On Fine-Tuned Deep Features for Unsupervised Domain Adaptation

Qian Wang
Department of Computer Science
Durham University
Durham, UK

Fanlin Meng
Alliance Manchester Business School
University of Manchester
Manchester, UK

Toby P. Breckon
Department of Computer Science
Department of Engineering
Durham University
Durham, UK

***Abstract*—Prior feature transformation based approaches to Unsupervised Domain Adaptation (UDA) employ the deep features extracted by pre-trained deep models without fine-tuning them on the specific source or target domain data for a particular domain adaptation task. In contrast, end-to-end learning based approaches optimise the pre-trained backbones and the customised adaptation modules simultaneously to learn domain-invariant features for UDA. In this work, we explore the potential of combining fine-tuned features and feature transformation based UDA methods for improved domain adaptation performance. Specifically, we integrate the prevalent progressive pseudo-labelling techniques into the fine-tuning framework to extract fine-tuned features which are subsequently used in a state-of-the-art feature transformation based domain adaptation method SPL (Selective Pseudo-Labeling). Thorough experiments with multiple deep models including ResNet-50/101 and DeiT-small/base are conducted to demonstrate the combination of fine-tuned features and SPL can achieve state-of-the-art performance on several benchmark datasets.**

***Index Terms*—Unsupervised domain adaption, deep convolutional neural network, fine-tuning, linear probing, self-training, selective pseudo labelling.**

I. Introduction

Deep learning has been a dominant technique for many computer vision tasks (e.g., image classification, object detection, semantic segmentation, etc.) in many real-world applications. A deep model (e.g., deep Convolutional Neural Networks [1] and Vision Transformers [2], [3]) usually has millions of parameters and training such a deep model requires sufficient training data (e.g., millions of annotated images). However, there exist situations where training data are limited or even unavailable. For example, in medical image processing, annotating data for training is non-trivial and cost-intensive. Training a deep model from scratch on a relatively small training data set cannot achieve satisfactory performance. A typical solution to this problem is transfer learning. A simple yet effective transfer learning technique is fine-tuning. Using a deep model pre-trained on a large dataset such as ImageNet and fine-tuning it on the training data of a particular task has been a de facto choice in many computer vision and image processing tasks.

In many real-world applications, however, there is no training data from the task domain (i.e. target domain) but abundant labelled data from a different relevant domain (i.e. source domain). To take advantage of the labelled data in the source domain, domain adaptation approaches have been proposed so that the task in the target domain can be better handled. In particular, Unsupervised Domain Adaptation (UDA) problems assume that: labelled source domain data and unlabelled target domain data are available during training; the test data in the target domain are available for training although their ground truth labels are unavailable hence the problem is under the transductive learning setting; the source and target domains share the same set of classes (i.e. closed-set domain adaptation).

Existing UDA approaches can be roughly categorised into two groups [4]: feature transformation based approaches and deep feature learning based approaches. The former type of approaches use features extracted by pre-trained deep models and learn linear projections to transform the deep features into a new feature space of good properties (e.g., discriminant, domain aligned). The latter type of approaches learn an end-to-end deep neural network consisting of modules for domain adaptation and the classifier. The former type of approaches have obvious limitations in that the deep models are only used as a feature extractor without being further fine-tuned on the data of specific tasks.

In this work, we aim at investigating the potential of feature transformation based approaches when the deep features are properly fine-tuned. Instead of designing complicated modules for domain adaptation, we employ a simple fine-tuning mechanism to update the pre-trained deep model using the data of target tasks. It is shown that fine-tuned deep features are indeed superior to the original deep features. The combination of fine-tuned deep features and feature transformation based UDA approaches can achieve comparable or better performance than state-of-the-art approaches.

II. Related Work

In this section, we give a brief review of prior works relating to ours. We first review recent approaches to UDA including both feature transformation based and deep learning based ones. Subsequently, we introduce the handling of batch normalisation layers in domain adaptation problems.

A. Unsupervised Domain Adaptation

As mentioned before, we categorise UDA approaches into two groups: feature transformation approaches [5]–[8] and

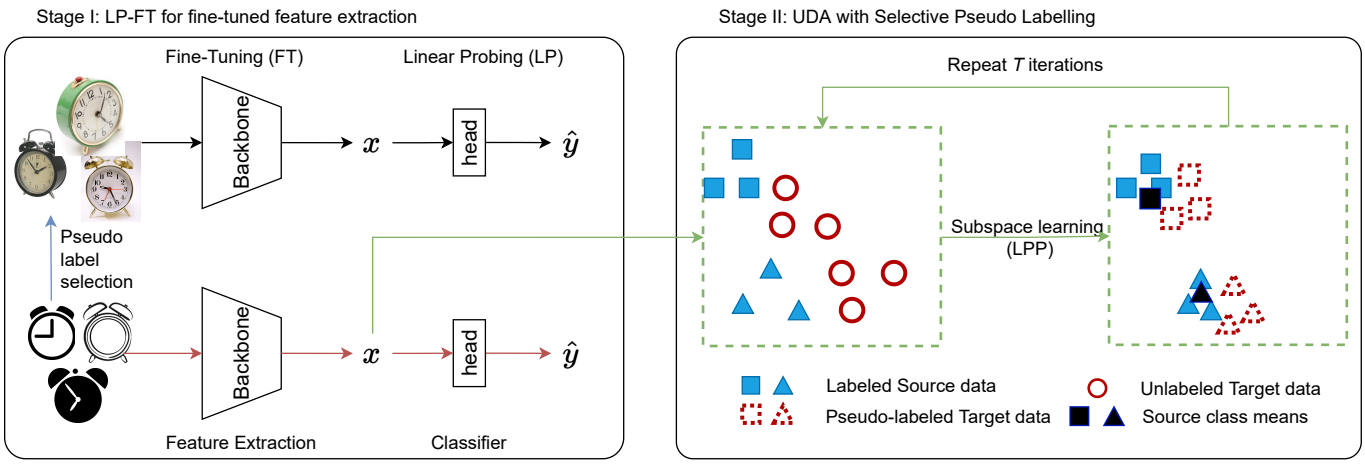

Fig. 1. The framework of our proposed approach to UDA. The left box illustrates the fine-tuning process of the backbone model for feature extraction. The right side box illustrates how the fine-tuned features are used in the Selective Pseudo-Labeling (SPL) UDA method.

deep feature learning approaches [9]–[12]. Feature transformation approaches aim at learning mapping functions between source and target domains or from the source/target domain to a common subspace. The deep features are usually extracted from pre-trained models without fine-tuning. Such features are usually better

Deep feature learning approaches to domain adaptation take advantage of the powerful representation learning capability of CNN models. The objectives are usually to learn domain-invariant features from raw image data in source and target domains in an end-to-end learning framework. For example, the gradient reversal layer has been widely employed for domain adaptation [9], [11]–[13]. However, training such models with only labelled source data biases to the source domain and leads to marginally better or even worse performance than the baseline (e.g., the vanilla ResNet50 without specific domain adaptation modules) for target domain sample classification.

Inspired by the success of transformers in vision tasks [14], [15], recently new approaches to UDA based on transformer frameworks [2], [3] have been proposed. These transformer based UDA approaches have demonstrated significantly improved performance on benchmark datasets. In our work, we show that the features extracted by the transformer based models (whether fine tuned or not) outperform their CNN based counterparts for UDA. The fine tuned features can usually achieve even higher performance.

B. Batch Normalisation in Domain Adaptation

Batch normalisation layers [16] are widely used in many modern CNN models including ResNet [1], DenseNet [17] and EfficientNet [18]. They need to be treated differently from other layers such as Convolutional layers and Dense/Fully-connected layers during fine-tuning [19]–[21]. Recall that the batch normalisation layer computes the output $\boldsymbol{y}$ of the input $\boldsymbol{x}$ in the following way:

$$\boldsymbol{y} = \gamma(\frac{\boldsymbol{x} - \mu}{\sigma + \epsilon}) + \beta \tag{1}$$

where γ and β are two trainable affine parameters; the mean μ and standard deviation σ are estimated during training. In existing works, different strategies have been employed to tackle the batch normalisation layers for UDA problems. Kanavati et al. [21] demonstrate that fine-tuning only the batch norm affine parameters leads to similar performance as to fine-tuning all of the model parameters for domain adaptation. Li et al. [22] replace the statistics of batch normalisation layers in the source domain with those in the target domain. As a result, the method is very simple and parameter-free in contrast to other UDA approaches. Romijnders et al. [23] use a domain agnostic normalisation layer which computes the statistics (i.e. μ and σ) based on the source domain and applies them to the target domain during inference. Klingner et al. [24] adapt the batch normalisation layer statistics by mixing the statistics from both domains. In our experiments, we fix the statistics from the pre-trained weights and directly apply them to both source and target domains during fine-tuning and inference.

III. Method

In this section, we present the details of how we fine-tune a pre-trained deep neural network and use the fine-tuned deep features to improve the unsupervised domain adaptation. Specifically, we first introduce the LP-FT (Linear Probing and Fine Tuning) scheme recently used in [21] and further analysed by Kumar et al. [25]. Subsequently, we introduce how to take advantage of labelled source data and pseudo-labelled target data during fine-tuning. In our empirical study, two types of deep neural networks ResNet [1] and DeiT [15] are considered. We describe a favourable trick needed to fine-tune networks containing batch normalisation layers (e.g., ResNet) for domain adaptation. Finally, to make the paper self-contained, we briefly describe the feature transformation based UDA approach SPL [26] which is employed in our experiments.

A. Linear Probing, Fine-Tuning and Batch Normalisation

A deep model pre-trained on ImageNet has been effectively used to extract deep features for downstream tasks. Although a model pre-trained on a large-scale dataset like ImageNet can generate features of good generalisation, fine-tuning it on the training data for the specific task can usually further improve the performance. To fine-tune a model on the training data of a specific downstream task, one needs to replace the head of the model with a new one suitable for the task. The head of the model can be a linear layer containing the same number of neurons as the number of classes for a classification task. The model is initialised with the pre-trained weights from which the model weights are gradually updated based on the training data using an optimiser such as Stochastic Gradient Descent (SGD) and Adam.

During the fine-tuning, one can choose to update model weights of specific layers [27]–[31] rather than all layers of the network. Linear probing (LP) only updates the weights of the last classification layer and freezes all the rest weights for feature extraction [25], [32]–[34]. Recently, Kumar et al. [25] demonstrate that LP-FT can outperform both linear probing and fine-tuning for out-of-distribution downstream tasks. LP-FT initialises the model with pre-trained weights and updates only the last classification layer (e.g. an L2 regularised logistic/softmax regression classifier) for some epochs. Subsequently, the whole network is fine tuned for weights in all layers to achieve the best performance [21], [25]. It is also a popular strategy to fine-tune different layers with different learning rates [19], [21]. For ResNet models, we use a learning rate $\eta^{LP} = 1e-2$ for the linear probing phase and adapt it to $\eta^{FT} = 1e-4$ for all layers including the last linear classification layer in the fine-tuning phase. For DeiT models, we follow the same settings used in [3].

In our employed LP-FT method, we set the batch norm layers trainable but in the inference mode. This means the trainable parameters in the batch norm layers (i.e. the coefficient γ and the bias β of the affine transformation) will be updated whilst the mean μ and standard deviation σ will be frozen.

B. Data for Fine-Tuning

For unsupervised domain adaptation, labelled source data are ready for use when fine-tuning the model. Fine-tuning the model with source data only enables the model to generate more discriminative features for source data but not necessarily for target data. Inspired by prior works on unsupervised domain adaptation [26], we select pseudo-labelled target samples progressively and add them to the training data set during fine-tuning. Specifically, after each epoch of fine-tuning, we select the most confidently predicted pseudo labels from the target data set. The confidence is based on the posterior probability $p(\hat{y}|x^t;\theta)$ output by the softmax layer. We employ classwise selection so that the numbers of selected pseudo-labelled target samples are balanced across all classes. In the following epoch of fine-tuning, the selected target samples along with their pseudo labels will be combined with the labelled source samples to form the new training data set. The number of selected pseudo-labelled target samples is linearly increased until all target samples are selected in the last epoch. In the early epochs of training, a small fraction of pseudo-labelled target samples are selected to participate in the fine-tuning to avoid the incorrect pseudo labels being reinforced in the following epochs. In the last epoch of fine-tuning, all target samples are exploited although incorrect pseudo labels still exist (unless the accuracy is 100%).

C. Method for UDA

To evaluate the fine-tuned features, we choose one feature transformation based UDA method in our study. To make the paper self-contained, we briefly describe the UDA approach Selective Pseudo Labelling (SPL) proposed in [26] and more details can be found in the original paper. The crucial components of SPL are subspace learning and selective pseudo labelling. A supervised Locality Preserving Projection (LPP) algorithm is employed to learn a common subspace into which both source and target data are projected. The learned subspace has favourable properties that projected samples in it are expected to be class discriminative and domain invariant. The subspace is learned based on the labelled source data and selected pseudo-labelled target samples. The pseudo labelled target samples are selected in a similar way to that used during model fine-tuning, i.e., the classwise selection is based on the pseudo label confidence and the number of selected pseudo labelled samples is linearly increased till the last iteration when all target samples are used for the subspace learning.

As for the classification, SPL uses the combination of Nearest Class Mean (NCM) [26] and the Structured Prediction (SP) algorithms. NCM calculates the class means in the learned subspace and assigns the nearest class to the target samples. SP takes advantage of the cluster structure of target samples in the learned subspace and finds the optimal one-to-one match between the clusters and classes.

IV. Experiments and Results

We conduct experiments on two benchmark datasets Office31 and Office-Home to show how the fine-tuned deep features can improve the UDA performance. We consider CNN models (i.e. ResNet50 and ResNet101 [1]) and transformer based models (i.e. DeiT-small and DeiT-base [15]) in our experiments. We will describe the details of datasets, experimental settings, experimental results and compare the results with other state-of-the-art methods in the following subsections.

A. Datasets

Office31 [44] consists of 4,110 images of 31 classes in three domains: Amazon (A), Webcam (W) and DSLR (D). Six domain adaptation tasks are used for the evaluation. **Office-Home** [45] consists of four domains: Artistic images (A), Clipart (C), Product images (P) and Real-World images (R). There are a total number of 15,588 images and 65 common object classes in four domains.

TABLE I
CLASSIFICATION ACCURACY (%) ON OFFICE31 DATASET USING VARYING MODELS AND METHODS.

Model	Method	A→W	D→W	W→D	A→D	D→A	W→A	Avg
ResNet50	noFT+SPL	92.7	98.7	99.8	93.0	76.4	76.8	89.6
	SourceOnlyFT	76.4	96.2	99.8	79.3	69.6	67.7	81.5
	SourceOnlyFT+SPL	93.0	98.6	99.8	95.0	78.0	77.2	**90.3**
	SourceTargetFT	81.9	97.0	99.8	82.9	68.7	66.7	82.8
	SourceTargetFT+SPL	92.3	98.7	99.8	93.8	78.5	77.4	90.1
ResNet101	noFT+SPL	90.1	99.0	99.8	92.4	79.8	78.5	89.8
	SourceOnlyFT	80.9	97.0	99.4	83.1	70.8	69.0	83.4
	SourceOnlyFT+SPL	93.2	99.0	99.8	94.4	80.3	78.3	**90.8**
	SourceTargetFT	83.1	93.0	98.2	87.8	69.5	68.3	83.3
	SourceTargetFT+SPL	93.1	99.0	99.8	94.8	79.5	78.4	**90.8**
DeiT-small	noFT+SPL	95.5	98.6	100.0	96.2	78.5	80.5	**91.6**
	SourceOnly	87.2	98.1	100.0	87.1	75.6	73.8	87.0
	SourceOnly+SPL	94.0	98.4	100.0	96.4	79.3	79.3	91.2
	SourceTargetFT	93.6	98.2	100.0	95.0	75.6	75.3	89.6
	SourceTargetFT+SPL	95.6	98.1	100.0	96.6	78.0	79.5	91.3
DeiT-base	noFT+SPL	96.9	99.1	100.0	96.4	82.4	81.0	92.6
	SourceOnlyFT	89.4	98.5	100.0	91.0	76.2	75.4	88.4
	SourceOnlyFT+SPL	97.9	99.1	100.0	97.6	82.6	81.1	93.0
	SourceTargetFT	94.7	98.4	100.0	96.6	77.1	76.5	90.6
	SourceTargetFT+SPL	97.6	99.1	100.0	98.0	81.3	82.4	**93.1**

TABLE II
CLASSIFICATION ACCURACY (%) ON OFFICE-HOME DATASET USING VARYING MODELS AND METHODS.

Model	Method	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Average
ResNet50	noFT+SPL	53.9	78.6	82.1	65.2	78.2	81.0	66.3	52.5	82.9	70.5	55.4	85.9	71.0
	SourceOnlyFT	44.3	67.2	74.4	55.1	66.9	67.5	55.0	42.2	74.3	64.2	44.9	76.4	61.0
	SourceOnlyFT+SPL	55.6	79.9	82.0	66.2	78.4	80.1	66.4	52.0	82.3	71.8	57.4	85.8	71.5
	SourceTargetFT	43.0	66.7	73.9	58.2	67.8	68.8	55.8	43.4	76.3	64.5	43.5	76.6	61.5
	SourceTargetFT+SPL	55.9	79.3	82.3	67.1	79.6	79.4	67.0	54.8	82.1	71.6	56.7	85.7	**71.8**
ResNet101	noFT+SPL	57.5	81.0	84.0	68.3	79.2	81.9	68.0	56.4	83.9	74.1	59.5	87.3	73.4
	SourceOnlyFT	48.3	69.2	76.0	57.8	67.7	70.5	55.6	44.6	75.6	66.6	48.1	78.8	63.2
	SourceOnlyFT+SPL	59.5	81.5	84.2	70.7	80.4	81.5	67.0	56.7	83.6	74.2	59.1	87.1	**73.8**
	SourceTargetFT	50.6	70.8	76.3	61.6	68.8	71.6	56.7	46.6	76.9	67.1	46.8	79.0	64.4
	SourceTargetFT+SPL	59.2	82.5	83.8	71.3	80.4	81.8	68.1	55.4	83.7	73.4	58.3	87.7	**73.8**
DeiT-small	noFT+SPL	58.5	85.6	85.2	73.3	84.2	84.6	71.0	58.5	85.5	76.3	59.3	87.2	75.8
	SourceOnly	56.4	75.7	81.7	71.0	75.0	78.0	67.7	52.0	81.8	74.3	53.2	83.7	70.9
	SourceOnly+SPL	62.7	82.9	85.1	76.5	84.1	83.8	75.0	55.1	85.4	76.8	58.4	87.0	76.1
	SourceTargetFT	59.6	77.3	82.9	73.6	78.4	80.3	71.3	56.7	83.7	73.4	54.8	85.3	73.1
	SourceTargetFT+SPL	63.8	82.9	85.2	76.2	84.4	83.8	75.2	61.5	85.8	77.4	60.0	87.2	**77.0**
DeiT-base	noFT+SPL	63.4	86.5	87.5	78.6	86.1	86.2	72.8	59.3	87.8	77.6	61.9	89.5	78.1
	SourceOnlyFT	60.9	79.2	84.1	73.4	78.6	80.7	71.0	55.1	84.3	77.6	58.4	85.8	74.1
	SourceOnlyFT+SPL	67.8	86.8	87.8	81.0	85.3	86.8	74.8	59.6	88.7	80.6	63.9	90.2	79.4
	SourceTargetFT	62.0	80.6	85.6	75.7	82.0	82.7	73.4	56.6	85.8	77.0	60.4	87.0	75.7
	SourceTargetFT+SPL	69.4	85.1	87.9	81.2	85.7	86.5	78.0	64.4	89.0	81.0	66.7	90.6	**80.5**

B. Experimental Settings

We conduct experiments with the following methods.

- **noFT+SPL**: the model pre-trained on ImageNet is used as a feature extractor to generate deep features (i.e. the activations of the second last layer) fed into the UDA approach SPL.
- **SourceOnlyFT**: the LP-FT scheme (c.f. Section III-A) is applied to the model and only the source data are used; the customized head (i.e. the last linear layer) serves as the domain-invariant classifier for the recognition of target samples and outputs the final results.
- **SourceOnlyFT+SPL**: the fine-tuned model (i.e. SourceOnlyFT) is used as a feature extractor to generate deep features (i.e. the activations of the second last layer) fed into the UDA approach SPL.
- **SourceTargetFT**: the LP-FT scheme (c.f. Section III-A) is applied to the model and the source data plus the progressively pseudo labelled target data are used; more specifically, the source data are used for the first 10 epochs of linear probing and the pseudo labelled target samples are progressively added for the fine-tuning from epoch 11 to 20; the customized head (i.e. the last linear layer) serves as the domain-invariant classifier for the recognition of target samples and outputs the final results.
- **SourceTargetFT+SPL**: the fine-tuned model (i.e. SourceTargetFT) is used as a feature extractor to generate deep features (i.e. the activations of the second last layer) fed into the UDA approach SPL.

C. Experimental Results

The experimental results of the dataset Office31 are shown in Table I. The classification accuracies of six adaptation tasks, as well as the average accuracy of six tasks, are reported. A

TABLE III
CLASSIFICATION ACCURACY (%) COMPARISON WITH STATE-OF-THE-ART METHODS ON THE OFFICE31 DATASET. * INDICATES THE RESULTS ARE PRODUCED BY [3].

Model (Backbone)	Method	A→W	D→W	W→D	A→D	D→A	W→A	Avg
ResNet50	RTN [35]	84.5	96.8	99.4	77.5	66.2	64.8	81.6
	MADA [11]	90.0	97.4	99.6	87.8	70.3	66.4	85.2
	GTA [36]	89.5	97.9	99.8	87.7	72.8	71.4	86.5
	iCAN [12]	92.5	98.8	100.0	90.1	72.1	69.9	87.2
	CDAN-E [37]	94.1	98.6	100.0	92.9	71.0	69.3	87.7
	JDDA [10]	82.6	95.2	99.7	79.8	57.4	66.7	80.2
	SymNets [38]	90.8	98.8	100.0	93.9	74.6	72.5	88.4
	TADA [39]	94.3	98.7	99.8	91.6	72.9	73.0	88.4
	MEDA [8]	86.2	97.2	99.4	85.3	72.4	74.0	85.7
	CAPLS [40]	90.6	98.6	99.6	88.6	75.4	76.3	88.2
	SPL [26]	92.7	98.7	99.8	93.0	76.4	76.8	89.6
	SourceTargetFT+SPL	92.3	98.7	99.8	93.8	78.5	77.4	90.1
ResNet101	SourceTargetFT+SPL	93.1	99.0	99.8	94.8	79.5	78.4	90.8
DeiT-small	CDTrans [3]	93.5	98.2	99.6	94.6	78.4	78.0	90.4
	SourceTargetFT+SPL	95.6	98.1	100.0	96.6	78.0	79.5	91.3
DeiT-base	SHOT* [41]	94.3	99.0	100.0	95.3	79.4	80.2	91.4
	CGDM* [42]	95.3	97.6	99.8	94.6	78.8	81.2	91.2
	CDTrans [3]	96.7	99.0	100.0	97.0	81.1	81.9	92.6
	SourceTargetFT+SPL	**97.6**	**99.1**	**100.0**	**98.0**	**81.3**	**82.4**	**93.1**

TABLE IV
CLASSIFICATION ACCURACY (%) COMPARISON WITH STATE-OF-THE-ART METHODS ON THE OFFICE-HOME DATASET. * INDICATES THE RESULTS ARE PRODUCED BY [3].

Model (Backbone)	Method	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Average
ResNet50	JAN [43]	45.9	61.2	68.9	50.4	59.7	61.0	45.8	43.4	70.3	63.9	52.4	76.8	58.3
	CDAN-E [37]	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
	SymNets [38]	47.7	72.9	78.5	64.2	71.3	74.2	64.2	48.8	79.5	74.5	52.6	82.7	67.6
	TADA [39]	53.1	72.3	77.2	59.1	71.2	72.1	59.7	53.1	78.4	72.4	60.0	82.9	67.6
	MEDA [8]	54.6	75.2	77.0	56.5	72.8	72.3	59.0	51.9	78.2	67.7	57.2	81.8	67.0
	CAPLS [40]	56.2	78.3	80.2	66.0	75.4	78.4	66.4	53.2	81.1	71.6	56.1	84.3	70.6
	SPL [26]	54.5	77.8	81.9	65.1	78.0	81.1	66.0	53.1	82.8	69.9	55.3	86.0	71.0
	SourceTargetFT+SPL	55.9	79.3	82.3	67.1	79.6	79.4	67.0	54.8	82.1	71.6	56.7	85.7	71.8
ResNet101	SourceTargetFT+SPL	59.2	82.5	83.8	71.3	80.4	81.8	68.1	55.4	83.7	73.4	58.3	87.7	73.8
DeiT-small	CDTrans [3]	60.6	79.5	82.4	75.6	81.0	82.3	72.5	56.7	84.4	77.0	59.1	85.5	74.7
	SourceTargetFT+SPL	63.8	82.9	85.2	76.2	84.4	83.8	75.2	61.5	85.8	77.4	60.0	87.2	77.0
DeiT-base	SHOT* [41]	67.1	83.5	85.5	76.6	83.4	83.7	76.3	**65.3**	85.3	80.4	66.7	83.4	78.1
	CGDM* [42]	67.1	83.9	85.4	77.2	83.3	83.7	74.6	64.7	85.6	79.3	**69.5**	87.7	78.5
	CDTrans [3]	68.8	85.0	86.9	**81.5**	**87.1**	**87.3**	**79.6**	63.3	88.2	**82.0**	66.0	**90.6**	**80.5**
	SourceTargetFT+SPL	**69.4**	**85.1**	**87.9**	81.2	85.7	86.5	78.0	64.4	**89.0**	81.0	66.7	**90.6**	**80.5**

few conclusions can be drawn from the results in Table I. First, when we use the customized head (i.e. the last linear layer for classification) as the classifier to output the final recognition results, the methods "SourceOnlyFT" and "SourceTargetFT" perform significantly worse than their counterparts with SPL. Second, when using SPL as the UDA approach, fine tuned deep features perform better than those extracted directly from the pre-trained model without fine-tuning. Whilst this is true for models like ResNet50 and ResNet101, there is no strong evidence for the models DeiT-small and DeiT-base. Third, the method "SourceTargetFT" outperforms "SourceOnlyFT", however, when the extracted features by these fine tuned models show no difference when fed into the UDA approach SPL, i.e., the method "SourceTargetFT+SPL" is no better than "SourceOnlyFT+SPL".

As for the experimental results on the Office-Home dataset (Table II), the first two conclusions we draw above still hold. However, the third conclusion needs to be revised since "SourceTargetFT" outperforms "SourceOnlyFT" in most cases whether the SPL approach is used or not.

The experimental results shown in Tables I and II provide evidence that properly fine tuned features are usually beneficial to UDA performance, especially when they are used in the feature transformation based UDA approaches such as SPL. In addition, using both source and target data in fine-tuning is usually superior to using source data only.

D. Comparison with SOTA

We compare our best method "SourceTargetFT+SPL" with state-of-the-art methods in Tables III and IV. For a fair comparison, the results of different methods are grouped according to the models (or backbones) and we compare results within and across groups.

For the Office31 dataset, when the ResNet50 model is used as the backbone, our best method achieves an average accuracy of 90.1% over six tasks and outperforms state-of-the-art methods. When the transformer based DeiT-small and DeiT-base models are employed, our best method achieves an average accuracy of 91.3% and 93.1%, respectively. The performance is higher than those of state-of-the-art methods

including the competitive CDTrans [3] which employs a triple-branch transformer framework.

On the Office-Home dataset, our best method also achieves the best performance regardless of the employed backbones. In particular, when the DeiT-small model is used, our method achieves an average accuracy of 77.0% which is significantly higher than the performance of the more complicated end-to-end deep learning based approach CDTrans (74.7%).

V. Conclusion

In this paper, we propose an effective fine-tuning mechanism for improved deep features for UDA. From the experimental results of our empirical studies, several conclusions can be drawn: properly fine-tuned deep features can improve the performance of feature transformation based domain adaptation approaches such as SPL [26]; it is usually beneficial to fine-tuning the models with both labelled source data and pseudo labelled target data; special care needs to be taken for batch normalisation layers in the deep CNN models during fine-tuning; and the performance gap between transformer and CNN based approaches is attributed to the improved deep features extracted from transformer based models.

In the future work, we will continue investigating how the feature transformation based UDA approaches can be enhanced with features extracted from more advanced deep models. In addition, we will investigate how to effectively integrate features extracted from different models into a unique feature transformation based framework for improved UDA.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[2] J. Yang, J. Liu, N. Xu, and J. Huang, "Tvt: Transferable vision transformer for unsupervised domain adaptation," *arXiv preprint arXiv:2108.05988*, 2021.

[3] T. Xu, W. Chen, P. Wang, F. Wang, H. Li, and R. Jin, "Cdtrans: Cross-domain transformer for unsupervised domain adaptation," in *International Conference on Learning Representations*, 2021.

[4] Q. Wang, F. Meng, and T. P. Breckon, "Data augmentation with norm-ae and selective pseudo-labelling for unsupervised domain adaptation," *Neural Networks*, vol. 161, pp. 614–625, 2023.

[5] J. Zhang, W. Li, and P. Ogunbona, "Joint geometrical and statistical alignment for visual domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2017, pp. 5150–5158.

[6] M. Ghifary, D. Balduzzi, W. B. Kleijn, and M. Zhang, "Scatter component analysis: A unified framework for domain adaptation and domain generalization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1414–1430, 2016.

[7] B. Sun, J. Feng, and K. Saenko, "Correlation alignment for unsupervised domain adaptation," in *Domain Adaptation in Computer Vision Applications*, 2017, pp. 153–171.

[8] J. Wang, W. Feng, Y. Chen, H. Yu, M. Huang, and P. S. Yu, "Visual domain adaptation with manifold embedded distribution alignment," in *2018 ACM Multimedia Conference on Multimedia Conference*. ACM, 2018, pp. 402–410.

[9] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International Conference on Machine Learning*, 2015, pp. 1180–1189.

[10] C. Chen, Z. Chen, B. Jiang, and X. Jin, "Joint domain alignment and discriminative feature learning for unsupervised deep domain adaptation," in *AAAI Conference on Artificial Intelligence*, 2019.

[11] Z. Pei, Z. Cao, M. Long, and J. Wang, "Multi-adversarial domain adaptation," in *AAAI Conference on Artificial Intelligence*, 2018.

[12] W. Zhang, W. Ouyang, W. Li, and D. Xu, "Collaborative and adversarial network for unsupervised domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3801–3809.

[13] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.

[14] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[15] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International Conference on Machine Learning*. PMLR, 2021, pp. 10 347–10 357.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[17] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[18] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.

[19] P. Peng and J. Wang, "How to fine-tune deep neural networks in few-shot learning?" *arXiv preprint arXiv:2012.00204*, 2020.

[20] J. Frankle, D. J. Schwab, and A. S. Morcos, "Training batchnorm and only batchnorm: On the expressive power of random features in cnns," in *International Conference on Learning Representations*, 2020.

[21] F. Kanavati and M. Tsuneki, "Partial transfusion: on the expressive influence of trainable batch norm parameters for transfer learning," in *Medical Imaging with Deep Learning*. PMLR, 2021, pp. 338–353.

[22] Y. Li, N. Wang, J. Shi, X. Hou, and J. Liu, "Adaptive batch normalization for practical domain adaptation," *Pattern Recognition*, vol. 80, pp. 109–117, 2018.

[23] R. Romijnders, P. Meletis, and G. Dubbelman, "A domain agnostic normalization layer for unsupervised adversarial domain adaptation," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 1866–1875.

[24] M. Klingner, J.-A. Termöhlen, J. Ritterbach, and T. Fingscheidt, "Unsupervised batchnorm adaptation (ubna): A domain adaptation method for semantic segmentation without using source domain representations," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 210–220.

[25] A. Kumar, A. Raghunathan, R. Jones, T. Ma, and P. Liang, "Fine-tuning can distort pretrained features and underperform out-of-distribution," in *International Conference on Learning Representations*, 2022.

[26] Q. Wang and T. P. Breckon, "Unsupervised domain adaptation via structured prediction based selective pseudo-labeling," in *AAAI Conference on Artificial Intelligence*, 2020.

[27] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," in *International Conference on Machine Learning*. JMLR. org, 2015, pp. 97–105.

[28] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang, "Convolutional neural networks for medical image analysis: Full training or fine tuning?" *IEEE transactions on medical imaging*, vol. 35, no. 5, pp. 1299–1312, 2016.

[29] E. Cetinic, T. Lipic, and S. Grgic, "Fine-tuning convolutional neural networks for fine art classification," *Expert Systems with Applications*, vol. 114, pp. 107–118, 2018.

[30] Y. Guo, H. Shi, A. Kumar, K. Grauman, T. Rosing, and R. Feris, "Spottune: transfer learning through adaptive fine-tuning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4805–4814.

[31] G. Vrbančič and V. Podgorelec, "Transfer learning with adaptive fine-tuning," *IEEE Access*, vol. 8, pp. 196 197–196 211, 2020.

[32] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever, "Generative pretraining from pixels," in *International conference on machine learning*. PMLR, 2020, pp. 1691–1703.

[33] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 9650–9660.

[34] J. Zhou, C. Wei, H. Wang, W. Shen, C. Xie, A. Yuille, and T. Kong, "ibot: Image bert pre-training with online tokenizer," in *International Conference on Learning Representations*, 2022.
[35] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Unsupervised domain adaptation with residual transfer networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 136–144.
[36] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa, "Generate to adapt: Aligning domains using generative adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.
[37] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Advances in Neural Information Processing Systems*, 2018, pp. 1647–1657.
[38] Y. Zhang, H. Tang, K. Jia, and M. Tan, "Domain-symmetric networks for adversarial domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5031–5040.
[39] X. Wang, L. Li, W. Ye, M. Long, and J. Wang, "Transferable attention for domain adaptation," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.
[40] Q. Wang, P. Bu, and T. P. Breckon, "Unifying unsupervised domain adaptation and zero-shot visual recognition." in *International Joint Conference on Neural Networks*, 2019.
[41] J. Liang, D. Hu, and J. Feng, "Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation," in *International Conference on Machine Learning*. PMLR, 2020, pp. 6028–6039.
[42] Z. Du, J. Li, H. Su, L. Zhu, and K. Lu, "Cross-domain gradient discrepancy minimization for unsupervised domain adaptation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 3937–3946.
[43] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *International Conference on Machine Learning*, 2017, pp. 2208–2217.
[44] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *European Conference on Computer Vision*. Springer, 2010, pp. 213–226.
[45] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5018–5027.