The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-20)

Multi-Source Domain Adaptation for Text Classification via DistanceNet-Bandits

Han Guo, Ramakanth Pasunuru, Mohit Bansal
UNC Chapel Hill
{hanguo, ram, mbansal}@cs.unc.edu

Abstract

Domain adaptation performance of a learning algorithm on a target domain is a function of its source domain error and a divergence measure between the data distribution of these two domains. We present a study of various distance-based measures in the context of NLP tasks, that characterize the dissimilarity between domains based on sample estimates. We first conduct analysis experiments to show which of these distance measures can best differentiate samples from same versus different domains, and are correlated with empirical results. Next, we develop a DistanceNet model which uses these distance measures, or a mixture of these distance measures, as an additional loss function to be minimized jointly with the task's loss function, so as to achieve better unsupervised domain adaptation. Finally, we extend this model to a novel DistanceNet-Bandit model, which employs a multi-armed bandit controller to dynamically switch between multiple source domains and allow the model to learn an optimal trajectory and mixture of domains for transfer to the low-resource target domain. We conduct experiments on popular sentiment analysis datasets with several diverse domains and show that our DistanceNet model, as well as its dynamic bandit variant, can outperform competitive baselines in the context of unsupervised domain adaptation.

1 Introduction

In situations where large-scale annotated datasets are available, supervised learning algorithms have achieved remarkable progress in various NLP challenges (LeCun, Bengio, and Hinton 2015). Most supervised learning algorithms rely on the assumption that data distribution during training is the same as that during test. However, in many real-life scenarios, the data distribution of interest at test-time might be different from that during training. The process of collecting new datasets that reflect the new distribution is usually not scalable due to monetary as well as time constraints. Hence, the goal of domain adaptation is to construct a learning algorithm, which, given samples of observations from a source domain, is able to adapt its performance to a target domain where the data distribution could be different.

Two major research areas in domain adaptation include supervised domain adaptation and unsupervised domain adaptation. In the former setup, limited training data from the target domain is available to provide supervision signals (Daumé III 2009), whereas in the latter case, only unlabeled data from the target domain is available (Ganin et al. 2016; Long et al. 2017; Bousmalis et al. 2016; Sun, Feng, and Saenko 2016; Sun and Saenko 2016; Tzeng et al. 2017). In this work, we focus on the unsupervised domain adaptation. It has been shown that the domain adaptation performance is influenced by three major (and orthogonal) factors (Ben-David et al. 2010). The first factor is the model performance on the source task, which benefits from recent advancements in neural models and is orthogonal to our focus. The second factor is the difference in the labeling functions across domains, which is inherent to the nature of the dataset and expected to be small in practice (Ben-David et al. 2010). The third factor represents a measure of divergence of data distributions – if the data distribution between the source and target domain is similar, we can reasonably expect a model trained on the source domain to perform well on the target domain. Our work primarily focuses on the last factor and aims to study the following two questions in the context of NLP: how to accurately estimate the dissimilarity between a pair of domains (Sec. 3 and Sec. 6), and how to leverage these domain dissimilarity measures to improve domain adaptation learning (Sec. 4 and Sec. 7).

To this end, we first provide a detailed study (comparison, models, and analyses) of several domain distance measures from the literature, with the goal of scalability (easy to calculate), differentiability (can be minimized), and interpretability (in a simple analytical form with well-studied properties), namely $\mathcal{L}_2$, Maximum Mean Discrepancy (MMD) (Gretton et al. 2012), Fisher Linear Discriminant (FDA) (Friedman, Hastie, and Tibshirani 2001), Cosine, and Correlation Alignment (CORAL) (Sun, Feng, and Saenko 2016). We start by defining these distance measures in Sec. 3, and provide a set of analyses to assess them in Sec. 6: (1) the ability of these distance measures to separate domains, and (2) the correlation between these distance measures and empirical results. From these analyses, we note that there does not exist a single best distance measure that fits all, and each measure pro-

vides an estimate of domain distance that could be complementary (e.g., based on discrepancy versus class separation). Thus, we also propose to use a mixture of distance measures, where we additionally introduce an unsupervised criterion to select the best distance measures so as to reduce the number of extra weight hyperparameters when mixing them.

Motivated by the aforementioned analysis, we next present a simple 'DistanceNet' model (in Sec. 4) that integrates these measures into the training optimization. In particular, we augment the classification task loss function with an additional distance measure. By minimizing the representational distances between features from source and target domains, the model learns better domain-agnostic features. Finally, when data from multiple source domains are present, we learn a dynamic scheduling of these domains that maximizes the learning performance on the no-training target task by framing the problem of dynamic domain selection as a multi-armed bandit problem, where each arm represents a candidate source domain.

We conduct our analyses and experiments on a popular sentiment analysis dataset with several diverse domains from Liu, Qiu, and Huang (2017), and present the domain adaptation results in Sec. 7. We first show that a subset of the domain discrepancy measures is able to separate samples from source and target domains. Then we show that our DistanceNet model, which uses one or a mixture of multiple domain discrepancies as an extra loss term, can outperform multiple competitive baselines. Finally, we show that our dynamic, bandit variant of the DistanceNet can also outperform a fairly comparable multi-source baseline that has access to the same amount of data.

We start by reviewing related work in Sec. 2, and then introduce both the distance measures as well as the domain adaptation models in Sec. 3-4. Finally, we present analyses on distance measures and experimental results in Sec. 5-7.

2 Related Work

Building an algorithm for domain adaptation is an open theoretical as well as practical problem (Blitzer, McDonald, and Pereira 2006; Pan and Yang 2010; Glorot, Bordes, and Bengio 2011; Blitzer, Kakade, and Foster 2011; Kulis, Saenko, and Darrell 2011; Saito et al. 2018; Kuroki et al. 2019; Lee et al. 2019).[1] When labeled data from target domain is available, supervised domain adaptation can achieve state-of-the-art results via fine-tuning, especially when source domain has orders of magnitude more data than target domain (Devlin et al. 2019; Radford et al. 2018). For unsupervised domain adaptation (no labels for target domains), there exist multiple approaches that have achieved remarkable progress, such as instance selection/reweighting (Huang et al. 2007; Gong, Grauman, and Sha 2013; Remus 2012) and feature space transformation (Pan et al. 2011; Baktashmotlagh et al. 2013). In this work we mainly focus on measuring domain discrepancy.

The works of Kifer, Ben-David, and Gehrke (2004), Ben-David et al. (2007), and Ben-David et al. (2010) provide an upper bound on the performance of a classifier under domain shift. They introduce the idea of training a binary classifier to distinguish samples from source/target domains, and the error $\mathcal{H}$-divergence provides an estimate of the discrepancy between domains. A tractable approximation, proxy $\mathcal{A}$-distance, applies a trained linear classifier to minimize a modified Huber loss (Ben-David et al. 2007).

Recent works further aim to provide more efficient estimates of the domain discrepancy. One popular choice is matching the distribution means in the kernel-reproducing Hilbert space (RKHS) (Huang et al. 2007; Gong, Grauman, and Sha 2013; Tzeng et al. 2014; Long et al. 2015; Bousmalis et al. 2016; Long et al. 2016; 2017; Zellinger et al. 2017; Rozantsev, Salzmann, and Fua 2018) using Maximum Mean Discrepancy (MMD) (Gretton et al. 2012). These methods have also been used in generative models (Li, Swersky, and Zemel 2015; Dziugaite, Roy, and Ghahramani 2015). Other methods explored in the literature include central moment discrepancy (CMD) (Zellinger et al. 2017), correlation alignment (CORAL) (Sun, Feng, and Saenko 2016; Sun and Saenko 2016), canonical correlation analysis (CCA) (Blitzer, Kakade, and Foster 2011), cosine similarity (Benaim and Wolf 2017). In addition to these directly-computable metrics, another successful approach is to encourage learned representations to fool a classifier whose goal is to distinguish samples from the source domain and target domain (Ganin et al. 2016; Shen et al. 2018).

When multiple domain adaptation criteria are available, Ruder and Plank (2017) use Bayesian optimization to decide the choice of metric, and Ying et al. (2018) use a meta-learning formulation. In our work, we provide a study of multiple domain distance measures (introduced in statistical learning/vision communities) in the context of NLP classification tasks such as sentiment analysis, where we analyze the domain-separability skills of these metrics and explore multiple ways of integrating them into the training dynamics (e.g., in the loss and as a multi-armed bandit).

Many problems can be cast as a multi-armed bandit problem. For example, Graves et al. (2017) use a multi-armed bandit (MAB) (Bubeck, Cesa-Bianchi, and others 2012) to learn a curriculum of tasks to maximize learning efficiency, Sharma and Ravindran (2017) use MAB to choose which domain of data to feed as input to a single model (in the context of Atari games), and Guo, Pasunuru, and Bansal (2019) use MAB for task selection during multi-task learning of text classification. In our work, we instead use a MAB controller with upper confidence bound (UCB) (Auer, Cesa-Bianchi, and Fischer 2002) for the task of multi-source domain selection for domain adaptation.

3 Domain Distance Measures

In Sec. 1, we described that domain adaptation performance is related to domain distance/dissimilarity. Here, we will first describe our individual distance measures. Then we will describe our mixture of distances. Later in Sec. 6, we will provide detailed analysis of these distance measures. Given source domain samples $X_s = \{x_1^s, x_2^s, ..., x_{n_s}^s\}$ as well as target domain samples $X_t = \{x_1^t, x_2^t, ..., x_{n_t}^t\}$, where we

[1] Due to AAAI page limit, we discuss the primary related work here, but we will add an extended version in the arxiv version.

assume $x \in \mathbb{R}^d$ are the embedding representations of the input data (e.g., sentences) produced from some feature extractors (e.g., LSTM-RNN), the goal of the distance measure is to estimate how different these two domains are. We will introduce five such methods: $\mathcal{L}_2$ distance, Cosine distance, Maximum Mean Discrepancy (MMD), Fisher Linear Discriminant (FLD), as well as CORAL.[2]

3.1 $\mathcal{L}_2$ Distance

The $\mathcal{L}_2$ distance measures the Euclidean distance between source domain and target domain samples. Define $\mu_s = \frac{1}{n_s}\sum_i x_i^s$ and $\mu_t = \frac{1}{n_t}\sum_i x_i^t$, the $\mathcal{L}_2$ distance is: $D_{\mathcal{L}_2}(X_s, X_t) = \|\mu_s - \mu_t\|_2$.

3.2 Cosine Distance

Cosine similarity is a measure of similarity between two vectors of an inner product space that measures the cosine of the angle of these vectors: $S_{\cos} = \frac{\mu_s \cdot \mu_t}{\|\mu_s\|_2 \|\mu_t\|_2}$, and cosine distance is $D_{\cos} = 1 - S_{\cos}$.

3.3 Maximum Mean Discrepancy (MMD)

Given two sets of source domain and target domain samples independently and identically distributed (i.i.d.) from $P_s(X)$ and $P_t(X)$, respectively. The statistical hypothesis testing is used to distinguish between the null hypothesis $\mathcal{H}_0$:P_s=P_t, and the alternative hypothesis $\mathcal{H}_A$:$P_s \neq P_t$ via comparing test statistic, which is described next. Maximum Mean Discrepancy or MMD (Gretton et al. 2012), also known as kernel two-sample test, is a frequentist estimator for answering the above question. MMD works by comparing statistics between the two samples, and if they are similar then they are likely to come from the same distribution. This is known as an integral probability metric (IPC) (Müller 1997) in statistics literature. Formally, let $\mathcal{F}$ be a class of functions $f : \mathcal{X} \to \mathbb{R}$, and the maximum mean discrepancy is:

$$\mathrm{MMD}_{\mathcal{F}}[P_s, P_t] = \sup_{f \in \mathcal{F}} \mathbb{E}_{x^s}[f(x^s)] - \mathbb{E}_{x^t}[f(x^t)]$$

Note that this equation involves a maximization over a family of functions. However, Gretton et al. (2012) show that when the function class $\mathcal{F}$ is the unit ball in a reproducing kernel Hilbert space (RKHS) endowed with a characteristic kernel k, this can be solved in closed form. A corresponding unbiased finite sample estimate is:

$$\mathrm{MMD}^2_{\mathcal{F}}[P_s, P_t] = \frac{1}{n_s^2}\sum_{i=1}^{n_s}\sum_{i'=1}^{n_s} k\left(x_i^s, x_{i'}^s\right) - \frac{2}{n_s n_t}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t} k\left(x_i^s, x_j^t\right) + \frac{1}{n_t^2}\sum_{j=1}^{n_t}\sum_{j'=1}^{n_t} k\left(x_j^t, x_{j'}^t\right)$$

For universal kernels like the Gaussian kernel $k(x, x') = \exp\left(-\frac{1}{2\sigma}|x - x'|^2\right)$ with bandwidth σ, minimizing MMD is analogous to minimizing a distance between all moments of the two distributions (Li, Swersky, and Zemel 2015). Here we will use $D_{\mathrm{MMD}}(X_s, X_t) = \mathrm{MMD}^2_{\mathcal{F}}[P_s, P_t]$.

3.4 Fisher Linear Discriminant

Fisher linear discriminant analysis (FLD) (Friedman, Hastie, and Tibshirani 2001) finds a projection (parameterized by w) where class separation is maximized. In particular, the goal of FLD is to give a large separation of class means while simultaneously keeping in-class variance small. This is formulated as $w^\star = \arg\max_w J(w) = \arg\max_w \frac{w^{\mathrm{T}} S_B w}{w^{\mathrm{T}} S_W w}$, where S_B is the between-class covariance matrix which is defined as $S_B = (\mu_s - \mu_t)(\mu_s - \mu_t)^T$, S_W is the within-class covariance matrix which is defined as $S_W = \sum_{c \in \{0,1\}} \sum_i (x_i^{(c)} - \mu_c)(x_i^{(c)} - \mu_c)^{\mathrm{T}}$, μ_c is the class mean and $\{0, 1\}$ here refers to source/target domain. The optimal $w^\star$ can be solved analytically as: $w^\star \propto S_W^{-1}(\mu_1 - \mu_2)$. Though the optimal $w^\star$ is usually desired, here we use the optimal J as a proxy of domain distance, and thus define our Fisher distance as $D_{\mathrm{FLD}}(X_s, X_t) = J(w^\star)$, which is a measure of difference between source/target representation means normalized by a measure of within-class scatter matrix. Note that computing the D_{FLD} is analogous to approximating the divergence between two domains by training an FLD to discriminate between unlabeled instances from source and target domains.

3.5 Correlation Alignment (CORAL)

The CORAL (correlation alignment) (Sun and Saenko 2016; Sun, Feng, and Saenko 2016) loss is defined as the distance between the second-order statistics of the source and target samples: $D_{\mathrm{CORAL}}(X_s, X_t) = \frac{1}{4d^2}\|C_s - C_t\|_F^2$, where $\|\cdot\|_F^2$ denotes the squared matrix Frobenius norm, d represents feature dimension, and C_s and C_t are the covariance matrices of source and target samples.

3.6 Mixture of Distances

As we will demonstrate in Sec. 6, no single distance measure outperforms all the others in our analyses. Also, note that while different distance measures provide different estimates of domain distances, each distance measure has its pathological cases. For example, samples from a Gaussian distribution and a Laplace distribution with same mean and variance might have small $\mathcal{L}_2$ distances even though they are different, whereas MMD can differentiate between them (Gretton et al. 2012). It is thus useful to consider a mixture of distances:

$$D_m(X_s, X_t) = \sum_k \alpha_k D_k(X_s, X_t) \tag{1}$$

where $\alpha_k \in \mathbb{R}$ is the coefficient for k-th distance. While appealing at first, naively adding all the distance measures to the mixture introduces unnecessary hyper-parameters. In Sec. 6.3, we will introduce simple unsupervised criteria to only include a subset of these distance measures.

4 Models

We will first describe the baseline and our DistanceNet model (based on a single source domain) which actively

[2] We also experimented with proxy $\mathcal{A}$-distance from Ben-David et al. (2007), which scored favorably on most of our evaluations. However, due to its non-differential nature as well as high computation cost, we do not include it here.

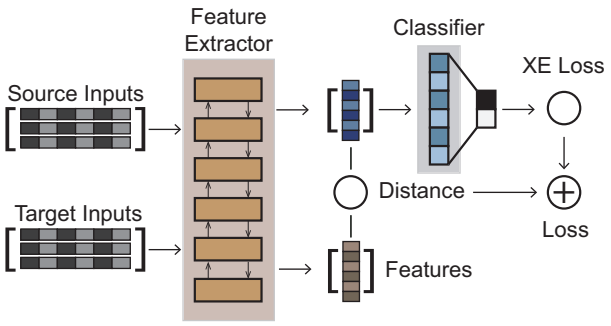

Figure 1: Overview of the DistanceNet model. The model takes both the source domain data (labeled) and target domain data (unlabeled), and computes feature representations. The distances, calculated via the distance measures, between the source and target samples are added to the (source) cross-entropy loss to be minimized jointly during the training.

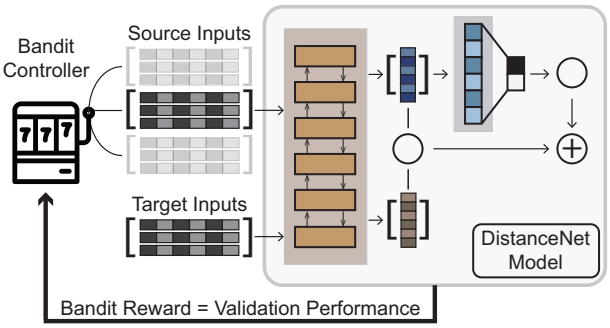

Figure 2: Overview of our multi-source DistanceNet model with controller. During the training, a multi-armed bandit controller dynamically selects the source domain from a set of candidate source domains. The controller updates its belief over the utility of each domain via receiving feedback on validation set.

minimizes the distance between the source and target domain during the model training for domain adaptation. Then we introduce the multi-source variant of DistanceNet that additionally utilizes a multi-armed bandit controller to learn a dynamic curriculum of multiple source domains for training a domain adaptation model.

4.1 Baseline Model

Given a sequence of tokens $\{w_0, w_1, ..., w_T\}$, we first embed these tokens into vector representations $\{e_0, e_1, ..., e_T\}$. Let $h_T = \text{LSTM}(\{e_t\}, \theta_1)$ be the output of the LSTM-RNN parameterized by θ_1. The probability distribution of labels is produced by $\hat{y} = \text{FC}(h_T, \theta_2)$, where FC is a fully connected neural network with parameters θ_2. The model is trained to minimize the cross entropy between predicted outputs $\hat{y}$ and ground truth y with N training examples and C classes: $L_{\text{XE}}(\hat{y}, y) = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j} \log \hat{y}_{i,j}$.

4.2 DistanceNet

The work of Ben-David et al. (2010) shows that domain adaptation performance is related to source domain performance and source/target domain distance. The first part (source domain performance) is already handled by the cross entropy loss (Sec. 4.1), and it is thus natural to additionally encourage the model to minimize the representational distances between source and target samples. To that end, we augment the classification task's loss function with a domain distance term. Given a sequence of tokens from the source domain $\{w_0^s, w_1^s, ..., w_{T_s}^s\}$, a sequence of tokens from the target domain $\{w_0^t, w_1^t, ..., w_{T_t}^t\}$, and model parameterized by (θ_1, θ_2), the new loss function for our DistanceNet (see Fig. 1) is then:

$$L(\hat{y}^s, y^s) = L_{\text{XE}}(\hat{y}^s, y^s) + \beta D_k(h_{T_s}^s, h_{T_t}^t) \qquad (2)$$

where $\hat{y}^s, y^s$ are the predicted and ground truth outputs of source domain, $h_{T_s}^s, h_{T_t}^t$ are the representations of source and target domain, and D_k is the choice of distance measure from Sec. 3.

4.3 Dynamic Multi-Source DistanceNet using Multi-Armed Bandit

In the previous section, we described our method for fitting a model on a pair of source/target domains. However, when we have access to multiple source domains, we need a better way to take advantage of these extra learning signals. One simple method is to treat these multiple source domains as a single (big) source domain, and apply algorithms described previously as usual. But as the model representation changes throughout the training, the domain that can provide the most informative training signal might change over time and based on the training curriculum history. This is also related to learning importance weights (Ben-David et al. 2010) of each source domain over time for the target domain. Thus, it might be more favorable to dynamically select the sequence of source domains to deliver the best outcome on the target domain task.

Here, we introduce a novel multi-armed bandit controller for dynamically changing the source domain during training (Fig. 2). We model the controller as an M-armed bandits (where M is the number of candidate domains) whose goal is to select a sequence of actions/arms to maximize the expected future payoffs. At each round, the controller selects an action (candidate domain) based on noisy value estimates and observes a reward. More specifically, as the training progresses, the controller picks one of the training domains and have the task model train on the selected domain using the loss function specified in Eq. 2, and the performance on the validation data will be used as the reward provided to the bandit as feedback. We use upper confidence bound (UCB) (Auer, Cesa-Bianchi, and Fischer 2002) bandit algorithm, which chooses the action (i.e., the source domain to use next) based on the performance upper bound: $a_t^{\text{UCB}} = \arg\max_{a \in \mathcal{A}} Q(a) + \sqrt{\frac{2 \log t}{N_t(a)}}$, where a_t represents the action at iteration time t, $N_t(a)$ counts the number of times the action has been selected, and $\mathcal{A}$ represents the set of candidate actions (i.e., the set of candidate source domains). $Q(a)$ represents the action-value of the action, and

is calculated as the running average of rewards.[3]

5 Experimental Setup

Dataset: We evaluate our methods using the datasets collected by Liu, Qiu, and Huang (2017)[4], which contains 16 datasets of product reviews (Blitzer, Dredze, and Pereira 2007) and movie reviews (Maas et al. 2011; Pang and Lee 2005), where the task is to classify these reviews as positive or negative. The performance of a model on this task is measured by accuracy. Since the number of experiments scales $\mathcal{O}(n^2)$ and $\mathcal{O}(n)$ for single- and multi-source experiments, we only evaluate on 3 and 5 datasets[5] for experiments in Sec. 7, respectively.[6] However, we still use the full set of domains for the analysis in Sec. 6.

Training Details: Our baseline model is similar to that of Liu, Qiu, and Huang (2017). We use a single-layer Bidirectional LSTM-RNN as sentence encoder and a two-layer fully-connected with ReLU non-linearity layer to produce the final model outputs. The word embeddings are initialized with GloVe (Pennington, Socher, and Manning 2014). We train the model using Adam optimizer (Kingma and Ba 2014). Following Ruder and Plank (2017) and Bousmalis et al. (2016), we chose to use a small number of target domain examples as validation set (for both tuning as well as providing rewards for the multi-armed bandit controller).[7] We use the adaptive experimentation platform Ax[8] to tune the rest of the hyperparameters and the search space for these hyperparameters are: learning rate $\in (10^{-4}, 10^{-3})$, dropout rate $\in (0.25, 0.75)$, $\beta \in (0.01, 1.0)$, and $\alpha_k \in (0.0, 1.0)$. We run each model for 3 times. We use the average validation performance as our validation criteria, and report average test performance.

6 Analysis of Distance Measures

Given our 5 distance measures (described in Sec. 3), we first want to ask which of these distance measures are able to measure domain (dis)similarities. Specifically, we conduct experiments to answer the following questions:

Q1. Is the distance measure able to differentiate samples from the same versus different domains?

Q2. Does the distance measure correlate well with empirical results?

These two questions are answered next in Sec. 6.1 and Sec. 6.2, respectively. After that, we will describe our unsupervised criteria for choosing a subset of distance measures (Sec. 6.3) to be used in the mixture of distance measures introduced in Sec. 3.6.

6.1 Domain Separability Test

Given two sets of source and target domain samples: $X_s{=}\{x_1^s, x_2^s, ..., x_{n_s}^s\}$ and $X_t{=}\{x_1^t, x_2^t, ..., x_{n_t}^t\}$, which are independently and identically distributed (i.i.d.) from $P_s(X)$ and $P_t(X)$, respectively. The goal here is to find whether these samples come from the same domain or not. For this, we compute the distance between the source and target samples, $d_k(P_s, P_t)$, via distance measure D_k (selected from the distance measures defined in Sec. 3):

$$d_k(P_s, P_t) = \mathop{\mathbb{E}}_{x_s \sim P_s, x_t \sim P_t}[D_k(x_s, x_t)] \tag{3}$$

For distance measure to estimate domain similarity, we expect $d_k(P_s, P_t)$ to be low when $P_s{=}P_t$, and high otherwise (similar to two sample test statistic (Gretton et al. 2012)).

Fig. 3 visualizes the results of our experiments, where the distance between exhaustive source/target domain pairs are measured on 16 datasets. We take 200 examples from each domain[9], and embed the sentences using pre-trained model[10], after which the distances are calculated. In particular, the entries on the diagonal refer to the in-domain distances (i.e., source and target domain is the same), and off-diagonal entries refer to the between-domain distances. As we want the in-domain distances to be small and between-domain distances to be large, we expect the visualization of a good distance measure to have a dark line on the diagonal (indicating low values) and bright otherwise. From the visualization plots (Fig. 3), we can see that $D_{\mathcal{L}_2}, D_{\cos}, D_{\mathrm{MMD}}$ and D_{FLD}, are able to separate domains well. However, all these measures have different scales and sensitivity, hence, we next define two statistics to quantitatively compare different distance measures d_k, which are denoted by z_1 and z_2 corresponding to method-1 and method-2, respectively. These statistics are shown in Table. 1. We can see that most of these methods are able to separate domains, with the exception of D_{CORAL}. Next, we describe these methods.

Method-1. The first method assess whether distances between samples from the same domain are lower than those between the samples from different domains, $d_k(P_i, P_i) \leq d_k(P_i, P_j) \forall i \neq j$. This statistic is appealing because it is invariant to scaling and translation, but does not concern how smaller in-domain distances are w.r.t. off-domain distances.

[3] One could also consider weighting each domain based on the distances, but these keep changing as DistanceNet's training evolves (which minimizes the distance). Further, our bandit decides the arm to pull based on DistanceNet's performance, thus already behaving similar to the distance-weighting approach (while also automatically learning these weights as a curriculum).

[4] The datasets include "unlabeled" split.

[5] MR, Apparel, Baby for single-source experiments. MR, Apparel, Baby, Books, Camera for multi-source experiments.

[6] Note that for n tasks, there will be $n{\times}(n-1)$ source/target domain pairs experiments, and n multi-source/single-target domain pairs experiments.

[7] Note that the two models in Table 5 should be fairly comparable, since they have access to the same validation dataset for tuning or "refining" their hyper-parameters or as weak reward feedback. Further, there are scenarios in which querying the scalar rewards on a small validation dataset is easier than accessing the rich gradient information through them (Bousmalis et al. 2016).

[8] https://github.com/facebook/Ax

[9] We take source domain samples from the training set and target domain samples from the validation set to avoid overlapping examples when sampling from the same domain.

[10] https://tfhub.dev/google/tf2-preview/nnlm-en-dim128/1

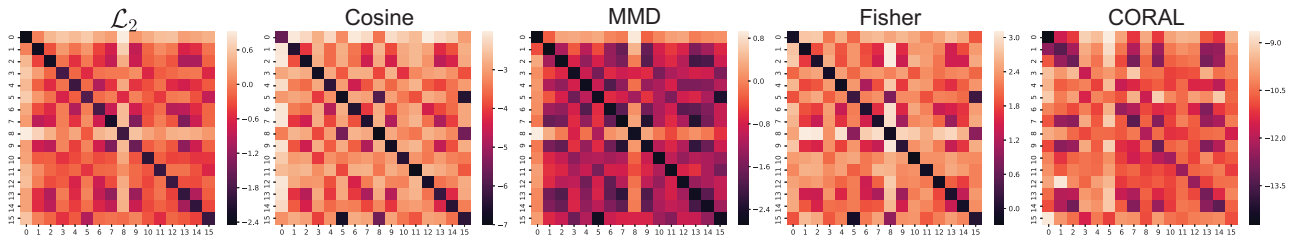

Figure 3: Domain separability test of distance methods. The order is (left to right): $D_{\mathcal{L}_2}, D_{\cos}, D_{\text{MMD}}, D_{\text{FLD}}, D_{\text{CORAL}}$. The value of each entry at position (i, j) refers to the distance between samples from i-th source domain and j-th target domain. In particular, the entries on the diagonal refer to the in-domain distances, and off-diagonal entries refer to the between-domain distances. Values shown are the $\log$ of the distances for visualization purposes.

Name	Method-1	Method-2	Result-Corr
$\mathcal{L}_2$	1.00	6.35×10^{-3}	0.67
Cosine	0.94	7.49×10^{-3}	0.79
MMD	0.94	6.83×10^{-3}	0.59
Fisher	0.88	5.99×10^{-3}	0.65
CORAL	0.75	9.11×10^{-3}	0.39

Table 1: Distance comparison statistics and result-correlations. Note that for Method-1 and Result-Corr, higher numerical values are better, however, for Method-2, lower numerical values are better.

Specifically, we compute the z_1 as:

$$z_1(d_k) = \frac{1}{N} \sum_i \mathbb{I}\big[d_k(P_i, P_i) \leq d_k(P_i, P_j) \wedge d_k(P_i, P_i) \leq d_k(P_j, P_i) \forall i \neq j\big]$$

We can see that $D_{\mathcal{L}_2}$ achieves the highest score, whereas D_{CORAL} achieves the lowest.

Method-2 The second method assesses how smaller the value of $d_k(P_i, P_i)$ is in comparison to $d_k(P_i, P_j) \forall i \neq j$. To compute this, we first standardize[11] the matrix $\{d_k(P_i, P_j), \forall i, j\}$, and then apply softmax function to ensure that all entries are positive. Then we compute the sum of the diagonal entries of the transformed matrix $\{d'_k(P_i, P_j), \forall i, j\}$ as our second quantitative assessment (z_2, note that smaller is better):

$$z_2(d_k) = \sum_i d'_k(P_i, P_i) \tag{4}$$

We can see that D_{FLD} obtains the lowest/best value, and D_{CORAL} scores the largest value.

6.2 Correlation With Results

The methods described previously answer the question of whether $P_s{=}P_t$ given the samples X_s and X_t. However, the assessment we are interested in ultimately is whether the distance measures correlate with the true domain distances. As the true domain distance is latent, here we will use a proxy. We denote $r(P_s, P_t)$ as the performance of the baseline model trained on the source domain and evaluated on the target domain. We want to measure the correlation between $d_k(P_s, P_t)$ and $r(P_s, P_t)$. Specifically, we train and evaluate baseline models on all source/target domain pairs, and then compute the Pearson correlation coefficient between the results (averaged over three runs) and distance measures. The values are shown in Table 1, where we can see that most of the distance measures are correlated with actual performance, with D_{CORAL} having the lowest correlation with empirical performance (hence we ignore D_{CORAL} for all future experiments, given that it is the worst by large margins on all 3 analysis methods above).

6.3 Informativeness of Mixture Components

Lastly, we present the basis for deciding which distances to (not) include in the mixture formulation described in Sec. 3.6. Specifically, our goal is to remove redundant distance measures from the mixture, subject to the constraint that the reduced mixture still provides sufficient information about the distances between two domains. We approach this problem via estimating the 'informativeness' of each distance measure. This is analogous to influence functions, a classic technique from robust statistics (Cook and Weisberg 1980; Koh and Liang 2017). To motivate our approach, let's say our mixture $\{D_k\}_{k=1}^K$ includes all distance measures which are previously defined (Sec, 3). Suppose we have a function $\phi(\{D_k\}_{i=1}^K)$ which can give us an estimate of the quality of the mixture. Now, we proceed by removing one metric (say D_m) from the mixture and apply the function ϕ to give us an estimate of the quality of the reduced mixture, $\phi(\{D_k\}_{k\neq m}^K)$. We can now define an estimate of distance measure's informativeness:

$$\mathcal{I}(D_m) = \phi(\{D_k\}_{i=1}^K) - \phi(\{D_k\}_{i\neq m}^K) \tag{5}$$

If $\mathcal{I}(D_m)$ is small, we can say the removed metric is not informative given other components in the mixture. Here, we use the optimal z_2 statistics[12] (which is unsupervised)

[11] Subtracting the means and normalize by standard deviations.

[12] We do not use z_1 because it is not differentiable (calculated as multiple binary comparisons), and z_1 already achieved almost-maximum scores (Table 1) thus making the optimization less useful. Also, since we evaluate using an unsupervised criterion, we decided not to use correlation because it is a supervised evaluation.

Name	Informativeness Estimate
$\mathcal{L}_2$	-2.086 $\times 10^{-3}$
Cosine	-0.001 $\times 10^{-3}$
MMD	-1.775 $\times 10^{-3}$
Fisher	-0.024 $\times 10^{-3}$

Table 2: Estimated importance of each distance measure as a component in the mixture. Values are negative because the full mixture achieves the lowest z_2. Lower (more negative) value means the distance measure provides more information. We did not include D_{CORAL} because it scored unfavorably in our previous assessments.

Model	MR	Aprl	Baby	Books	Camera
Liu (2017)	74.7	86.0	83.5	81.0	86.0
Ours	73.8	87.2	85.2	81.4	88.1

Table 3: Performance of our baseline compared with previous work (Liu, Qiu, and Huang 2017).

defined in Sec. 6.1 as the mixture evaluation function:

$$\phi(\{D_k\}_{i=1}^K) = \max_{\alpha_1,\ldots,\alpha_k} z_2\Big(\sum_k \alpha_k D_k(X_s, X_t)\Big)$$

where we estimate the maximum value using gradient descent (via the JAX library). We found that removing $D_{\cos}$ has far lower impact on the optimal z_2, and thus in our experiments using mixture of distances, we do not include $D_{\cos}$ (see Table 2 for detailed scores of informativeness for all the distance measures).

7 DistanceNet and Bandit Results

In this section, we show domain-adaptation experimental results for the sentiment classification task on the target domain (using out-of-domain source training data). We start with comparing our (in-domain) baseline to previous work, where the source and target domain are the same. Then we will show the results of our DistanceNet (with both single distance and mixture-of-distance measures), when the source domain and target domain is different. Lastly, we will show the results of our multi-source DistanceNet baseline versus our multi-source DistanceNet bandit model which dynamically selects source domains. Based on the results of Sec. 6, we do not include D_{CORAL} in our DistanceNet experiments, and do not include both D_{CORAL} and $D_{\cos}$ in our DistanceNet with mixture-of-distance experiments.

7.1 Single Source DistanceNet Results

Baseline Results. In Table 3 we show the results of our (in-domain) baseline compared with similar models in Liu, Qiu, and Huang (2017). We can see that our baseline is stronger than comparable previous work in four of the five domains we considered.

DistanceNet Results. Table 4 shows the results of baselines and DistanceNet models when the source and target domain is different, where the last column shows the average results.[13] First, comparing the numbers to those in Table 3, we can see that performance drops when there is a shift in the data distribution. Next, we can see that by adding our domain distance measure as an additional loss term, the model is able to reduce the gap between in-domain performance and out-of-domain performance. In particular, all of our models perform better than our baseline in terms of average results, with MMD model better than the baseline by one corresponding standard deviation.[14]

Source	MR(M)		Aprl(A)		Baby(B)		Avg
Target	A	B	M	B	M	A	
DataSel	68.1	65.2	64.3	74.3	65.6	78.9	69.39
DANN	69.9	65.3	63.7	78.2	65.5	80.0	70.46
Baseline	67.3	66.5	65.8	78.2	64.6	78.1	70.08
$\mathcal{L}_2$	70.9	66.5	64.7	76.6	65.3	78.2	70.37
Cosine	70.2	66.2	64.6	78.3	65.3	78.2	70.48
MMD	69.9	67.1	64.3	77.1	66.0	78.1	70.42
Fisher	69.1	64.2	64.6	77.9	65.4	79.4	70.10
Mixture	70.4	67.1	65.6	79.0	66.5	79.3	71.32

Table 4: Performance comparison of previous works (DataSel: Remus (2012) ; DANN: Ganin et al. (2016)), single-source baseline, and DistanceNet models.

Mixture DistanceNet Results. Table 4 shows the results of our DistanceNet with mixture of distance measures experiments. From the results, we can see that leveraging the power of multiple distance measures additionally improves the results in out-of-domain settings, and achieving the highest average results (better than baseline by two standard deviations). We also compare our DistanceNet models to other domain-adaptation approaches. DANN encourages similar latent features by augmenting the model with a few standard layers and a new gradient reversal layer (Ganin et al. 2016). DataSel instead relies on data selection based on domain similarity and complexity variance (Remus 2012). From the results, we can see that our DistanceNet with mixture of distance measures outperforms these approaches (better w.r.t. standard deviation margins).

7.2 Multi-Source DistanceNet-Bandit Results

Table 5 shows the results for our multi-source experiments, where the source domains include all but the target domain, thus we have one result for each target domain. Here the baseline is the DistanceNet with mixture of distance measures, which selects domains in a round-robin fashion. Our model instead applies a dynamic controller to select the

[13] Note that the single-distance methods, e.g., MMD, have been used in previous works (Bousmalis et al. 2016; Tzeng et al. 2014; Benaim and Wolf 2017) and can also be considered as baselines.

[14] To calculate the standard deviation of the average results, we first compute the average results for each run, and compute the standard deviation of the average results. This is equivalent to computing the standard deviation of a single large prediction by concatenating model outputs for all tasks as a single output.

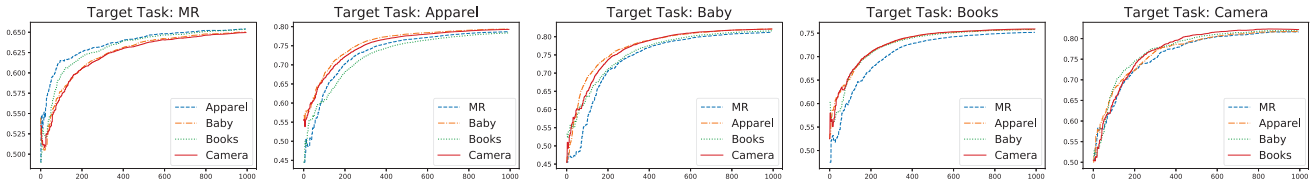

Figure 4: Visualization examples of multi-armed bandits. Each line represents an arm (a source domain), X-axis refers to time, and Y-axis refers to the values of each arm (higher value of an arm corresponds to potentially more usefulness of the task).

Model	MR	Aprl	Baby	Books	Camera	Avg
Mixture	69.8	80.8	82.5	77.0	80.9	78.20
+Bandit	72.0	82.3	82.8	78.3	81.3	79.30

Table 5: Multi-source DistanceNet versus bandit.

source domain to use. We can see from the results that using the dynamic controller improves the individual results, and the average results (better by two standard deviations).[15] In general, we observed that the bandit always improves over the non-bandit baseline (with two std. deviations) even when we simply reuse the best hyperparameters found in the single-source experiments, and when we employ a bandit without the DistanceNet loss (i.e., just cross-entropy).

7.3 Multi-Armed Bandit Visualization

Fig. 4 provides example visualizations of the usefulness of each source domain for a given target domain during the training trajectory of multi-source bandit experiments. We provide a brief summary of our observations from these examples here. When the target task is "MR", we observed that "Books" and "Apparel" are more beneficial. When the target task is "Apparel", we found that "Camera" as well as "Baby" are beneficial; moreover, there the bandit learns to switch between "Books" and "MR" over time. When the target task is "Baby", we see that "Camera" and "Apparel" are beneficial. When "Books" is the target task, we found that "MR" seemed to be less helpful. Finally, when the target-task is "Camera", we see that "Books" had the highest value.

8 Conclusion

In this work, we presented a study of multiple domain distance measures to address the problem of domain adaptation. We provided analyses of these measures based on their ability to separate same/different domains and correlation with results. Next, we introduced our model, DistanceNet, which augments the loss function with the distance measures. Later, we extended our DistanceNet to the multi-source setup via a multi-armed bandit controller. Our experiment results suggest that our DistanceNet, as well as its variant with the multi-armed bandit, is able to outperform corresponding baselines.

[15]Our single-source experiments suggested that "MR" and "Books" are not helpful for the learning of the other three tasks, thus we mask the DistanceNet loss from these domains when the target domain is not "MR" or "Books".

Acknowledgments

We thank the reviewers and Boyang Li for their helpful comments. This work was supported by DARPA (YFA17-D17AP00022), NSF-CAREER Award #1846185, ONR Grant #N00014-18-1-2871, Google, Facebook, Baidu, and Salesforce. The views contained in this article are those of the authors and not of the funding agency.

References

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time analysis of the multiarmed bandit problem. *Machine learning*.

Baktashmotlagh, M.; Harandi, M. T.; Lovell, B. C.; and Salzmann, M. 2013. Unsupervised domain adaptation by domain invariant projection. In *ICCV*.

Ben-David, S.; Blitzer, J.; Crammer, K.; and Pereira, F. 2007. Analysis of representations for domain adaptation. In *NeurIPS*.

Ben-David, S.; Blitzer, J.; Crammer, K.; Kulesza, A.; Pereira, F.; and Vaughan, J. W. 2010. A theory of learning from different domains. *Machine learning*.

Benaim, S., and Wolf, L. 2017. One-sided unsupervised domain mapping. In *NeurIPS*.

Blitzer, J.; Dredze, M.; and Pereira, F. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*.

Blitzer, J.; Kakade, S. M.; and Foster, D. P. 2011. Domain adaptation with coupled subspaces. In *AISTATS*.

Blitzer, J.; McDonald, R.; and Pereira, F. 2006. Domain adaptation with structural correspondence learning. In *EMNLP*.

Bousmalis, K.; Trigeorgis, G.; Silberman, N.; Krishnan, D.; and Erhan, D. 2016. Domain separation networks. In *NeurIPS*.

Bubeck, S.; Cesa-Bianchi, N.; et al. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*.

Cook, R. D., and Weisberg, S. 1980. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*.

Daumé III, H. 2009. Frustratingly easy domain adaptation. *arXiv:0907.1815*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Dziugaite, G. K.; Roy, D. M.; and Ghahramani, Z. 2015. Training generative neural networks via maximum mean discrepancy optimization. In *UAI*.

Friedman, J.; Hastie, T.; and Tibshirani, R. 2001. *The elements of statistical learning*. Springer.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *JMLR*.

Glorot, X.; Bordes, A.; and Bengio, Y. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*.

Gong, B.; Grauman, K.; and Sha, F. 2013. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*.

Graves, A.; Bellemare, M. G.; Menick, J.; Munos, R.; and Kavukcuoglu, K. 2017. Automated curriculum learning for neural networks. In *ICML*.

Gretton, A.; Borgwardt, K. M.; Rasch, M. J.; Schölkopf, B.; and Smola, A. 2012. A kernel two-sample test. *JMLR*.

Guo, H.; Pasunuru, R.; and Bansal, M. 2019. AutoSeM: Automatic task selection and mixing in multi-task learning. In *NAACL*.

Huang, J.; Gretton, A.; Borgwardt, K.; Schölkopf, B.; and Smola, A. J. 2007. Correcting sample selection bias by unlabeled data. In *NeurIPS*.

Kifer, D.; Ben-David, S.; and Gehrke, J. 2004. Detecting change in data streams. In *VLDB*.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv:1412.6980*.

Koh, P. W., and Liang, P. 2017. Understanding black-box predictions via influence functions. In *ICML*.

Kulis, B.; Saenko, K.; and Darrell, T. 2011. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*.

Kuroki, S.; Charoenphakdee, N.; Bao, H.; Honda, J.; Sato, I.; and Sugiyama, M. 2019. Unsupervised domain adaptation based on source-guided discrepancy. In *AAAI*.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *nature*.

Lee, J.; Charoenphakdee, N.; Kuroki, S.; and Sugiyama, M. 2019. Domain discrepancy measure using complex models in unsupervised domain adaptation. *arXiv:1901.10654*.

Li, Y.; Swersky, K.; and Zemel, R. 2015. Generative moment matching networks. In *ICML*.

Liu, P.; Qiu, X.; and Huang, X. 2017. Adversarial multi-task learning for text classification. In *ACL*.

Long, M.; Cao, Y.; Wang, J.; and Jordan, M. 2015. Learning transferable features with deep adaptation networks. In *ICML*.

Long, M.; Zhu, H.; Wang, J.; and Jordan, M. I. 2016. Unsupervised domain adaptation with residual transfer networks. In *NeurIPS*.

Long, M.; Zhu, H.; Wang, J.; and Jordan, M. I. 2017. Deep transfer learning with joint adaptation networks. In *ICML*.

Maas, A. L.; Daly, R. E.; Pham, P. T.; Huang, D.; Ng, A. Y.; and Potts, C. 2011. Learning word vectors for sentiment analysis. In *ACL*.

Müller, A. 1997. Integral probability metrics and their generating classes of functions. *ADV APPL PROBAB*.

Pan, S. J., and Yang, Q. 2010. A survey on transfer learning. *TKDE*.

Pan, S. J.; Tsang, I. W.; Kwok, J. T.; and Yang, Q. 2011. Domain adaptation via transfer component analysis. *TNNLS*.

Pang, B., and Lee, L. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL*.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training. *OpenAI Blog*.

Remus, R. 2012. Domain adaptation using domain similarity-and domain complexity-based instance selection for cross-domain sentiment analysis. In *ICDM workshop*.

Rozantsev, A.; Salzmann, M.; and Fua, P. 2018. Beyond sharing weights for deep domain adaptation. *TPAMI*.

Ruder, S., and Plank, B. 2017. Learning to select data for transfer learning with bayesian optimization. In *EMNLP*.

Saito, K.; Watanabe, K.; Ushiku, Y.; and Harada, T. 2018. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*.

Sharma, S., and Ravindran, B. 2017. Online multi-task learning using active sampling. *CoRR*.

Shen, J.; Qu, Y.; Zhang, W.; and Yu, Y. 2018. Wasserstein distance guided representation learning for domain adaptation. In *AAAI*.

Sun, B., and Saenko, K. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *ECCV*.

Sun, B.; Feng, J.; and Saenko, K. 2016. Return of frustratingly easy domain adaptation. In *AAAI*.

Tzeng, E.; Hoffman, J.; Zhang, N.; Saenko, K.; and Darrell, T. 2014. Deep domain confusion: Maximizing for domain invariance. *arXiv:1412.3474*.

Tzeng, E.; Hoffman, J.; Saenko, K.; and Darrell, T. 2017. Adversarial discriminative domain adaptation. In *CVPR*.

Ying, W.; Zhang, Y.; Huang, J.; and Yang, Q. 2018. Transfer learning via learning to transfer. In *ICML*.

Zellinger, W.; Grubinger, T.; Lughofer, E.; Natschläger, T.; and Saminger-Platz, S. 2017. Central moment discrepancy (cmd) for domain-invariant representation learning. *arXiv:1702.08811*.