Domain Adaptation Using Small Group Learning

Aadil Ahamed
UC San Diego
aahamed@ucsd.edu

JiMin Mun
UC San Diego
jmmun@ucsd.edu

John O'Boyle
UC San Diego
jboboyle@ucsd.edu

Alex Tung
UC San Diego
aktung@ucsd.edu

Abstract

Domain adaptation is an unsupervised deep learning technique aimed to extract a meaningful representation of a target domain $\mathcal{D}^T$ using a model previously trained on a source domain $\mathcal{D}^S$. The method relies on the assumption that features common to both $\mathcal{D}^S$ and $\mathcal{D}^T$ have invariant properties. The existence of such features allows for disentanglement of the underlying factors of variation across the source and target domains. Different approaches to domain adaption have been implemented with varying degrees of success. In the current work, we will explore the small group learning approach in domain adaptation and evaluate its performance.

1 What research problem are you aiming to solve?

Transfer learning broadly describes the application of a trained neural network to another dataset. This technique can be classified by whether the adaptation is performed over the learning domain (i.e. feature space and distribution) or the learning task (i.e. label space, objective function)[1], of which we are interested in exploring the first. Domain adaptation transfers a model trained to a set task $\mathcal{T}$ on a source domain $\mathcal{D}^S$ to the same task on a different domain (target domain) $\mathcal{D}^T$ with the same feature space but a different marginal distribution of features [2].

2 Why is this problem important?

Domain adaptation is important as source and target distributions can differ in practice even when performing a conventional learning task with a constant dataset. Though network performance can be predicted by portioning training data into validation and test groups, the network may still suffer if deployed in an environment that diverges from the entire set[1]. As its name implies, optimization of a network using domain adaptation methods can improve its external validity to adapt to different domains, which in turn improves its robustness. With regard to supervised learning research in new fields, it can be expensive to label large amounts of data from the target domain if there are no existing datasets to train from – a problem that domain adaptation can also address.

Small group learning (SGL hereafter) is a newly proposed search method in which multiple networks, termed learners, are first independently trained on a labeled dataset, then cross-trained with datasets with pseudo-labels from other learners, and finally optimized on a validation set [3]. Following the terminology given above, this procedure is considered a semi-supervised and transductive form of domain adaptation because the intermediate step involves labels generated locally by each learner.

Given the nature of domain adaptation in exploring datasets that a given network has not been exposed to, any expansion in the understanding of SGL and its implementation would develop the relatively young field. In addition, the original authors implemented this topology in the context of a DARTS architecture search [4], but it would be interesting to explore different weight initializations

[1] Under the Bayesian formulation of learning problems, this divergence is most commonly characterized as a change in prior or conditional probability distribution.

Preprint. Under review.

to examine whether individual learners arriving at local loss minima can further converge towards a global minimum when cross-trained in a small group. This is of particular interest when optimizing non-convex problems.

3 Related Work

Domain adaptation was originally applied to the problem of sentiment classification across text reviews of different goods [5], but the principle has since expanded to include other sub-approaches. In their survey, Wilson et al. broadly classify these approaches into the following categories: domain invariant feature learning, domain mapping, normalization statistics and ensemble methods [1]. We will focus primarily on domain invariant feature learning (DIFL hereafter) as it relates closely to the discussions in the formulation of SGL. DIFL attempts to extract features from the input that generalize well to different domains [1]. The following sections summarize three common feature extraction methods used in DIFL.

The divergence method aims to minimize the distance between source and target distributions based on some metric. Common metrics used in this approach are maximum mean discrepancy, correlation alignment, contrastive domain discrepancy and the Wasserstein metric [1] [6].

The reconstruction method attempts to build a feature representation that can both classify labels against data from the source domain and reconstruct the original input from the source and target domains. There have been broad efforts made to adapt autoencoders to different domains altogether without necessarily keeping the feature spaces constant [7].

The adversarial learning method consists of a feature extractor and domain classifier acting as adversaries, as the name suggests. The domain classifier tries to classify whether an input sample originated from the source or target domain and the feature extractor tries to extract domain invariant features that make it difficult for the domain classifier to classify correctly [8].

4 Proposal

We propose to try applying the SGL learning framework to the problem of domain adaptation. Specifically, this learning framework can be used in combination with some existing models that have worked well for domain adaptation. For example, the two-learner system in the original SGL framework could be adapted, where each learner uses an adversarial model with different architectures and weights.

5 Novelty

Our proposal will be the first attempt to tackle the problem of domain adaptation using SGL. The original authors have shown that SGL can perform better than state of the art models in tasks such as image classification in conjunction with architecture search and it is plausible that it can provide strong results for different domains as well. It also may be interesting to evaluate the performance of SGL in a strictly transductive sense by using the framework to extract features of other image-like inputs (e.g. audio spectrograms, GIS data), though it will be outside the scope of this study.

6 Tentative Timeline

Refer to project proposal for timeline.

7 Conceptualization of Methods

7.1 DANN: Domain Adversarial Neural Network

One possible learner model is an adversarial network composed of a label classifier G_y, a binary domain classifier G_d and a feature extractor G_f [9]. For input space X, feature space f, label space Y, and domain label space $d \in \{source, target\}$ the classifiers can be expressed in function notation as follows:

$$G_f : X \mapsto f, \quad G_y : f \mapsto Y, \quad G_d : f \mapsto d$$

This network has two learning objectives. The first objective is preserved from the original SGL objective, which is to minimize the label prediction loss L_y. The second objective is to make the extracted features f domain invariant. More formally, we want to make the distribution of features extracted for the source domain $S(f) = \{G_f(x, \theta_f)|x \sim S(x)\}$ similar to the distribution of features extracted from the target domain $T(f) = \{G_f(x, \theta_f)|x \sim T(x)\}$. The performance of an optimized domain classifier G_d, is used as a measure of similarity between the two feature distributions: A domain classifier that performs poorly indicates that the extracted features are domain invariant, while one that performs well indicates that they are not. Hence, the second learning objective requires G_f and G_d to act as adversaries: The system searches for parameters θ_f that maximize L_d while simultaneously searching for parameters θ_d that minimize L_d. Combining these two learning objectives yields the following objective function to be optimized:

$$E(\theta_f, \theta_y, \theta_d) = \sum_{\substack{i=1\ldots N, \\ d_i = source}} L_y^i(\theta_f, \theta_y) - \lambda \sum_{i=1\ldots N} L_d^i(\theta_f, \theta_d) \tag{1}$$

Specifically, we want to find the saddle point of $E(\theta_f, \theta_y, \theta_d)$:

$$\theta_f, \theta_y = \underset{\theta_f, \theta_y}{\operatorname{argmin}} E(\theta_f, \theta_y, \theta_d) \qquad \theta_d = \underset{\theta_d}{\operatorname{argmax}} E(\theta_f, \theta_y, \theta_d) \tag{2}$$

The stochastic gradient descent update for the parameters then becomes:

$$\theta_f \leftarrow \theta_f - \mu\left(\frac{\partial L_y^i}{\partial \theta_f} - \lambda \frac{\partial L_d^i}{\partial \theta_f}\right) \qquad \theta_y \leftarrow \theta_y - \mu \frac{\partial L_y^i}{\partial \theta_y} \qquad \theta_d \leftarrow \theta_d - \mu \frac{\partial L_d^i}{\partial \theta_y} \tag{3}$$

where μ is the learning rate and the loss functions L_y and L_d are multinomial and binomial cross entropy losses, respectively. The $-\lambda \frac{\partial L_d^i}{\partial \theta_f}$ term in the update for θ_f in Eq. 3 can be implemented through a gradient reversal layer. This layer can be represented mathematically as a pseudo-function $R_\lambda(x)$:

$$R_\lambda(x) = x \qquad \frac{\mathrm{d}R_\lambda(x)}{\mathrm{d}x} = -\lambda \tag{4}$$

Hence, during forward propagation, $R_\lambda(x)$ acts as an identity transform and during backpropagation it scales L_d by a factor of $-\lambda$. The diagram in 1 illustrates the forward propagation of the inputs and backward propagation of gradients through the model.

7.2 SGL: Small Group Learning

SGL [3] is a multi-stage learning framework involving K learners, each with their own architectures A_k and two sets of weights $\{V_k, W_k\}$, that train cooperatively to learn some task T. Prior to training, the training data is split into three subsets: training data D^{tr}, unlabeled data D^{ul}, and validation data D^{val}. Each learner then goes through a pretraining stage, where they each update their weights V_k using the training data D^{tr} for some threshold number of epochs. Once pretraining completes, SGL, which consists of three stages, begins. In the first stage each learner keeps their architecture A_k fixed and updates their first set of weights V_k to V_k^* by minimizing the training loss on D^{tr}. In the second stage, each learner creates a pseudolabeled dataset D_k^{pl} by labeling D^{ul} using (A_k, V_k^*). Note here that the pseudolabels are represented as J dimensional vectors (J being the number of classes) where each class gets a probability between $[0, 1]$. Each learner then trains its second set

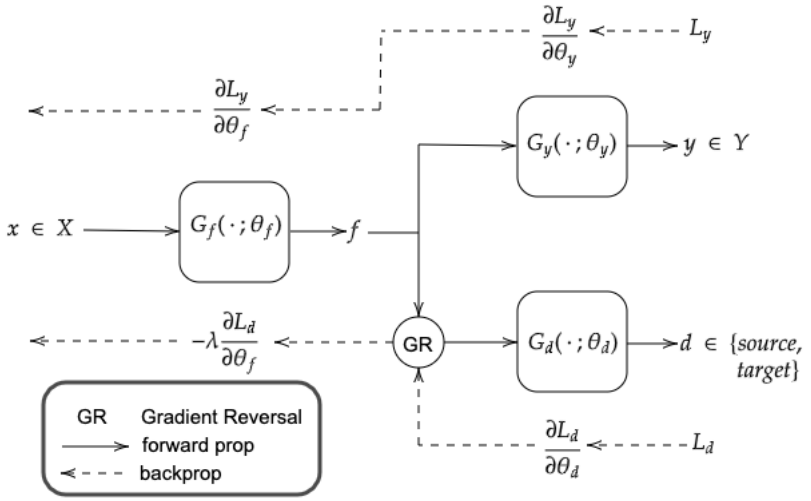

Figure 1: Adversarial network for domain adaptation

of weights W_k to W_k^* by minimizing the training loss on $D^{tr} \cup \{D_{k'}^{pl}\}_{k' \neq k}^{K}$. In the final stage, each learner minimizes the validation loss on D^{val} by updating its architecture A_k with weights W_k^* using a differentiable search technique such as DARTS [4].

The learning framework can be written mathematically as:

$$\begin{aligned} &\min_{\{A_k\}_{k=1}^K} \sum_{k=1}^{K} L(W_k^*, A_k, D^{val}) \\ s.t. \quad &\{W_k^*\}_{k=1}^K = \min_{\{W_k\}_{k=1}^K} \sum_{k=1}^{K} L(W_k, A_k, D^{tr}) + \lambda \sum_{j \neq k}^{K} L(W_k, A_k, D_j^{pl}(D^{ul}, V_j^*)) \\ s.t. \quad &\{V_k^*\}_{k=1}^K = \min_{\{V_k\}_{k=1}^K} \sum_{k=1}^{K} L(V_k, A_k, D^{tr}) \end{aligned} \tag{5}$$

7.3 DARTS: Differentiable Architecture Search

DARTS is a neural architecture search method that relaxes the candidate search space to be continuous such that the architecture can be optimized via gradient descent. The advantage of using DARTS is that it can achieve results comparable to state of-the-art architecture search methods such as reinforcement learning and Bayesian learning while requiring significantly less computational resources. [4]. Following the approach used in SGL, the network architecture A_k and weights W_k are in the final stage of the learning framework via DARTS. A_k consists of 3 components: feature extractor A_f, label classifier A_y, and domain classifier A_d, we plan to evaluate multiple update update policies for A_k. One approach might be updating each of the components A_f, A_y and A_d. Another approach could be to hold both classifier architectures A_y and A_d fixed while only updating the feature extractor A_f. With final model architecture update policy will selected once performance is tested.

7.4 SGL + DANN

In this project we propose to apply the SGL learning framework to the problem of domain adaptation. We accomplish this by using $K = 2$ learners, each using the adversarial model described previously, but with slightly different architectures and weight initialization. Since the adversarial model consists of three subnetworks (feature extractor, label classifier and domain classifier), the parameters of each learner can also be broken into three sets of subparameters: $V_k = (\theta_f, \theta_y, \theta_d)_k$, $W_k = (\theta'_f, \theta'_y, \theta'_d)$

and $A_k = (A_f, A_y, A_d)$. Each learner will use the objective function $E(\theta_f, \theta_y, \theta_d)$ described in Eq. 1 as the loss function for each of the optimization steps presented in Eq. 5 to update the subparameters of V_k, W_k and A_k.

7.5 Color Space Mapping

In domain adaptation problems, there are numerous approaches in defining a source and domain-shifted target distribution. Common domain adaptation applications use distributions varying in sparsity and environment. Although the original DANN implementation is evaluated by learning across different numerical image sets (e.g. MNIST, SVHN), the choice was made to evaluate the current work using CIFAR, which is the dataset of choice by the SGL authors. We are interested in shifting the distributions by changing the way image information is stored. Our approach is to use the CIFAR-10 dataset as $\mathcal{D}^S$ encoded in RGB format and create a modified CIFAR-10 dataset as $\mathcal{D}^T$ encoded in HSV[2] format.

We implement color space mapping from RGB to HSV to create an augmented dataset with different distribution for domain adaptation task. HSV stands for hue (dominant color), saturation (purity of the color), and value (brilliance of the color). These values represent the color space in the closest form to what humans actually experience and is derived from concepts in color psychology.

It is worth noting that CIFAR-10 is in the sRGB format, so a preliminary conversion must be made to convert sRGB representation to RGB by linearizing it by power-law of 2.2 [10]. The following steps are followed to yield an augmented CIFAR-10 set in HSV format.

$$\hat{R}, \hat{G}, \hat{B} = \frac{R}{255}, \frac{G}{255}, \frac{B}{255} \tag{6}$$

$$C_{max} = max(\hat{R}, \hat{G}, \hat{B}) \tag{7}$$

$$C_{min} = min(\hat{R}, \hat{G}, \hat{B}) \tag{8}$$

$$\Delta = range(\hat{R}, \hat{G}, \hat{B}) \tag{9}$$

$$H = \begin{cases} 60^\circ \cdot (\frac{\hat{G}-\hat{B}}{\Delta} mod(6)) & \text{if label}(C_{max}) = \hat{R} \\ 60^\circ \cdot (\frac{\hat{B}-\hat{R}}{\Delta} + 2) & \text{if label}(C_{max}) = \hat{G} \\ 60^\circ \cdot (\frac{\hat{R}-\hat{G}}{\Delta} + 4) & \text{if label}(C_{max}) = \hat{B} \end{cases} \tag{10}$$

$$S = \begin{cases} \frac{\Delta}{C_{max}} & \text{if } C_{max} \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

$$V = C_{max} \tag{12}$$

Our model aims to build features invariant to each images pixel encoding such that classification can be achieved on distributions of various image representation. Should it be found that the model performs well on RGB to HSV domain shifts, the current exploration can be expanded in scope to include domains represented in other color spaces (e.g. CMYK, YPbPr).

[2]Hue, saturation, value. "Value" is commonly substituted with "lightness", so HSV and HSL can be used interchangeably.

References

[1] Wilson, G., D. J. Cook. A survey of unsupervised deep domain adaptation, 2020.

[2] Wang, M., W. Deng. Deep visual domain adaptation: A survey, 2018.

[3] Du, X., P. Xie. Small-group learning, with application to neural architecture search, 2020.

[4] Liu, H., K. Simonyan, Y. Yang. Darts: Differentiable architecture search, 2019.

[5] Glorot, X., A. Bordes, Y. Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 513–520. Omnipress, Madison, WI, USA, 2011.

[6] Olkin, I., F. Pukelsheim. The distance between two random vectors with given dispersion matrices. *Linear Algebra and its Applications*, 48:257 – 263, 1982.

[7] Ghifary, M., W. B. Kleijn, M. Zhang, et al. Domain generalization for object recognition with multi-task autoencoders. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2551–2559. 2015.

[8] Goodfellow, I. J., J. Pouget-Abadie, M. Mirza, et al. Generative adversarial networks, 2014.

[9] Ganin, Y., V. Lempitsky. Unsupervised domain adaptation by backpropagation, 2015.

[10] Gowda, S. N., C. Yuan. Colornet: Investigating the importance of color spaces for image classification. *CoRR*, abs/1902.00267, 2019.