Latent space decomposition into task-specific and domain-specific subspaces for domain adaptation

Takaya Ueda
Ritsumeikan University
Shiga, Japan
0000-0001-5715-1609

Ikuko Nishikawa
Ritsumeikan University
Shiga, Japan
0000-0003-4780-0155

***Abstract*—In this study, we propose a novel method of acquiring the latent space (composed of task-specific and domain-specific subspaces) of data. The layered neural networks acquire the effective features for a given task in its internal layers as a latent space. If the acquired latent space is relevant only to the task and invariant to the data domain, it can be adapted to a new domain. Therefore, our goal is to acquire the latent space in which all the features relevant to the task are contained in a task-specific subspace, whereas all the features relevant to the data domain are contained in a domain-specific subspace. The method iterates two simple training steps. First, a classifier acquires a task-specific space in its final layer through a conventional supervised learning. Whereas a domain-specific space is obtained as its complementary subspace in the complete latent space. The latter is realized by an autoencoder training using both subspaces. In the second step, two input data are used, and two task-specific representations and two domain-specific representations are exchanged to generate two composite data. Further training of the classifier using the composite data forces each subspace to be more specific to either the task or the domain. Computer experiments are conducted on three famous datasets of handwritten digits as three different domains. The visualization of the obtained task-specific space shows the clusters corresponding to the digit classes, whereas the domain-specific space shows the areas corresponding to the datasets. The domain adaptation to an unlearned dataset using the task-specific representation obtained by our method demonstrates a higher performance than an alternative method.**

***Index Terms*—latent space decomposition, composite data, domain invariance, Wasserstein GAN, domain adaptation**

I. Introduction

Supervised training of neural networks generally requires a large amount of data to attain a high performance in a recognition task. However, it is often difficult and costly to obtain an adequate amount of the labeled data in real world problems. One of the solutions is developed as an unsupervised domain adaptation. The basic idea is to find a similar but different data domain for which the label is easier to obtain for the same task. If it is found, a neural network is trained by the labeled data in the found domain. Then, the features obtained in the internal layers are expected to be effective also for the original problem with the same task. For this adaptation to work well, the acquired features should be relevant only to the task and be invariant to the data domain. Therefore, the extraction of domain-invariant features is indispensable for the domain adaptation. Not limited to the domain adaptation, the domain-invariant representation is effective to cope with the variations of the data distribution referred to as the domain shift[1].

The current study proposes a novel method to obtain the domain-invariant representation. II introduces the recent related works. The proposed algorithm is described in III, and it is applied to the benchmark datasets in IV to evaluate its effectiveness.

II. Related studies on task-specific domain-invariant representation

The acquisition of a domain-invariant representation of data is indispensable for unsupervised domain adaptation. A task-specific representation without detrimental variations in the data is important in machine learning. Some recent related studies are briefly introduced in the context of the domain adaptation in II-A, and the representation learning in II-B.

Let us define some important terms and abbreviations to simplify the following descriptions. Data $\mathbf{x}$ is mapped by the encoder E to its internal representation $\mathbf{z}$. This is passed to classifier C to output the class label y. This indicates that $\mathbf{z} = E(\mathbf{x})$ and $y = C(\mathbf{z})$. The decoder D is an inverse mapping of E which referred to as $\mathbf{x} = D(\mathbf{z})$. The objective of this study is to obtain an effective $\mathbf{z}$ for classification using C. All of E, C, and D are given by deep neural networks. These networks are composed of convolution, fully connected or deconvolution layers in the following studies.

A. Domain-invariant representation for domain adaptation

Domain adaptation is to utilize a classifier composed of E and C through $\mathbf{z}$, which is trained by data $\mathbf{x}$ in one domain, in a similar domain. The former is denoted as a source domain whereas the latter is denoted as a target domain. It is useful when it is difficult to obtain an adequate amount of labeled data for the training in the target domain.

One of the main unsupervised approaches is to obtain E and $\mathbf{z}$, which are commonly useful in both domains[2], [3], [4]. M. Ghifary et al.[3] obtained a shared representation $\mathbf{z}$ by the trainings of the classification in a source domain and of the reconstruction in a target domain. Moreover, because the data distributions in both domains are expected to be similar, the similarity between both distributions in $\mathbf{z}$ is used as the measure to be minimized by the unsupervised training.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

E. Tzeng et al.[5] used maximum mean discrepancy (MMD) proposed by A. Gretton et al.[6] to measure the distance between the distributions in the final layer of E. In addition, M. Long et al.[7] extended the definition of MMD to multiple kernels, and applied it to multiple layers. Y. Ganin et al.[8] introduced an adversarial training between E and a domain classifier in order to exclude any domain-relevant features from $\mathbf{z}$. Moreover, the adversarial discriminative domain adaptation (ADDA) proposed by E. Tzeng et al.[9] introduced a domain encoder instead of sharing one encoder by both domains to be trained in an adversarial way[10] against a discriminator that measures the distance between source and target distributions in the common $\mathbf{z}$.

The above methods assume a source as a single domain whereas our method does not, and the effectiveness for a source with multiple domains is demonstrated in IV.

Another idea is to directly transform the data $\mathbf{x}$ between the two domains. K. Bousmalis et al.[11] proposed the generative adversarial networks (GAN)-based method to generate a target data from a source data with the same label y. Moreover, J. Hoffman et al.[12] used the CycleGAN-based method to obtain a transformation in both directions between a source and a target, and used the label y of the source data to the corresponding target data.

B. Task-specific representation without detrimental variations

There are many attempts to learn the representation without detrimental variations in the data. Data variations such as a noise or a lighting condition in a photo image are both irrelevant and detrimental or nuisance factors for the classification. The following have been proposed to discard such factors $\mathbf{s}$ contained in the $\mathbf{x}$, not to be mapped to $\mathbf{z}$. Here, $\mathbf{s}$ is assumed to be known in advance. The variational auto-encoder[13]-based variational fair auto-encoder by C. Louizos et al.[14] forces the distribution in $\mathbf{z}$ to be independent of $\mathbf{s}$ based on the MMD. In an adversarial approach proposed by Q. Xie et al.[15], E learns to exclude the information on $\mathbf{s}$ through an adversarial training against a discriminator that conversely infers $\mathbf{s}$ from $\mathbf{z}$. Whereas A. Jaiswal et al.[16] proposed an unsupervised adversarial training to obtain the representation that is divided into invariant factors and nuisance factors under the constraint of mutual independence.

Our proposed method is a straightforward training of E, C, and D to obtain a task-specific representation. The key idea is to generate composite data from the roughly decomposed representations to use them for the further fine decomposition into two subspaces.

III. Proposed method of latent space decomposition

A. Basic idea of the decomposition through training by composite data

Let us consider a task to classify data $\mathbf{x}$ to class y. Then, a classifier needs to extract effective features from $\mathbf{x}$. Here, we are making reference to the lower dimensional representation $\mathbf{z}$ of data $\mathbf{x}$ from which $\mathbf{x}$ can be reconstructed as a complete latent representation $\mathbf{z}$ of $\mathbf{x}$. Both the encoder E and the decoder D are defined in the same way as in II. Given the classification task, the goal of the proposed algorithm is to decompose $\mathbf{z}$ into two subspaces: task-specific $\mathbf{z}_T$ and domain-specific $\mathbf{z}_D$ subspaces in which all the features necessary for the classification are included in $\mathbf{z}_T$, and not in $\mathbf{z}_D$. The decomposition indicates $\mathbf{z} = \mathbf{z}_T \times \mathbf{z}_D$, where $\mathbf{z}_D$ is the complement of $\mathbf{z}_T$, and does not contain any feature relevant to the task. However, any detrimental factors should be contained in $\mathbf{z}_D$.

The proposed training for the decomposition iterates the following two steps (Fig. 1). The first step of the decomposition is described in III-B. Task-specific $\mathbf{z}_T$ and its encoder E_T are acquired through a supervised training of the classification, and the domain-specific $\mathbf{z}_D$ and its encoder E_D are acquired through a training of the reconstruction of $\mathbf{x}$ from $\mathbf{z}_T \times \mathbf{z}_D$. III-C describes the second step to enforce the independence between $\mathbf{z}_T$ and $\mathbf{z}_D$. This indicates that all the task-related features should be contained in $\mathbf{z}_T$ and not in $\mathbf{z}_D$. The additional training of the classifier uses the composite data that are generated from $\mathbf{z}_T$ and $\mathbf{z}_D$ obtained in the first step. The iteration of the above two steps leads to the further independence of the two subspaces. Finally, the obtained task-specific $\mathbf{z}_T$ and its encoder E_T are expected to possess a generalization ability for the task, and be effective for the adaptation to a new domain. However, $\mathbf{z}_D$ is supposed to contain factors that are irrelevant or detrimental to the task and related to the domain.

III-D describes the unsupervised domain adaptation based on the obtained $\mathbf{z}_T$. E_T is used as a source encoder, and a target encoder for any new domain is acquired by the adversarial training on the common space $\mathbf{z}_T$.

B. Rough decomposition by the training of the classification and the reconstruction

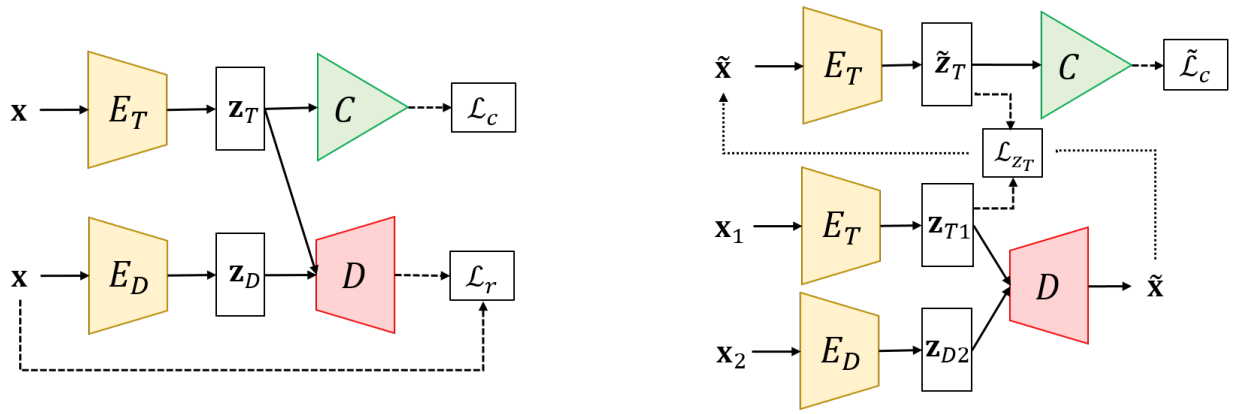
First step of the training for rough decomposition is shown in Fig. 1(a). The data $\mathbf{x}$ is input to both encoders E_T and E_D to be mapped to $\mathbf{z}_T$ and $\mathbf{z}_D$ respectively. The classifier C gets $\mathbf{z}_T$ to output y whereas the decoder D gets both $\mathbf{z}_T$ and $\mathbf{z}_D$ to reconstruct the original $\mathbf{x}$.

The loss functions $\mathcal{L}_c$ and $\mathcal{L}_r$ (for the classification and reconstruction respectively) to be minimized for the training of E_T, E_D, C, and D are given by

$$\mathcal{L}_c = -\mathbb{E}_{(\mathbf{x},y)\sim P(\mathbf{x},y)}[\sum_{k=1}^{K} \mathbf{1}_{[k=y]} \log C(E_T(\mathbf{x}))\ ]\ , \quad (1)$$

$$\mathcal{L}_r = \mathbb{E}_{\mathbf{x}\sim P(\mathbf{x},y)}[|\mathbf{x} - D(E_T(\mathbf{x}), E_D(\mathbf{x}))|^2]\ , \quad (2)$$

where $P(\mathbf{x}, y)$ is the data distribution on which the expectation $\mathbb{E}$ is defined, and $\mathbf{1}_{[k=y]}$ is indicated to take the output value of C for only the correct class y. K is the number of classes. The classification loss $\mathcal{L}_c$ is given by the cross entropy, and the reconstruction loss $\mathcal{L}_r$ is given by the L2 norm. Therefore, the total loss for the training is given by the weighted summation of $\alpha\mathcal{L}_r + \beta\mathcal{L}_c$ with the hyper parameters α and β.



(a) First step (b) Second step

Fig. 1. Two steps of the training in the proposed method (a) First step to acquire the task- and domain-specific spaces $\mathbf{z}_T$ and $\mathbf{z}_D$ respectively by the supervised trainings of the classification by E_T and C and of the reconstruction by E_T, E_D, and D (b) Second step for higher separability by the additional training of the classifier using the composite data $\tilde{\mathbf{x}}$ generated from $\mathbf{x}_1$ and $\mathbf{x}_2$

C. Fine decomposition by the additional training of the classification using artificial composite data

The second step enforces the separation of task-relevant and task-irrelevant features by further training the classifier C and the task-specific encoder E_T. The key idea is to use artificial composite data for the training. The composite data is generated by the decoding from $\mathbf{z}_T \times \mathbf{z}_D$, where $\mathbf{z}_T$ and $\mathbf{z}_D$ are encoded from two different inputs: $\mathbf{x}_1$ and $\mathbf{x}_2$ respectively. If the composite data successfully inherits the task-specific features from $\mathbf{x}_1$ and the domain-specific features from $\mathbf{x}_2$, then it is expected to possess the same $\mathbf{z}_T$ with $\mathbf{x}_1$ and the same $\mathbf{z}_D$ with $\mathbf{x}_2$.

Fig. 1(b) shows the training by the composite data in the second step. Two different data $\mathbf{x}_1, \mathbf{x}_2 \sim P(\mathbf{x}, y)$ are input to encoders E_T and E_D respectively, and mapped to $\mathbf{z}_{T1} \equiv E_T(\mathbf{x}_1)$ and $\mathbf{z}_{D2} \equiv E_D(\mathbf{x}_2)$. Then, the composite data $\tilde{\mathbf{x}}$ is generated by the decoder D as $D(\mathbf{z}_{T1}, \mathbf{z}_{D2})$. The composite data $\tilde{\mathbf{x}}$ is again input to the task-specific encoder E_T, and mapped to $\tilde{\mathbf{z}}_T \equiv E_T(\tilde{\mathbf{x}})$. In order to enforce that $\tilde{\mathbf{x}}$ inherits the task-specific features only from $\mathbf{x}_1$ through $\mathbf{z}_{T1}$ and not from $\mathbf{x}_2$ through $\mathbf{z}_{D2}$, the classifier C is trained to output the same class with $\mathbf{x}_1$, and encoder E_T is trained to map to the same point $\mathbf{z}_{T1}$.

Therefore, the loss functions $\tilde{\mathcal{L}}_c$ and $\mathcal{L}_{z_T}$ to be minimized for the additional training of C and E_T are given by

$$\tilde{\mathcal{L}}_c = -\mathbb{E}_{(\tilde{\mathbf{x}}, y_1)}[\sum_{k=1}^{K} \mathbf{1}_{[k=y_1]} \log C(E_T(\tilde{\mathbf{x}}))\ ]\ , \tag{3}$$

$$\mathcal{L}_{z_T} = \mathbb{E}_{\tilde{\mathbf{x}}}[\ |\mathbf{z}_{T1} - E_T(\tilde{\mathbf{x}})|^2\ ]\ , \tag{4}$$

where y_1 is the correct class of $\mathbf{x}_1$. The total loss is given by the weighted summation of $\gamma\tilde{\mathcal{L}}_c + \delta\mathcal{L}_{z_T}$ with the hyper parameters γ and δ. The training of C and E_T by the back-propagation for $\tilde{\mathbf{x}}$ should not propagate to the D and E_T which generate $\tilde{\mathbf{x}}$.

The two steps of the training are iterated. To ensure the reconstruction accuracy in the first step, a three-times longer training is taken for the first step in the computer experiments in IV.

D. Domain adaptation using task-specific space $\mathbf{z}_T$

The main purpose of the above decomposition is to obtain the task-specific space $\mathbf{z}_T$ for the classification task independent of the domain. The obtained $\mathbf{z}_T$ is used for the adaptation to a different domain.

Several approaches of the domain adaptation are proposed to train the encoder E_T in the adversarial way as described in II. Among these approaches, adversarial discriminative domain adaptation (ADDA) by E. Tzeng et al.[9] is explained in detail with the loss function being used in the training. This is adopted in our model. This indicates that the encoder E_T obtained in III-B and III-C is used as a source encoder, and it is trained to acquire the target encoder $\mathcal{E}_T$ using the adversarial training against the discriminator $\mathcal{D}$.

The target encoder $\mathcal{E}_T$ is trained to encode the data $\boldsymbol{x}$ in a new domain (i.e., a target domain) to the same task-specific space $\mathbf{z}_T$. The initial state of $\mathcal{E}_T$ is made the same as source encoder E_T. Therefore, the distribution on $\mathbf{z}_T$ mapped by $\mathcal{E}_T$ according to the target data distribution ($\boldsymbol{x} \sim P_t(\boldsymbol{x})$) differs from that of the source data $\mathbf{x}$. The discriminator $\mathcal{D}$ measures this difference in the two distributions to distinguish the two domains whereas the target encoder $\mathcal{E}_T$ is trained to decrease the difference.

The loss function of this adversarial training for the target encoder $\mathcal{E}_T$ to be minimized and for discriminator $\mathcal{D}$ to be maximized is given by

$$\mathcal{L}_{DA} = \mathbb{E}_{\mathbf{x}\sim P(\mathbf{x})}[\mathcal{D}(E_T(\mathbf{x}))] - \mathbb{E}_{\boldsymbol{x}\sim P_t(\boldsymbol{x})}[\mathcal{D}(\mathcal{E}_T(\boldsymbol{x}))] - \lambda(|\nabla_{\hat{\boldsymbol{z}}}\mathcal{D}(\hat{\boldsymbol{z}})|_2 - 1)^2\ , \tag{5}$$

where $P(\mathbf{x})$ and $P_t(\boldsymbol{x})$ are the distributions of the source data $\mathbf{x}$ and the target data $\boldsymbol{x}$ respectively. $\hat{\boldsymbol{z}}$ is an internal point in $\mathbf{z}_T$ defined by $\hat{\boldsymbol{z}} \equiv \varepsilon E_T(\mathbf{x}) + (1-\varepsilon)\mathcal{E}_T(\boldsymbol{x}),\ \ \varepsilon \in U[0,1]$.

In the following adversarial training, the Wasserstein distance is used for the loss according to Wasserstein GAN[17]. Moreover, the improved method WGAN-GP[18] is applied in the experiments in IV. In addition, the hyper parameter is set to $\lambda = 10$, and the training for discriminator $\mathcal{D}$ takes five times longer than that for $\mathcal{E}_T$.

IV. Computer experiments

The proposed method is applied to a well-known classification problem of hand-written digits image data, where a task is a classification into ten classes. Three different datasets are used as the three domains.

First, the method is applied to the mixture of two datasets. The acquired two subspaces $\mathbf{z}_T$ and $\mathbf{z}_D$ are analyzed by the generated composite data, and the visualizations of the data distributions in each subspace using t-SNE[19] are conducted in IV-C. Thereafter, the obtained $\mathbf{z}_T$ is used for an adaptation to the third dataset by the ADDA in IV-D, and the classification accuracy is compared to an alternative method in IV-E.

A. Three datasets of hand-written digits

The hand-written digit image data are obtained from the following three datasets: MNIST[20], USPS, as well as SVHN[21], and they are used for the classification into ten digits of zero to nine. Seven hundred data for each digit (i.e., 7,000 data in total) are obtained from each dataset for the training. Because the image data are colored in the SVHN whereas they are in a gray scale in the MNIST and the USPS, three identical images are fed into the RGB channels for the latter datasets. The image size is normalized to 32×32 pixels.

B. Settings of the neural networks

The neural network structure of each encoder E_T, E_D, and $\mathcal{E}_T$ is based on LeNet[20], and the number of output units (i.e., the dimension of each subspace) is set to 500. The structures of both the classifier C and the discriminator $\mathcal{D}$ are fully connected three-layered networks with the number of units at each layer being $500 - 500 - 10$, and the activation function is rectified linear (Relu). The network of decoder D has one fully connected layer and two deconvolution layers.

The training of all the above networks is optimized by Adam[22], and the training rate is 0.001. The batch size is 128, and all the training data are input once per each epoch in a randomized order. The results obtained after 200 epochs of the training are shown below. The values of the hyper parameters in the proposed method are $\alpha = 0.1, \beta = 100, \gamma = 1$ and $\delta = 10$.

C. Decomposition into two subspaces

The training data are sampled from both the USPS and the SVHN in the experiment.

First, the reconstructed data after the training are shown in Fig. 2. Figs. 2(a) and (b) show the original test input data and their reconstruction from the decoder D for USPS and SVHN respectively. Each data is well reconstructed in both color and shape. Whereas Fig.2(c) shows the composite data that are decoded from the task-specific representation of the USPS data and the domain-specific representation of the SVHN data in their corresponding positions in (a) and (b). The obtained images indicate that the subspaces $\mathbf{z}_T$ and $\mathbf{z}_D$ are highly specific to task and domain respectively.

Thereafter, the data distribution in each 500-dimensional subspace is visualized in a two-dimensional plane using t-SNE[19]. The test data for the visualization are sampled from the USPS and the SVHN data, 100 data for each of ten classes from each dataset.

Fig. 3 shows the distribution in the domain-specific subspace $\mathbf{z}_D$. The two clusters that correspond to the two datasets of the USPS and the SVHN are found in Fig. 3(a). However, no structure corresponds to the class found in (b). Therefore, the domain-specific subspace $\mathbf{z}_D$ possesses features related to the dataset (i.e., the domain); however, few of the features are relevant to the classification.

The distribution in the task-specific subspace $\mathbf{z}_T$ is visualized in IV-D with the third domain data before and after the domain adaptation.

D. Adaptation to a new domain

Thereafter, the obtained $\mathbf{z}_T$ is used for an adaptation to the third dataset, the MNIST by ADDA.

The visualization of the data distribution in $\mathbf{z}_T$ using the t-SNE is shown in Fig. 4. One thousand test data of the MNIST are added to those of the USPS and the SVHN. Fig. 4 shows the distribution of 1,000 data from three datasets in their common $\mathbf{z}_T$. The upper row (a)-(c) visualizes the $\mathbf{z}_T$ that is obtained by an ordinary classifier training, while the lower row (d)-(f) shows the $\mathbf{z}_T$ that is obtained by the proposed method. The left and middle columns ((a), (b) and (d), (e)) are the distributions mapped by the source encoder E_T before the adaptation, while the target data distribution in the right column ((c) and (f)) is mapped by the target encoder $\mathcal{E}_T$ after the adaptation. The color of the data indicates the dataset in the left and right columns ((a), (c) and (d), (f)), while the color indicates ten different digits in the middle column ((b) and (e)).

First, the comparison between the upper and lower figures shows that the clusters are formed for each class of digit (e) but not for each dataset (d), especially in the proposed method. The difference is clearly seen for SVHN, comparing (a) and (d). Moreover, within the cluster of each digit class, three regions are formed, corresponding to three datasets in (b) and (e), especially for digits '1', '2' and '8'.

Second, the mappings by the target encoder $\mathcal{E}_T$ after the adaptation (in both (c) and (f)) show that each class of target dataset of the MNIST becomes overlapped with the corresponding cluster of the source datasets because of the adversarial training in ADDA.

E. Comparison of classification accuracy after adaptation

Evaluation of the domain adaptation is given by the classification accuracy in the target domain. In addition to the adaptation from the source USPS and SVHN to the target

(a) $\mathbf{x}_1$ and $D(E_T(\mathbf{x}_1), E_D(\mathbf{x}_1))$ for $\mathbf{x}_1 \in$ USPS (b) $\mathbf{x}_2$ and $D(E_T(\mathbf{x}_2), E_D(\mathbf{x}_2))$ for $\mathbf{x}_2 \in$ SVHN (c) $D(E_T(\mathbf{x}_1), E_D(\mathbf{x}_2))$

Fig. 2. Output images from decoder D after the training. (a) Input (left) and its reconstruction (right) for the USPS data. Each row of 5×10 images corresponds to class $y = 0, \ldots, 9$. (b) Input (left) and its reconstruction (right) for the SVHN data (c) Composite data from the USPS in (a) for $\mathbf{z}_T = E_T(\mathbf{x}_1)$ and the SVHN in (b) for $\mathbf{z}_D = E_D(\mathbf{x}_2)$

(a) Two clusters corresponding to two datasets (b) No cluster for each class

Fig. 3. Visualization of the data distribution on $\mathbf{z}_D$ using t-SNE. (a) Colored by each dataset. The USPS data are shown in blue whereas the SVHN data are shown in orange. (b) Colored by each class. Ten classes are indicated by ten colors.

MNIST in IV-D, the other two ways of adaptation: USPS, MNIST $\rightarrow$ SVHN, and MNIST, SVHN $\rightarrow$ USPS are studied. The UAI by Jaiswal et al.[16] is compared to the proposed method as an alternative approach to obtain the domain invariant representation $\mathbf{z}_T$.

Fig. 5 shows the classification accuracy in the target domain during the 200 epochs of the training by ADDA from the source USPS and SVHN to the target MNIST. Three methods (a classifier training, UAI, and our method) to obtain $\mathbf{z}_T$ are compared. The accuracy increases during ADDA in our method whereas it decreases in the other two methods. Ten trials of ADDA experiments show the same tendency. The obtained results show that the domain invariant representation $\mathbf{z}_T$ is an essential requirement for domain adaptation based on the distance between the data distributions, and our method provides a more effective representation for the task, particularly when the source domain is not single.

Table I shows the classification accuracy obtained for three ways of adaptation, comparing a simple classifier training, the UAI, our method with only the first step, and our method with the iterative two steps. The accuracies before and after the adaptation are indicated by the arrow. A simple average of the maximum and the minimum accuracies among the ten trials is shown with the deviation from the average. The improvement by the adaptation is significantly seen in our proposed method. Though our method fails when it is without the second step of the training in the case of MNIST, SVHN $\rightarrow$ USPS, it (our method) never fails in any trails with the iteration of two steps, and never shows a drop in the accuracy during the ADDA training that is found in the other methods in Fig. 5.

V. Conclusion

A novel method has been proposed to obtain the domain invariant representation by layered neural networks. The latent representation is decomposed into domain-invariant and domain-specific subspaces, and the composite data generated from the two subspaces are used for further fine decomposition. Therefore, the domain-invariant representation is extracted even if the training data are sampled from multiple domains. The computer experiments demonstrate the effectiveness by the classification accuracy after the domain adaptation. The subsequent steps include the extension of the proposed method to multiple target domains and its application to the real world problems.

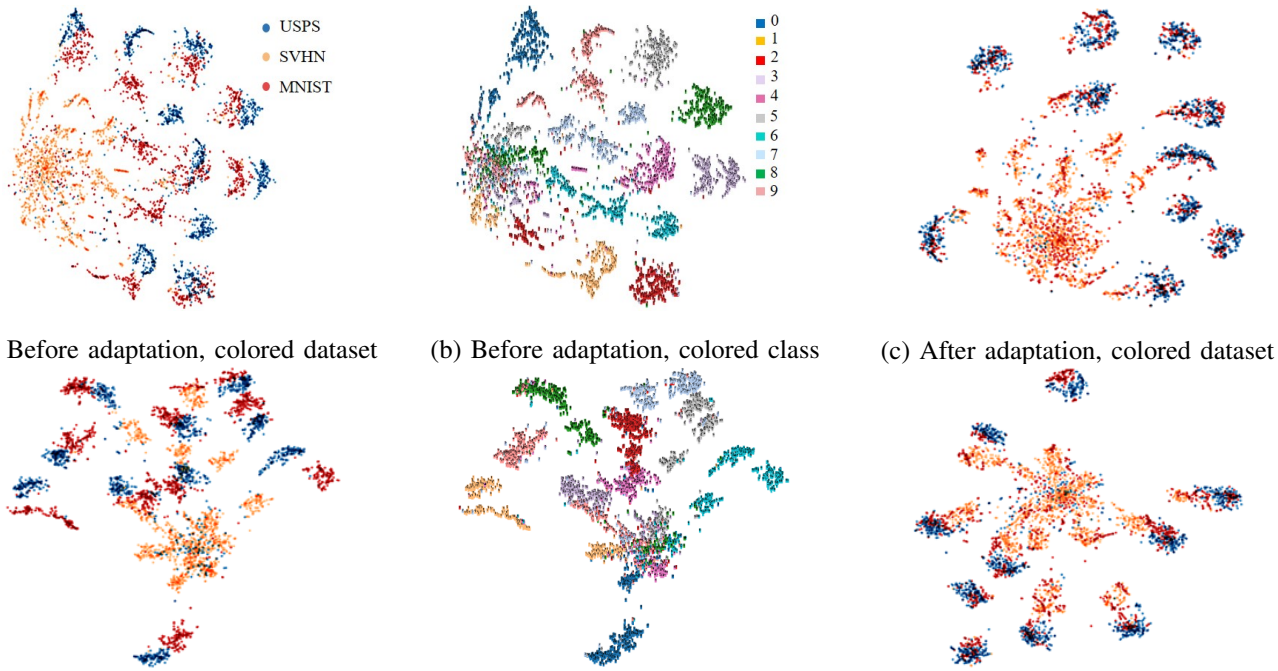
Acknowledgment

This work was supported in part by JSPS KAKENHI Grant No. 19K12164.

References

[1] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. M. Borgwardt, and B. Schölkopf, "Covariate shift and local learning by distribution matching," MIT Press, pp. 131–160, 2009.

[2] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," Advances in Neural Information Processing Systems, pp. 343–351, 2016.



(a) Before adaptation, colored dataset (b) Before adaptation, colored class (c) After adaptation, colored dataset

(d) Before adaptation, colored dataset (Our method) (e) Before adaptation, colored class (Our method) (f) After adaptation, colored dataset (Our method)

Fig. 4. Visualization of the data distribution on $\mathbf{z}_T$ using t-SNE. Upper: $\mathbf{z}_T$ obtained by an ordinary classifier training. Lower: $\mathbf{z}_T$ obtained by the proposed method. Left and middle: Before the domain adaptation, mapped by E_T. Right: After the domain adaptation, mapped by $\mathcal{E}_T$. Left and right: Each colored dataset: USPS in blue, SVHN in orange and MNIST in red. Middle: Each colored by each class. Ten classes are indicated in ten colors.

TABLE I

CLASSIFICATION ACCURACY IN EACH TARGET DOMAIN (%). THE ARROW INDICATES THE CHANGE IN ACCURACY BEFORE AND AFTER THE ADAPTATION.

Method	USPS,SVHN → MNIST	USPS,MNIST → SVHN	MNIST,SVHN → USPS
ADDA only	76.2 ± 1.7 → 79.3 ± 7.8	17.2 ± 4.3 → 21.2 ± 2.5	81.3 ± 1.5 → 68.3 ± 3.1
UAI → ADDA	77.1 ± 0.4 → 75.4 ± 7.3	13.5 ± 6.6 → 13.5 ± 6.7	77.2 ± 2.3 → 62.2 ± 5.3
Ours (Only 1st step) → ADDA	77.8 ± 1.6 → 85.9 ± 2.2	15.0 ± 3.0 → 30.0 ± 6.0	80.4 ± 3.7 → 68.9 ± 5.0
Ours (Iterative 2 steps) → ADDA	79.1 ± 3.1 → 89.3 ± 1.2	19.3 ± 1.8 → 26.5 ± 2.8	78.3 ± 1.8 → 78.4 ± 2.7

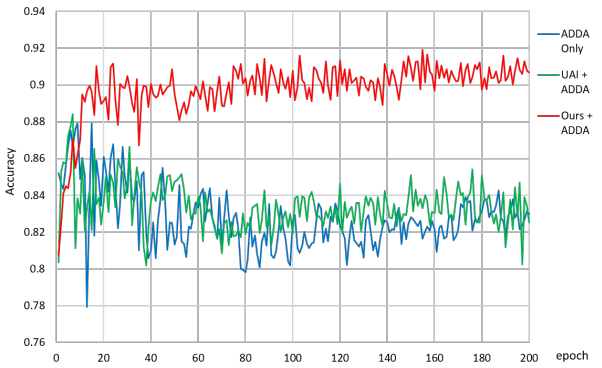


Fig. 5. Classification accuracy in the target domain during ADDA

[3] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, "Deep reconstruction-classification networks for unsupervised domain adaptation," 14th European Conference on Computer Vision, vol. 9908, pp. 597–613, 2016.

[4] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and S. Saminger-Platz, "Central moment discrepancy (cmd) for domain-invariant representation learning," International Conference on Learning Representations, 2017.

[5] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," Preprint arXiv:1412.3474, 2014.

[6] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A Kernel Two-Sample Test," The Journal of Machine Learning Research, vol. 13, pp. 723–773, 2012.

[7] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," Proceedings of the 32nd International Conference on International Conference on Machine Learning, vol. 37, pp. 97–105, 2015.

[8] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," The Journal of Machine Learning Research, vol. 17, pp. 2096–2030, 2016.

[9] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial Discriminative Domain Adaptation," 2017 IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[10] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets,"

Advances in Neural Information Processing Systems, pp. 2672–2680, 2014.
[11] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," IEEE Conference on Computer Vision and Pattern Recognition, pp. 95–104, 2017.
[12] J. Hoffman, E. Tzeng, T. Park, J. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, "CyCADA: Cycle-Consistent Adversarial Domain Adaptation," Proceedings of the 35th International Conference on Machine Learning, vol. 80, pp. 1989–1998, 2018.
[13] D. P. Kingma, and M. Welling, "Auto-Encoding Variational Bayes," International Conference on Learning Representations, 2014.
[14] C. Louizos, K. Swersky, Y. Li, M. Welling, and R. Zeme, A. Courville, "The variational fair autoencoder," 14th European Conference on Computer Vision, 2016.
[15] Q. Xie, Z. Dai, Y. Du, E. Hovy, and G. Neubig, "Controllable invariance through adversarial feature learning," , Advances in Neural Information Processing Systems 30, pp. 585–596, 2017.
[16] A. Jaiswal, Y. Wu, W. AbdAlmageed, and P. Natarajan, "Unsupervised Adversarial Invariance," Advances in Neural Information Processing Systems, 2018.
[17] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein Generative Adversarial Networks," Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 214–223, 2017.
[18] I. Gulrajani, F. Ahmed, M. Arjovsky, and V. Dumoulin, A. Courville, "Improved Training of Wasserstein GANs," Advances in Neural Information Processing Systems 30, 2017.
[19] L. Maaten, and G. Hinton, "Visualizing data using t-sne," Journal of machine learning research, vol. 9, pp. 2579–2605, 2008.
[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, pp. 2278–2323, 1998.
[21] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading Digits in Natural Images with Unsupervised Feature Learning," NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2011.
[22] D. Kingma, and J. Ba, "Adam: A method for stochastic optimization," International Conference on Learning Representations, 2015.