS-D	82.80	79.93	81.77	83.98	83.44	80.61
T-S	83.79	81.09	82.68	84.66	84.85	82.57
TO-S	82.87	81.43	82.07	83.98	84.98	82.90
T+TO-S	82.87	81.13	82.97	84.65	84.43	82.43
T-S+D	83.72	81.60	82.80	84.62	85.44	82.87
TO-S+D	82.60	80.83	84.04	84.45	85.89	82.33
RANDOM	81.28	83.41	81.03	82.67	82.46	80.74
ALL	**85.65**	**87.78**	**86.07**	**87.27**	85.51	85.56
SDG (JS)	84.03	85.98	84.17	86.25	**86.22**	85.24
SDG (MMD)	84.19	86.25	84.87	86.80	85.57	84.37
SDG (RÉNYI)	84.55	85.11	85.27	86.93	85.65	**85.79**
SDG (LOSS)	83.97	85.86	84.05	86.21	86.03	84.98

Table 3: Dependency parsing results (LAS).

bution discrepancy metrics outperforms all baselines based on the same predictor. This observation demonstrates the excellent adaptability of our approach in this task although there is complicated structural variance in sentences. Among the four metrics, Rényi divergence achieve the best overall performance, which is slightly surpassed by MMD in the ANSWER and WEBLOGS domain. We observe around 50 epochs of training to reach convergence of our approach. As a result, 50% training data in the source domain are selected.

3.4 Dependency Parsing

The Predictor The Bi-LSTM parser proposed by Kiperwasser and Goldberg (2016) is the predictor.

Baselines For dependency parsing, we use the same baselines introduced in the POS tagging task.

Results The performance (labeled attachment scores, LAS) of dependency parsing is reported in Table 3. Similar to POS tagging, the term distribution-based method (T-S) as well as its combination with diversity features (T-S+D) outperform other Bayesian optimization baselines. Our models are also shown to be superior than measurement-based as well as neural models significantly in most domains. However, different from POS tagging, in this task, the predictor trained on the entire source data still performs the best on some domains, which can be explained by the complexity of the task. To precisely predict structured parsing results, in spite of noise from different domains, large amount of data might be more helpful because various contextual information is beneficial in text representation learning (Song et al., 2018). In this case, selection based methods sacrifice accuracy for their efficiency with less data.

	B	D	E	K
JS-E	72.49	68.21	76.78	77.54
JS-D	75.28	73.75	72.53	80.05
T-S	75.39	76.27	81.91	83.41
TO-S	76.07	75.92	81.69	83.06
T+TO-S	75.75	76.62	81.74	83.39
T-S+D	76.20	77.60	82.66	84.98
TO-S+D	77.16	79.00	81.92	84.29
SCL	74.57	76.30	78.93	82.07
SST	76.32	78.77	83.57	85.19
DAM	75.61	77.57	82.79	84.23
SDAMS-LS	77.95	78.80	83.98	85.96
SDAMS-SVM	77.86	79.02	**84.18**	85.78
RANDOM	76.78	75.28	78.25	82.27
ALL	78.48	79.68	80.58	84.50
SDG (JS)	79.37	81.06	82.38	85.78
SDG (MMD)	79.57	81.08	82.68	85.69
SDG (RÉNYI)	**80.07**	**82.07**	82.28	**86.18**
SDG (LOSS)	79.57	80.58	81.88	85.08

Table 4: Sentiment analysis results (accuracy %).

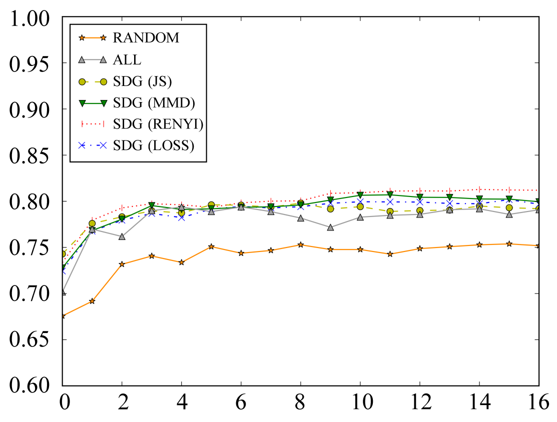
Yet, our models, e.g., the SDG (JS) and SDG (RÉNYI), outperform the ALL model in the last two domains, with only half of the source domain data used. We observe that averagely 60 epochs of training is required to obtain the best model.

3.5 Sentiment Analysis

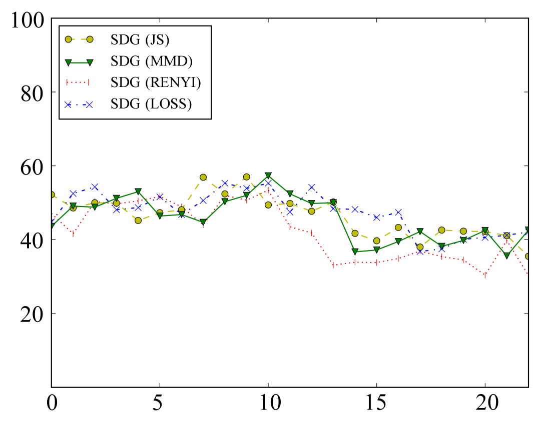
The Predictor We adopt the CNN classifier proposed by Kim (2014) as the predictor in this task.

Baselines In addition to the baselines for POS tagging and dependency parsing, we use a series of extra baselines from previous studies: 1) **SCL**, the structural correspondence learning proposed by Blitzer et al. (2006); 2) **SST**, the sentiment sensitive thesaurus method (Bollegala et al., 2011); 3) **DAM**, a general-purpose multi-source domain adaptation method proposed by Mansour et al. (2008); 4) **SDAMS-LS** and **SDAMS-SVM**, the specially designed sentiment domain adaptation approach (Wu and Huang, 2016) for multiple sources with square loss and hinge loss, respectively.

Results Table 4 presents the results for sentiment analysis. Similar to previous tasks, it is observed that our approach still performs well in this task, even though compared with the algorithms particularly designed for sentiment analysis (e.g., SDAMS). A potential reason for our weaker results on ELECTRONICS domain is that SDAMS methods use relation graphs among key words as prior knowledge, while our model does not need that and aims for a wider application without such task-specific consideration. Slightly different from previous tasks, in this task, around 40% source data



(a) Accuracies against training epochs.



(b) % data selected against training epochs.

Figure 2: Investigation curves of using different models on the DVD domain for sentiment analysis.

are selected upon the convergence of our approach with around 15 epochs of training. Note that, although there exist other recent domain adaptation methods exclusively designed for sentiment analysis (Barnes et al., 2018; Ziser and Reichart, 2018) with stronger results, their setting mainly focused on single source domain adaptation. Thus they are not directly compared with our models and baselines in Table 4, which is for a more general and challenging setting with multiple source domains.

3.6 Discussion

In all three tasks, our approach achieve the best overall performance when there are half or less than half source domain data selected to train the predictor. The comparisons between our approach and the basic distribution measure-based methods, the general-purpose multi-source approach as well as models from previous studies (in sentiment analysis) across all tasks illustrate the superiority of our approach in selecting the most useful instances for the target domain while eliminating negative effects. However, domain variance is task-specific and still plays an important role affecting model performance. Compared to POS tagging and dependency parsing, in sentiment analysis, there exists more significant bias across domains, e.g., words such as "*small*" and "*cheap*" could be positive in one domain but negative in another. As a result, topic relevant domains express similar sentiment expressions. The investigation on the selected data indicates that our approach chooses more instances from the similar domains in sentiment analysis (e.g., BOOK $\Rightarrow$ DVD), while the selected instances in POS tagging and dependency parsing are more balanced across domains. This observation suggests the effectiveness of our approach in adapting different tasks with the most appropriate strategy.

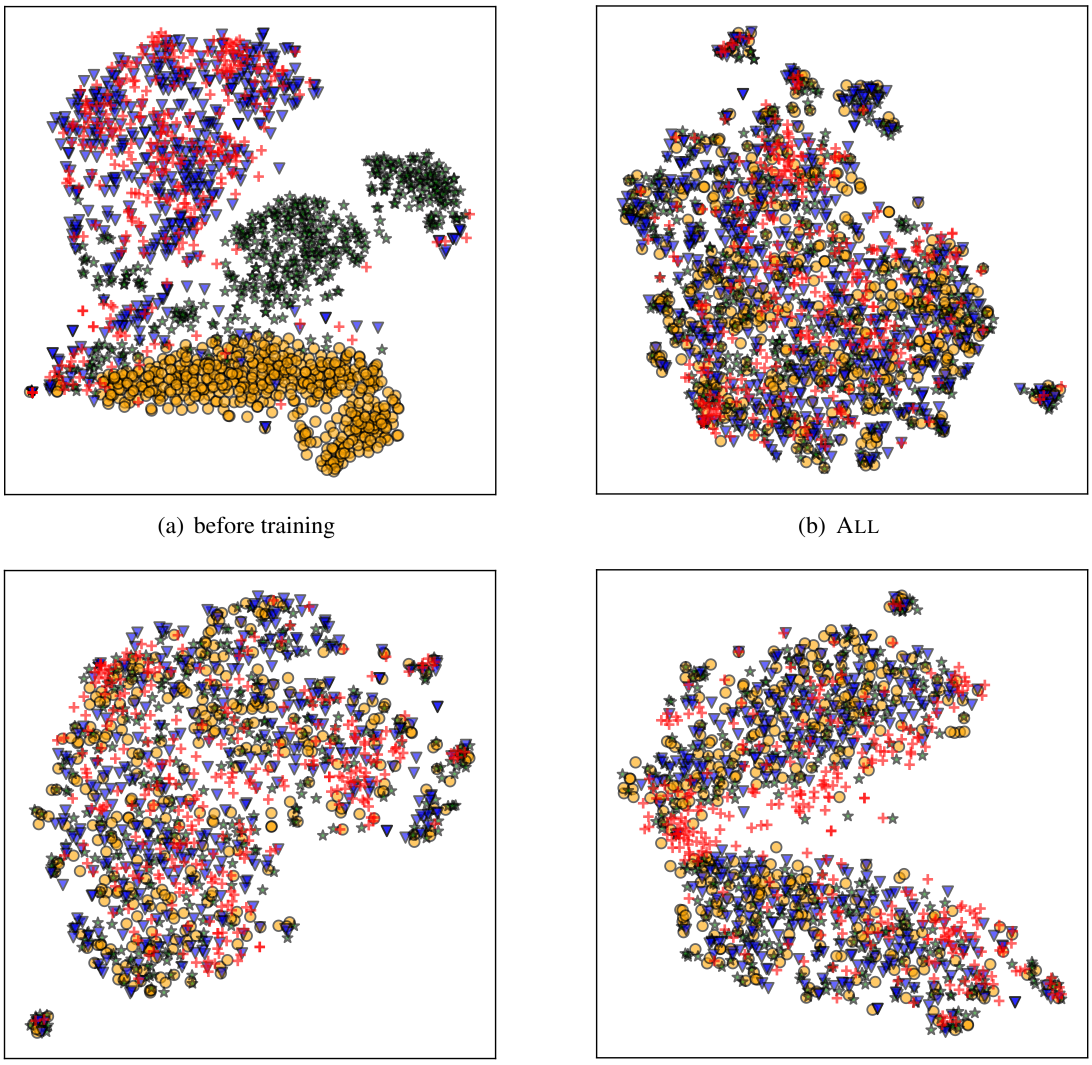
Yet, in addition, there still exist side effects on noise filtering and relevant instance selection, which can be observed from the slightly weaker results on ELECTRONICS domain in sentiment analysis as well as the fact that our approach is outperformed by training on all source data (in some domains) in parsing task. Such phenomenon implies that filtering irrelevant instances may lose intrinsic beneficial information for the target domain. Moreover, policy gradient method with partial data may sometimes converges to a local optima when learning on structured data because there exist many indirect relations among the learning instances.

4 Ablation Studies

4.1 Performance and Efficiency Analysis

To better understand the behavior of our model with different measurements, we investigate their performance through a case study on the DVD domain in sentiment analysis. We draw accuracy curves of different models with respect to their training epochs, as shown in Figure 2(a). In general, our models present similar performance and are significantly better than the RANDOM one. Interestingly, their curves are similar to the ALL model but show a much stable fluctuation with epoch increasing. This observation demonstrates that there exist noise when directly using all source data, while our models are able to overcome such limitation.

Another investigation is to study how much data are selected by different variants of our model. We display the number of instances selected by the four measurements in Figure 2(b), using the same do-


(a) before training

(b) ALL

(c) JS-E

(d) SDG (RÉNYI)

Figure 3: t-SNE visualization of features (data representations) from the feature extractor in different scenarios for sentiment analysis on the DVD domain. Red cross, blue triangle, green star, and orange circle symbols represent samples from DVD, BOOKS, ELECTRONICS, and KITCHEN domain, respectively.

main and task setting as that in Figure 2(a). Overall our models with different measurements share similar behavior in selecting source data in terms of selection numbers. They tend to select more data at the beginning stage, i.e., before 10 epochs, then reduce selected instances to a smaller set and maintain the performance of the predictor (comparing with curves in Figure 2(a)). Among all measurements, Rényi divergence tend to select less data while achieving a better performance when matching its curve with the results reported in Table 4. In addition, we perform an early stop when the decrement of the training error falls below a preset threshold. Alternatively, to avoid over-selection, one can follow Klein et al. (2017) to predict the development of the performance curve so that TDS can be done more efficiently in fewer epochs.

4.2 Distribution Visualization

To better demonstrate the effectiveness of our approach, we still use the sentiment analysis for the DVD domain with SDG (RÉNYI) for visualized comparison among the distributions of the features (data representations) in different scenarios. Following Tzeng et al. (2014), we plot in Figure 3 the t-SNE visualizations of the features learned from the feature extractor in four settings: features before training (initialized weights) (Figure 3(a)), directly trained on all source data (Figure 3(b)), trained with JS-E (Figure 3(c)), and trained with SDG (Rényi) (Figure 3(d)). It is observed that, for original features, DVD and BOOKS are similar, while ELECTRONICS and KITCHEN are different from them as well as to each other. When trained with all source data, features are visualized with some changes in their distributions where instances from different domains are mixed and closer to the target domain. In the case where JS divergence is minimized for each instance, we can see a further mixture with closer representation matching. The figures indicate that, for both ALL and JS-E models, their domain adaptation ability is limited since the learned representations are not optimized for the target domain. On the contrary, when trained with our approach, the selected instances result in a highly similar distribution as that in the target domain (Figure 3(d)), with matched shape between the points in red and other colors. Such visualization confirms that our TDS framework not only selects the most appropriate instances (similar in the distribution shape), but also learns better representations (located at the similar positions of target domain instances) for them with respect to the target domain, which further illustrates the validity and effectiveness of joint selecting and learning from training instances for domain adaptation.

5 Related Work

Many studies have been conducted recently for domain adaptation with neural networks (Long et al., 2015, 2017; Shu et al., 2018; Shankar et al., 2018). Their methodologies follow several mainstreams such as representation learning (Glorot et al., 2011; Chen et al., 2012; Baktashmotlagh et al., 2013; Song and Shi, 2018; Zhao et al., 2017), reweighing samples from the source domain (Borgwardt et al., 2006; Daumé III, 2007; Song and Xia, 2013), and feature space transformation (Gopalan et al., 2011; Pan et al., 2011; Long et al., 2013), etc.

Normally, the transferable knowledge across domains are derived from some certain data, while others contribute less and are costly to be learned from (Axelrod et al., 2011; Ruder and Plank, 2017). Thus, previous studies conduct domain adaptation through selecting relative and informative source data according to the nature of the target domain, via entropy-based methods (Song et al., 2012), Bayesian optimization (Ruder and Plank, 2017), etc. Particularly for NLP, TDS are proved to be effective in various tasks, such as in language modeling (Moore and Lewis, 2010), word segmentation (Song and Xia, 2012; Song et al., 2012), machine translation (Chen et al., 2016; van der Wees et al., 2017), and multilingual NER (Murthy et al., 2018).

Recently, RL and representation learning provided new possibilities for TDS. For example, Fan et al. (2017) proposed to allocate appropriate training data at different training stages, which helps achieving comparative accuracy with less computational efforts compared with the model trained on the entire data. Feng et al. (2018) used sequential one-step actions for each single instance where every action is decided based on the previous one. As a result, their selection becomes a consuming process where the complexity is determined by the amount of the source data. For representation learning based approaches, there are studies such as Mansour et al. (2008); Gopalan et al. (2014); Pei et al. (2018) that adapted representations across domains, which is a widely adopted strategy for domain adaptation on neural models. Moreover, a similar work (Dong and Xing, 2018) adopted reinforced sampling strategy specifically for one-shot scenarios. Compared to aforementioned previous work, the proposed approach in this paper combines TDS and transferable representation learning in a unified RL framework, and is conducted in an effective way using data batches.

6 Conclusion

In this paper, we proposed a general TDS framework for domain adaptation via reinforcement learning, which matches the representations of the selected data from the source domain and the guidance set from the target domain and pass the similarity at different steps as rewards to guide a selection distribution generator. Through the generator, different instances from the source domain are selected to train a task-specific predictor. To this end, not only those data relevant to the target domain are selected, but also task- and domain-specific representations are learned for them. Experimental results from three NLP tasks, i.e., POS tagging, dependency parsing, and sentiment analysis, demonstrate that our models outperform various baselines across domains, especially (in most cases) the same predictor trained on all source data. Ablation studies on model convergence, selection numbers, as well as distribution visualizations further confirmed the validity and effectiveness of our approach.

References

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain Adaptation via Pseudo in-domain

Data Selection. In *Proceedings of the conference on empirical methods in natural language processing*, pages 355–362.

Mahsa Baktashmotlagh, Mehrtash Tafazzoli Harandi, Brian C. Lovell, and Mathieu Salzmann. 2013. Unsupervised domain adaptation by domain invariant projection. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 769–776.

Jeremy Barnes, Roman Klinger, and Sabine Schulte im Walde. 2018. Projecting embeddings for domain adaption: Joint modeling of sentiment analysis in diverse domains. *arXiv preprint arXiv:1806.04381*.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007a. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007b. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain Adaptation with Structural Correspondence Learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128.

Danushka Bollegala, David J. Weir, and John A. Carroll. 2011. Using multiple sources to construct a sentiment sensitive thesaurus for cross-domain sentiment classification. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 132–141.

Karsten M. Borgwardt, Arthur Gretton, Malte J. Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alexander J. Smola. 2006. Integrating Structured Biological Data by Kernel Maximum Mean Discrepancy. In *Proceedings 14th International Conference on Intelligent Systems for Molecular Biology 2006, Fortaleza, Brazil, August 6-10, 2006*, pages 49–57.

Eric Brochu, Vlad M Cora, and Nando De Freitas. 2010. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*.

Boxing Chen, Roland Kuhn, George Foster, Colin Cherry, and Fei Huang. 2016. Bilingual Methods for Adaptive Training Data Selection for Machine Translation. In *Proc. of AMTA*, pages 93–103.

Minmin Chen, Zhixiang Eddie Xu, Kilian Q. Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*.

Gabriela Csurka, editor. 2017. *Domain Adaptation in Computer Vision Applications*. Advances in Computer Vision and Pattern Recognition. Springer.

Hal Daumé III. 2007. Frustratingly Easy Domain Adaptation. In *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*.

Nanqing Dong and Eric P Xing. 2018. Domain adaption in one-shot learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 573–588. Springer.

Yang Fan, Fei Tian, Tao Qin, Jiang Bian, and Tie-Yan Liu. 2017. Learning what data to learn. *arXiv preprint*, abs/1702.08635.

Jun Feng, Minlie Huang, Li Zhao, Yang Yang, and Xiaoyan Zhu. 2018. Reinforcement learning for relation classification from noisy data. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 513–520.

Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. 2011. Domain Adaptation for Object Recognition: An Unsupervised Approach. In *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pages 999–1006.

Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. 2014. Unsupervised Adaptation Across Domain Shifts by Generating Intermediate Data Representations. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2288–2302.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *TACL*, 4:313–327.

Aaron Klein, Stefan Falkner, Jost Tobias Springenberg, and Frank Hutter. 2017. Learning Curve Prediction with Bayesian Neural Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017*.

Jianhua Lin. 1991. Divergence Measures Based on the Shannon Entropy. *IEEE Trans. Information Theory*, 37(1):145–151.

Miaofeng Liu, Jialong Han, Haisong Zhang, and Yan Song. 2018. Domain Adaptation for Disease Phrase Matching with Adversarial Networks. In *Proceedings of the BioNLP 2018 workshop*, pages 137–141, Melbourne, Australia.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. 2015. Learning Transferable Features with Deep Adaptation Networks. *arXiv preprint arXiv:1502.02791*.

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.

Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. 2017. Deep Transfer Learning with Joint Adaptation Networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 2208–2217, Sydney, Australia.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. 2008. Domain adaptation with multiple sources. In *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, pages 1041–1048.

Robert C Moore and William Lewis. 2010. Intelligent Selection of Language Model Training Data. In *Proceedings of the ACL 2010 conference short papers*, pages 220–224. Association for Computational Linguistics.

Rudra Murthy, Anoop Kunchukuttan, and Pushpak Bhattacharyya. 2018. Judicious Selection of Training Data in Assisting Language for Multilingual Neural NER. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 401–406.

Sinno Jialin Pan, Ivor W. Tsang, James T. Kwok, and Qiang Yang. 2011. Domain adaptation via transfer component analysis. *IEEE Trans. Neural Networks*, 22(2):199–210.

Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. 2018. Multi-adversarial Domain Adaptation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Slav Petrov and Ryan McDonald. 2012. Overview of the 2012 shared task on parsing the web. In *Notes of the first workshop on syntactic analysis of non-canonical language (sancl)*, volume 59. Citeseer.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. *arXiv preprint arXiv:1604.05529*.

Barbara Plank and Gertjan Van Noord. 2011. Effective measures of domain similarity for parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1566–1576.

Alfréd Rényi. 1961. On measures of entropy and information. Technical report, HUNGARIAN ACADEMY OF SCIENCES Budapest Hungary.

Michael T Rosenstein, Zvika Marx, Leslie Pack Kaelbling, and Thomas G Dietterich. 2005. To transfer or not to transfer. In *NIPS 2005 workshop on transfer learning*, volume 898, pages 1–4.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with Bayesian Optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 372–382, Copenhagen, Denmark.

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

Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. 2018. Generalizing across domains via cross-gradient training. *arXiv preprint arXiv:1804.10745*.

Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. 2018. A dirt-t approach to unsupervised domain adaptation. *arXiv preprint arXiv:1802.08735*.

Anders Søgaard. 2011. Data point selection for cross-language adaptation of dependency parsers. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 682–686, Portland, Oregon, USA.

Yan Song, Prescott Klassen, Fei Xia, and Chunyu Kit. 2012. Entropy-based Training Data Selection for Domain Adaptation. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 1191–1200, Mumbai, India.

Yan Song and Shuming Shi. 2018. Complementary Learning of Word Embeddings. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4368–4374.

Yan Song, Shuming Shi, Jing Li, and Haisong Zhang. 2018. Directional Skip-Gram: Explicitly Distinguishing Left and Right Context for Word Embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 175–180, New Orleans, Louisiana.

Yan Song and Fei Xia. 2012. Using a Goodness Measurement for Domain Adaptation: A Case Study on Chinese Word Segmentation. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 3853–3860, Istanbul, Turkey.

Yan Song and Fei Xia. 2013. A Common Case of Jekyll and Hyde: The Synergistic Effect of Using Divided Source Training Data for Feature Augmentation. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 623–631, Nagoya, Japan.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pages 1057–1063.

Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. 2014. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint*, abs/1412.3474.

Vincent Van Asch and Walter Daelemans. 2010. Using domain similarity for performance estimation. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, pages 31–36.

Marlies van der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic Data Selection for Neural Machine Translation. *arXiv preprint arXiv:1708.00712*.

Fangzhao Wu and Yongfeng Huang. 2016. Sentiment Domain Adaptation with Multiple Sources. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 301–310.

Han Zhao, Shanghang Zhang, Guanhang Wu, João P. Costeira, José M. F. Moura, and Geoffrey J. Gordon. 2017. Multiple Source Domain Adaptation with Adversarial Training of Neural Networks. *arXiv preprint*, abs/1705.09684.

Yftah Ziser and Roi Reichart. 2018. Pivot based Language Modeling for Improved Neural Domain Adaptation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 1241–1251.