2023 IEEE 33rd International Workshop on Machine Learning for Signal Processing (MLSP) | 979-8-3503-2411-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/MLSP55844.2023.10285908

A DISCRIMINATIVE APPROACH TO UNSUPERVISED DOMAIN ADAPTATION IN COARSE-TO-FINE CLASSIFIERS

Ismail R. Alkhouri [1,*] *Akram S. Awad* [1,*] *Connor Hatfield* [1] *George K. Atia* [1,2]

[1] Department of Electrical and Computer Engineering, University of Central Florida, Orlando FL, USA
[2] Department of Computer Science, University of Central Florida, Orlando FL, USA
Email: {ismail.alkhouri, akram.awad, co418391, george.atia}@ucf.edu

ABSTRACT

Conventional machine learning models often exhibit poor performance when tested on data that comes from a different probability distribution than that of the training data. This phenomenon is known as 'domain shift'. To overcome this problem, domain adaptation (DA) methods have been proposed to reduce the discrepancy between the two domains and improve the performance on the target domain. However, most of the existing DA techniques for classification have focused on One Level Classifiers (OLCs), which are not suitable for data that follows a hierarchical classification structure. In this paper, we propose a discriminative DA approach for coarse-to-fine (C2F) classifiers, which provide both coarse and fine labels. Our approach learns an invariant feature representation of the data across domains for both the coarse and finer levels. Our experimental results on well-known digit datasets demonstrate that the proposed algorithm outperforms the base C2F model in terms of the classification accuracy of the target domain data.

***Index Terms*—** Unsupervised Domain Adaptation, Coarse-to-Fine Classification, Discriminative Models, Representation Learning, Neural Networks

1. INTRODUCTION

Conventional machine learning models rely on the assumption that the training and test data follow the same distribution, which should result in good performance on new data. However, in practice, these methods can perform poorly due to distributional mismatches, also known as domain shift, between the training and test data. The discrepancy between the two domains can prevent the trained models from adapting to new data, leading to a decline in performance. To overcome this issue, adaptive methods that can adjust to the target domain are needed.

Domain adaptation (DA) tackles the issue of distribution mismatch between the training and test data, by adjusting a model trained on a specific source domain to be able to perform well on a different target domain with a different distribution [1]. DA has been shown to significantly improve learning performance by reducing the discrepancy between related but distinct domains, leading to its adoption in various fields [2–4]. DA methods can be classified into two categories: unsupervised domain adaptation (UDA) [5] and semi-supervised domain adaptation [6], depending on whether the label information is available for the target domain. UDA methods are used when no labels are available in the target domain, whereas semi-supervised methods are applied when labels are only partially available.

The literature proposes various approaches to DA. One of the main approaches is the instance-based method, which involves learning distinct weights for ranking training examples in a source domain. When utilized in a target domain, this method can enhance learning performance [7–9]. However, accurately estimating the weights becomes increasingly challenging as the data dimensionality increases [10]. Alternatively, feature-based approaches attempt to learn a shared feature structure across different domains, enabling knowledge transfer between them [11–13]. This can be achieved by finding a feature transformation that minimizes domain differences while preserving statistical properties [11, 14], or by using adversarial DA, as in the two-player game of Generative Adversarial Networks (GANs) [15]. With adversarial DA, a domain discriminator is trained to reduce classification errors between the source and target domains, while deep neural networks learn transferable domain-invariant representations that cannot be differentiated by the domain discriminator [16–18]. The UDA problem has two main discriminative approaches: the Domain-Adversarial Neural Network (DANN) [17] and the Adversarial Discriminative Domain Adaptation (ADDA) [16]. Unlike ADDA, the DANN approach requires an additional network to model a generator function. However, studies have shown that ADDA outperforms all other methods, including DANN, in terms of classification accuracy and run-time.

This work was supported by NSF Award CCF-2106339, NSF CAREER Award CCF-1552497, and DOE Award DE-EE0009152.

* Indicates an equal contribution.

Adversarial DA has been widely used for One-Level Classifiers (OLCs), but to the best of our knowledge, no existing studies have investigated DA in the context of Coarse-to-Fine (C2F) or Hierarchical Classifiers (HCs) [19]. A HC consists of multiple classification levels, where the coarse level distinguishes between major categories or classes, and the fine level refines the classification to more specific subcategories or subclasses. HCs have applications in various fields such as speech classification [20, 21], computer vision [22, 23] and hierarchical fault diagnosis systems [24]. Therefore, it is crucial to develop effective DA methods for HCs, which is the primary focus of this work. Here, we propose an adversarial DA approach for C2F classifiers, however, our approach can be extended to other HC models. We aim to learn an invariant feature representation of the data across the two domains for both the coarse and finer levels. Our approach improves the classification performance on the target domain, as demonstrated by a series of experiments on well-known benchmark datasets (e.g., MNIST and USPS) with different groupings.

2. BACKGROUND: DISCRIMINATIVE DOMAIN ADAPTATION FOR ONE LEVEL CLASSIFIERS

Consider a labeled source dataset (X_s, Y), where X_s is the set of data points and Y is the corresponding set of true labels. Each data point $x_s \in X_s \subseteq \mathbb{R}^N$ has a label $y \in [K]$, where K is the number of classes and $[K] := \{1, \dots, K\}$. In addition to the source dataset, (X_s, Y), we assume that we have an unlabeled target dataset X_t with entries $x_t \in \mathbb{R}^N$.

An OLC consists of two stages: a feature extraction stage and a classification stage. The feature extraction stage is defined by a function $M_{\theta_s} : \mathbb{R}^N \to \mathbb{R}^F$, parameterized by θ_s, that extracts $F < N$ features. The classification stage uses a function $C_\phi : \mathbb{R}^F \to \Delta^K$, parameterized by ϕ, to predict the class label of a given data point. Here, Δ^K denotes the probability simplex of dimension K. The predicted label is obtained by selecting the class with the highest probability, given by

$$p(x; \theta_s, \phi) = \underset{k \in [K]}{\operatorname{argmax}} \, C_\phi\big(M_{\theta_s}(x)\big)_k \,, \tag{1}$$

where $C_\phi\big(M_{\theta_s}(x)\big)_k$ is the k^{th} entry of the discriminant vector $C_\phi\big(M_{\theta_s}(x)\big) \in \Delta^K$. The goal of UDA in OLCs is to find feature extraction and classification functions that enable accurate labeling of the *target* dataset.

The ADDA method proposed in [16] is one of the most efficient UDA techniques for OLCs. It comprises two stages. In the first stage, known as the pre-training stage, a feature extractor function M_{θ_s} and a classification model C_ϕ are trained on the source dataset using supervised learning, i.e., the conventional deep learning training method. The optimization problem to be solved with respect to θ_s and ϕ is formulated using the cross-entropy loss and the one-hot encoding representation for the true label as follows:

$$\min_{\theta_s, \phi} \mathbb{E}_{(x_s, y) \sim (X_s, Y)} - \log\Big(C_\phi\big(M_{\theta_s}(x)\big)_y\Big) \,. \tag{2}$$

In the second stage, referred to as the adversarial adaptation stage, domain adaptation of the source and target features takes place. For this stage, two additional functions are introduced: the target feature extraction function $M_{\theta_t} : \mathbb{R}^N \to \mathbb{R}^F$ parameterized by θ_t, and the discriminator function $D_\psi : \mathbb{R}^F \to \mathbb{R}$ parameterized by ψ. These functions are learned using the target dataset X_t, the source data points set X_s (without the label set Y), and the learned source data feature extraction function M_{θ_s} from the first ADDA stage, where θ_s is the solution of (2). The discriminator function is a binary classifier, and its learning takes place by solving the two unconstrained optimization problems (3) and (4) simultaneously using cross-entropy losses. The solution can be obtained using alternating minimization.

$$\begin{aligned} \min_{\psi} & - \mathbb{E}_{x_s \sim X_s} \log\Big(D_\psi\big(M_{\theta_s}(x_s)\big)\Big) \\ & - \mathbb{E}_{x_t \sim X_t} \log\Big(1 - D_\psi\big(M_{\theta_t}(x_t)\big)\Big) \end{aligned} \tag{3}$$

$$\min_{\theta_t} - \mathbb{E}_{x_t \sim X_t} \log\Big(D_\psi\big(M_{\theta_t}(x_t)\big)\Big) \,. \tag{4}$$

In summary, the standard training process yields C_ϕ and M_{θ_s} from the source dataset (X_s, Y) through conventional training, as described by the function in (5).

$$C_\phi, M_{\theta_s} = \text{StandardTraining}(X_s, Y) \,. \tag{5}$$

Then, the adversarial adaptation process leverages X_s, X_t, and M_{θ_s} from (5) to obtain M_{θ_t}, as summarized in (6).

$$M_{\theta_t} = \text{AdversarialAdaptation}(X_s, X_t, M_{\theta_s}) \,. \tag{6}$$

At inference time, the trained target feature extraction function M_{θ_t} from (6) and the trained classification function from (5) are used to predict the label of any $x \in \mathbb{R}^N$, given by

$$p(x; \theta_t, \phi) = \underset{k \in [K]}{\operatorname{argmax}} \, C_\phi\big(M_{\theta_t}(x)\big)_k \,. \tag{7}$$

3. DISCRIMINATIVE DOMAIN ADAPTATION FOR C2F CLASSIFIERS

In this section, we describe our problem setup and present a discriminative model for UDA that aims to predict both coarse and fine labels of a HC in the target domain.

3.1. Source and Target Data for UDA in C2F Setting

Assume that we have a labeled source dataset $D_s = \{(x_1, y_1, z_1), \dots, (x_n, y_n, z_n)\}$, where $y_i \in Y$ represents the true coarse label of data point $x_i \in X_s$, and Y is a set of

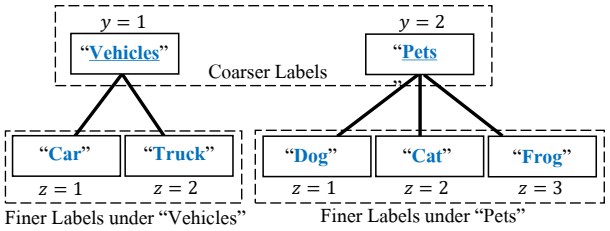

Fig. 1: An example of a tree of coarse-to-fine labels, where $L = 2$, $L_1 = 2$, and $L_2 = 3$. The sets $Z_1 = \{1, 2\}$ and $Z_2 = \{1, 2, 3\}$ represent the possible fine labels for the two coarse labels. The coarse and fine labels are shown in blue font. For instance, if we have two data points $(x_1, 1, 1)$ and $(x_2, 2, 1) \in D_s$, where the coarse and fine label pairs are given by $(1, 1)$ and $(2, 1)$, respectively, they correspond to ('Vehicles', Car') and ('Pets', 'Dog'), respectively.

coarse labels with L unique values. For each coarse label $y \in Y$, there exists a set of corresponding fine labels denoted as Z_y. Thus, each data point in X_s has a fine label $z_i \in Z_{y_i}$ according to a predefined tree structure that maps coarse-to-fine classes. See Figure 1 and its caption for an example of a tree of coarse-to-fine classes. We use L_y to represent the number of fine labels for a given coarse label y. In addition to the source data, we have an unlabeled target data X_t. The goal is to develop a method for UDA to obtain the coarse and fine labels of the target data.

3.2. C2F Classification Model

Next, we present our two-stage prediction model that consists of one coarse and L fine classifiers. Let $M_{\theta_{s,c}} : \mathbb{R}^N \to \mathbb{R}^F$, parameterized by $\theta_{s,c}$, be the feature extraction function, and $C_{\phi_c} : \mathbb{R}^F \to \Delta^L$, parameterized by ϕ_c, be its corresponding classification function, which classifies the input into one of L coarse classes. For each $i \in [L]$, we define functions $M_{\theta_{s,f_i}} : \mathbb{R}^N \to \mathbb{R}^F$, parameterized by θ_{s,f_i}, and $C_{\phi_{f_i}} : \mathbb{R}^F \to \Delta^{L_i}$, parameterized by ϕ_{f_i}, to represent the feature extraction and classification stages of the i^{th} finer classifier, respectively. The predicted coarse label is then obtained as

$$r(x; \theta_{s,c}, \phi_c) = \underset{i \in [L]}{\operatorname{argmax}} \, C_{\phi_c}\left(M_{\theta_{s,c}}(x)\right)_i , \tag{8}$$

Based on the coarse prediction in (8), and the finer classification model, the finer level prediction is found as

$$h(x, r; \theta_{s,f_i}, \phi_{f_i}) = \underset{j \in [L_i]}{\operatorname{argmax}} \, C_{\phi_{f_i}}\left(M_{\theta_{s,f_i}}(x)\right)_j . \tag{9}$$

A block diagram of the C2F model is given in Figure 2.

3.3. Proposed Algorithm

In this section, we describe the different stages of the proposed algorithm for UDA in the C2F setting.

3.3.1. Stage 1: Source feature extraction and classification

In this stage, we use the data points and the coarse labels of the source data, i.e., X_s and Y, to learn the feature extraction function $M_{\theta_{s,c}}$, and the classification function C_{ϕ_c} as

$$C_{\phi_c}, M_{\theta_{s,c}} = \text{StandardTraining}(X_s, Y) . \tag{10}$$

In order to obtain the extraction and classification functions of the finer level classifiers, first, the true coarse labels are used to split the source dataset (X_s, Y) into L disjoint subsets. Specifically, for every $i \in [L]$, we obtain

$$X_s^{(i)} = \{x_j \in X_s : y_j = i\} , \tag{11}$$

such that $X_s = \bigcup_{i \in [L]} X_s^{(i)}$.

We also denote by $Z^{(i)}$ the fine labels of the data points in $X_s^{(i)}$. We can readily use standard training on the fine level classification models to obtain, for each $i \in [L]$,

$$C_{\phi_{f_i}}, M_{\theta_{s,f_i}} = \text{StandardTraining}(X_s^{(i)}, Z^{(i)}) . \tag{12}$$

3.3.2. Stage 2: Domain-adaptive feature extraction

Having obtained the coarse and fine source feature extraction and classification functions, in this step, we leverage an adversarial adaptation function for the C2F prediction model. We first use the adversarial adaptation function to obtain the parameters of the coarse target feature extraction function. This is accomplished by

$$M_{\theta_{t,c}} = \text{AdversarialAdaptation}(X_s, X_t, M_{\theta_{s,c}}) . \tag{13}$$

Using (13), we can obtain the predicted coarse labels of the target data, which is the first set of labels we need for the UDA problem in the C2F model we consider in this paper. Next, we use the predicted coarse labels of the target data to split X_t into L disjoint subsets in a similar fashion as was done for the source data. Hence, for each $i \in [L]$, we obtain

$$X_t^{(i)} = \{x \in X_t : \underset{j \in [L]}{\operatorname{argmax}} \, C_{\phi_c}\left(M_{\theta_{t,c}}(x)\right)_j = i\} , \tag{14}$$

such that $X_t = \bigcup_{i \in [L]} X_t^{(i)}$.

After splitting the data, for each $i \in [L]$, we utilize the adversarial adaptation method to obtain the feature extraction function of the target data as

$$M_{\theta_{t,f_i}} = \text{AdversarialAdaptation}(X_s^{(i)}, X_t^{(i)}, M_{\theta_{s,f_i}}) . \tag{15}$$

The complete procedure of our proposed method is outlined in Algorithm 1.

3.4. Inference Time

At inference time, the UDA-trained C2F model predicts coarse and fine labels using (16), which depends on the feature extraction functions obtained from Algorithm 1. This is

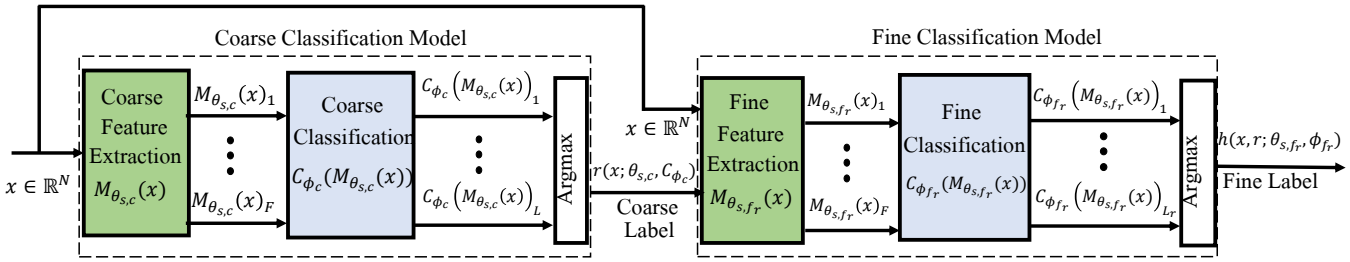

Fig. 2: A block diagram of the two-stage C2F classification model.

Table 1: Performance results for the C2F base and Algorithm 1 models for the USPS adaptation to MNIST (USPS → MNIST).

C2F Model	C2F Scenario	CA_c (%)	T_1 (%)	T_2 (%)	CA_{f_1} (%)	CA_{f_2} (%)	CCA (%)
Base	$\{0,2,4,6,8\}$,$\{1,3,5,7,9\}$	73.08	**99.8**	47.2	84.62	70.33	58.44
Algorithm 1	$\{0,2,4,6,8\}$,$\{1,3,5,7,9\}$	**94.18**	99	**89.5**	**96.27**	**97.82**	**91.37**
Base	$\{0,1,2,3,4\}$,$\{5,6,7,8,9\}$	76.33	92.2	59.5	79.37	**78.49**	60.30
Algorithm 1	$\{0,1,2,3,4\}$,$\{5,6,7,8,9\}$	**87.64**	**99.2**	**75.4**	**99.11**	63.84	**73.92**

Algorithm 1 UDA in C2F Algorithm

Input: Source labeled dataset (X_s, Y, Z), Target dataset X_t, number of coarse classes L.

Output: Trained C2F model that consists of the coarse functions C_{ϕ_c} and $M_{\theta_{t,c}}$, and the finer functions $C_{\phi_{f_i}}$ and $M_{\theta_{t,f_i}}, \forall i \in [L]$.

1: **Obtain** C_{ϕ_c} and $M_{\theta_{s,c}}$ using X_s and Y using (10)

2: **Split** the source dataset as in (11)

3: **For** all fine classifiers $i \in [L]$

4: **Obtain** $C_{\phi_{f_i}}$ and $M_{\theta_{s,f_i}}$ using $X_s^{(i)}$ and $Z^{(i)}$ using (12)

5: **Obtain** $M_{\theta_{t,c}}$ using X_s, X_t, and $M_{\theta_{s,c}}$ using (13)

6: **Split** the target dataset as in (14)

7: **For** all fine classifiers $i \in [L]$

8: **Obtain** $M_{\theta_{t,f_i}}$ using $X_s^{(i)}$, $X_t^{(i)}$, and $M_{\theta_{s,f_i}}$ using (15)

in contrast to the base model where no UDA approach is used, and coarse and fine label predictions are made using equations (8) and (9).

$$r(x;\theta_{t,c},\phi_c) = \underset{i\in[L]}{\operatorname{argmax}}\, C_{\phi_c}\left(M_{\theta_{t,c}}(x)\right)_i ,$$
$$h(x,r;\theta_{t,f_r},\phi_{f_r}) = \underset{j\in[L_r]}{\operatorname{argmax}}\, C_{\phi_{f_r}}\left(M_{\theta_{t,f_r}}(x)\right)_j . \quad (16)$$

4. EXPERIMENTAL EVALUATION

In this section, we evaluate the effectiveness of our proposed algorithm in various C2F classification settings through a series of experiments.

Experimental Setting: The MNIST and USPS datasets [25] are used for evaluation, where each example is a 28×28 grayscale matrix of values in $[0, 1]$, representing a handwritten digit from '0' to '9' with corresponding class labels. We investigate two domain adaptation scenarios: MNIST→ USPS (source dataset is USPS and target dataset is MNIST) and USPS→ MNIST (source dataset is MNIST and target dataset is USPS).

Moreover, we consider two hierarchical classification settings in our experiments. In the first setting (C2F scenario), we divide the class labels into two sets: even numbers {0,2,4,6,8} and odd numbers {1,3,5,7,9}, referred to as the even-odd case. In the second setting, we split the labels around 5, creating two sets: {0,1,2,3,4} and {5,6,7,8,9}.

To perform feature extraction, classification, and discrimination functions, we employ standard CNN models. Specifically, we use the modified LeNet architecture available in the Caffe source code [26] for each classifier. In addition, each domain discriminator comprises three fully connected layers, with the first two layers containing 500 hidden units, and the final layer producing the discriminator output. ReLU is used as the activation function in each of the 500-unit layers. More details and the code for these networks are available online[1].

Baseline and Evaluation Metric: The proposed algorithm is compared with the base C2F classifier, which serves as the baseline method. Evaluation metrics are based on multiple measures of classification accuracy (CA). We define $T_i, \forall i \in [L]$ as the percentage of correctly classified samples by the coarse classifier as label i. This means that T_i represents the accuracy of class i. The coarse classifier's overall CA, denoted by CA_c, is calculated as $CA_c = \sum_{i\in[L]} \alpha_i T_i$ where α_i represents the ratio of the number of samples of class i to the total number of samples. For the overall CA of the finer classifier i, we use CA_{f_i}. To measure the overall CA of the C2F model, we use the combined classifica-

[1] https://github.com/ialkhouri/DiscriminativeUDAinC2Fs

Table 2: Performance results for the C2F base and Algorithm 1 models for the MNIST adaptation to USPS (MNIST $\rightarrow$ USPS).

C2F Model	C2F Scenario	CA_c (%)	T_1 (%)	T_2 (%)	CA_{f_1} (%)	CA_{f_2} (%)	CCA (%)
Base	$\{0,2,4,6,8\}$,$\{1,3,5,7,9\}$	92.33	**93.4**	91.00	84.56	74.32	73.81
Algorithm 1	$\{0,2,4,6,8\}$,$\{1,3,5,7,9\}$	**94.22**	92.5	**96.3**	**86.55**	**91.14**	**83.57**
Base	$\{0,1,2,3,4\}$,$\{5,6,7,8,9\}$	**89.64**	**93.3**	84.3	**92.38**	76.98	**77.48**
Algorithm 1	$\{0,1,2,3,4\}$,$\{5,6,7,8,9\}$	89.49	84.3	**97**	83.01	**90.94**	77.43

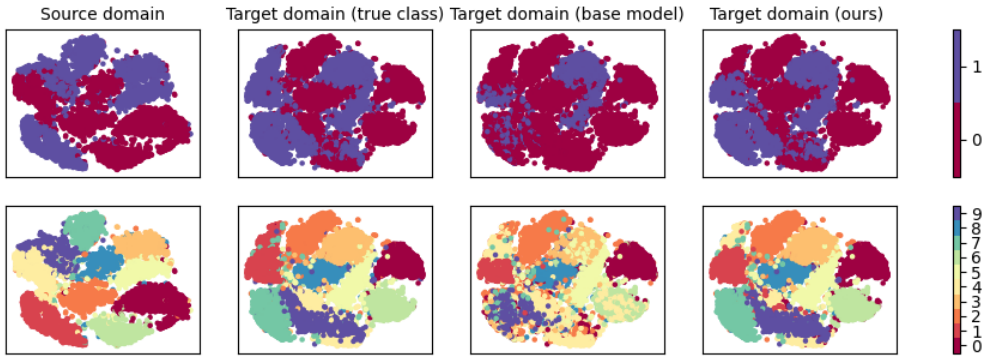

Fig. 3: Visualisation of the source (USPS) and target (MNIST) domains. The first row represents the coarse level classes (even ='0', odd ='1'), while the second one represents the finer classes.

tion accuracy (CCA), which is defined as the percentage of samples correctly classified by both the coarse and fine classifiers. Based on the notation introduced, the CCA is calculated as $\text{CCA} = \sum_{i\in[L]} \alpha_i \; T_i \; \text{CA}_{f_i}$. The predicted labels are obtained using (8) and (9) for the base model and (16) for the Algorithm 1 C2F model. Specifically, the base model was trained using solely the available source domain dataset, where each classifier was trained independently. These classifiers are utilized at inference time to predict the target domain labels.

Results: We present the classification performance results of the target data of the C2F model obtained from our proposed algorithm compared to the base C2F model, which are given in Tables 1 and 2.

We observe that the model trained by Algorithm1 outperforms the base model in terms of CA performance for most of considered C2F scenarios (listed in the last column of Table 1 and 2) and adaptation directions (USPS $\rightarrow$ MNIST in Table 1 and MNIST $\rightarrow$ USPS in Table 2). This improvement is demonstrated not only by the values of the combined CA (CCA), but also by most of the overall CA values of the coarse and finer classifiers individually. For example, in the first two rows of Table 1, the CCA for the base model is only 58.44%. However, when using our algorithm, the CCA of the C2F model becomes 91.37%. Furthermore, we observe that the overall coarse CA (CA_c) increases from 73.08% to 94.18%. This improvement is also observed for the overall CA of the finer 1 (finer 2) classifier, where CA_{f_1} (CA_{f_2}) increases from 84.62% (70.33%) to 96.27% (97.82%).

Although Algorithm 1 consistently produces a trained model with an overall CCA that is higher than or on par with the base model, we observe that the even-odd C2F setting shows a larger performance improvement compared to the two-sided case. This means that the effectiveness of our algorithm depends on the hierarchy of the C2F classification and the source and target domains. In Figure 3, we present a visualization of the source and target domain datasets (for USPS $\rightarrow$ MNIST, even and odd splitting) using t-SNE, to visualize high-dimensional data. As observed, our methods outperform the base model for both coarse and finer classification.

5. CONCLUSION & FUTURE WORK

In this paper, we have developed a discriminative and adversarial approach to address the unsupervised domain adaptation problem in coarse-to-fine (C2F) classification models. Our proposed algorithm leverages the standard training of the source dataset and employs an adversarial adaptation procedure between the source and target datasets using feature extraction and discriminator functions to achieve higher overall classification accuracy on the target data. Through our experimental results, we have demonstrated the efficacy of our proposed algorithm in significantly improving the overall and individual performance of the C2F model in most of the scenarios considered, including both the coarse and finer levels.

In the future, we plan to extend our work to validate the proposed algorithm on larger datasets, non-image classification tasks, and multi-stage coarse-to-fine classification settings. Additionally, we aim to explore the potential of incorporating other techniques such as meta-learning and model compression to further enhance the performance of the C2F model in unsupervised domain adaptation scenarios.

6. REFERENCES

[1] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020.

[2] Meng Yang, Lei Zhang, Xiangchu Feng, and David Zhang, "Fisher discrimination dictionary learning for sparse representation," in *International Conference on Computer Vision*, 2011, pp. 543–550.

[3] Han Guo, Ramakanth Pasunuru, and Mohit Bansal, "Multi-source domain adaptation for text classification via distancenet-bandits," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, vol. 34, pp. 7830–7838.

[4] Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin, "How transferable are neural networks in NLP applications?," *arXiv preprint arXiv:1603.06111*, 2016.

[5] Xuemiao Xu, Hai He, Huaidong Zhang, Yangyang Xu, and Shengfeng He, "Unsupervised domain adaptation via importance sampling," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 12, pp. 4688–4699, 2019.

[6] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko, "Semi-supervised domain adaptation via minimax entropy," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 8050–8058.

[7] Sirvan Khalighi, Bernardete Ribeiro, and Urbano J Nunes, "Importance weighted import vector machine for unsupervised domain adaptation," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3280–3292, 2016.

[8] Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul Buenau, and Motoaki Kawanabe, "Direct importance estimation with model selection and its application to covariate shift adaptation," *Advances in Neural Information Processing Systems*, vol. 20, 2007.

[9] Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex Smola, "Correcting sample selection bias by unlabeled data," *Advances in neural information processing systems*, vol. 19, 2006.

[10] Masashi Sugiyama, Motoaki Kawanabe, and Pui Ling Chui, "Dimensionality reduction for density ratio estimation in high-dimensional spaces," *Neural Networks*, vol. 23, no. 1, pp. 44–59, 2010.

[11] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu, "Transfer feature learning with joint distribution adaptation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2200–2207.

[12] Christian Pölitz, Wouter Duivesteijn, and Katharina Morik, "Interpretable domain adaptation via optimization over the stiefel manifold," *Machine Learning*, vol. 104, pp. 315–336, 2016.

[13] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang, "Domain adaptation via transfer component analysis," *IEEE Transactions on Neural Networks*, vol. 22, no. 2, pp. 199–210, 2010.

[14] Jing Zhang, Wanqing Li, and Philip Ogunbona, "Joint geometrical and statistical alignment for visual domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1859–1867.

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[16] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7167–7176.

[17] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[18] Yaroslav Ganin and Victor Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International Conference on Machine Learning*. PMLR, 2015, pp. 1180–1189.

[19] Carlos N Silla and Alex A Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, pp. 31–72, 2011.

[20] Ofer Dekel, Joseph Keshet, and Yoram Singer, "An online algorithm for hierarchical phoneme classification," in *Machine Learning for Multimodal Interaction: First International Workshop*. Springer, 2005, pp. 146–158.

[21] Zafer Barutcuoglu and Christopher DeCoro, "Hierarchical shape classification using bayesian aggregation," in *IEEE International Conference on Shape Modeling and Applications (SMI'06)*, 2006.

[22] Pavel Zhdanov, Adil Khan, Adín Ramírez Rivera, and Asad Masood Khattak, "Improving human action recognition through hierarchical neural network classifiers," in *IEEE International Joint Conference on Neural Networks (IJCNN)*, 2018, pp. 1–7.

[23] Yong Xu, Qi Zhu, Zizhu Fan, Minna Qiu, Yan Chen, and Hong Liu, "Coarse to fine K nearest neighbor classifier," *Pattern Recognition Letters*, vol. 34, no. 9, pp. 980–986, 2013.

[24] Akram S Awad, Ismail R Alkhouri, and George K Atia, "Adversarial attacks on multi-level fault detection and diagnosis systems," in *IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*, 2021, pp. 1–6.

[25] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[26] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675–678.