GradMix: Multi-source Transfer across Domains and Tasks

Junnan Li*[1], Ziwei Xu*[1], Yongkang Wang[1], Qi Zhao[2], and Mohan S. Kankanhalli[1]

[1]School of Computing, National University of Singapore
[2]Department of Computer Science and Engineering, University of Minnesota
{lijunnan,ziwei.xu}@u.nus.edu, yongkang.wong@nus.edu.sg, qzhao@cs.umn.edu,
mohan@comp.nus.edu.sg

Abstract

The computer vision community is witnessing an unprecedented rate of new tasks being proposed and addressed, thanks to the deep convolutional networks' capability to find complex mappings from $\mathcal{X}$ to $\mathcal{Y}$. The advent of each task often accompanies the release of a large-scale annotated dataset, for supervised training of deep network. However, it is expensive and time-consuming to manually label sufficient amount of training data. Therefore, it is important to develop algorithms that can leverage off-the-shelf labeled dataset to learn useful knowledge for the target task. While previous works mostly focus on transfer learning from a single source, we study multi-source transfer across domains and tasks (MS-DTT), in a semi-supervised setting. We propose GradMix, a model-agnostic method applicable to any model trained with gradient-based learning rule, to transfer knowledge via gradient descent by weighting and mixing the gradients from all sources during training. GradMix follows a meta-learning objective, which assigns layer-wise weights to the source gradients, such that the combined gradient follows the direction that minimize the loss for a small set of samples from the target dataset. In addition, we propose to adaptively adjust the learning rate for each mini-batch based on its importance to the target task, and a pseudo-labeling method to leverage the unlabeled samples in the target domain. We conduct MS-DTT experiments on two tasks: digit recognition and action recognition, and demonstrate the advantageous performance of the proposed method against multiple baselines.

1. Introduction

Deep convolutional networks (ConvNets) have significantly improved the state-of-the-art for visual recognition, by finding complex mappings from $\mathcal{X}$ to $\mathcal{Y}$. Unfortunately, these impressive gains in performance come only when massive amounts of paired labeled data (x, y) s.t. $x \in \mathcal{X}, y \in \mathcal{Y}$ are available for supervised training. For many application domains, it is often prohibitive to manually label sufficient training data, due to the significant amount of human efforts required or the concern of violating individual's privacy. Hence, there is strong incentive to develop algorithms that can reduce the burden of manual labeling, typically by leveraging off-the-shelf labeled datasets from other related domains and tasks.

There has been a large amount of efforts in the research community to address adapting deep models across domains [7, 21, 39], to transfer knowledge across tasks [23, 8, 42], and to learn efficiently in a few shot manner [5, 29, 30]. However, most works focus on a single-source and single-target scenario. Recently, some works [41, 25, 43] propose deep approaches for multi-source domain adaptation, but assume that the source and target domains have shared label space (task).

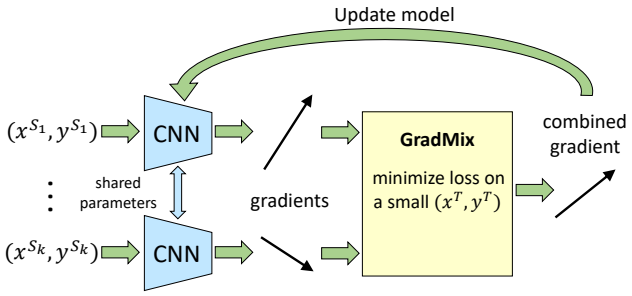

Figure 1: High-level overview of the proposed method. We transfer knowledge to the target domain by weighting and mixing gradients from source domains, such that the combined gradient should minimize the loss for a few validation samples from the target domain.

*equal contribution

In many computer vision applications, there often exist multiple labeled datasets available from different domains and/or tasks related to the target application. Hence, it is important and practically valuable that we can transfer knowledge from as many source datasets as possible. In this work, we formalize this problem as multi-source domain and task transfer (MS-DTT). Given a set of labeled source dataset, $\mathcal{S} = \{S_1, S_2, ..., S_k\}$, we aim to transfer knowledge to a sparsely labeled target dataset $\mathcal{T}$. Each source dataset S_i could come from a different domain compared to $\mathcal{T}$, having a different task, or different in both domain and task. We focus on a semi-supervised setting where only few samples in $\mathcal{T}$ have labels.

Most works achieve domain transfer by aligning the feature distribution of source domain and target domain [20, 21, 7, 38, 25, 41]. However, this method could be suboptimal for MS-DTT. The reason is that in MS-DTT, the distribution of source data $p(x^{S_i}, y^{S_i})$ and target data $p(x^{\mathcal{T}}, y^{\mathcal{T}})$ could be significantly different in both input space and label space, thus feature alignment may generate indiscriminative features for the target classes. In addition, feature alignment introduces additional layers and loss terms, which require careful design to perform well.

In this work, we propose a generic and scalable method, namely GradMix, for semi-supervised MS-DTT. GradMix is a model-agnostic method, applicable to any model that uses gradient-based learning rule. Our method does not introduce extra layers or loss functions for feature alignment. Instead, we perform knowledge transfer via gradient descent, by weighting and mixing the gradients from all the source datasets during training. We follow a meta-learning paradigm and model the most basic assumption: *the combined gradient should minimize the loss for a set of unbiased samples from the target dataset* [31]. We propose an online method to weight and mix the source gradients at each training iteration, such that the knowledge most useful for the target task is preserved through the gradient update. Our method can adaptively adjust the learning rate for each mini-batch based on its importance to the target task. In addition, we propose a pseudo-labeling method based on model ensemble to learn from the unlabeled data in target domain. We perform extensive experiments on two sets of MS-DTT task, including digit recognition and action recognition, and demonstrate the advantageous performance of the proposed method compared to multiple baselines.

2. Related Work

2.1. Domain Adaptation

Domain adaptation seeks to address the domain shift problem [4] and learn from source domain a model that performs well on the target domain. Most existing works focus on aligning the feature distribution of the source domain and the target domain. Several works attempt to learn domain-invariant features by minimizing Maximum Mean Discrepancy [20, 21, 36]. Other methods propose adversarial discriminative models, which try to learn domain-agnostic representations by maximizing a domain confusion loss [7, 38, 23].

Recently, multi-source domain adaptation with deep model has been studied. Mancini *et al*. [25] use DA-layers [3, 18] to minimize the distribution discrepancy of network activations. Xu *et al*. [41] propose multi-way adversarial domain discriminator that minimizes the domain discrepancies between the target and each of the sources. Zhao *et al*. [43] propose multisource domain adversarial networks that approach domain adaptation by optimizing domain-adaptive generalization bounds. However, all of these methods [25, 41, 43] assume that the source and target domains have a shared label space.

2.2. Transfer Learning.

Transfer learning extends domain adaptation into more general cases, where the source and target domain could be different, in both input space and label space [28, 40, 16, 14]. In computer vision, transfer learning has been widely studied to overcome the deficit of labeled data by adapting models trained for other tasks. With the advance of deep supervised learning, ConvNets trained on large datasets such as ImageNet [32] have achieved state-of-the-art performance when transfered to other tasks (*e.g*. object detection [8], semantic segmentation [19], etc.) by simple fine-tuning. In this work, we focus on the setting where source and target domains have the same input space and different label spaces.

2.3. Meta-Learning.

Meta-learning aims to utilize knowledge from past experiences to learn quickly on target tasks, from only a few annotated samples. Meta-learning generally seeks performing the learning at a level higher than where conventional learning occurs, *e.g*. learning the update rule of a learner [29], or finding a good initialization point that is more robust [17] or can be easily fine-tuned [5]. Li *et al*. [13] propose a meta-learning method to train models with good generalization ability to novel domains. Franceschi *et al*. [6] introduce a framework based on bilevel programming that unifies gradient-based hyperparameter optimization and meta-learning. Sun *et al*. [37] propose a meta-transfer learning method to address the few-shot learning task. Ren *et al*. [31] propose example reweighting in a meta-learning framework. Our method follows the meta-learning paradigm that uses validation loss as the meta-objective. However, different from [31] which reweight samples in a batch for robust learning against noise, we reweight source domain gradients layer-wise for transfer learning. Gradient alignment

has also been used to enhance learning congruency in [22].

3. Method

3.1. Problem Formulation

We first formally introduce the semi-supervised MS-DTT problem. Assume that there exists a set of k source domains $\mathcal{S} = \{S_1, S_2, ..., S_k\}$ and a target domain $\mathcal{T}$. Each source domain S_i contains N^{S_i} images, $x^{S_i} \in \mathcal{X}^{S_i}$, with associated labels $y^{S_i} \in \mathcal{Y}^{S_i}$. Similarly, the target domain consists of $N^{\mathcal{T}}$ unlabeled images, $x^{\mathcal{T}} \in \mathcal{X}^{\mathcal{T}}$, as well as $M^{\mathcal{T}}$ labeled images with associated labels $y^{\mathcal{T}} \in \mathcal{Y}^{\mathcal{T}}$. We assume target domain is only sparsely labeled, *i.e.* $M^{\mathcal{T}} \ll N^{\mathcal{T}}$. Our goal is to learn a strong target classifier that can predict labels $y^{\mathcal{T}}$ given $x^{\mathcal{T}}$.

Different from standard domain adaptation approaches that assume a shared label space between each source and target domain ($\mathcal{Y}^{S_i} = \mathcal{Y}^{\mathcal{T}}$), we study the problem of joint transfer across domains and tasks. In our setting, only one of the source domain needs to have the same label space as the target domain ($\exists S_i \; s.t. \; \mathcal{Y}^{S_i} = \mathcal{Y}^{\mathcal{T}}$). Other source domains could either have a partially overlapping label space with the target domain ($\mathcal{Y}^{S_i} \cap \mathcal{Y}^{\mathcal{T}} \subset \mathcal{Y}^{\mathcal{T}}$ and $\mathcal{Y}^{S_i} \cap \mathcal{Y}^{\mathcal{T}} \neq \varnothing$), or a non-overlapping label space ($\mathcal{Y}^{S_i} \cap \mathcal{Y}^{\mathcal{T}} = \varnothing$).

3.2. Meta-learning Objective

Let Θ denote the network parameters for our model. We consider a loss function $\mathcal{L}(x, y; \Theta) = f(\Theta)$ to minimize during training. For deep networks, stochastic gradient descent (SGD) or its variants are commonly used to optimize the loss functions. At every step n of training, we forward a mini-batch of samples from each of the source domain $\{S_i\}_{i=1}^k$, and apply back-propagation to calculate the gradients w.r.t the parameters Θ_n, $\nabla f_{s_i}(\Theta_n)$. The parameters are then adjusted according to the sum of the source gradients. For example, for vanilla SGD:

$$\Theta_{n+1} = \Theta_n - \alpha \sum_{i=1}^{k} \nabla f_{s_i}(\Theta_n), \tag{1}$$

where α is the learning rate.

In semi-supervised MS-DTT, we also have a small validation set $\mathcal{V}$ that contains few labeled samples from the target domain. We want to learn a set of weights for the source gradients, $w = \{w_{s_i}\}_{i=1}^k$, such that when taking a gradient descent using their weighted combination $\sum_{i=1}^k w_{s_i} \nabla f_{s_i}(\Theta_n)$, the loss on the validation set is minimized:

$$\Theta^*(w) = \Theta_n - \alpha \sum_{i=1}^{k} w_{s_i} \nabla f_{s_i}(\Theta_n), \tag{2}$$

$$w^* = \arg\min_{w, w \geq 0} f_{\mathcal{V}}(\Theta^*(w)) \tag{3}$$

3.3. Layer-wise Gradient Weighting

Calculating the optimal w^* requires two nested loops of optimization, which can be computationally expensive. Here we propose an approximation to the above objective. At each training iteration n, we do a forward-backward pass using the small validation set $\mathcal{V}$ to calculate the gradient, $\nabla f_{\mathcal{V}}(\Theta_n)$. We take a first-order approximation and assume that adjusting Θ_n in the direction of $\nabla f_{\mathcal{V}}(\Theta_n)$ can minimize $f_{\mathcal{V}}(\Theta_n)$. Therefore, we find the optimal w^* by maximizing the cosine similarity between the combined source gradient and the validation gradient:

$$w^* = \arg\max_{w, w \geq 0} \text{cossim} \left[\sum_{i=1}^{k} w_{s_i} \nabla f_{s_i}(\Theta_n), \nabla f_{\mathcal{V}}(\Theta_n) \right], \tag{4}$$

where the cosine similarity between two vectors is defined as:

$$\text{cossim}[a, b] = \frac{a \cdot b}{\|a\| \, \|b\|}. \tag{5}$$

Instead of using a global weight value for each source gradient, we propose a layer-wise gradient weighting, where the gradient for each network layer are weighted separately. This enables a finer level of gradient combination. Specifically, in our MS-DTT setting, all source domains and the target domain share the same parameters up to the last fully-connected (fc) layer, which is task-specific (the target domain shares its last layer only with the source domain that has the same label space as the target). Therefore, for each layer l with parameter θ^l, and for each source domain S_i, we have a corresponding weight $w_{s_i}^l$. We can then write Equation 4 as:

$$w^* = \arg\max_{w, w \geq 0} \sum_{l=1}^{L-1} \text{cossim} \left[\sum_{i=1}^{k} w_{s_i}^l \nabla f_{s_i}(\theta_n^l), \nabla f_{\mathcal{V}}(\theta_n^l) \right], \tag{6}$$

where L is the total number of layers for the ConvNet. We constrain $w_{s_i}^l \geq 0$ for all i and l, since negative gradient update can usually result in unstable behavior. To efficiently solve the above constrained non-linear optimization problem, we utilize a sequential quadratic programming method, SLSQP, implemented in NLopt [10].

In practice, we normalize the weights for each layer across all source domains so that they sum up to one:

$$\tilde{w}_{s_i}^l = \frac{w_{s_i}^l}{\sum_{i=1}^k w_{s_i}^l} \tag{7}$$

The computational overhead of GradMix mainly comes from optimizing w and calculating $\nabla f_{\mathcal{V}}$. Compared to source-only training, GradMix increases the training time per-batch by approximately 40%.

3.4. Adaptive Learning Rate

Intuitively, certain mini-batches from the source domains contain more useful knowledge that can be transferred to the target domain, whereas some mini-batches contain less. Therefore, we want to adaptively adjust our training to pay more attention to the important mini-batches. To this end, we measure the importance score ρ of a mini-batch using the cosine similarity between the optimally combined gradient and the validation gradient:

$$\rho = \sum_{l=1}^{L-1} \text{cossim}\left[\sum_{i=1}^{k} \tilde{w}_{s_i}^{l} \nabla f_{s_i}(\theta_n^l), \nabla f_{\mathcal{V}}(\theta_n^l)\right] \quad (8)$$

Based on ρ, we calculate a scaling term η bounded between 0 and 1:

$$\eta = \frac{1}{1 + e^{-(\beta\rho - \gamma)}}, \quad (9)$$

where β controls the rate of saturation for η, and γ controls the shift along the horizontal axis (*i.e.* when $\beta\rho = \gamma$, $\eta = 0.5$). We determine the value of β and γ empirically through experiments.

Finally, we multiply η to the learning rate α, and perform SGD to update the parameters:

$$\theta_{n+1}^l = \theta_n^l - \eta\alpha \sum_{i=1}^{k} \tilde{w}_{s_i}^{l} \nabla f_{s_i}(\theta_n^l), \text{ for } l = 1, 2, ..., L-1 \quad (10)$$

3.5. Pseudo-label with Ensembles

In our semi-supervised MS-DTT setting, there also exists a large set of unlabeled images in the target domain, denoted as $\mathcal{U} = \{(x_n^{\mathcal{T}})\}_{n=1}^{N^{\mathcal{T}}}$. We want to learn target-discriminative knowledge from $\mathcal{U}$. To achieve this, we propose a method to calculated pseudo-labels $\hat{y}_n^{\mathcal{T}}$ for the unlabeled images, and construct a pseudo-labeled dataset $S_u = \{(x_n^{\mathcal{T}}, \hat{y}_n^{\mathcal{T}})\}_{n=1}^{N^p}$. Then we leverage S_u using the same gradient mixing method as described above. Specifically, we consider to minimize a loss $\mathcal{L}_u(x, \hat{y}; \Theta)$ during training where $(x, \hat{y}) \in S_u$. At each training iteration n, we sample a mini-batch from S_u, calculate the gradient $\nabla f_{s_u}(\Theta_n)$, and combine it with the source gradients $\{\nabla f_{s_i}(\Theta_n)\}_{i=1}^{k}$ using the proposed layer-wise weighting method.

In order to acquire the pseudo-labels, we perform a first step to train a model using the source domain datasets following the proposed gradient mixing method, and use the learned model to label $\mathcal{U}$. However, the learned model would inevitably create some false pseudo-labels. Previous studies found that ensemble of models helps to produce more reliable pseudo-labels [34, 11]. Therefore, in our first step, we train multiple models with different combination of β and γ in Equation 9. Then we pick the top R models with the best accuracies on the hyper-validation set (we set $R = 3$ in our experiments), and use their ensemble to create pseudo-labels. The difference in hyper-parameters during training ensures that different models learn significantly different sets of weight, hence the ensemble of their prediction is less biased.

Here we propose two approaches to create pseudo-labels, namely hard label and soft label:

Hard label. Here, we assume that the pseudo-label is more likely to be correct if all the models can reach an agreement with high confidence. We assign a pseudo-label $\hat{y} = C$ to an image $x \in \mathcal{U}$, where C is a class index, if the two following conditions are satisfied. First, all of the R models should predict C as the class with maximum probability. Second, for all models, the probability for C should exceed certain threshold, which is set as 0.8 in our experiments. If these two conditions are satisfied, we will add $(x, \hat{y})$ into S_u. During training, the loss $\mathcal{L}_u(x, \hat{y}; \Theta)$ is the standard cross entropy loss.

Soft label. Let p_r denote the output from the r-th model's softmax layer for an input x, which represents the probability over classes. We calculate the average of p_r across all of the R models as the soft pseudo-label for x, *i.e.* $\hat{y} = \frac{1}{R}\sum_{r=1}^{R} p_r$. Every unlabeled image $x \in \mathcal{U}$ will be assigned a soft label and added to S_u. During training, let p_Θ be the output probability from the model, we want to minimize the KL-divergence between p_Θ and the soft pseudo-label for all pairs $(x, \hat{y}) \in S_u$. Therefore, the loss is $\mathcal{L}_u(x, \hat{y}; \Theta) = D_{KL}(p_\Theta, \hat{y})$.

For both hard label and soft label approach, after getting the pseudo-labels, we train a model from scratch using all available datasets $\{S_i\}_{i=1}^{k}$, S_u and $\mathcal{V}$. Since the proposed gradient mixing method relies on $\mathcal{V}$ to estimate the model's performance on the target domain, we enlarge the size of $\mathcal{V}$ to 100 samples per class, by adding hard-labeled images from S_u using the method described above. The enlarged $\mathcal{V}$ can represent the target domain with less bias, which helps to calculate better weights on the source gradients, such that the model's performance on the target domain is maximized.

3.6. Incorporating Semi-supervised Learning

We can further exploit the unlabeled target domain data $\mathcal{U}$ by leveraging semi-supervised learning (SSL) methods. Specifically, we incorporate two state-of-the-art SSL methods, virtual adversarial training [26] and MixMatch [2], into our GradMix method, by adding an additional unlabeled loss term on $\mathcal{U}$ during training. The details of the unlabeled loss can be found in the original papers [26, 2].

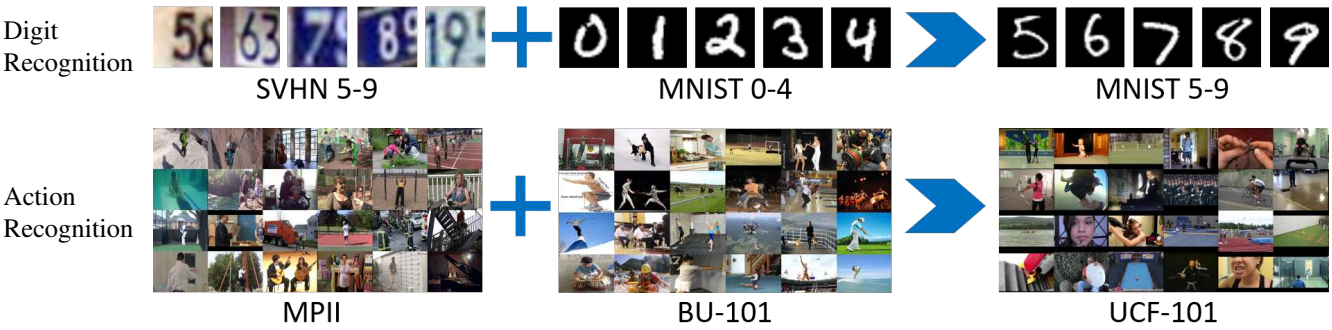

Figure 2: An illustration of the two experimental settings for multi-source domain and task transfer (MS-DTT). Our method effectively transfers knowledge from multiple sources to the target task.

4. Experiment

4.1. Experimental Setup

Datasets. In our experiment, we perform MS-DTT across two different groups of data settings, as shown in Figure 2. First, we do transfer learning across different digit domains using MNIST [12] and Street View House Numbers (SVHN) [27]. MNIST is a popular benchmark for handwritten digit recognition, which contains a training set of 60,000 examples and a test set of 10,000 examples. SVHN is a real-word dataset consisting of images with colored background and blurred digits. It has 73,257 examples for training and 26,032 examples for test.

For our second setup, we study MS-DTT from human activity images in MPII dataset [1] and human action images from the Web (BU101 dataset) [24], to video action recognition using UCF101 [35] dataset. MPII dataset consists of 28,821 images covering 410 human activities including home activities, religious activities, occupation, etc. UCF101 is a benchmark action recognition dataset collected from YouTube. It has 13,320 videos from 101 action categories, captured under various lighting conditions with camera motion and occlusion. We take the first split of UCF101 for our experiment. BU101 contains 23,800 images collected from the Web, with the same action categories as UCF101. It contains professional photos, commercial photos, and artistic photos, which differ significantly from video frames.

Network and implementation details. For digit recognition, we use the same ConvNet architecture as [23], which has 4 Conv layers and 2 fc layers. We randomly initialize the weights, and train the network using SGD with learning rate $\alpha = 0.05$, and a momentum of 0.9. For fine-tuning we reduce the learning rate to 0.005. For action recognition, we use ResNet-18 [9] architecture. We initialize the network with ImageNet pre-trained weights, which is important for all baseline methods to perform well. The learning rate is 0.001 for training and $5\mathrm{e}{-5}$ for fine-tuning.

4.2. SVHN 5-9 + MNIST 0-4 $\rightarrow$ MNIST 5-9

Experimental setting. In this experiment, we define four sets of training data: (1) labeled images of digits 5-9 from the training split of SVHN dataset as the first source S_1, (2) labeled images of digits 0-4 from the training split of MNIST dataset as the second source S_2, (3) few labeled images of digits 5-9 from the training split of MNIST dataset as the validation set $\mathcal{V}$, (4) unlabeled images from the rest of the training split of MNIST 5-9 as $\mathcal{U}$. We subsample k examples from each class of MNIST 5-9 to construct the unbiased validation set $\mathcal{V}$. We experiment with $k = 2, 3, 4, 5$, which corresponds to $10, 15, 20, 25$ labeled examples. Since $\mathcal{V}$ is randomly sampled, we repeat our experiment 10 times with different $\mathcal{V}$. In order to monitor training progress and tune hyper-parameters (*e.g.* α, β, γ), we split out another 1000 labeled samples from MNIST 5-9 as the hyper-validation set. The hyper-validation set is the traditional validation set and is fixed across 10 runs.

Baselines. We compare the proposed method to multiple baseline methods:

- *Target only*: the model is trained using $\mathcal{V}$.
- *Source only*: the model is trained using S_1 and S_2 without gradient reweighting.
- *Fine-tune*: the *Source only* model is fine-tuned using $\mathcal{V}$.
- *MME* [33]: Minimax Entropy is a state-of-the-art method for single-source semi-supervised domain adaptation. We use S_1 (SVHN 5-9) as the source domain because it is has the same label space as the target task.
- *MDDA* [25]: Multi-domain domain alignment layers that shift the network activations for each domain using a parameterized transformation equivalent to batch normalization.
- *DCTN* [41]: Deep Cocktail Network, which uses multi-way adversarial adaptation to align the distribution of multiple source domains and the target domain.

Table 1: Classification accuracy (%) of the baselines and our method on the test split of MNIST 5-9. We report the mean and the standard error of each method across 10 runs with different randomly sampled $\mathcal{V}$.

Method	**Datasets**	**k=2**	**k=3**	**k=4**	**k=5**
Target only	$\mathcal{V}$	71.35±1.85	77.15±1.36	81.43±1.41	84.83±1.10
Source only	S_1, S_2	82.39	82.39	82.39	82.39
Fine-tune	$S_1, S_2, \mathcal{V}$	89.94±0.35	89.86±0.46	90.89±0.48	91.96±0.39
GradMix SGD [31]	$S_1, S_2, \mathcal{V}$	89.30±0.73	89.78±0.72	91.70±0.45	92.05±0.29
GradMix w/o AdaLR	$S_1, S_2, \mathcal{V}$	90.10±0.37	90.22±0.62	92.14±0.43	92.92±0.29
GradMix	$S_1, S_2, \mathcal{V}$	**91.17±0.37**	**91.45±0.52**	**92.14±0.40**	**93.06±0.46**
MME [33]	$S_1, \mathcal{V}, \mathcal{U}$	90.25±0.31	90.37±0.36	91.38±0.29	91.76±0.24
MDDA [25]	$S_1, S_2, \mathcal{V}, \mathcal{U}$	90.23±0.40	90.28±0.50	91.45±0.37	91.85±0.31
DCTN [41]	$S_1, S_2, \mathcal{V}, \mathcal{U}$	91.81±0.26	92.34±0.28	92.42±0.39	92.97±0.37
GradMix w/ soft label	$S_1, S_2, \mathcal{V}, \mathcal{U}$	94.62±0.18	95.03±0.30	95.26±0.17	95.74±0.21
GradMix w/ hard label	$S_1, S_2, \mathcal{V}, \mathcal{U}$	96.02±0.24	96.24±0.33	96.63±0.17	96.84±0.20
GradMix w/ VAT [26]	$S_1, S_2, \mathcal{V}, \mathcal{U}$	96.23±0.21	96.35±0.31	**96.87±0.19**	96.94±0.20
GradMix w/ MixMatch [2]	$S_1, S_2, \mathcal{V}, \mathcal{U}$	**96.30±0.23**	**96.43±0.32**	96.85±0.19	**97.02±0.21**

We also evaluate different variants of our model with and without certain component to show its effect:

- *GradMix SGD*: instead of calculating the optimal weights w^* by maximizing cosine similarity of gradients (Equation 6), we follow the method in [31] and perform SGD on w to directly minimize the validation error in Equation 3.

- *GradMix w/o AdaLR*: the method in Section 3.3 without the adaptive learning rate (Section 3.4).

- *GradMix*: the proposed method that uses S_1, S_2 and $\mathcal{V}$ during training.

- *GradMix w/ hard label*: using the hard label approach to create pseudo-labels for $\mathcal{U}$, and train a model with all available datasets.

- *GradMix w/ soft label*: using the soft label approach to create pseudo-labels for $\mathcal{U}$, and train a model with all available datasets.

- *GradMix w/ VAT*: incorporating VAT [26] into GradMix.

- *GradMix w/ MixMatch*: incorporating MixMatch [2] into GradMix.

Results. Table 1 shows the results for methods described above. We report the mean and standard error of classification accuracy across 10 runs with randomly sampled $\mathcal{V}$. Methods in the upper part of the table do not use the unlabeled target domain data $\mathcal{U}$. Among these methods, the proposed GradMix has the best performance. If we remove the adaptive learning rate, the accuracy would decrease. As expected, the performance improves as k increases, which indicates more samples in $\mathcal{V}$ can help the GradMix method to better combine the gradients during training.

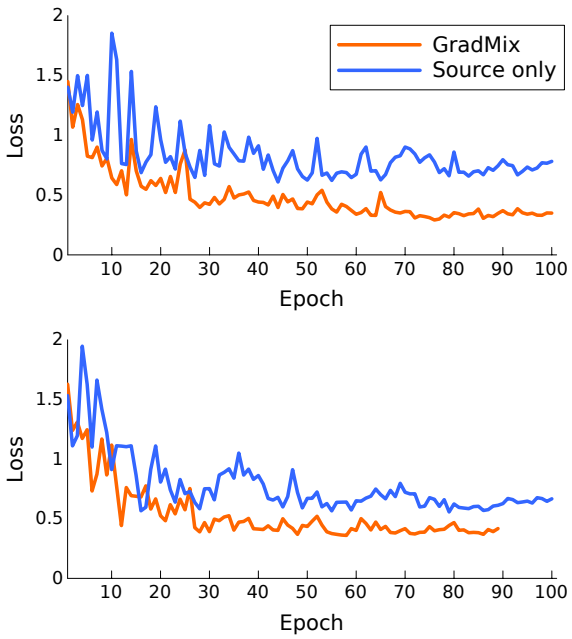

Figure 3: Loss on the hyper-validation set as training proceeds on digit recognition task. Top row is with $k = 2$ whereas the bottom row is with $k = 5$. We define 1 epoch as training for 100 mini-batches (gradient descents).

The lower part of the table shows methods that leverage the unlabeled target data $\mathcal{U}$. MME [33] only uses S_1, whereas other methods use both S_1 and S_2. The proposed GradMix without $\mathcal{U}$ can achieve comparable performance with state-of-the-art baselines that use $\mathcal{U}$ (MME, MDDA and DCTN). Using pseudo-label with model ensemble significantly improves performance compared to

Table 2: Results of GradMix using different β and γ when $k = 3$. Numbers indicate the test accuracy (%) on MNIST 5-9 (averaged across 10 runs). The ensemble of the top three models is used to create pseudo-labels.

	$\gamma = 0$	$\gamma = 0.1$	$\gamma = 0.2$	$\gamma = 0.3$	$\gamma = 0.4$	$\gamma = 0.5$	$\gamma = 0.6$	$\gamma = 0.7$	$\gamma = 0.8$
$\beta = 5$	90.92	**90.96**	90.95	90.58	90.75	90.75	90.51	90.63	**91.12**
$\beta = 6$	90.41	90.75	89.95	90.79	90.59	89.95	90.58	90.63	90.56
$\beta = 7$	89.76	90.44	90.42	90.94	90.28	90.40	90.52	90.70	90.66
$\beta = 8$	90.05	90.89	90.93	90.57	90.77	90.69	89.99	90.58	90.71
$\beta = 9$	90.32	90.70	90.48	90.94	90.47	90.92	90.20	90.23	90.86
$\beta = 10$	90.52	90.03	89.67	90.01	89.84	90.51	**91.45**	90.58	90.70

baseline methods. Comparing soft label to hard label, the hard label approach achieves better performance. More detailed results about model ensemble for pseudo-labeling is shown later in the ablation study. Furthermore, both VAT [26] and MixMatch [2] can achieve performance improvement by effectively utilizing the unlabeled data $\mathcal{U}$.

Ablation Study. In this section, we perform ablation experiments to demonstrate the effectiveness of our method and the effect of different hyper-parameters. First, Figure 3 shows two examples of the hyper-validation loss as training proceeds. We show the loss for the *Source only* baseline and the proposed GradMix, where we perform hyper-validation every 100 mini-batches (gradient descents). In both examples with different k, GradMix achieves a quicker and steadier decrease in the hyper-validation loss.

In Table 2, we show the results using GradMix with different combination of β and γ when $k = 3$. We perform a grid search with $\beta = [5, 6, ..., 10]$ and $\gamma = [0, 0.1, ..., 0.8]$. The accuracy is the highest for $\beta = 10$ and $\gamma = 0.6$. The top three models are selected for ensemble to create pseudo-labels for the unlabeled set $\mathcal{U}$.

In addition, we perform experiments with various number of models used for ensemble when creating pseudo-labels for the unlabeled set $\mathcal{U}$. Figure 4 shows the results for $R = 1, 2, 3, 4, 5$ across all values of k. $R = 3$ has the best overall performance and a moderate computational cost. Therefore, we use the ensemble of the top three models to create reliable pseudo-labels.

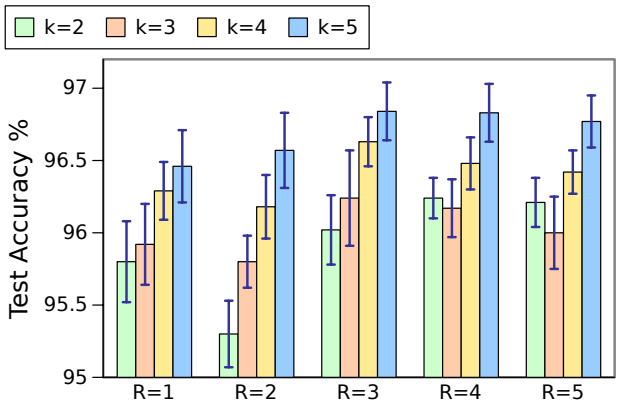

Figure 4: Results of GradMix w/ hard label using various number of pre-trained models (R) for ensemble on digit recognition task. k is the number of labeled samples per class in $\mathcal{V}$.

4.3. MPII + BU101 → UCF101

Experimental setting. In the action recognition experiment, we have four sets of training data similar to the digit recognition experiment, which include (1) S_1: labeled images from the training split of MPII, (2) S_2: labeled images from the training split of BU101, (3) $\mathcal{V}$: k labeled video clips per class randomly sampled from the training split of UCF101, (4) $\mathcal{U}$: unlabeled images from the rest of the training split of UCF101. We experiment with $k = 3, 5, 10$ which corresponds to $303, 505, 1010$ video clips. Each experiment is run two times with different $\mathcal{V}$. We report the mean accuracy across the two runs for both per-frame classification and per-video classification. Per-frame classification is the same as doing individual image classification for every frame in the video, and per-video classification is done by averaging the softmax score for all the frames in a video as the video's score.

Baselines. We compare our method with multiple baselines described in Section 4.2, including *Target only*, *Source only*, *Fine-tune*, *MDDA* [25] and *DCTN* [41]. In addition, we evaluate another baseline for knowledge transfer in action recognition, namely EnergyNet [15]: The ConvNet (ResNet-18) is first trained on MPII and BU101, then knowledge is transfered to UCF101 through spatial attention maps using a Siamese Energy Network.

Results. Table 3 shows the results for action recognition. *Target only* has better performance compared to *Source only* even for $k = 3$, which indicates a strong distribution shift between source data and target data for actions in the wild. For all values of k, the proposed GradMix outperforms baseline methods that use S_1, S_2 and $\mathcal{V}$ for training in both per-frame and per-video accuracy. GradMix also has comparable performance with MDDA that uses the

Table 3: Classification accuracy (%) of the baselines and our method on the test split of UCF101. We report the mean accuracy of each method across two runs with different randomly sampled $\mathcal{V}$.

Method	**Datasets**	per-frame			per-video		
		k=3	**k=5**	**k=10**	**k=3**	**k=5**	**k=10**
Target only	$\mathcal{V}$	42.58	53.31	63.05	43.74	55.50	64.74
Source only	S_1, S_2	41.96	41.96	41.96	43.46	43.46	43.46
Fine-tune	$S_1, S_2, \mathcal{V}$	55.86	60.55	66.77	58.57	66.01	70.21
EnergyNet [15]	$S_1, S_2, \mathcal{V}$	55.93	60.82	66.73	58.70	66.23	70.25
GradMix	$S_1, S_2, \mathcal{V}$	**56.25**	**61.73**	**67.30**	**59.41**	**66.27**	**71.49**
MDDA [25]	$S_1, S_2, \mathcal{V}, \mathcal{U}$	56.65	61.58	67.65	60.00	65.14	71.54
DCTN [41]	$S_1, S_2, \mathcal{V}, \mathcal{U}$	57.88	61.97	68.46	61.64	66.59	72.85
GradMix w/ hard label	$S_1, S_2, \mathcal{V}, \mathcal{U}$	68.92	68.76	69.25	72.58	72.34	73.48
GradMix w/ VAT [26]	$S_1, S_2, \mathcal{V}, \mathcal{U}$	69.02	69.59	**70.11**	73.35	73.05	**73.71**
GradMix w/ MixMatch [2]	$S_1, S_2, \mathcal{V}, \mathcal{U}$	**69.33**	**69.88**	70.09	**73.57**	**73.46**	73.68

unlabeled dataset $\mathcal{U}$. The proposed pseudo-label method achieves significant gain in accuracy by assigning hard labels to $\mathcal{U}$ and learn target-discriminative knowledge from the pseudo-labeled dataset. Futhermore, performance improved is achieved by incorporating state-of-the-art semi-supervised learning methods.

5. Conclusion

In this work, we propose GradMix, a method for semi-supervised MS-DTT: multi-source domain and task transfer. GradMix assigns layer-wise weights to the gradients calculated from each source objective, in a way such that the combined gradient can optimize the target objective, measured by the loss on a small validation set. GradMix can adaptively adjust the learning rate for each mini-batch based on its importance to the target task. In addition, we assign pseudo-labels to the unlabeled samples using model ensembles, and consider the pseudo-labeled dataset as a source during training. We validate the effectiveness our method with extensive experiments on two MS-DTT settings, namely digit recognition and action recognition. GradMix is a generic framework applicable to any models trained with gradient descent. For future work, we intend to extend GradMix to other problems where labeled data for the target task is expensive to acquire, such as image captioning.

Acknowledgment

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Strategic Capability Research Centres Funding Initiative. The computational work for this article was partially performed on resources of the National Supercomputing Centre, Singapore (https://www.nscc.sg).

References

[1] M. Andriluka, L. Pishchulin, P. V. Gehler, and B. Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *CVPR*, pages 3686–3693, 2014. 5

[2] D. Berthelot, N. Carlini, I. J. Goodfellow, N. Papernot, A. Oliver, and C. Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, 2019. 4, 6, 7, 8

[3] F. M. Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. R. Bulò. Autodial: Automatic domain alignment layers. In *ICCV*, pages 5077–5085, 2017. 2

[4] G. Csurka. A comprehensive survey on domain adaptation for visual applications. In *Domain Adaptation in Computer Vision Applications*, pages 1–35. Springer, 2017. 2

[5] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pages 1126–1135, 2017. 1, 2

[6] L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, and M. Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *ICML*, pages 1563–1572, 2018. 2

[7] Y. Ganin and V. S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, pages 1180–1189, 2015. 1, 2

[8] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. In *ICCV*, pages 2980–2988, 2017. 1, 2

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 5

[10] S. G. Johnson. The NLopt nonlinear-optimization package, 2008. 3

[11] S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. In *ICLR*, 2017. 4

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[13] D. Li, Y. Yang, Y. Song, and T. M. Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018. 2

[14] J. Li, J. Liu, Y. Wong, S. Nishimura, and M. S. Kankanhalli. Self-supervised representation learning using 360° data. In *ACM Multimedia*, pages 998–1006, 2019. 2

[15] J. Li, Y. Wong, Q. Zhao, and M. S. Kankanhalli. Attention transfer from web images for video recognition. In *ACM Multimedia*, pages 1–9, 2017. 7, 8

[16] J. Li, Y. Wong, Q. Zhao, and M. S. Kankanhalli. Unsupervised learning of view-invariant action representations. In *NeurIPS*, pages 1262–1272, 2018. 2

[17] J. Li, Y. Wong, Q. Zhao, and M. S. Kankanhalli. Learning to learn from noisy labeled data. In *CVPR*, pages 5051–5059, 2019. 2

[18] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou. Revisiting batch normalization for practical domain adaptation. In *ICLR*, 2017. 2

[19] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015. 2

[20] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, pages 97–105, 2015. 2

[21] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *NeurIPS*, pages 136–144, 2016. 1, 2

[22] Y. Luo, Y. Wong, M. S. Kankanhalli, and Q. Zhao. Direction concentration learning: Enhancing congruency in machine learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2019. 3

[23] Z. Luo, Y. Zou, J. Hoffman, and F. Li. Label efficient learning of transferable representations acrosss domains and tasks. In *NeurIPS*, pages 164–176, 2017. 1, 2, 5

[24] S. Ma, S. A. Bargal, J. Zhang, L. Sigal, and S. Sclaroff. Do less and achieve more: Training cnns for action recognition utilizing action images from the web. *Pattern Recognition*, 68:334–345, 2017. 5

[25] M. Mancini, L. Porzi, S. R. Bulò, B. Caputo, and E. Ricci. Boosting domain adaptation by discovering latent domains. In *CVPR*, pages 3771–3780, 2018. 1, 2, 5, 6, 7, 8

[26] T. Miyato, S. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *TPAMI*, 41(8), 2019. 4, 6, 7, 8

[27] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS workshop*, 2011. 5

[28] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, 2010. 2

[29] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 1, 2

[30] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J. B. Tenenbaum, H. Larochelle, and R. S. Zemel. Meta-learning for semi-supervised few-shot classification. In *ICLR*, 2018. 1

[31] M. Ren, W. Zeng, B. Yang, and R. Urtasun. Learning to reweight examples for robust deep learning. In *ICML*, pages 4331–4340, 2018. 2, 6

[32] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 2

[33] K. Saito, D. Kim, S. Sclaroff, T. Darrell, and K. Saenko. Semi-supervised domain adaptation via minimax entropy. In *ICCV*, 2019. 5, 6

[34] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. In *ICML*, pages 2988–2997, 2017. 4

[35] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 5

[36] B. Sun and K. Saenko. Deep CORAL: correlation alignment for deep domain adaptation. In *ECCV Workshops*, pages 443–450, 2016. 2

[37] Q. Sun, Y. Liu, T.-S. Chua, and B. Schiele. Meta-transfer learning for few-shot learning. In *CVPR*, June 2019. 2

[38] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *ICCV*, pages 4068–4076, 2015. 2

[39] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *CVPR*, pages 2962–2971, 2017. 1

[40] K. R. Weiss, T. M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *Journal of Big Data*, 3:9, 2016. 2

[41] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *CVPR*, pages 3964–3973, 2018. 1, 2, 5, 6, 7, 8

[42] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese. Taskonomy: Disentangling task transfer learning. In *CVPR*, pages 3712–3722, 2018. 1

[43] H. Zhao, S. Zhang, G. Wu, J. M. F. Moura, J. P. Costeira, and G. J. Gordon. Adversarial multiple source domain adaptation. In *NeurIPS*, pages 8568–8579, 2018. 1, 2