Sample Selection for Universal Domain Adaptation

Omri Lifshitz, Lior Wolf
Tel-Aviv University
omri.lifshtz@gmail.com, liorwolf@gmail.com

Abstract

This paper studies the problem of unsupervised domain adaptation in the universal scenario, in which only some of the classes are shared between the source and target domains. We present a scoring scheme that is effective in identifying the samples of the shared classes. The score is used to select samples in the target domain for which to apply specific losses during training; pseudo-labels for high scoring samples and confidence regularization for low scoring samples. Taken together, our method is shown to outperform, by a sizeable margin, the current state of the art on the literature benchmarks.

Introduction

In real world situations, the necessity of applying domain adaptation is the rule and not the exception, since "no man ever steps in the same river twice". This is true not only for the input samples, whose distribution is likely to change both because of the shifting setting and due to the practical considerations of collecting training samples, but also with regards to the output labels. In many cases, the classes seen and labeled during training differ from those encountered during the deployment phase.

Unsupervised domain adaptation seeks to learn a classifier in a source domain in which supervised training samples exist, such that it would be effective in a target domain for which only unsupervised samples exist. Universal domain adaptation (UniDA) adds the challenge that some of the classes in the source domain do not appear in the target domain and vice versa. Therefore, the classifier, when applied to the target domain, has to classify only according to the relevant classes, and also identify the samples that belong to the classes that are unique to the target domain.

Our method is based on three losses. The first loss is the conventional domain confusion loss, which encourages the representation of the samples to be domain agnostic. The second is the pseudo-labeling loss, which is a very common loss in semi-supervised learning and, in particular, in unsupervised domain adaptation. However, the application of pseudo-labels in the UniDA setting requires additional care, since labeling every sample is almost guaranteed to lead to adverse results. In other words, assigning wrong pseudo-labels leads to "negative transfer", which occurs when incorrectly applying the knowledge from the source domain to the target domain, thus lowering the classification accuracy. We, therefore, propose to identify the samples in the target domain for which the labels are likely to be in the set of shared classes.

The third loss we apply is the confidence regularization loss, which encourages target samples that are likely to be from classes that appear only in the target domain ("private classes") to be classified with lower confidence. This regularization term is especially important in the UniDA case, in which there is no prior assumption on the label set relation between the source and target domains and thus there is a high risk of negative transfer. Similarly to the pseudo-labeling scheme above, sample selection is needed, since lowering the confidence of samples from the shared set of classes would lead to a decrease in performance.

In our work, the samples from the target domain that are likely to be from the shared classes are identified based on two signals. The first is the certainty of the classifier, assuming that the classifier is more likely to be confused when encountering samples from unseen classes. The second is the similarity between samples from source and target domains, assuming that there is a higher similarity between samples from the shared classes. We, therefore, suggest a scoring scheme that combines the outputs of both the label classifier and the domain classifier.

Our experiments show that our scoring scheme based on the aforementioned signals together with the three loss terms improves the state of the art accuracy in the UniDA scenario.

Our main contributions are: (i) a direct method for UniDA, which employs selective pseudo-labels as the main loss, (ii) encouraging score separation using the confidence regularization, (iii) a new sample scoring scheme that outperforms the weights employed in the literature, and (iv) state of the art results across datasets and benchmarks.

Related Work

The problem of unsupervised domain adaptation can be divided into four different categories, based on the relation between the label sets of the source and target domains: closed-set, open-set, partial and universal.

Closed-set domain adaptation is a scenario where the source and target domains share the same label set. The main challenge in this scenario is to overcome the *domain gap* that comes as a result of the samples being taken from different distributions. There are two common approaches to the close-set problem: feature adaptation and generative models. Generative based approaches (Bousmalis et al. 2017; Sankaranarayanan et al. 2018; Hu et al. 2018; Liu et al. 2018; Murez et al. 2018; Huang et al. 2018; Volpi et al. 2018) attempt to generate labeled target samples from the source samples. Methods based on CycleGAN (Zhu et al. 2017) generate synthetic target-like samples from the source domain and source-like samples from the target to train classifiers on each of the domains (Hoffman et al. 2018; Russo et al. 2018).

Methods based on feature adaptation aim to reduce the discrepancy between the feature distribution of samples from the source and target domains. In work by Ganin et al. (2016), a domain adversarial network is introduced and added to a classifier network with the purpose of creating features that are indiscriminate with respect to a shift between domains, yet still discriminative for the main classification task. By introducing a gradient reversal unit, the feature extractor is trained to produce features that confuse the domain classifier. **Open-set** domain adaptation, proposed by Busto, Iqbal, and Gall (2020) assumes knowledge of the shared label set between the source and target domains, while all private label sets are marked as "unknown". A modification by Saito et al. (2018) requires no data from the private source label set.**Partial-set** domain adaptation assumes that the target domain's label set is a subset of the source's label set. Cao et al. (2018) employ adversarial distribution matching by using a number of domain discriminators together with a weighting scheme at both the class and instance level. Zhang et al. (2018) use an adversarial method to identify the source samples that are potentially from the private target label set. Cao et al. (2018) further improve the results by using a single domain adversarial network and down-weighting the data of the source private set during training. **Universal** domain adaptation was introduced by You et al. (2019) and unlike the aforementioned scenarios, it does not assume any prior knowledge about the relation between the source and target label sets. This setting is also addressed by Fu et al. (2020) and by Saito et al. (2020) via neighborhood clustering. Kundu et al. (2020) introduce a two-stage learning process where only one domain is available at each stage.

Method Comparison We employ the universal setting, which is the most generic one. Our method greatly differs technically from the previous work in this and in other settings. In Tab. 1 we summarize the main differences from a representative selection of other methods (including all UniDA work we are aware of). The table presents the number of domain classifiers used as well as the weighting or scoring scheme (if applicable) and the loss terms. We can observe the uniqueness of our method, as well as the diversity in the existing literature. One thing that separates our method from other previously used methods is the use of "selective-sample" losses. Instead of down-weighting the losses according to a weighting scheme, we use the scoring scheme in order to select samples for calculating the losses. This idea goes hand in hand with the use of pseudo-labels and our regularization term. In addition, we are the only UniDA-setting method to employ a single domain classification loss, where the other methods either omit this loss or use it twice. With the exception of the domain classification loss, our losses are entirely different from previous works.

Method	Uni	DA	Weight/scoring scheme	Loss terms
DANN (Ganin et al. 2016)		1	None	CE, DA
OSBP (Saito et al. 2018)		0	None	CE, binary CE
PADA (Cao et al. 2018)		1	Class weights: $\gamma = \frac{1}{n_t} \sum_{i=1}^{n_t} \bar{y}_i$	CE, weighted DA
UAN(You et al. 2019)	✓	2	Weight for source: $w^s(x) = \frac{H(\bar{y})}{\log \lvert Y_S \rvert} - d(x)$ Weight for target: $w^t(x) = d(x) - \frac{H(\bar{y})}{\log \lvert Y_S \rvert}$	CE, DA, weighted domain non-adversarial
USFDA (Kundu et al. 2020)	✓	0	Weight for positive: $w(x) = \exp max(\bar{y})$ Weight for negative: $w(x) = \exp max(1 - \bar{y})$	Generated negative samples, weighted CE, weighted entropy
Ours	✓	1	No weighting, selection scores: $s(x) = d(x) + \max \bar{y}(x)$	CE, sample-selective pseudo-labels, DA, sample-selection confidence regularization

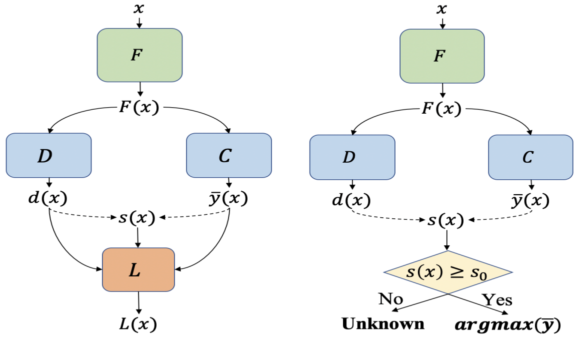
Table 1: Domain adaptation methods. Uni=universal. DA=domain adversarial terms. CE=cross entropy.

Pseudo-labels refers to the use of predicted labels as though they were the correct labels during training. This is a simple yet effective tool used in closed-set domain adaptation, in order to learn categorical representation of the target domain (French, Mackiewicz, and Fisher 2018; Saito, Ushiku, and Harada 2017; Sener et al. 2016; Shu et al. 2018; Zhang et al. 2018; Choi et al. 2019). Although the use of pseudo-labels during training can greatly improve the final outcome of the network, false pseudo-labels may lead to negative transfer, which is a major concern in UniDA.

Method

We follow the setting of UniDA proposed by You et al. (2019). During training, we are provided with a source domain $D_s = \{(x_i^s, y_i^s) \sim p\}_{i=1}^{n_s}$ of labeled data sampled from distribution p and a target domain $D_t = \{(x_i^t) |\ x_i^t \sim q^x\}_{i=1}^{n_t}$ of unlabeled data sampled from distribution q^x, which is the marginalization of the distribution q of samples and their labels in the target domain. We denote by Y_s (Y_t) the label set of the source (target) domain. The shared label set is denoted by $Y = Y_s \cap Y_t$. For convenience, we denote the private label sets of the source and target domain in the following manner: $\overline{Y_s} = Y_s \setminus Y$ and $\overline{Y_t} = Y_t \setminus Y$, respectively.

UniDA generalizes all other variants of domain adaptation. Namely, the partial-set case in which the target classes are a subset of the source classes (closed-set is a special case of partial-set), and the open-set case in which the source classes



(a) (b)

Figure 1: Architecture of the network during training and deployment. Components in green are encoders, blue are classifiers, orange are loss components and yellow are conditions. During the training stage (a), the score along with the label and domain classification is used to calculate the loss. During the deployment stage (b) the scores are used as a threshold to decide whether the sample is from the shared label set or should be marked as unknown.

are a subset of the target classes. The latter case is the more challenging of the two as some of the target domain samples cannot be adapted to match the samples seen during training.

The Jaccard index of the label sets of the two domains, $\xi = \frac{|Y|}{|Y_s \cup Y_t|}$, is used to measure the overlap in classes. The objective in the UniDA scenario is to create a model g that maximizes the target classification on the shared label set, as well as distinguishes between samples with labels from Y and those in $\overline{Y_t}$. i.e.

$$\max_g E_{(x,y)\sim q}[g(x) = t(y)] \tag{1}$$

where

$$t(y) = \begin{cases} y & \text{if } y \in Y \\ \tau & \text{if } y \in \overline{Y_t} \end{cases} \tag{2}$$

and τ is the symbol used to mark unknown classes not seen in the labeled training set D_s.

The Sub-networks The architecture we employ is shown in Fig. 1. It consists of a domain classifier D, a feature extractor F, and a label classifier C. By using one adversarial domain classifier D, our method is simpler than previous work (You et al. 2019), which uses two domain classifiers.

Input x (from both domains) is fed into the feature extractor F, yielding the feature vector $F(x)$. $F(x)$ is, in turn, fed to both the domain classifier D and the label classifier C. The label classifier outputs the label prediction of classes from the source domain $\overline{y}(x) = C(F(x)) \in \mathbb{R}^{|Y_s|}$, which is a vector of pseudo probabilities obtained by the softmax function. The adversarial domain classifier yields the probability of the sample being from the source domain $d(x) = D(F(x)) \in [0, 1]$. The results from both classifiers are used for calculating the *sample transfer score* and for calculating the losses.

The sample transfer score, $s(x)$, estimates the confidence that x is from the shared label set. The score is calculated using the prediction $\max \overline{y}(x)$ and the domain classification $d(x)$ as detailed below. A higher value of $s(x)$ indicates that the sample x appears to be from the shared label set and that the correct label was identified.

During the deployment stage, the test sample undergoes the same path as before, but rather than calculating losses, we use the score $s(x)$ as a threshold to decide whether we should predict a class or label the sample as the symbol τ that represents all labels unseen during training. We use a hyper-parameter s_0 and output the class label according to the following:

$$y(x) = \begin{cases} \arg\max \overline{y} & s(x) > s_0 \\ \tau & \text{otherwise} \end{cases} \tag{3}$$

The Sample Transfer Score

We define a scoring mechanism that represents the confidence that a sample x is from the shared label set Y. This score is used in both training and deployment. During training, the scores are used as a threshold for losses on samples from the target domain, as explained in the following sections. During deployment, the scores are used in order to decide whether or not a sample should be labeled as τ or predicted from one of the classes in the source label set, as shown in Eq. 3.

The score is a combination of two signals: (i) the confidence in the classification label, as it manifests itself in the vector of pseudo probabilities $\overline{y}(x)$, and (ii) the estimation of the probability of it being in the source domain, as is estimated by $d(x)$. The usage of the second signal on target domain samples, is meant to measure the similarity of these samples to the source domain samples. Naturally, target samples that are more similar to the source domain samples are more likely to be in the shared label set.

It is reasonable to expect that

$$\begin{aligned} \mathbb{E}_{(x,y)\in p} \max \overline{y}(x) &> \mathbb{E}_{(x,y)\in q | y \in Y} \max \overline{y}(x) \\ &> \mathbb{E}_{(x,y)\in q | y \in \overline{Y_t}} \max \overline{y}(x) \end{aligned} \tag{4}$$

In other words, the maximal value of the pseudo probability can be used as a measure for identifying the target samples that have labels in Y. We, therefore, derive the following scoring mechanism for target samples:

$$s(x) = d(x) + \max \overline{y}(x) \tag{5}$$

Let us notice that as $d(x) \in [0, 1]$ (higher values for source samples) and $\max \overline{y}(x) \in [0, 1]$ it holds that $s(x) \in [0, 2]$.

You et al. (2019) propose to use a different scoring scheme and apply it as weights for training the second domain classifier they use (we do not employ this component). Their scoring scheme employs the following scores to target domain samples

$$w_t(x) = d(x) - \frac{H(\overline{y}(x))}{\log |Y_s|} \tag{6}$$

where $H(\overline{y}(x))$ is the entropy of vector $\overline{y}(x)$. In their work, source domain samples are also scored, by the score $w_s(x) = -w_t(x)$, while we only select target samples as detailed below. Nevertheless, despite using scoring for completely different losses and to different sets of samples, we explore empirically the replacement of our scoring mechanism $s(x)$ with their $w_t(x)$ and demonstrate that our scheme is superior by a sizable margin.

Sample Selective Pseudo-labels In order to utilize the unlabeled data as much as possible, we opt to use pseudo-labels. As explained by Choi et al. (2019), pseudo-labels can be an extremely simple yet effective tool when training a network in a semi-supervised scenario. The difficulty with pseudo-labels in the UniDA scenario is the high risk of negative transfer, i.e., decreasing the classifier's performance due to the incorporation of false supervision. In the universal scenario, the target label set is unknown and, therefore, assuming that the network's classification is correct is even more likely to be detrimental than in the conventional domain adaptation case.

In order to deal with the risk of negative transfer, our approach is to use pseudo-labels only on high confidence samples that are likely to be in the shared label set Y. As a confidence measure, we employ the sample's transfer score, $s(x)$, and only use pseudo-labels for samples where $s(x)$ is above a certain threshold. We use the following dynamic threshold, $s_\alpha(t)$ during training:

$$s_\alpha(t) = \frac{2+s_0}{2} - \frac{t}{T} \cdot (\frac{2+s_0}{2} - s_0) \tag{7}$$

where t is the current training step and T is the total number of training steps. A dynamic threshold is used to avoid negative transfer; we begin at the midpoint between s_0 and the maximal value of $s(x)$, and as the training advances the network better classifies samples with the threshold s_0 and thus it is reasonable to lower the threshold further.

Our pseudo-label classification loss is the following:

$$\begin{aligned} L_C =& \mathbb{E}_{(x,y)\sim p}[L_{CE}(y, \bar{y}(x))] + \\ & \gamma \cdot \mathbb{E}_{(x,y)\sim q}[\mathbb{1}_{s(x)>s_\alpha(t)} \cdot L_{CE}(\arg\max \bar{y}(x), \bar{y}(x))] \end{aligned} \tag{8}$$

where L_{CE} is the cross-entropy, γ is a trade-off parameter and $\mathbb{1}$ is the 0-1 indicator function.

Sample Selective Confidence Regularization In order to enforce a better separation between the scores of samples from the private and shared sets and to reduce the amount of negative transfer, we employ a novel regularization term aimed to decrease the network's confidence for samples likely to be from the target private set. Recall from Eq. 5 that the score for any given sample is the result of its domain similarity and the predicted class confidence. Thus, in order to better separate between samples from the shared and private classes we encourage samples with a low score to have a low predicted class confidence.

As with the pseudo-labels, we opt for a sample selective approach in which we only apply the confidence-lowering loss to a subset of the target samples in the batch, specifically to samples for which the sample transfer score $s(x)$ is below a threshold s_β. Denote by $\overline{B_t}$ the target sample in the batch B for which $s(x) < s_\beta$. We define the following loss term:

$$L_{CR} = \sum_{j=1}^{|Y_s|} \Big(\frac{1}{|\overline{B_t}|} \sum_{i \in \overline{B_t}} \bar{y}(x_i)_j \Big)^2 \tag{9}$$

where $\bar{y}(x_i)_j$ denotes the classifier's pseudo-probability associated with label j given a sample x_i. Adding this penalty to the network's objective reduces the prediction confidence of samples with low transfer scores by encouraging a solution that is more uniformly distributed across the different classes.

Here we also use a dynamic threshold during training, $s_\beta(t)$, that changes according to the following:

$$s_\beta(t) = \frac{s_0}{2} + \frac{t}{T} \cdot (s_0 - \frac{s_0}{2}) \tag{10}$$

where t is the current training step and T is the total number of training steps. Using the same reasoning as before, at first the threshold is set at the midpoint between s_0 and 0 in order to decrease the confidence only for samples that already have a very low transfer score. As the training advances, the classification to private-set and shared-set samples, that occurs by comparing $s(x)$ with s_0 becomes more accurate, and the threshold becomes closer to s_0.

Domain Adversarial Loss

In addition to the losses described above, we also use the conventional adversarial domain loss first introduced by Ganin et al. (2016). The domain classifier's network, D, is trained with a binary cross-entropy loss and a gradient reversal layer is used when backpropagating to network F.

$$L_{DA} = \mathbb{E}_{(x,y)\sim p} L_{CE}(1, d(x)) + \mathbb{E}_{(x,y)\sim q} L_{CE}(0, d(x)) \tag{11}$$

The final loss has the following unweighted components:

$$L = L_C + L_{CR} - L_{DA} \tag{12}$$

Experiments

Following You et al. (2019), we use four datasets. **Office-Home** (Venkateswara et al. 2017) is a dataset made up of 65 different classes from four domains: Artistic (Ar), Clipart (Cl), Product (Pr) and Real-world images (RW). Keeping in line with You et al. (2019), we test each combination of source and target domain by setting the first 10 classes in alphabetical order as the shared label set Y, the next five as the source private, $\overline{Y_s}$, and the rest of the classes (50 classes) are the private target, $\overline{Y_t}$. **Office31** (Saenko et al. 2010) consists of three domains, each with 31 classes. The domains are Amazon (**A**), DSLR (**D**) and Webcam (**W**). The 10 shared classes between this dataset and Caltech-256 (Griffin, Holub, and Perona 2006) are used as the shared label set. Aside from these classes, we set the first 10 classes in alphabetical order as $\overline{Y_s}$ and the last 11 classes as $\overline{Y_t}$. **VisDA2017** (Peng et al. 2018) is a dataset with a single source and target domain testing the ability to perform transfer learning from synthetic images to natural images. The dataset has 12 classes identical in each domain; we use the first six as the shared label set, the next three as the private source label set and the last three as the private target label set. **ImageNet-Caltech** employs Imagenet-1K (Deng et al. 2009) with 1000 different classes and Caltech-256 (Griffin, Holub, and Perona 2006) with 256 classes. The shared label set is comprised of the 84 shared classes between the two datasets, while the source and target private label sets are all other classes in each dataset.

Evaluation protocol The protocol of the Open-Set challenge in VisDA2018 is employed. After the training stage, the model is tested only on samples from the target domain. The network must classify the test data into $|Y|+1$ different classes, where the last label τ contains all labels from the target domain's private label set. As detailed above, our network tries to classify using the labels from the source domain and only classifies into the "unknown" class if the sample's transfer score is lower than a predetermined threshold.

Implementation details The architecture of F, C, and D follows that of You et al. (2019) in order to provide a direct comparison with this previous work. The method is implemented in Pytorch using a ResNet-50 model (He et al. 2016), pretrained on ImageNet (Deng et al. 2009), as the backbone feature extractor F. The label classifier network, C, is a fully connected network with a single layer used to classify the features $F(x)$. The domain classifier network, D, is a three-layer MLP with ReLU activations.

Our method enjoys a very limited number of hyperparameters. Early on during the development process, we fixed the following hyperparameters across all datasets: $\gamma = 0.6$ and $s_0 = 1.0$. We provide parameter sensitivity experiments to demonstrate the robustness of the method to its parameters.

Classification Results

We compare our approach with prior methods in the UniDA setting. Tab. 2, 3 present the results on the acceptable benchmarks of the field. The success rate for methods other than ours (with the exception of USFDA (Kundu et al. 2020)) is taken from You et al. (2019). As can be observed, our approach achieves state of the art results on the majority of the domain adaptation tasks across the different datasets.

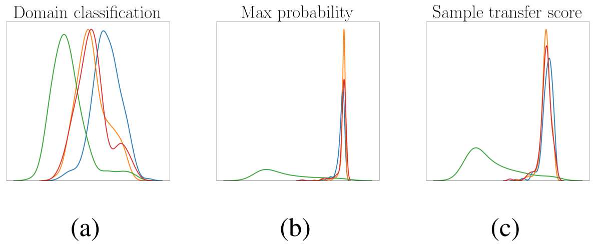
Scoring Scheme Analysis

In Fig. 2 we present the estimated probability density function for the different components of $s(x)$ on the Office31 dataset for the domain shift **W**→**D**. $d(x)$, shown in Fig. 2(a), displays the following expected behavior:

$$\mathbb{E}_{(x,y)\in p|y\in\overline{Y_s}}d(x) > \mathbb{E}_{(x,y)\in p|y\in Y}d(x) \approx \mathbb{E}_{(x,y)\in q|y\in Y}d(x) > \mathbb{E}_{(x,y)\in q|y\in\overline{Y_t}}d(x) \qquad (13)$$

In Fig. 2(b) we analyze the max probability of the classifier, $\max \bar{y}(x)$, validating the hypothesis in Eq. 4 and justifying using this component as part of our scoring scheme. Finally, in Fig. 2(c), we present the full sample transfer score $s(x)$. The results show that target samples with higher scores $s(x)$ are typically from the shared label set. This justifies the use of our scoring scheme to distinguish between samples that we can predict correctly and those that should be labeled τ.

Comparing Scoring Schemes We next compare our proposed scoring scheme $s(x)$, as shown in Eq. 5, to a scoring scheme that is based on the weight proposed by You et al. (2019), $w_t(x)$ given by Eq. 6. In order to compare the two scoring schemes, we use the score $w_t(x)$ proposed on the target samples instead of $s(x)$. The method that uses $w_t(x)$ was tuned to optimize its performance. In addition to the scoring scheme based on the weight $w_t(x)$, we also compare to an entropy based one, since entropy has been shown to be a good criterion in domain adaptation (Grandvalet and Bengio 2004; Long et al. 2016). Based on the assumption that the target samples from the shared label set are similar to the source samples and will thus have a lower entropy, we define the following scoring scheme:



Domain classification

Max probability

Sample transfer score

Figure 2: Distributions of the different components of the scoring scheme on the four following sample groups: source samples in Y (orange), source sample in $\overline{Y_s}$ (blue), target samples in Y (red) and target samples in $\overline{Y_t}$ (green). (a) Distribution of the domain classifier's output $d(x)$. (b) The label classifier's maximum probability, $\max \bar{y}(x)$. (c) The score $s(x)$, which combines both. All distributions shown are presented using the Gaussian kernel density estimator.

$$s_h(x) = 1 - \frac{H(\bar{y}(x))}{\log |Y_s|} \qquad (14)$$

The comparison on the Office31 dataset is shown in Tab. 4. Clearly, our scoring scheme produces superior results across the entire dataset. We thus conclude that our scoring mechanism outperforms the one proposed by You et al. (2019) and $s_h(x)$ when used in the context of our method.

Tab. 4 also presents an ablation study on the components of the scoring mechanism. "$s(x)$ w/o $d(x)$" refers to the score function when removing the domain factor $d(x)$ from Eq. 5 and "$s(x)$ w/o $\max \bar{y}(x)$" to the score function when removing the classification component. The results show that both components are necessary for achieving our final results. However, the classification component is more crucial to the success of our scoring mechanism, and by itself already outperforms the state of the art.

Comparing Regularization and Pseudo-labels We next compare our regularization term and pseudo-labels scheme to similar methods from the literature. We compare the confidence regularization scheme to the widely used entropy maximization loss and our selective pseudo-labels to that proposed by Zou et al. (2019). The results for the Office31 dataset are shown in Tab. 5. It is clear that our method achieves better results across the entire dataset.

Parameter Sensitivity

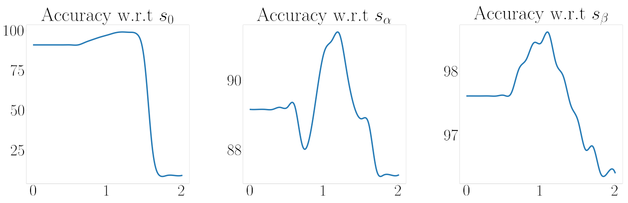
Pseudo-label Threshold Analysis We study the sensitivity of our method to the threshold s_α, which is used to determine whether or not the pseudo-label of a target sample should be taken into consideration when calculating its loss. While our method employs a dynamic threshold, in order to obtain a clearer image, we perform the experiment when the threshold is fixed. We compare the average accuracy on the

Method	Office-Home												
	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg
ResNet (He et al. 2016)	59.37	76.58	87.48	68.86	71.11	81.66	73.72	56.30	86.07	78.68	59.22	78.59	73.22
DANN (Ganin et al. 2016)	56.17	81.72	85.87	68.67	73.38	83.76	69.92	56.84	85.80	79.41	57.26	78.26	73.17
RTN (Long et al. 2016)	50.46	77.80	86.90	65.12	73.40	85.07	67.86	45.23	85.50	79.20	55.55	78.79	70.91
IWAN (Zhang et al. 2018)	52.55	81.40	86.51	70.58	70.99	85.29	74.88	57.33	85.07	77.48	59.65	79.91	73.39
PADA (Cao et al. 2018)	39.59	69.37	76.26	62.57	67.39	77.47	48.39	35.79	79.60	75.94	44.50	78.10	62.91
ATI (Busto, Iqbal, and Gall 2020)	52.90	80.37	85.91	71.08	72.41	84.39	74.28	57.84	85.61	76.06	60.17	78.42	73.29
OSBP (Saito et al. 2018)	47.75	60.90	76.78	59.23	61.58	74.33	61.67	44.50	79.31	70.59	54.95	75.18	63.90
UAN (You et al. 2019)	63.00	82.83	87.85	76.88	78.70	85.36	78.22	58.59	86.80	**83.37**	**63.17**	79.43	77.02
USFDA (Kundu et al. 2020)	63.35	83.3	89.35	70.96	72.34	86.09	78.53	**60.15**	87.35	81.56	**63.17**	88.23	77.03
Ours	**65.07**	**86.38**	**91.41**	**79.45**	**84.86**	**89.61**	**82.00**	56.80	**89.81**	79.52	61.47	**89.03**	**79.62**

Table 2: Average class accuracy (%) on the Office-Home ($\xi = 0.15$). The results for all methods besides USFDA (Kundu et al. 2020) and ours are taken from You et al. (2019)

Method	Office31							ImageNet-Caltech			VisDA2017
	A → W	D → W	W → D	A → D	D → A	W → A	Avg	I → C	C → I	Avg	
ResNet (He et al. 2016)	75.94	89.60	90.91	80.45	78.83	81.42	82.86	70.28	65.14	67.71	52.80
DANN (Ganin et al. 2016)	80.65	80.94	88.07	82.67	74.82	83.54	81.78	71.37	66.54	68.96	52.94
RTN (Long et al. 2016)	85.70	87.80	88.91	82.69	74.64	83.26	84.18	71.94	66.15	69.05	53.92
IWAN (Zhang et al. 2018)	85.25	90.09	90.00	84.27	84.22	86.25	86.68	72.19	66.48	69.34	58.72
PADA (Cao et al. 2018)	85.37	79.26	90.91	81.68	55.32	82.61	79.19	65.47	58.73	62.10	44.98
ATI (Busto, Iqbal, and Gall 2020)	79.38	92.60	90.08	84.40	78.85	81.57	84.48	71.59	67.36	69.48	54.81
OSBP (Saito et al. 2018)	66.13	73.57	85.62	72.92	47.35	60.48	67.68	62.08	55.48	58.78	30.26
UAN (You et al. 2019)	85.62	94.77	97.99	86.50	85.45	85.12	89.24	75.28	70.17	72.73	60.83
USFDA (Kundu et al. 2020)	85.56	95.20	97.79	88.47	87.5	86.61	90.19	**76.85**	72.13	74.49	63.92
Ours	**90.11**	**95.33**	**98.18**	**90.56**	**90.03**	**90.45**	**92.44**	76.13	**73.75**	**74.94**	**66.83**

Table 3: Average class accuracy for Office31($\xi = 0.32$), ImageNet-Caltech ($\xi = 0.07$) and VisDA2017($\xi = 0.50$)



(a) (b) (c)

Figure 3: (a) Accuracy w.r.t s_0 on Office31 W to D. (b) Accuracy w.r.t s_α on Office31 A to D. (c) Accuracy w.r.t. threshold s_β on Office31 W to D.

Office31 dataset with the domain shift **A→D**. The tests are conducted by fixing all other hyperparameters to the default values and only changing the value of s_α. The results are presented in Fig. 3(b). By taking the lowest threshold possible, $s_\alpha = 0$, we allow the use of pseudo-labels on every sample seen during the training stage. As can be seen from the performance graph, this yields a lower result than higher thresholds, probably due to negative transfer. This is also evident when examining the results of Tab. 6 when setting the threshold $s_\alpha = 0$. The second edge case is $s_\alpha = 2$, which is the maximal value that the $s(x)$ can have. With this threshold, no score will ever satisfy $s(x) > s_\alpha$ and thus it is equivalent to not using pseudo-labels at all. From Fig. 3(b) one can observe that $s_\alpha = 2$ yields lower results, meaning that the use of pseudo-labels does, in fact, help train the network. Tab. 6 shows the results on the OfficeHome and Office31 datasets in the case where pseudo-labels are applied as suggested in our approach and when they are not applied at all. These results show that the use of pseudo-labels during training does improve the accuracy during the deployment stage.

We also analyze the advantage of employing a dynamic threshold for the use of pseudo-labels. The analysis is done on the Office31 and OfficeHome datasets by fixing a set threshold, $s_\alpha = 1.20$ (which was found to provide the optimal value). The results are reported in Tab. 6. As can be seen, the dynamic threshold does seem to give better results overall. This is probably due to the fact that we are able to use more samples for which the transfer score, $s(x)$, is above s_0 and below the static s_α at later parts of the training.

Confidence Regularization Threshold Analysis We next analyze the threshold s_β used to determine which of the target samples are used when calculating the confidence regularization. We compare the average accuracy when only changing the threshold s_β while all other hyper-parameters are set to the default value. The results can be found in Fig. 3(c). The accuracy varies by around 2% and it is clear that the use of this regularization term under a relatively stable threshold value does improve the final result. For larger

	A → W	D → W	W → D	A → D	D → A	W → A	Avg
UAN(You et al. 2019)	85.62	94.77	97.99	86.50	85.45	85.12	89.24
USFDA (Kundu et al. 2020)	85.56	95.20	97.79	88.47	87.5	86.61	90.19
Ours with $w_t(x)$	86.23	93.26	91.79	84.31	86.09	85.41	87.84
Ours with $s_h(x)$	85.26	93.81	95.16	82.84	85.31	83.51	87.65
Ours, $s(x)$ w/o $d(x)$	88.73	**95.70**	96.46	88.69	89.61	80.83	81.67
Ours, $s(x)$ w/o $\max \bar{y}(x)$	78.99	89.91	87.91	83.73	81.80	82.55	84.15
Ours with $s(x)$	**90.11**	95.33	**98.18**	**90.56**	**90.03**	**90.45**	**92.44**

Table 4: Comparison on Office31 between UAN (You et al. 2019), USFDA (Kundu et al. 2020) and our approach when using either $s(x)$ (with ablation on the score's components) or UAN's $w_t(x)$ as the scoring scheme, as well as other variants.

	Accuracy
Ours using entropy maximization	91.70
Ours using pseduo-labels of Zou et al. (2019)	88.26
Ours	**92.44**

Table 5: Variants of our method: entropy maximization and the pseudo-labeling scheme of Zou et al. (2019). (Office31; see appendix for detailed results).

	Office-Home	Office31
Ours w/o pseudo-labels	76.689	89.89
Ours, $s_\alpha = 0$	77.13	88.19
Ours, static $s_\alpha = 1.2$	78.46	**92.48**
Ours	**79.62**	92.44

Table 6: Average results when using different thresholds schemes for pseudo-labels on Office-Home and Office31.

values of s_β the accuracy of the model drops as a result of the regularization term lowering the prediction confidence for samples that are likely to be in the shared label set.

In Tab. 7 we analyze the effect of the dynamic threshold s_β compared to a static one. The analysis is done on the Office31 and OfficeHome datasets by fixing a set threshold, $s_\beta = 1.00$ (which was found to provide the optimal value). As can be seen from the results, the dynamic threshold yields slightly higher results across different datasets, but not always. We note that on top of improved average performance, the dynamic threshold reduces the number of parameters. When removing the confidence regularization all together or applying it to all target samples, the performance further drops. Tab. 7 also shows the ratio between the mean sample score for shared and private samples for the domain shift **W→D**. Evidently, the regularization term increases the ratio and thus helps distinguish between the private and shared classes.

Decision Threshold Analysis Another component of the network we analyze is the decision threshold s_0, which is used to decide whether the model would label a sample as τ or use the predicted label. The analysis is done in a similar manner to the two previous sections.

As is evident from the results in Fig. 3(a), there is little variance in the results for a threshold in a wide range between 0 and 1.4. For thresholds higher than 1.4, we see a sharp drop in the accuracy until finally reaching the lowest possible value at $s_0 = 2$. This drop in accuracy occurs because only a very small number of samples have a transfer score, $s(x)$, higher than the threshold and thus most samples are labeled τ. The extreme case, as seen in the graph, is $s_0 = 2$ where no sample can pass this threshold and all are labeled τ, leading to an accuracy score that is ξ the fraction of samples from novel target classes in this benchmark.

	O-H	Office31	Score ratio
No CR	78.04	91.12	1.38 : 0.88
CR, all target samples	78.18	91.07	1.42 : 0.87
CR, all samples	78.56	91.91	1.41 : 0.88
CR, static $s_\beta = 1.0$	79.10	**92.55**	1.40 : 0.81
CR, dynamic s_β (ours)	**79.62**	92.44	**1.41 : 0.80**

Table 7: Variants of the confidence regularization (CR) for Office31 and Office-Home (O-H). The score ratio column presents the ratio between the mean score for shared and private samples in the target domain for W → D on the Office31 dataset. See appendix for full results.

Conclusions

We study unsupervised domain adaptation in the challenging case where there is a partial overlap between the source and target domain classes. Our method adapts through the usage of pseudo-labels and a confidence regularization loss. However, since some of the samples of the target domain cannot be properly labeled by any of the source labels, we propose to score the target samples and apply a threshold in order to select those that would lead to positive transfer.

Our scoring takes into consideration the confidence of the label classifier, as well as the confidence of the domain discriminator. The more certain the first classifier is in its prediction and the less certain the latter is that the sample is from the target domain, the more likely the target domain sample is from the shared label set.

The method obtains state of the art results by a sizable margin on the relevant literature benchmarks, despite being simpler than previous work. We also demonstrate that our scoring scheme is superior to the values given by the weighting schemes previously proposed.

Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation program (grant ERC CoG 725974).

References

Bousmalis, K.; Silberman, N.; Dohan, D.; Erhan, D.; and Krishnan, D. 2017. Unsupervised Pixel-Level Domain Adaptation With Generative Adversarial Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Busto, P. P.; Iqbal, A.; and Gall, J. 2020. Open Set Domain Adaptation for Image and Action Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42: 413–429.

Cao, Z.; Long, M.; Wang, J.; and Jordan, M. I. 2018. Partial Transfer Learning with Selective Adversarial Networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2724–2732.

Cao, Z.; Ma, L.; Long, M.; and Wang, J. 2018. Partial Adversarial Domain Adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV).*

Choi, J.; Jeong, M.; Kim, T.; and Kim, C. 2019. Pseudo-Labeling Curriculum for Unsupervised Domain Adaptation. In *BMVC*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

French, G.; Mackiewicz, M.; and Fisher, M. 2018. Self-ensembling for visual domain adaptation. In *International Conference on Learning Representations*. URL https://openreview.net/forum?id=rkpoTaxA-.

Fu, B.; Cao, Z.; Long, M.; and Wang, J. 2020. Learning to Detect Open Classes for Universal Domain Adaptation. In Vedaldi, A.; Bischof, H.; Brox, T.; and Frahm, J.-M., eds., *Computer Vision – ECCV 2020*, 567–583. Cham: Springer International Publishing.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *Journal of Machine Learning Research* 17(59): 1–35.

Grandvalet, Y.; and Bengio, Y. 2004. Semi-supervised Learning by Entropy Minimization. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'04, 529–536. Cambridge, MA, USA: MIT Press. URL http://dl.acm.org/citation.cfm?id=2976040.2976107.

Griffin, G.; Holub, A.; and Perona, P. 2006. Caltech256 Image Dataset URL http://www.vision.caltech.edu/Image_Datasets/Caltech256/. Last accessed: 2021-03-19.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hoffman, J.; Tzeng, E.; Park, T.; Zhu, J.-Y.; Isola, P.; Saenko, K.; Efros, A.; and Darrell, T. 2018. CyCADA: Cycle-Consistent Adversarial Domain Adaptation. volume 80 of *Proceedings of Machine Learning Research*, 1989–1998. Stockholmsmässan, Stockholm Sweden: PMLR. URL http://proceedings.mlr.press/v80/hoffman18a.html.

Hu, L.; Kan, M.; Shan, S.; and Chen, X. 2018. Duplex Generative Adversarial Network for Unsupervised Domain Adaptation. In *CVPR*, 1498–1507.

Huang, S.-W.; Lin, A.; Chen, S.-P.; Wu, Y.-Y.; Hsu, P.-H.; and Lai, S.-H. 2018. AugGAN: Cross Domain Adaptation with GAN-based Data Augmentation. In *ECCV*.

Kundu, J. N.; Venkat, N.; V, R. M.; and Babu, R. V. 2020. Universal Source-Free Domain Adaptationn URL /https://openreview.net/forum?id=B1gd0nEFwS. Last accessed: 2021-03-19.

Liu, Y.-C.; Yeh, Y.-Y.; Fu, T.-C.; Wang, S.-D.; Chiu, W.-C.; and Wang, Y.-C. F. 2018. Detach and Adapt: Learning Cross-Domain Disentangled Deep Representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Long, M.; Zhu, H.; Wang, J.; and Jordan, M. I. 2016. Unsupervised Domain Adaptation with Residual Transfer Networks. In Lee, D. D.; Sugiyama, M.; Luxburg, U. V.; Guyon, I.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 29*, 136–144. Curran Associates, Inc. URL http://papers.nips.cc/paper/6110-unsupervised-domain-adaptation-with-residual-transfer-networks.pdf.

Murez, Z.; Kolouri, S.; Kriegman, D.; Ramamoorthi, R.; and Kim, K. 2018. Image to Image Translation for Domain Adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Peng, X.; Usman, B.; Kaushik, N.; Wang, D.; Hoffman, J.; and Saenko, K. 2018. VisDA: A Synthetic-to-Real Benchmark for Visual Domain Adaptation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2018, Salt Lake City, UT, USA, June 18-22, 2018*, 2021–2026. IEEE Computer Society. doi:10.1109/CVPRW.2018.00271.

Russo, P.; Carlucci, F. M.; Tommasi, T.; and Caputo, B. 2018. From Source to Target and Back: Symmetric Bi-Directional Adaptive GAN. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Saenko, K.; Kulis, B.; Fritz, M.; and Darrell, T. 2010. Adapting Visual Category Models to New Domains. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, ECCV'10, 213–226. Berlin, Heidelberg: Springer-Verlag. ISBN 3-642-15560-X, 978-3-642-15560-4. URL http://dl.acm.org/citation.cfm?id=1888089.1888106.

Saito, K.; Kim, D.; Sclaroff, S.; and Saenko, K. 2020. Universal Domain Adaptation through Self Supervision. *ArXiv* abs/2002.07953.

Saito, K.; Ushiku, Y.; and Harada, T. 2017. Asymmetric Tri-Training for Unsupervised Domain Adaptation. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, 2988–2997. JMLR.org.

Saito, K.; Yamamoto, S.; Ushiku, Y.; and Harada, T. 2018. Open Set Domain Adaptation by Backpropagation. *CoRR* abs/1804.10427. URL http://arxiv.org/abs/1804.10427.

Sankaranarayanan, S.; Balaji, Y.; Castillo, C. D.; and Chellappa, R. 2018. Generate to Adapt: Aligning Domains Using Generative Adversarial Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sener, O.; Song, H. O.; Saxena, A.; and Savarese, S. 2016. Learning Transferrable Representations for Unsupervised Domain Adaptation. In Lee, D. D.; Sugiyama, M.; Luxburg, U. V.; Guyon, I.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 29*, 2110–2118. Curran Associates, Inc. URL http://papers.nips.cc/paper/6360-learning-transferrable-representations-for-unsupervised-domain-adaptation.pdf.

Shu, R.; Bui, H. H.; Narui, H.; and Ermon, S. 2018. A DIRT-T Approach to Unsupervised Domain Adaptation. *International Conference on Learning Representations (ICLR)* .

Venkateswara, H.; Eusebio, J.; Chakraborty, S.; and Panchanathan, S. 2017. Deep Hashing Network for Unsupervised Domain Adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Volpi, R.; Morerio, P.; Savarese, S.; and Murino, V. 2018. Adversarial Feature Augmentation for Unsupervised Domain Adaptation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* 5495–5504.

You, K.; Long, M.; Cao, Z.; Wang, J.; and Jordan, M. I. 2019. Universal Domain Adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, J.; Ding, Z.; Li, W.; and Ogunbona, P. 2018. Importance Weighted Adversarial Nets for Partial Domain Adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhang, W.; Ouyang, W.; Li, W.; and Xu, D. 2018. Collaborative and Adversarial Network for Unsupervised Domain Adaptation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3801–3809. doi: 10.1109/CVPR.2018.00400.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Zou, Y.; Yu, Z.; Liu, X.; Kumar, B. V.; and Wang, J. 2019. Confidence Regularized Self-Training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.