Towards All-around Knowledge Transferring: Learning From Task-irrelevant Labels

Yinghui Li¹, Ruiyang Liu², ZiHao Zhang³, Ning Ding¹,², Ying Shen⁴, Linmi Tao², Hai-Tao Zheng¹
¹Tsinghua Shenzhen International Graduate School, Tsinghua University
²Department of Computer Science and Technology, Tsinghua University
³Institute for Interdisciplinary Information Sciences, Tsinghua University
⁴School of Intelligent Systems Engineering, Sun Yat-Sen University
{liyinghu20, lry20, zihao-zh20}@mails.tsinghua.edu.cn

Abstract
Deep neural models have hitherto achieved significant performances on numerous classification tasks, but meanwhile require sufficient manually annotated data. Since it is extremely time-consuming and expensive to annotate adequate data for each classification task, learning an empirically effective model with generalization on small dataset has received increased attention. Existing efforts mainly focus on transferring task-relevant knowledge from other similar data to tackle the issue. These approaches have yielded remarkable improvements, yet neglecting the fact that the task-irrelevant features could bring out massive negative transfer effects. To date, no large-scale studies have been performed to investigate the impact of task-irrelevant features, let alone the utilization of this kind of features. In this paper, we firstly propose Task-Irrelevant Transfer Learning (TIRTL) to exploit task-irrelevant features, which mainly are extracted from task-irrelevant labels. Particularly, we suppress the expression of task-irrelevant information and facilitate the learning process of classification. We also provide a theoretical explanation of our method. In addition, TIRTL does not conflict with those that have previously exploited task-relevant knowledge and can be well combined to enable the simultaneous utilization of task-relevant and task-irrelevant features for the first time. In order to verify the effectiveness of our theory and method, we conduct extensive experiments on facial expression recognition and digit recognition tasks. Our source code will be also available in the future for reproducibility.

Introduction
With sufficient manually annotated training samples, deep neural networks could automatically perform feature extraction and achieve unprecedented performances on various classification tasks (Gu et al. 2018). However, collecting and annotating adequate training data is an extremely time-consuming and expensive process, leading that in many instances the training samples are insufficient and even noisy (Zhang et al. 2020). Under this circumstance, the performance of deep models always drops gravely on most classification tasks (Ajoye, Abdullah-Arshan, and Hongwu 2015; Prusa, Khoshgoftaar, and Selija 2015). The essential reason for the phenomenon is that the goal of "minimizing the empirical risk" is not reliable when the training data is inadequate (Wang et al. 2020), as a result, deep neural models will easily overfit the training data. Therefore, training effective deep neural models with remarkable generalization performance on small training samples is of great practical importance in terms of vastly expanding the scalability of deep learning methods.

Many techniques have been developed to tackle the issue. Some approaches directly expanding training samples (Cubuk et al. 2018; Bowles et al. 2018; Chen et al. 2020; Zhang et al. 2014; Shinohara 2016a) to alleviate the shortage of annotated data, but may be limited because expansion based on small-scale training samples is only of theoretical significance and not operability. On the other hand, previous studies also attempt to break through the limitation of a specific dataset and find training samples in a broad sense. Based on this motivation, transfer learning (Wang and Hebert 2016; Luo et al. 2017) exploits knowledge from additional datasets with relevant content and labels, achieving remarkable improvements on the target task. However, they only achieve good results when labels are relevant enough since irrelevant labels could bring out massive negative transfer (Weiss, Khoshgoftaar, and Wang 2016; Rosenstein et al. 2005). And to date, there have been few study points that those task-irrelevant features could even facilitate the deep learning models. In this paper, we argue that while transferring task-relevant labels from other datasets, task-irrelevant labels could also be utilized to improve the generalization of classification on small datasets without conflicts.

Our discussion is further motivated by considering a running example of computer vision: facial expression recognition (FER). In this particular task, our goal is to develop a model that could accurately predict the expression given by an unseen facial picture. At the scope of feature level, it is easy to distinguish that smiling is a task-relevant feature, and hair color is a proper task-irrelevant feature. Assume we have a small FER dataset, but unfortunately, since the dataset is extremely small, there happen to exist non-negligible bias of the distribution of task-irrelevant features, i.e., hair color. Specifically, most people with black hair are labeled as happy and people with white hair are exactly labeled as sad. This kind of bias has been observed in many previous works (Tommasi et al. 2017; Torralba and Efros 2011), thus, as a matter of fact, it is a ubiquitous problem when training...
samples are insufficient. According to the principle of minimizing the empirical risk, the task-irrelevant features create a substantial barrier for the learning process, and models would inevitably learn the hair features and become overfitting. Subsequently, the performances on unseen facial expression samples are severely affected. An intuitive method to address the issue is to make use of massive related facial information from other facial datasets by transfer learning, but as we mentioned, there would be severe negative transfer when using irrelevant facial information (hair color). Our methodology is motivated by the predicament, which is epitomized as the utilization of vast task-irrelevant (hair color) features for the overfitting problem.

In this paper, we propose Task-Irrelevant Transfer Learning (TIRTL) to explore task-irrelevant features from large-scale datasets with the same content. By minimizing the Wasserstein distance between the distribution of representation extracted from samples with a fixed task-irrelevant label and distribution from the entire dataset, consequently, we reduce the influence of task-irrelevant features. And this suppression of task-irrelevant features plays a “two negatives make a positive” role that further highlights the representation ability of task-relevant features and finally improve model generalization.

We make three-fold contributions in the paper:

• We propose TIRTL, firstly utilizing task-irrelevant labels to improve the model generalization. Furthermore, we provide a theoretical explanation of our method.

• We develop a comprehensive transfer learning framework integrated both TIRTL and previous transferring method without conflict, which could simultaneously utilize the task-relevant and task-irrelevant labels and further improve generalization performance.

• We conduct experiments and analyses in detail on FER and digit recognition. Results show that TIRTL brings significant improvements in a variety of models.

Related Work

Because of the unreliable empirical risk minimization especially when the training sample size is inadequate (Wang et al. 2020), some studies focus on extracting additional knowledge from auxiliary samples or labels, thereby helping the model utilize task-relevant features and reduce the interference of task-irrelevant features, ideally improving the generalization performance of the model on unseen test data. There are two types of methods according to the source of additional knowledge:

Obtaining additional knowledge from expanding training samples: When the training samples are insufficient, one intuitive way is to expand the dataset directly. New training samples are automatically generated from the source training samples, e.g., the data augmentations based on basic image manipulations (Cubuk et al. 2018), Moreno-Barea et al. 2018, and GAN-based data augmentations (Bowles et al. 2018, Frid-Adar et al. 2018), which have made some progress but are not safe (Shorten and Khoshgoftaar 2019). Additionally, knowledge can also be obtained from extra labels. Some multi-task learning methods (based on hard parameter sharing) (Caruana 1997, Zhang and Yang 2017, Ruder 2017) use both task labels and extra task-relevant labels of the training samples simultaneously to enhance the influence of task-relevant features. Some use several related labels and share the model feature extraction module during training (Zhang et al. 2014). Some adversarial multi-task learning using the additional irrelevant label on samples as a source for transferring (Shinohara 2016a). But in general, these kinds of the method are theoretical and not available in practice when the training sample set is very small, as the size of samples which are available to add other labels is insufficient.

Obtaining additional knowledge from labels of auxiliary samples: The shortcomings of the above methods encourage researchers to break through the limitation of a specific dataset and exploit training samples in a broad sense. One possible way is to utilize a relevant dataset. Transfer learning transfers the knowledge from the relevant source dataset with task-relevant labels to the target task, thereby improving the performance on the target dataset (Wang and Hebert 2016, Luo et al. 2017, Deng et al. 2009, Marmanis et al. 2015). In addition, another type of multi-task learning methods (based on soft parameter sharing) train several related tasks at the same time to learn common task-relevant features and improve the performance on all of these tasks (Yan, Yap, and Mori 2015, Yang and Hospedales 2016). Both multi-task learning and transfer learning require a strong correlation between the source tasks (or labels) and the target task (Weiss, Khoshgoftaar, and Wang 2016), and many studies have empirically shown that negative transfer may happen when two tasks are unrelated (Rosenstein et al. 2005, Wang et al. 2019a).

Overall, as far as we know, few methods could obtain and utilize the knowledge directly from task-irrelevant labels. The studies of negative transfer (Ge et al. 2014, Seah, Ong, and Tsang 2012, Wang et al. 2019b) focus on how to reduce the knowledge that would trigger negative transfer, instead of using them. Our method is different in that we directly utilize this knowledge to improve the performance of the deep models. Compared with these works, our work provides a deeper insight into knowledge from task-irrelevant labels.

Methodology

Task-Irrelevant Transfer Learning (TIRTL) focuses on exploiting the knowledge contained in task-irrelevant labels from auxiliary samples and transferring it to the target task. With this knowledge, we measure the influence of task-irrelevant features, and then use the adversarial learning method based on Wasserstein distances to limit such influence during feature extraction, thereby helping the model focus on task-relevant features.

Task-Irrelevant Labels

In the first place, task-irrelevant features are defined as the features irrelevant to the target task, and similarly, task-relevant features are features relevant to the target task. For example, hair color is a task-irrelevant feature for the FER
task. Correspondingly, if samples from some irrelevant tasks contain the same content but different labels with the target task, the labels are termed as task-irrelevant labels, and the samples are termed as task-irrelevant samples.

For a specific target task, and a dataset with task-irrelevant (TIR) labels, \( S = \{(x, y_{tir})\} \), we denote the feature space by \( X \), \( x \in X \), and the label space of task-irrelevant labels by \( Y; \ Y = \{y_1, \cdots, y_n\}; \ y_{tir} \in Y \). Input \( x \) contains some features that are task-relevant, denoted by \( x_{tr} \) and the distribution of \( x_{tr} \) in \( S \) is denoted by \( \mathbb{D}_S \).

We select all samples whose \( y_{tir} = y_1 \) from \( S \), denoted by \( S_{y_1} \):

\[
S_{y_1} = \{(x, y_{tir})|y_{tir} = y_1, (x, y_{tir}) \in S\},
\]

and the distribution of \( x_{tr} \) in \( S_{y_1} \) is denoted by \( \mathbb{D}_{S_{y_1}} \). Since \( y_{tir} \) is irrelevant with \( x_{tr} \), we have:

\[
\mathbb{D}_S = \mathbb{D}_{S_{y_1}}.
\]

In the deep models, the feature extractor is used to extract representation from input data. We denote the feature extractor by \( f_e \) and the extracted representation by \( r \). Here, \( R \) and \( R_{y_1} \) are extracted from \( S \) and \( S_{y_1} \),

\[
R = \{(r, y_{tir})|r = f_e(x), (x, y_{tir}) \in S\}
\]

\[
R_{y_1} = \{(r, y_{tir})|r = f_e(x), (x, y_{tir}) \in S_{y_1}\}.
\]

Similarly, the distributions of \( r \) in \( R \) and \( R_{y_1} \) are denoted by \( \mathbb{D}_R \) and \( \mathbb{D}_{R_{y_1}} \).

Under ideal conditions, a good extractor only extracts task-relevant features, so \( \mathbb{D}_R = \mathbb{D}_{R_{y_1}} \). However, in reality, the extractor inevitably extracts task-irrelevant features and leads to \( \mathbb{D}_R \neq \mathbb{D}_{R_{y_1}} \). The greater the difference between \( \mathbb{D}_R \) and \( \mathbb{D}_{R_{y_1}} \), the more the extractor is influenced by task-irrelevant features. To reduce this difference, TIRTL aims to minimize the divergence between these two distributions,

\[
\min(\text{WD}(\mathbb{D}_R, \mathbb{D}_{R_{y_1}})),
\]

where Wasserstein distance, denoted by WD, is used to measure the divergence between the two distributions. In Appendix A, we provide a further theoretical analysis of Wasserstein distance’s efficacy and its generalization bound.

**TIRTL Framework**

Figure 1 shows the framework of TIRTL. Generally, the framework is divided into three parts: a feature extractor \( f_e \), a linear classifier \( f_c \), and an additional discriminator \( f_d \). We denote the dataset of the target task by \( S_{tgt} \), and the task-irrelevant dataset by \( S_{src} \). To utilize both \( S_{tgt} \) and \( S_{src} \), two optimization goals are set:

- **Goal 1** is the empirical risk minimization that we minimize the classification error so that the model learns the knowledge contained in the samples of \( S_{tgt} \).
- **Goal 2** aims to minimize the Wasserstein distance as Equation 4 so that reduces the influence of task-irrelevant features during feature extraction with the helping of \( S_{src} \).

**Goal 1.** For training sample \( x \) and label \( y, (x, y) \in S_{tgt} \), the classification probability \( p = \text{softmax}(f_c(f_e(x))) \), and
the classification loss is calculated by cross-entropy, which is consistent with the loss in empirical risk.

\[
\mathcal{L}_{\text{classification}} = - \sum_{x, y : y \in S_{\text{tgt}}} y \log (\text{softmax}(f_c(f_e(x)))).
\]

(5)

The model uses Goal 1 to learn the mapping of input features to labels on the target dataset, but the generalization ability of the model trained on goal 1 depends heavily on the consistency of the training data distribution and the real data distribution, which is usually hard to achieve on a small dataset. Thus, a new optimization goal should be imported to utilize large-scale task-relevant or task-irrelevant data to reduce this bias.

**Goal 2.** This goal aims to minimize the Wasserstein distance between \(\mathbb{D}_{R_A}\) and \(\mathbb{D}_R\) as Equation 4, and it is necessary to estimate the two distributions by sampling. For \(S_{\text{src}}\), we randomly select two groups of samples, one from samples with a specific task-irrelevant label \(y_1\) in \(S_{\text{src}}\), denoted by \(A\), and the other from the entire \(S_{\text{src}}\), denoted by \(B\).

For \(A\) and \(B\), the representations extracted by \(f_e\) are denoted by \(R_A\) and \(R_B\):

\[
R_A = \{f_e(x) | x \in A\}
\]

(6)

\[
R_B = \{f_e(x) | x \in B\}.
\]

When the sizes of \(A\) and \(B\) are large enough, the representation distributions of \(R_A\) and \(R_B\), denoted by \(\mathbb{D}_{R_A}\) and \(\mathbb{D}_{R_B}\), are used as reasonable estimations of \(\mathbb{D}_{R_{\text{src}}}\) and \(\mathbb{D}_R\). Thus, Equation 4 is rewritten as:

\[
\min \sup_{\|f_d\|_{L_\infty}} \mathbb{E}_{x \in A} [f_d(f_e(x))] - \mathbb{E}_{x \in B} [f_d(f_e(x))].
\]

(7)

We substitute the WD with Wasserstein’s Kantorovich-Rubinstein duality form,

\[
\min \sup_{\|f_d\|_{L_1}} \mathbb{E}_{x \in A} [f_d(f_e(x))] - \mathbb{E}_{x \in B} [f_d(f_e(x))].
\]

(8)

Inspired by WGAN [Arjovsky, Chintala, and Bottou 2017], Equation 8 could be divided into two steps. Firstly, the discriminator \(f_d\) is trained by:

\[
\max_{f_d} \mathbb{E}_{x \in A} [f_d(f_e(x))] - \mathbb{E}_{x \in B} [f_d(f_e(x))].
\]

(9)

Secondly, we select a group of training samples, \(C\), from training dataset of target task \(S_{\text{tgt}}\). The feature representations extracted by \(f_e\) are denoted by \(R_C\),

\[
R_C = \{f_e(x) | x \in C\},
\]

(10)

then, the extractor \(f_e\) is trained by:

\[
\min_{f_e} - \mathbb{E}_{x \in C} [f_d(f_e(x))].
\]

(11)

and the corresponding Wasserstein loss is:

\[
\mathcal{L}_{\text{wasserstein}} = - \frac{1}{|C|} \sum_{x \in C} |f_d(f_e(x))|,
\]

(12)

where \(|C|\) is the size of the sample group \(C\).

Combining \(\mathcal{L}_{\text{classification}}\) and \(\mathcal{L}_{\text{wasserstein}}\), the complete loss function is written as follows, where \(\lambda_1\) and \(\lambda_2\) are weighting factors:

\[
\mathcal{L}_{\text{TIRTL}} = \lambda_1 \mathcal{L}_{\text{classification}} + \lambda_2 \mathcal{L}_{\text{wasserstein}}.
\]

(13)

In practice, the training algorithm is shown in the Alg. 1.

**Algorithm 1 Algorithm of TIRTL.**

**Require:** target training sample set \(S_{\text{tgt}}\); task-irrelevant sample set \(S_{\text{src}}\); learning rate for discriminator \(\alpha_1\); learning rate for feature extractor and classifier \(\alpha_2\); batch size \(m\); iteration numbers \(n_1, n_2\).

**Notation:** function fitted by the feature extractor, \(f_e\); function fitted by the classifier, \(f_c\); function fitted by the discriminator, \(f_d\).

1: Initialize feature extractor, linear classifier, discriminator with random weights \(\theta_e, \theta_c, \theta_d\).
2: for \(n_1\) steps do
3: \hspace{1em} Sample minibatch \(A = \{a(i)\}_{i=1}^m\) from \(S_{\text{src}}\) with a fixed task-irrelevant label.
4: \hspace{1em} Sample minibatch \(B = \{b(i)\}_{i=1}^m\) from \(S_{\text{src}}\) with random task-irrelevant labels.
5: \hspace{1em} \(g_d \leftarrow \nabla d \frac{1}{m} \sum_{i=1}^m (f_d(f_e(a(i))) - f_d(f_e(b(i))))\)
6: \hspace{1em} \(\theta_d \leftarrow \theta_d + \alpha_1 \text{Adam}(\theta_d, g_d)\)
7: for \(n_2\) steps do
8: \hspace{1em} Sample minibatch \(C = \{x(i), y(i)\}_{i=1}^m\) from \(S_{\text{tgt}}\).
9: \hspace{1em} \textbf{Goal 1:}
10: \hspace{2em} \(g_c \leftarrow \nabla c \frac{1}{m} \sum_{i=1}^m (y(i) \cdot \log f_c(f_e(x(i))))\)
11: \hspace{2em} \(\theta_c \leftarrow \theta_c + \alpha_2 \text{SGD}(\theta_c, g_c)\)
12: \hspace{2em} \(g_e \leftarrow \nabla e \frac{1}{m} \sum_{i=1}^m (y(i) \cdot \log f_c(f_e(x(i))))\)
13: \hspace{2em} \(\theta_e \leftarrow \theta_e + \alpha_2 \text{SGD}(\theta_e, g_e)\)
14: \hspace{1em} \textbf{Goal 2:}
15: \hspace{2em} \(g_e \leftarrow \nabla e \frac{1}{m} \sum_{i=1}^m (-f_d(f_e(x(i))))\)
16: \hspace{2em} \(\theta_e \leftarrow \theta_e + \alpha_2 \text{SGD}(\theta_e, g_e)\)

**All-around Knowledge Transferring**

TIRTL can be integrated with other transfer learning methods without conflicts. As shown in Figure 1 to combine TIRTL with task-relevant transfer learning, the straightforward way is to perform a mature transfer learning method with task-relevant samples at first and then use TIRTL. The combined model could be jointly optimized:

\[
\mathcal{L} = \mathcal{L}_{\text{TIRTL}} + \mathcal{L}_{\text{TRTL}},
\]

(14)

where \(\mathcal{L}_{\text{TIRTL}}\) denotes the loss of task-relevant transfer learning (TRTL). Here, \(\mathcal{L}_{\text{TIRTL}}\) utilizes training samples as well as task-irrelevant samples, and \(\mathcal{L}_{\text{TRTL}}\) utilizes training samples as well as task-relevant samples. The integrated model makes better use of the knowledge in both task-irrelevant samples and task-relevant samples, and empirical studies show that the integrated model achieves better generalization performance than solely using transfer learning.

**Experiment**

In this section, we conduct extensive experiments to evaluate the effectiveness of our proposed TIRTL in improving the model generalization. A series of subsidiary experiments are carried out for deep analysis and some of the results are reported in Appendix D.

**Experimental Setup**

**Model.** The framework of TIRTL consists of three parts: a feature extractor, a linear classifier, and a discriminator. The
Table 1: Results (Accuracy %) on FER task, where TL refers to transfer learning, MTL refers to multi-task learning, and AMTL refers to adversarial multi-task learning. ck+→mmi means model is trained on ck+ and tested on mmi dataset. TL (Hair Color) means using hair color label in transfer learning.

| Feature Extractor | Train → Test Dataset | Goal 1-Only  | TL (Hair Color) | MTL (Hair Color) | AMTL (Hair Color) | TIRTL (Hair Color) |
|-------------------|----------------------|--------------|-----------------|------------------|-------------------|-------------------|
| AlexNet           | ck+ → mmi            | 35.91        | 38.28           | 33.73            | 34.57             | 37.61             |
|                   | ck+ → oulu           | 34.75        | 25.54           | 33.08            | 33.15             | 38.31             |
|                   | mmi → ck+            | 56.36        | 44.97           | 57.09            | 57.94             | 61.45             |
|                   | mmi → oulu           | 22.12        | 19.89           | 23.66            | 28.73             | 22.61             |
|                   | oulu → ck+           | 55.03        | 54.42           | 54.67            | 43.52             | 59.64             |
|                   | oulu → mmi           | 39.46        | 40.00           | 40.30            | 28.33             | 45.36             |
|                   | Average              | 40.61        | 37.18           | 40.42            | 37.37             | 44.16             |
| ResNet34          | ck+ → mmi            | 50.42        | 46.54           | 46.54            | 44.35             | 50.42             |
|                   | ck+ → oulu           | 50.94        | 54.29           | 47.24            | 43.75             | 51.78             |
|                   | mmi → ck+            | 65.33        | 68.73           | 66.18            | 64.61             | 69.21             |
|                   | mmi → oulu           | 44.87        | 46.13           | 40.96            | 42.15             | 42.43             |
|                   | oulu → ck+           | 73.45        | 73.33           | 72.93            | 66.18             | 80.12             |
|                   | oulu → mmi           | 54.13        | 49.01           | 51.43            | 41.99             | 54.30             |
|                   | Average              | 56.52        | 56.34           | 54.21            | 50.51             | 58.04             |
| VggNet19          | ck+ → mmi            | 45.53        | 41.10           | 37.10            | 33.05             | 45.53             |
|                   | ck+ → oulu           | 56.66        | 44.52           | 32.24            | 47.66             | 57.22             |
|                   | mmi → ck+            | 65.33        | 62.30           | 64.61            | 57.82             | 66.79             |
|                   | mmi → oulu           | 45.08        | 32.17           | 46.69            | 40.20             | 46.76             |
|                   | oulu → ck+           | 76.73        | 71.27           | 72.12            | 31.88             | 78.91             |
|                   | oulu → mmi           | 45.03        | 43.68           | 40.00            | 20.24             | 51.43             |
|                   | Average              | 55.73        | 49.17           | 48.79            | 38.48             | 57.77             |

Table 1: Results (Accuracy %) on FER task, where TL refers to transfer learning, MTL refers to multi-task learning, and AMTL refers to adversarial multi-task learning. ck+→mmi means model is trained on ck+ and tested on mmi dataset. TL (Hair Color) means using hair color label in transfer learning.

**Evaluation Metrics.** To evaluate the model generalization, the cross-dataset test results are taken as evaluation metrics. This is based on a reasonable assumption: the training set and the real data distribution are different, and the unseen test set is sampled from the real data distribution, so there exists a difference between the distribution of test data and the distribution of training data. Therefore, the higher the accuracy in our test means the better the generalization performance of the model.

**Hyperparameters.** We implement all experiments without data augmentation. The model is trained by an SGD optimizer with an initial learning rate of 0.001, the momentum of 0.9, StepLR(step size is 7), and γ of 0.1. We apply widely used classification models (e.g ResNet) as a feature extractor and set the output dimension of the penultimate layer to 128. The classifier is a fully connected layer and the output dimension is equal to the number of classes (7 for FER and 10 for digit recognition). The discriminator consists of four fully connected layers with 512, 256, 10, 1 nodes, the discriminator is trained using an Adam with the same learning rate, and a weight limit 0.1.

In addition, the step ratio of $n_2$ in Alg.1 is 3. The factors $\lambda_1$ and $\lambda_2$ are both 1. We use batch size 32 in FER, and 128 in digit recognition. All results are obtained after the model has been trained for 50 epochs.

**Experiments on Facial Expression Recognition**

In this part, we compare TIRTL with some methods related to our work on the FER task. These methods are briefly described below, and their implementation details are shown in Appendix B:

- **Goal 1 only**: As an empirical baseline, we use labeled training data to train the model with the optimization goal 1 only.
- **Transfer learning**: We use the hair color recognition task to pre-train the network, and then fine-tune on FER task.
- **Multi-task learning**: We train FER and hair color recognition tasks using a shared feature extractor.
- **Adversarial multi-task learning**: We add a gradient inversion layer for the hair color recognition task on the multi-task learning method above.

**Datasets and Feature Extractors.** In FER experiments, we use small-scale datasets, including ck+ (Lucey et al. 2010), oulu (Zhao et al. 2011) and mmi (Pantic et al. 2005) for cross-dataset evaluation. In addition, we select AlexNet (Krizhevsky, Sutskever, and Hinton 2012), VggNet19 (Simonyan and Zisserman 2015) and ResNet34 (He et al. 2016) as feature extractors. The datasets used in the experiment are detailed in Appendix C.
In more detail, we use $224 \times 224$ resolution for all RGB pictures and preprocess them through MTCNN (Zhang et al., 2016) for alignment. More importantly, the hair color labels in CelebA (Liu et al., 2015) are regarded as task-irrelevant labels. Therefore, we select samples with fixed hair color labels and samples randomly selected from CelebA, and then using TIRTL on them.

**Results.** Table 1 is the comparison between TIRTL and baseline methods. Our method achieves better results in 14 of 18 cross-dataset tests. The improvement of model generalization could be attributed to extra knowledge from task-irrelevant features.

The results in Table 1 also show that using transfer learning, multi-task learning or adversarial multi-task learning methods reduce the model generalization, while using our TIRTL improve the model generalization. A likely explanation is that since the hair color is not relevant to FER, it leads to negative transfer. So the generalization of the model is getting worse when using transfer learning or multi-task learning. The performance of the adversarial multi-task learning is the worst because this method requires training samples with both the task label and the task-irrelevant label. However, in our experiments, the training samples are only annotated with task labels, and the task-irrelevant labels are from additional samples. The results indicate that TIRTL successfully deals with negative transfer and make use of the knowledge from task-irrelevant labels to improve the generalization performance of the model.

**Experiments on Digit Recognition**

In this part, we evaluate TIRTL on the Digit Recognition task. In this experiment, we add another baseline, DANN (Ajanan et al., 2014), a domain transfer framework based on adversarial multi-task learning. The model details of DANN and the experiment results to verify the correctness of the DANN we implement are shown in Appendix B and Appendix D, respectively.

**Datasets and Feature Extractors.** In the digit recognition experiments, regarding the background color as the task-irrelevant features, we use 20000 pictures of MNIST (LeCun et al., 1998) as the training set to test on SVHN (Netzer et al., 2011) and MNIST-M (Ganin et al., 2016). The two groups of samples, one is sampled from MNIST, and the other is randomly sampled from the combination of MNIST and SVHN. Besides, all pictures are converted to RGB pictures and we use $32 \times 32$ resolution for them. The datasets used in the experiment are detailed in Appendix C.

As for feature extractor, we select ResNet18 (He et al., 2016), VggNet11 (Simonyan and Zisserman, 2015), DenseNet121 (Huang et al., 2017), SeNet (Hu, Shen, and Sun, 2018), EfficientNet (Tan and Le, 2019), MobileNetV2 (Sandler et al., 2018) and ShuffleNetV2 (Ma et al., 2018).

**Results.** Table 2 is the comparison between our method and the two baselines. Compared with these two baselines, the experimental results show that TIRTL achieves a better result than DANN and the “Goal 1 only” method. Although transferring from MNIST to SVHN is difficult since SVHN is more complex than MNIST, TIRTL facilitates this process by using knowledge of task-irrelevant features and improve model generalization.

**Analysis**

**The impact of different levels of task-irrelevance.** As different task-relevan can affect the performance of transfer learning, it is essential to study the impact of different task-irrelevance on TIRTL. Except for the hair color label, we use the smiling label from CelebA. We apply these labels to the TIRTL framework and compare their performance to

| Train → Test | Feature Extractor | Goal 1 Only | TIRTL (Background Color) | TIRTL (Smiling) | TIRTL (Hair Color) |
|--------------|-------------------|------------|--------------------------|-----------------|-------------------|
| MNIST → SVHN | DenseNet121       | 13.85      | 10.64                    | 16.44           |                   |
|              | EfficientNet      | 22.99      | 13.99                    | 33.74           |                   |
|              | MobileNetV2       | 20.06      | 11.35                    | 22.11           |                   |
|              | ResNet18          | 15.01      | 12.00                    | 18.83           |                   |
|              | SeNet             | 15.44      | 10.69                    | 18.22           |                   |
|              | ShuffleNetV2      | 14.29      | 11.98                    | 20.72           |                   |
|              | VggNet11          | 19.50      | 13.79                    | 25.71           |                   |
|              | Average           | 17.31      | 12.06                    | 22.25           |                   |
| MNIST → MNIST-M | DenseNet121       | 25.26      | 14.28                    | 35.21           |                   |
|              | EfficientNet      | 40.78      | 38.71                    | 57.64           |                   |
|              | MobileNetV2       | 29.77      | 21.75                    | 48.76           |                   |
|              | ResNet18          | 25.10      | 22.54                    | 37.64           |                   |
|              | SeNet             | 17.24      | 15.83                    | 39.71           |                   |
|              | ShuffleNetV2      | 27.96      | 20.34                    | 47.65           |                   |
|              | VggNet11          | 39.18      | 42.09                    | 48.98           |                   |
|              | Average           | 32.93      | 25.08                    | 45.08           |                   |

**Table 2: Results (Accuracy %) on digit recognition task.**

| Feature Extractor | Train → Test | Goal 1 Only | TIRTL (Smiling) | TIRTL (Hair Color) |
|-------------------|--------------|-------------|-----------------|-------------------|
| MNIST → SVHN      | DenseNet121  | ck+ → mmi  | 35.91           | 37.61             |
|                   |              | ck+ → oulu | 34.75           | 38.31             |
|                   |              | mmi → ck+  | 56.36           | 61.45             |
|                   |              | Average    | 40.61           | 44.16             |
| MNIST → MNIST-M   | DenseNet121  | ck+ → mmi  | 50.42           | 50.42             |
|                   |              | ck+ → oulu | 50.94           | 51.78             |
|                   |              | mmi → ck+  | 65.33           | 69.21             |
|                   | ResNet34     | mmi → oulu | 44.87           | 42.43             |
|                   |              | oulu → ck+ | 73.45           | 80.12             |
|                   |              | oulu → mmi | 54.13           | 54.30             |
|                   |              | Average    | 56.52           | 58.04             |
| MNIST → SVHN      | AlexNet      | ck+ → mmi  | 45.53           | 45.53             |
|                   |              | ck+ → oulu | 56.66           | 57.22             |
|                   |              | mmi → ck+  | 65.33           | 66.79             |
|                   | DenseNet121  | mmi → oulu | 45.08           | 46.76             |
|                   |              | oulu → ck+ | 76.73           | 78.91             |
|                   |              | oulu → mmi | 45.03           | 51.43             |
|                   |              | Average    | 55.73           | 57.77             |

**Table 3: Results (Accuracy %) on FER task, using different labels (smiling and hair color) in TIRTL.**
evaluate the impact of different task-irrelevance.

Table 3 indicates that the experiment results that compared with the "Goal 1 only" method, the performance of TIRTL remains the same or gets worse when using the smiling label, but have significant improvement when using the hair color label. The experiment results suggest that using a more irrelevant label is better for the TIRTL framework to improve the performance since the TIRTL framework tries to use the knowledge of task-irrelevant features. On the other hand, if we misuse task-relevant samples in TIRTL, it will have a negative impact. This is similar to the negative transfer while using task-irrelevant labels in transfer learning. In general, The phenomenon means the TIRTL is different from transfer learning, and it benefits from the irrelevance, instead of relevance.

PCA analysis on feature representations. Based on the maximum variance theory (Tipping and Bishop 1999), if the percentage of variance explained by principal components increase, the representation vector will contain more information about task-relevant features, and task-relevant information will be better retained during the process of dimensionality reduction. Due to the paper’s length limitation, we only use the results of ResNet18 as an example, and the other results are listed in Appendix D.

![PCA results of ResNet18 on the digit recognition task.](image)

Figure 2: PCA results of ResNet18 on the digit recognition task. The larger the proportion of the dark green part (principal component 1 & 2), the larger the proportion of the components that are decisive to the task in the representation.

Figure 2 shows the proportion of principal components of different methods in digit recognition tasks. Compared with other methods (Goal 1 only and DANN), the proportion of task-relevant information is much higher in TIRTL, which indicates that the feature representation vectors are easier to aggregate in TIRTL. A possible explanation is that TIRTL gets knowledge from task-irrelevant labels and avoid extracting task-irrelevant features during feature extraction, corresponding to the effect of increasing the proportion of task-relevant features.

Combining TIRTL with Task-relevant Transfer Learning. The above experiments point that TIRTL enables task-irrelevant samples to positively influence the target learner. Recall that existing transfer learning methods utilize task-relevant samples to improve the performance of the target learner. This prompted us to consider combining both task-relevant and task-irrelevant samples in the target task to obtain better performance. Owing to the scalability and flexibility of TIRTL, it is easily integrated with existing transfer learning methods. Specifically, we pre-train the model on a large FER dataset, RAF (Li, Deng, and Du 2017a), and then finetune it on a small training dataset by TIRTL.

Table 4 indicates that although task-relevant transfer learning brings great performance improvement without the participation of TIRTL, combining with the TIRTL framework can further yields better performance of model generalization. The results are heartening enough to drive future development of all-around transfer learning combining task-relevant and task-irrelevant samples.

### Conclusion

In order to solve the problem caused by the unreliable empirical risk minimization when the training dataset is small, we propose a novel framework, TIRTL, to exploit the knowledge from additional task-irrelevant labels. Based on the detailed theoretical analysis, we illustrate that samples with task-irrelevant labels can be used in improving the generalization performance of the model, and propose a novel task-irrelevant transfer learning framework (TIRTL), which directly obtains and utilizes knowledge from a wide range of task-irrelevant labels. Furthermore, we combine TIRTL with the existing task-relevant transfer learning method, which truly realizes all-around knowledge transferring. We believe that the exploration of task-irrelevant labels in our work has profound guiding significance for the improvement of the transfer learning system.
References

Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; and Marchand, M. 2014. Domain-adversarial neural networks. arXiv preprint arXiv:1412.4446.

Ajiboye, A.; Abdullah-Arshah, R.; and Hongwu, Q. 2015. Evaluating the effect of dataset size on predictive model using supervised learning technique.

Arbelaez, P.; Maire, M.; Fowlkes, C.; and Malik, J. 2010. Contour detection and hierarchical image segmentation. IEEE transactions on pattern analysis and machine intelligence 33(5): 898–916.

Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein gan. arXiv preprint arXiv:1701.07875.

Bowles, C.; Chen, L.; Guererro, R.; Bentley, P.; Gunn, R.; Hammers, A.; Dickie, D. A.; Hernández, M. V.; Wardlaw, J.; and Rueckert, D. 2018. Gan augmentation: Augmenting training data using generative adversarial networks. arXiv preprint arXiv:1810.10863.

Caruana, R. 1993. Multitask Learning: A Knowledge-Based Source of Inductive Bias. In Proceedings of the Tenth International Conference on Machine Learning, 41–48. Morgan Kaufmann.

Caruana, R. 1997. Multitask learning. Machine learning 28(1): 41–75.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. arXiv preprint arXiv:2002.05709.

Cubuk, E. D.; Zoph, B.; Mane, D.; Vasudevan, V.; and Le, Q. V. 2018. Autoaugment: Learning augmentation policies from data. arXiv preprint arXiv:1805.09501.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, 248–255. Ieee.

Frid-Adar, M.; Klang, E.; Amitai, M.; Goldberger, J.; and Greenspan, H. 2018. Gan-based data augmentation for improved liver lesion classification.

Ganin, Y.; and Lempitsky, V. 2015. Unsupervised domain adaptation by backpropagation. In International conference on machine learning, 1180–1189. PMLR.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. The Journal of Machine Learning Research 17(1): 2096–2030.

Ge, L.; Gao, J.; Ngo, H.; Li, K.; and Zhang, A. 2014. On handling negative transfer and imbalanced distributions in multiple source transfer learning. Statistical Analysis and Data Mining: The ASA Data Science Journal 7(4): 254–271.

Gu, J.; Wang, Z.; Kuen, J.; Ma, L.; Shahroudy, A.; Shuai, B.; Liu, T.; Wang, X.; Wang, G.; Cai, J.; et al. 2018. Recent advances in convolutional neural networks. Pattern Recognition 77: 354–377.

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.

Hu, J.; Shen, L.; and Sun, G. 2018. Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, 7132–7141.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, 4700–4708.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In Pereira, F.; Burges, C. J. C.; Bottou, L.; and Weinberger, K. Q., eds., Advances in Neural Information Processing Systems 25, 1097–1105. Curran Associates, Inc. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

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

Li, S.; Deng, W.; and Du, J. 2017a. Reliable crowdsourcing and deep locality-preserving learning for expression recognition in the wild. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2852–2861.

Li, S.; Deng, W.; and Du, J. 2017b. Reliable Crowdsourcing and Deep Locality-Preserving Learning for Expression Recognition in the Wild. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2584–2593. IEEE.

Luo, Z.; Luo, P.; Wang, X.; and Tang, X. 2015. Deep Learning Face Attributes in the Wild. In Proceedings of International Conference on Computer Vision (ICCV).

Lucey, P.; Cohn, J. F.; Kanade, T.; Saragih, J.; Ambadar, Z.; and Matthews, I. 2010. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In 2010 ieee computer society conference on computer vision and pattern recognition workshops, 94–101. Ieee.

Ma, N.; Zhang, X.; Zheng, H.-T.; and Sun, J. 2018. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Proceedings of the European conference on computer vision (ECCV), 116–131.

Marmanis, D.; Datcu, M.; Esch, T.; and Stilla, U. 2015. Deep learning earth observation classification using imagenet pretrained networks. IEEE Geoscience and Remote Sensing Letters 13(1): 105–109.

Mohri, M.; Rostamizadeh, A.; and Talwalkar, A. 2012. Foundations of Machine Learning. Adaptive computation and machine learning. MIT Press. ISBN 978-0-262-01825-8.
Moreno-Barea, F. J.; Strazzer, F.; Jerez, J. M.; Urda, D.; and Franco, L. 2018. Forward noise adjustment scheme for data augmentation. In 2018 IEEE Symposium Series on Computational Intelligence (SSCI), 728–734. IEEE.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading digits in natural images with unsupervised feature learning.

Pantic, M.; Valstar, M.; Rademacher, R.; and Maat, L. 2005. Web-based database for facial expression analysis. In 2005 IEEE international conference on multimedia and Expo, 5–pp. IEEE.

Prusa, J.; Khoshgoftaar, T. M.; and Seliya, N. 2015. The effect of dataset size on training tweet sentiment classifiers. In 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), 96–102. IEEE.

Rosenstein, M. T.; Marx, Z.; Kaelbling, L. P.; and Dietterich, T. G. 2005. To transfer or not to transfer. In NIPS 2005 workshop on transfer learning, volume 898, 1–4.

Ruder, S. 2017. An overview of multi-task learning in deep neural networks. arXiv preprint arXiv:1706.05098.

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition, 4510–4520.

Seah, C.-W.; Ong, Y.-S.; and Tsang, I. W. 2012. Combating negative transfer from predictive distribution differences. IEEE transactions on cybernetics 43(4): 1153–1165.

Shen, J.; Qu, Y.; Zhang, W.; and Yu, Y. 2017. Wasserstein distance guided representation learning for domain adaptation. arXiv preprint arXiv:1707.01217.

Shinohara, Y. 2016a. Adversarial Multi-Task Learning of Deep Neural Networks for Robust Speech Recognition. In Interspeech, 2369–2372. San Francisco, CA, USA.

Shinohara, Y. 2016b. Adversarial Multi-Task Learning of Deep Neural Networks for Robust Speech Recognition. In Interspeech, 2369–2372. San Francisco, CA, USA.

Shorten, C.; and Khoshgoftaar, T. M. 2019. A survey on image data augmentation for deep learning. Journal of Big Data 6(1): 60.

Simonyan, K.; and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In International Conference on Learning Representations.

Tan, M.; and Le, Q. V. 2019. Efficientnet: Rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946.

Tipping, M. E.; and Bishop, C. M. 1999. Mixtures of probabilistic principal component analyzers. Neural computation 11(2): 443–482.

Tommasi, T.; Patricia, N.; Caputo, B.; and Tuytelaars, T. 2017. A deeper look at dataset bias. In Domain adaptation in computer vision applications, 37–55. Springer.

Torralba, A.; and Efros, A. A. 2011. Unbiased look at dataset bias. In CVPR 2011, 1521–1528. IEEE.
Appendices

A. Theoretical Analysis

We begin with a brief review of the unreliable empirical risk minimizer, then make theoretical analysis for the proposed Task-Irrelevant Transfer Learning (TIRTL).

Unreliability of Minimizing Empirical risk

For a learning task \( T \), we are given a training sample set, which includes \( I \) labeled instances \( D_{\text{train}} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_I, y_I)\} \). Let \( p(x, y) \) be the ground-truth joint probability distribution of sample \( x \) and label \( y \). We denote the space of input instance by \( \mathcal{X} \), and the space of task labels by \( \mathcal{Y} \), \( x \in \mathcal{X} \) and \( y \in \mathcal{Y} \). The goal of deep learning is to learn a model \( f(x; \theta) : \mathcal{X} \rightarrow \mathcal{Y} \) parameterized by \( \theta \in \Theta \) from training data to minimize the expected risk, i.e.,

\[
\min_{f} R(f) = \int L(f(x; \theta), y)dp(x, y),
\]

where \( p \) refers to a certain loss function, e.g., mean squared error or cross entropy loss.

As \( p(x, y) \) is unknown, the most classical approach is to approximate the expected risk by minimizing the empirical risk \( R_{\text{emp}}(f) \) may then be far from being a good approximation of the expected risk \( R(f) \). The empirical risk minimizer is no longer reliable (Wang et al. 2020). The bias of task-irrelevant features in the training dataset is one of the main causes of the deviation between training data distribution and \( p(x, y) \). We believe that weakening the impact of task-irrelevant features will help train models with better generalization performance, which requires introducing knowledge of task-irrelevant features from samples with task-irrelevant labels.

Theoretical Analysis of TIRTL

This theoretical analysis is based on (Shen et al. 2017), which gives a generalization bound of error in the domain adaptation problem. We make some slight modifications and make it suitable for our method.

The classification model is divided into two parts, a classifier \( f_c \), and a feature extractor \( f_e \). We denote the feature space of input instances by \( \mathcal{X} \). After sending an input instance to the extractor \( f_e \), we get a representation \( r \). The space of the representation is denoted by \( \mathcal{R} \), and \( f_e : \mathcal{X} \rightarrow \mathcal{R} \). Besides, the label space is denoted by \( \mathcal{Y} \), and \( f_e : \mathcal{R} \rightarrow \mathcal{Y} \).

\( H \) is a hypothesis class that for every \( h \in H \), \( h : \mathcal{R} \rightarrow \mathcal{Y} \) and \( h \) is \( K \)-Lipschitz continuous. In neural networks, we limit the scale of the weights so that \( f_e \) is \( K \)-Lipschitz continuous and \( f_e \in H \). For every distribution \( \mathcal{D} \) on \( \mathcal{X} \), the corresponding distribution of representation is denoted by \( \mathcal{R} \), \( \mathcal{R} = f_e(\mathcal{D}) \). The difference of \( h_1 \) and \( h_2 \) \((h_1, h_2 \in H)\) on \( \mathcal{R} \) are defined by:

\[
\epsilon_\mathcal{R}(h_1, h_2) = \mathbb{E}_{r \sim \mathcal{R}} ||h_1(r) - h_2(r)||.
\]

Similarly,

\[
\epsilon_\mathcal{D}(h_1 \circ f_e, h_2 \circ f_e) = \mathbb{E}_{x \sim \mathcal{D}} ||h_1(f_e(x)) - h_2(f_e(x))||.
\]

**Theorem 1** For representation distributions \( \mathcal{R}_1, \mathcal{R}_2 \) on \( \mathcal{R} \), and \( h_1, h_2 \in H \). Then the following holds:

\[
\epsilon_{\mathcal{R}_1}(h_1, h_2) \leq \epsilon_{\mathcal{R}_2}(h_1, h_2) + 2K \cdot WD(\mathcal{R}_1, \mathcal{R}_2).
\]

**Proof.** To begin with, we need to prove that \( |h_1 - h_2| \) is \( 2K \)-Lipschitz continuous. Using the triangle inequality, we have:

\[
||h_1(r) - h_2(r)|| \leq ||h_1(r) - h_1(r')|| + ||h_1(r') - h_2(r)||
\]

\[
\leq ||h_1(r) - h_1(r')|| + ||h_1(r') - h_2(r')|| + ||h_2(r) - h_2(r')||,
\]

thus,

\[
||h_1(r) - h_2(r)|| - ||h_2(r') - h_2(r')|| \leq ||h_1(r) - h_1(r')|| + ||h_1(r) - h_2(r')||.
\]

then, because both \( h_1 \) and \( h_2 \) are \( K \)-Lipschitz continuous,

\[
\frac{||h_1(r) - h_2(r)|| - ||h_1(r') - h_2(r')||}{\rho(r, r')} \leq \frac{||h_1(r) - h_1(r')|| + ||h_2(r) - h_2(r')||}{\rho(r, r')}
\]

\[
\leq 2K.
\]
Thus, $|h_1 - h_2|$ is $2K$-Lipschitz continuous. Then we have:

$$
\epsilon_{R_1}(h_1, h_2) - \epsilon_{R_2}(h_1, h_2) = \mathbb{E}_{r \in R_1} [||h_1(r) - h_2(r)||] - \mathbb{E}_{r \in R_2} [||h_1(r) - h_2(r)||]
$$

$$
\leq \sup_{\|f\| \leq 2K} \mathbb{E}_{r \sim R_1} [f(r)] - \mathbb{E}_{r \sim R_2} [f(r)]
$$

$$
= 2K \cdot \text{WD} (R_1, R_2).
$$

So far, theorem 1 is proven.

Next, we give the upper bound of generalization error by Wasserstein distance. Denoting the ideal classifier for a specific classification task by $f^*_c$, both $f_c$ and $f_c^*$ are $K$-Lipschitz continuous. In order to get better performance, $f_c$ need to approach the ideal classifier $f^*_c$. So the error of $f_c$ on distribution $R$ is defined as $\gamma(f_c)$:

$$
\gamma_{R_c}(f_c) = \epsilon_{R}(f_c, f_c^*).
$$

For two distribution of input instances, denoted by $\mathbb{D}_1$ and $\mathbb{D}_2$, their corresponding representation distributions are $R_1$ and $R_2$. According to Theorem 1, we have:

$$
\epsilon_{R_1}(f_c, f_c^*) \leq \epsilon_{R_2}(f_c, f_c^*) + 2K \cdot \text{WD}(R_1, R_2),
$$

thus,

$$
\gamma_{R_1}(f_c) \leq \gamma_{R_2}(f_c) + 2K \cdot \text{WD} (R_1, R_2).
$$

Correspondingly,

$$
\gamma_{\mathbb{D}_1}(f_c \circ f_c) \leq \gamma_{\mathbb{D}_2}(f_c \circ f_c) + 2K \cdot \text{WD}(\mathbb{D}_1, \mathbb{D}_2).
$$

Thus, for an unknown test distribution $\mathbb{D}_1$, minimizing error on $\mathbb{D}_1$ can be divided into two goals. Firstly, we minimize error on the given training sample distribution $\mathbb{D}_2$ (as goal 1). Secondly, we minimize the Wasserstein distance between $R_1$ and $R_2$ (as goal 2).

### B. Experimental implementation details

We provide implementation details of the comparison method used in our experiments.

#### Facial Expression Recognition

- **Transfer learning**: We adopt the pre-training and fine-tuning approach to transfer learning based on shared parameters (Yosinski et al. 2014). We first pre-train the feature extractor as well as the classifier on CelebA for hair color classification and then train the whole model on the expression dataset.

- **Multi-task learning**: We adopt the multi-task learning approach with hard parameter sharing of hidden layers (Caruana 1993). Specifically, a shared feature extractor accepts both images with hair color tags from CelebA and images with expression tags from a training set. And two independent classifier networks make separate predictions for hair color and expressions. The entire network gets trained by backpropagating the final loss which is calculated by adding up the classification loss of hair color and the classification loss of expression.

- **Adversarial Multi-task Learning**: Based on the multi-tasking learning model, a new gradient reversal layer (GRL) is added between the feature extractor and the hair color classifier, where the idea is consistent with Shinohara (Shinohara 2016b).

#### Digit Recognition

- **DANN**: DANN is a framework for unsupervised domain adaptation based on adversarial multitasking learning proposed by Ganin (Ganin et al. 2016). As a comparison method for the simultaneous use of MNIST and SVHN data, it is implemented by adding a new gradient reversal layer (GRL) and a domain classifier with 2-dimensional outputs for domain classification to the existing feature extractor and classifier. The feature extractor accepts both images from MNIST and SVHN. One classifier then classifies the feature representation of the MNIST data from 0 to 9, and another classifier distinguishes whether the feature representation originates from MNIST or SVHN.

### C. Datasets Details

#### Facial Expression Recognition

- **CK+** (we use 927 images): The Extended Cohn-Kanade (CK+) dataset has the facial behavior of 210 adults was recorded using two hardware synchronized Panasonic AG-7500 cameras. Participants were 18 to 50 years of age, 69% female, 81%, Euro-American, 13% Afro-American, and 6% other groups (Lucey et al. 2010). CK+ is available at [http://www.consortium.ri.cmu.edu/ckagree/](http://www.consortium.ri.cmu.edu/ckagree/)
• **Oulu** (we use 1440 images): The Oulu Multi-pose Eye Gaze Dataset includes 200 image sequences from 50 subjects (For each subject it includes four image sequences). Each sequence consists of 225 frames captured when people are fixating on 10 targeting points on the screen (Zhao et al. 2011). Oulu is available at [https://sites.google.com/site/oulunpudatabase/](https://sites.google.com/site/oulunpudatabase/).

• **MMI** (we use 639 images): The MMI Facial Expression Database, which includes more than 1500 samples of both static images and image sequences of faces in frontal and in profile view displaying various expressions of emotion, single and multiple facial muscle activation (Pantic et al. 2005). MMI is available at [https://mmifacedb.eu/](https://mmifacedb.eu/).

• **CelebA**: CelebFaces Attributes Dataset (CelebA) is a large-scale face attributes dataset with more than 200K celebrity images, each with contains 40 attribute annotations without detailed emotion labels. It is noteworthy that there is a ”smile” expression label in CelebA, but we do not use it in our experiment (Liu et al. 2015). CelebA is available at [http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html](http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html).

• **RAF**: Real-world Affective Faces Database (RAF) is a large-scale dataset of facial expressions containing approximately 30,000 diverse facial images downloaded from the Internet, where includes a 7-dimensional expression distribution vector for each image (Li, Deng, and Du 2017b). RAF is available at [http://whdeng.cn/RAF/model1.html](http://whdeng.cn/RAF/model1.html).

---

**Figure 3**: Examples of datasets used in the facial expression recognition experiments, which have been preprocessed through MTCNN for alignment. Notice that the characters have a more homogenous hair color (black in CK+ and brown in Oulu), while the hair color is richer in MMI. Hair is usually not visible in the task-relevant dataset RAF, whereas hair color is varied in CelebA.

---

**Digit Recognition**

• **MNIST**: The MNIST handwritten digit database has a training set of 60,000 samples and a test set of 10,000 samples (LeCun et al. 1998). The numbers have been size-standardized to a $28 \times 28$ image size and are centered on a fixed digital center. MNIST is available at [http://yann.lecun.com/exdb/mnist/](http://yann.lecun.com/exdb/mnist/).

• **SVHN**: The Street View House Numbers Dataset (SVHN) is a real-world image dataset (Netzer et al. 2011). It can be seen as similar in flavor to MNIST (e.g., the images are of small cropped digits), but incorporates an order of magnitude more
Figure 4: Examples of datasets used in the digital recognition experiments. Note that MNIST contains only the black background and white numbers, while SVHN and MNIST contain a variety of colored backgrounds and numbers.

labeled data (over 600,000 digit images) and comes from a significantly harder, unsolved, real-world problem (recognizing digits and numbers in natural scene images). SVHN is available at [http://ufldl.stanford.edu/housenumbers/](http://ufldl.stanford.edu/housenumbers/).

- **MNIST-M**: The acquisition of MNIST-M is the same as [Ganin et al. (2016)](https://arxiv.org/abs/1603.08155). Specifically, we blend digits from the MNIST over patches randomly extracted from color photos from BSDS500 [Arbelaez et al. (2010)](https://www.cs.cmu.edu/~arbelaez/Publications/2010paper.pdf).

## D. Additional Experimental Results

### Verify the correctness of the DANN implementation

The method (DANN) proposed in [Ajakan et al. (2014)](https://arxiv.org/abs/1406.1469) to achieve unsupervised domain adaptation based on the adversarial multi-task learning framework has achieved great success and has been widely used. We found in experiments (refer to experiments on digit recognition) that it is very difficult to implement domain adaptation from MNIST to SVHN, and the DANN framework will even weaken the original performance of the model. For this reason, we add an experiment that realizes domain adaptation from SVHN to MNIST dataset to verify the correctness of our DANN’s implementation. The experimental results are shown in Table 1.

| Train → Test Dataset | Feature Extractor | Goal 1 Only | DANN (Background Color) | TIRTL (Background Color) |
|----------------------|-------------------|-------------|-------------------------|--------------------------|
| SVHN → MNIST         | DenseNet121       | 67.36       | **83.36**                | 63.05                    |
|                      | EfficientNet      | 50.63       | 55.23                   | 64.38                    |
|                      | MobileNetV2       | 62.50       | 64.20                   | **66.99**                |
|                      | ResNet18          | 59.80       | 60.36                   | **62.73**                |
|                      | SeNet             | 50.72       | **77.34**               | 57.04                    |
|                      | ShuffleNetV2      | 47.52       | 48.94                   | 49.70                    |
|                      | VggNet11          | **62.96**   | 54.14                   | **61.07**                |
|                      | Average           | 57.36       | **65.89**               | 60.60                    |
| SVHN → MNIST-M       | DenseNet121       | 39.80       | **44.38**               | 40.06                    |
|                      | EfficientNet      | 35.50       | 35.80                   | **46.62**                |
|                      | MobileNetV2       | 40.86       | 40.82                   | **42.08**                |
|                      | ResNet18          | 35.52       | 35.91                   | **40.94**                |
|                      | SeNet             | 31.14       | 37.34                   | **40.48**                |
|                      | ShuffleNetV2      | 33.70       | **38.81**               | 33.42                    |
|                      | VggNet11          | **43.13**   | 41.20                   | 42.14                    |
|                      | Average           | 37.09       | **39.18**               | **40.82**                |

Table 5: Additional results (Accuracy %) on digit recognition task.

In the additional experiment, the model using DenseNet121 as the feature extractor achieved an accuracy of 83.36 on the MNIST test dataset under the DANN framework, which is higher than the result (73.85) reported in the original paper [Ganin and Lempitsky (2015)](https://arxiv.org/abs/1503.03175). Note that we only use 20,000 pictures from SVHN as a training dataset. The experiment also shows that
the DANN framework would greatly improve some models (DenseNet121, SeNet, and ShuffleNetV2), but the improvement effect is not obvious for other models. In contrast, TIRTL is significantly more general. Also noticed that although the average performance of TIRTL under the MNIST test dataset is not as good as that of DANN, TIRTL has achieved better performance on the MNIST-M test dataset. This is because DANN as a domain adaptation method focuses on improving the model in a known target domain, while TIRTL focuses on improving the generalization performance of the model under the unseen test samples. Also, it is worth noting that the background and foreground colors of the pictures in SVHN are more evenly distributed than MNIST, which is the main reason why the improvement effect of TIRTL on SVHN is not obvious.

Other PCA analysis results
Due to the paper’s length limitation, we only use the results of ResNet18 as an example, and the other results are shown in Figure 5. It can be guided from Figure 5 that the results on other models are the same as our results using ResNet as an example. The proportion of task-relevant information is much higher in TIRTL on all models, which indicates that the feature representation vectors are easier to aggregate in TIRTL.

![PCA results of other feature extractors on the digit recognition task.](image-url)