COSINE SIMILARITY-BASED ADVERSARIAL PROCESS

Anonymous authors
Paper under double-blind review

ABSTRACT

An adversarial process between two deep neural networks is a promising approach to train robust networks. In this study, we propose a framework for training networks that eliminates subsidiary information via the adversarial process. The objective of the proposed framework is to train a primary model that is robust to existing subsidiary information. This primary model can be used for various recognition tasks, such as digit recognition and speaker identification. Subsidiary information refers to the factors that might decrease the performance of the primary model such as channel information in speaker recognition and noise information in digit recognition. Our proposed framework comprises two discriminative models for the primary and subsidiary task, as well as an encoder network for feature representation. A subsidiary task is an operation associated with subsidiary information such as identifying the noise type. The discriminative model for the subsidiary task is trained for modeling the dependency of subsidiary class labels on codes from the encoder. Therefore, we expect that subsidiary information could be eliminated by training the encoder to reduce the dependency between the class labels and codes. In order to do so, we train the weight parameters of the subsidiary model; then, we develop the codes and the parameters of subsidiary model to make them orthogonal. For this purpose, we design a loss function to train the encoder based on cosine similarity between the weight parameters of the subsidiary model and codes. Finally, the proposed framework involves repeatedly performing the adversarial process of modeling the subsidiary information and eliminating it. Furthermore, we discuss possible applications of the proposed framework: reducing channel information for speaker identification and domain information for unsupervised domain adaptation.

1 INTRODUCTION

A generative adversarial network (GAN) is a framework for estimating generative models using adversarial processes (Goodfellow et al. (2014)). In this framework, a generative model G and discriminative model D are trained simultaneously. In particular, G estimates the data distribution of the target data to generate samples that are similar to the training data, whereas D determines whether the input sample is one that is generated by G or was part of the actual training data. Thus, the procedure of training G and D is an adversarial process in a manner similar to a minimax game. The GAN framework has been successful in efficiently training generative models; it has been widely used in many research fields (Goodfellow et al. (2016), Isola et al. (2017), Ledig et al. (2016)).

In this study, we propose a framework to eliminate subsidiary information in input data using cosine similarity-based adversarial process. The information to be eliminated typically degrades performance of the discriminative model for the primary task of classification. For example, channel or noise information should be reduced when the primary task is speaker recognition or image recognition, respectively. Thus, our proposed framework differs from existing GANs in that it trains the discriminative model by using the cosine similarity-based adversarial process.

2 COSINE SIMILARITY-BASED ADVERSARIAL NETWORK

Fig. 1 shows the overall architecture of the proposed cosine similarity-based adversarial network (CAN) framework. The CAN is composed of three components: an encode network E, a discriminative model M for the primary task, and a discriminative model S for the subsidiary task. E encodes

input data into a fixed-dimensional code; it can be composed of convolutional layers as well as fully-connected layers. M takes the code from E as input and performs classification based on the primary task. S receives the code generated by E as well and performs classification based on the information to be eliminated. An overall network can be configured to perform multi-task learning by concatenating E, M, and S.

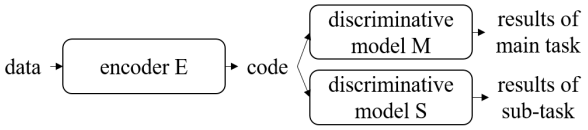


Figure 1: Overall architecture of the proposed CAN framework

The objective of the CAN framework is to eliminate subsidiary information from input data using E, so that the performance of S degrades, and the performance of M increases, consequently. The training procedure to achieve this objective is as follows. First, only E and M are trained to perform the primary task; this process is the same as training a typical deep neural network as a discriminative model. Thus, although E is trained for the primary task, subsidiary information is retained in the codes, because subsidiary information is not considered separately. Next, only S is trained while E and M are left unchanged. While training S, the amount of subsidiary information present in the codes can be evaluated; in addition, the weight parameters of the subsidiary model can be identified. For example, we can imagine the simplest S with just the output layer. The trained S receives 100-dimensional codes as input and classifies them into one of five classes. Considering this, S is composed of a weight matrix with size 100×5, and a five-dimensional bias term. If S was appropriately trained to perform a subsidiary task, the weight matrix of S can be treated as five 100-dimensional column vectors for modeling the dependency of each subsidiary class on the codes. Therefore, we can use the following loss function $\mathcal{L}_E$ when training E to eliminate subsidiary information.

$$\mathcal{L}_E = \mathbb{E}_{\boldsymbol{x} \sim X} \frac{1}{N_c} \sum_{i}^{N_c} CS(E(\boldsymbol{x}), W_i^{sub})^2, \tag{1}$$

where N_c is the number of the column in the weight matrix, $CS(\cdot, \cdot)$ is the cosine similarity function between two vectors, $E(\boldsymbol{x})$ is the output of the encoder network E when $\boldsymbol{x}$ is the input data, and W_i is the ith column of the weight matrix. This loss functions sets the absolute values of the cosine similarity between codes and columns to zero, thus rendering the codes almost orthogonal to the weight parameters of the subsidiary model. To avoid the problem of non-division and improve efficiency in outlier training, the square function is used instead of the absolute values. One can expect that the process of making codes and subsidiary parameters independent of each other would eliminate the subsidiary information contained in the codes.

The main concept of the proposed CAN framework can be summarized as followings. The objective of the CAN is to eliminate subsidiary information by making the codes from E independent with the subsidiary class label y^{sub}. The dependency between the codes and the label y^{sub} can be modeled by weight parameters of S. We expected that making the codes orthogonal to the weight parameters of S would make the codes independent with the class label y^{sub}, consequently, eliminating the subsidiary information contained in the codes. Finally, the cosine similarity applied in the equation (1) makes the codes and weight parameters to be orthogonal.

However, some subsidiary information might still be retained, because eliminating subsidiary information depends on the weight parameters of the model S. Therefore, S is trained again to find another set of parameters, and the subsidiary information is eliminated again based on these new parameters. It is expected that subsidiary information could be eliminated more efficiently by repeating the adversarial process of training and eliminating it by S and E, respectively. This adversarial process is depicted in Fig. 2. Furthermore, it should be noted that E is re-trained with model M to perform primary task separately.

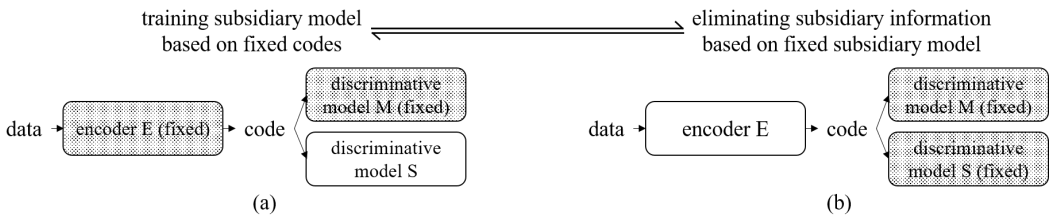


Figure 2: Description of the adversarial process for the proposed CAN framework. (a) Training the model S (subsidiary parameters) based on fixed codes. (b) Training the encoder E to eliminate subsidiary information based on the fixed model S

Several steps can be taken to increase the efficiency of the proposed CAN framework. First, leaky-rectified linear unit (ReLU) (Maas et al. (2013)) can be used as the activation function at the code layer instead of ReLU; the code layer is the last layer of the encoder E. This is because, based on our empirical experiments, we found that when ReLU is used as the activation at the code layer, the codes become excessively sparse. We assume that it is because setting the output of the code layer to all zeros is the simplest method to minimize the proposed loss function, given by Equation (1). Therefore, because of thresholding in ReLU function, the codes might not be orthogonal to the weight vectors of S; in addition, it might lead to drawbacks such as the codes approaching the zero vector. This drawback can be avoided by using leaky-ReLU as the activation function.

Second, L2 normalization can be applied to the codes (Wan et al. (2017)). In particular, L2 normalization ensures that the models S and M do not take into account the scale of the codes. In our proposed CAN framework, the loss function is designed to only adjust the direction of the codes without considering the scale of the codes. Therefore, the effectiveness of the proposed loss function can be increased by fixing the scale of the codes.

The third step involves the use of multiple hidden layers for S. The loss function defined by Equation (1) as well as the aforementioned example are based on a simple S with only an output layer and no hidden layers. However, because the behavior of the CAN framework depends on model S, using a deeper model can eliminate subsidiary information more efficiently. In order to use a deeper model, it is necessary to include methods for treating multiple weight matrices from the model and reflecting them in the loss function. In this study, we used a method to calculate the cosine similarity only in the last hidden layer and output layer as follows.

$$\mathcal{L}_E = \mathbb{E}_{\boldsymbol{x}\sim X} \frac{1}{N_c} \sum_{i}^{N_c} CS(S_{last}(\boldsymbol{x}), W_i^{sub})^2, \tag{2}$$

where N_c is the number of columns in the weight matrix of the output layer of S, $S_{last}(\cdot)$ indicates the output of the last hidden layer of S, and W_i^{sub} indicates the ith column of the weight matrix of the output layer of S. This modified loss function serves to make the output of the last hidden layer and the associated weight parameters of the output layer almost orthogonal. It is important to note that this modified loss function is used for training E, but not S.

3 Related works

In this section, we introduce several studies related to the adversarial process. In particular, we focus on studies about training the discriminative models, although there are other studies based on the adversarial process. The objective of these studies is to train models that are robust to subsidiary information. For example, Yu et al. (2017) built a noise-robust system for speaker verification; for this purpose, a subsidiary model was trained to classify the input speech into N+1 classes, representing N noise types and the clean case. After training the subsidiary model, the encoder was trained to fool the sub-model with fake labels (in this case, all labels were set as clean). Then, the encoder is trained to eliminate noise information by repeating the adversarial process that uses noise type labels and

fake labels. However, in their research, the performances of the framework were evaluated assuming that there are target samples, such as clean speech; therefore, it is difficult to predict the operation of the framework when there is no target sample. In particular, our proposed CAN framework is designed to work without target samples. Furthermore, performance comparison was conducted through experiments for speaker identification, which is discussed in the next section.

In addition, there are studies that use the adversarial process in a manner that interrupts the operation of the subsidiary task. In Shinohara (2016), the encoder network is connected to subsidiary model via a gradient reversal layer (GRL) to extract the channel-invariant feature; the GRL multiplies the gradient by a certain negative constant during training of the encoder network. Gradient reversal leads to similar code distributions over different channel; consequently, this make the codes as indistinguishable as possible for the subsidiary model and channel-invariant feature.

The operation of the GRL can be implemented simply by reversing the loss function of the subsidiary model. Assuming that the loss of the subsidiary model is defined based on softmax, the GRL changes the loss function to softmin. In particular, the softmax function is used to enhance the discriminative power of the model and can be interpreted as a soft version of the argmin function (Goodfellow et al. (2016)). Therefore, applying GRL to the softmax-based loss function is equivalent to using the soft version of argmin function as the loss function. Consequently, the operation of the GRL, in effect, interferes with the operation of the subsidiary model. In contrast, the effect of our proposed framework CAN on the subsidiary model is different from GRL as follows. Assuming that the proposed framework minimizes the loss defined in Equation (1) to zero, the classification of the subsidiary task will be impossible. The output of the subsidiary model, defined by the weights matrix W^{sub} and bias term b^{sub}, is calculated as, $\boldsymbol{o}_S = E(\boldsymbol{x})W^{sub} + b^{sub}$. In addition, the term $E(\boldsymbol{x})W^{sub}$ will be zero when the loss function defined by Equation (1) is zero. Thus, the output is referred to as the bias term, which is a constant term; consequently, the subsidiary model has no discriminative power for any input $\boldsymbol{x}$.

4 APPLICATIONS

In this section, we introduce applications of the proposed CAN framework. In order to apply the CAN framework, speaker identification and digit recognition are set as the primary tasks; in addition, channel information and domain information are set as the subsidiary information for each task, respectively. All deep neural networks used for these applications were implemented using Keras (Chollet et al. (2015)) with a TensorFlow (Abadi et al. (2015)) backend. In addition, we used Kaldi (Povey et al. (2011)), an open-source speech recognition toolkit, for speech signal processing such as i-vector (Dehak et al. (2011)) extraction.

4.1 REDUCING CHANNEL INFORMATION

Here, we introduce speaker identification with exclusion in channel information as one of the applications of the proposed CAN framework. Speaker identification is a task that identifies the person who spoke the input speech. In a typical speaker identification task, it is assumed that only the speech of known speakers is inputted, thus limiting the candidate group to be identified. Channel information is well-known as one of factors that decreases speaker identification performance (Solomonoff et al. (2004)). Furthermore, channel information varies depending on the recording devices and transmission method. Therefore, even if utterances are from a known speaker, they can have different characteristics owing to the channel information. The purpose of applying the proposed CAN framework is to reduce channel information. Reducing channel information from speech data is a challenging task, because there is no specific target sample. For example, we can create noisy speech by inserting noise into clean speech; the clean and noisy speeches can be used as the target and source sample, respectively, for the task of reducing noise from speech. Thus, if there are target and source samples, it is possible to train models, such as a stacked denoising autoencoder, to reduce noise information. However, there is no target sample (clean speech) for the task of reducing channel information, because, as previously mentioned, the characteristics of the recording device are inevitably reflected in the process of recording the speech.

The experiments for speaker identification were designed using the RSR2015 dataset (Larcher et al. (2014)) as follows. Six mobile devices (five smartphones and one tablet) were used for collecting

Table 1: Experimental results for speaker identification using the three frameworks compared in our study

	w/o adversarial process	w/ Yu et al. (2017)	w/ Shinohara (2016)	w/ proposed CAN
Error rate(%)	5.69 $\pm$ 0.13	5.53 $\pm$ 0.09	5.32 $\pm$ 0.09	4.88 $\pm$ 0.12

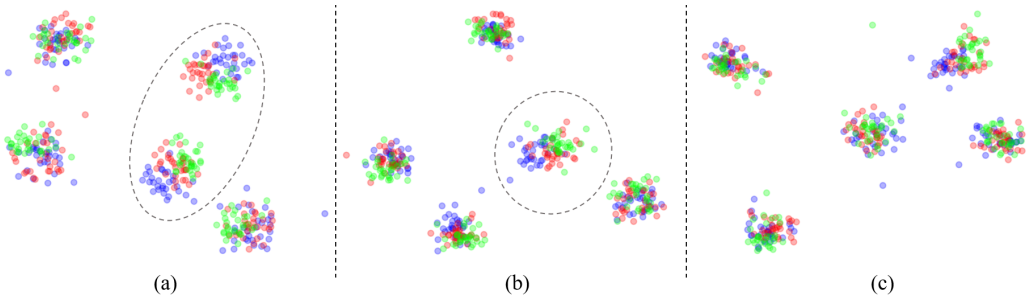
speech data for the RSR2015. Three devices were assigned to each speaker. The performance of speaker identification was evaluated for 143 female speakers among a total of 300 speakers. For training, 10 utterances for each speaker were used. Then, identification was performed using utterances that were about one second long. In addition, as input for the encoder, 200-dimensional i-vectors were extracted for each utterance. The i-vector is a technique that is used for representing utterances of varying lengths as a vector with fixed dimensionality (Dehak et al. (2011)). In particular, the encoder E receives an i-vector and outputs the code. The two models (M and S) receive the code and perform speaker identification and channel identification, respectively. Details about the entire network, including E, M, and S, are presented in appendix-A. In our study, we report the average accuracy over 5 networks that were randomly selected after sufficient training (Liu & Tuzel (2016)). In order to validate the performance of the proposed CAN framework, we compared the performances of the system without the adversarial process with the conventional adversarial process as well as the proposed process.

Table 1 lists the experimental results for speaker identification. The system without adversarial process means a network consisting of only E and M. The framework of Yu et al. (2017) did not show a significant performance improvement over the system without the adversarial process; this can be attributed to the fact that, in their system, there is no target sample in the task for reducing the channel information. The GRL-based framework (Shinohara (2016)) showed a relative error exclusion of about 6.5% compared with the system without the adversarial process. In contrast, our proposed CAN framework showed the lowest error rate, with a relative error exclusion of about 8.2% compared with the GRL-based framework.

Fig. 3 shows examples of the two-dimensional codes obtained using various encoders. The codes were extracted from the training utterances of the randomly selected five speakers in the RSR2015 dataset; these codes were visualized using the t-SNE technique (Maaten & Hinton (2008)). Each utterance is indicated by a red, green, or blue dot, depending on the channel (recording device). Overall, five clusters corresponding to the five speakers were apparent. However, it can be seen that the results of the system without the adversarial process (Fig. 3-(a)) have the largest intra-class variability. In addition, it can be observed that the large variability was caused by the channel information of each utterance. The influence of the channel information in the clusters is highlighted using dotted ellipses in Fig. 3. These clusters show that each utterance can be clearly distinguished based on the channel information even if the utterances are from the same speaker. In contrast, the results of the system based on our CAN framework (Fig. 3-(c)) show the lowest intra-class variability. Furthermore, it is difficult to distinguish the utterance based on channel information. Therefore, we can deduce that the proposed CAN framework led to improvements in speaker identification because of the exclusion in channel information.

4.2 Unsupervised domain adaptation

Unsupervised domain adaptation is a task that makes a classifier suitable for a target domain. The main challenge of this task is that only unlabeled data are provided from target domain. Therefore, the classifier, which is trained using labeled data of the source domain, should be generalized to the target domain. However, because of the phenomenon of dataset bias, a classifier from one domain does not generalize well to another domain (Torralba & Efros (2011)). We assume that the dataset bias is caused by domain information included in the data. Considering this assumption, the domain information represents characteristics of each dataset. Therefore, we performed the unsupervised domain adaptation task using the proposed CAN framework to eliminate domain information. We designed the unsupervised domain adaptation task using the MNIST (LeCun & Cortes (2010)), USPS, and SVHN (Netzer et al. (2011)) datasets to validate the performance of our proposed CAN framework. Several examples of each dataset are depicted in Fig. 4.



(a) (b) (c)

Figure 3: Visualization of codes obtained using the different encoders (colors indicate different channels), (a) When the speaker identifier is trained without the adversarial process, (b) When the GRL is applied to the model, (c) When the proposed CAN framework is applied to the model.

(a) (b) (c)

Figure 4: Examples from the (a) MNIST, (b) USPS and (c) SVHN datasets.

For this performance comparisons, we conducted two kinds of experiments about domain adaptation. As a first experiment, we adopted the experimental protocol described in (Tzeng et al. (2017)). We randomly selected 2000 and 1800 images from the MNIST and USPS datasets, respectively, to train the model as well as for domain adaptation. The selected images of the MNIST dataset were re-sized to 16 $\times$16 to ensure that they are the same size as those of the USPS dataset. In particular, we used the MNIST dataset as the source domain, whereas the USPS dataset as the target domain for domain adaptation. We additionally conducted domain adaptation experiment using different, larger dataset, SVHN instead of USPS with MNIST dataset. In this experiment, the SVHN dataset is used as source domain, and the MNIST dataset is used as target domain. Such setting was introduced in Ganin et al. (2016), as adaptation between domains which are significantly different in appearance.

Though both images and labels were used to train the source domain model, only the images were used for adaptation to the target domain. We compared the performances of the model trained using only source data with the adapted model, which also used the target data. Domain adaptation was conducted using the proposed CAN framework and another method (Tzeng et al. (2017)) to validate the performance of the proposed framework. Adversarial discriminative domain adaptation (ADDA) (Tzeng et al. (2017)) was used to compare the performance of the proposed framework with the other framework. Tzeng et al. (2017) summarized the various adversarial processes that can be used for domain adaptation and proposed a GAN loss-based adaptation framework. In order to validate the performance of the proposed framework, we compared the performances of the system without the adaptation, with ADDA, with the proposed process, along with the performance reported in (Tzeng et al. (2017)).

Unlike previous studies that used LeNet (LeCun et al. (1998)) for digit recognition, we used a model based on the maximum feature map (MFM) (Wu et al. (2015)). The MFM replaces the activation function, such as ReLU, with element-wise maximum operation between feature maps. We expected that the competitive relationship in the MFM network leads to a more appropriate model to represent data from different domains. The competitive relationship requires selecting one of the parameters that are trained to be suitable for each domain using the max operation. Based on our empirical experimental results, we confirmed that the MFM-based model could reduce the size of the model by about half compared with the one using LeNet, consequently, improving the accuracy by about 30%. Details about the entire network are presented in appendix-B.

Table 2: Experimental results for digit recognition using different frameworks.

Error rate(%)	Baseline (src only train)	ADDA (Tzeng et al. (2017))	ADDA (our implementation)	Proposed CAN
Target domain (USPS)	17.59 $\pm$ 0.39	10.60 $\pm$ 0.20	8.51 $\pm$ 0.30	5.51 $\pm$ 0.33
Source domain (MNIST)	2.62 $\pm$ 0.23	-	-	2.50 $\pm$ 0.23
Target domain (MNIST)	30.76 $\pm$ 1.30	24.00 $\pm$ 1.80	19.69 $\pm$ 1.21	16.74 $\pm$ 1.03
Source domain (SVHN)	5.70 $\pm$ 0.16	-	-	5.40 $\pm$ 0.15

Table 2 lists the experimental results for domain adaptation. We confirmed that the performance of the ADDA implemented by us is higher than the previously reported performance of ADDA (Tzeng et al. (2017)). From this result, we found that the MFM-based model is more suitable for the domain adaptation task than the one using LeNet even if the size of the MFM-based model is smaller than the LeNet. Our proposed CAN framework showed the lowest error rate with a relative error exclusion about 35% compared with the ADDA framework in the experiments about adaptation to USPS from MNIST. The proposed framework does not generate a separate encoder for the target domain data during the domain adaptation process. Therefore, it is possible to recognize the source domain data using the model adapted to the target domain. In contrast, the ADDA generates an additional encoder for the target domain; therefore, the adapted model cannot recognize data from the source domain. Using the same model for the target and source domains has advantages, including eliminating the size of the model and increasing its availability. However, model optimization might be performed poorly because the same model must process images from two separate domains (Tzeng et al. (2017)). The results obtained for the performance of the proposed framework in the case of the source and target domains are contrary to this conventional knowledge. In particular, the proposed CAN framework maintained the performance on the source domain and successfully adapted the model to the target domain simultaneously. We could confirm the similar trend in the experiments about adaptation to MNIST from SVHN. Our proposed CAN framework showed the lowest error rate with a relative error exclusion about 15% compared with the ADDA framework in target domain. In addition, we also confirmed that the error rate of the source domain is reduced by about 5% through the process of eliminating domain information.

Fig. 5 shows examples of the two-dimensional codes from the various encoders considered in our study. The codes are extracted from the training data of the MNIST and USPS datasets, and visualized using the t-SNE technique. The target and source domain codes are indicated using red and green dots, respectively. The visualization result of the baseline (Fig. 5-(a)) shows that the codes from the source domain are represented by 10 clusters (referred to as 10 classes) while the codes from the target domain are represented by 9 clusters. This phenomenon considerably degrades the performance of the model M on the target domain even if the distributions of the source and target domains seem similar. The result of the ADDA system (Fig. 5-(b)) shows that the codes of the target domain are expressed in 10 clusters, thus the model is well adapted to the target domain. However, it was observed that there is a difference in the distributions of the codes from the source and target domains because the domain information is still retained in the codes. In contrast, considering the result of our proposed CAN framework (Fig. 5-(c)), it can be observed that the distribution of the source and target domains almost overlap.

5 Discussion

We expected that the proposed CAN could focus on eliminating subsidiary information by using cosine similarity-based loss, while the other methods focus on confusing S. However, it would be difficult to analyze and compare the internal behavior of the deep neural networks. Therefore, we tried to demonstrate the strength of the proposed method through an experiment to recognize the subsidiary information of the codes from E. For this purpose, Es from the experiments about adaptaion to USPS from MNIST in section 4.2 were used; Es were adapted by CAN or ADDA, or not adapted. In particular, recognizing the subsidiary information means identifying the domain of codes (MNIST or USPS). We observed the training loss while training a new model S_2 to identify the domain of the codes. The model S_2 has the same structure with S and performs the same operation, but the

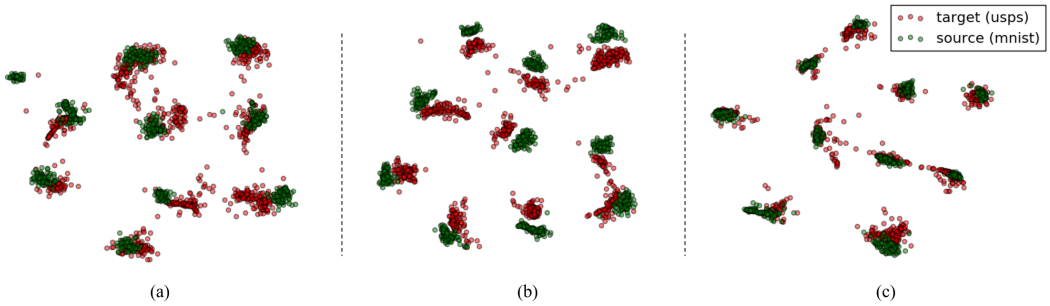


Figure 5: Visualization of codes from various encoders (colors indicate different domains). (a) No adaptation, (b) Adaptation based on ADDA, (c) Adaptation based on the proposed CAN framework

parameters were re-initialized and trained. We expected that the training losses of S_2 could reflect the amount of subsidiary information remaining in the codes. Figure 6 shows that recognizing the domain of codes from E adapted by the proposed method is harder than other cases. Therefore, we concluded that using the proposed loss is more appropriate for reducing the subsidiary information and domain adaptation, even if the subsidiary information could be reduced through the process of confusing S as the conventional method.

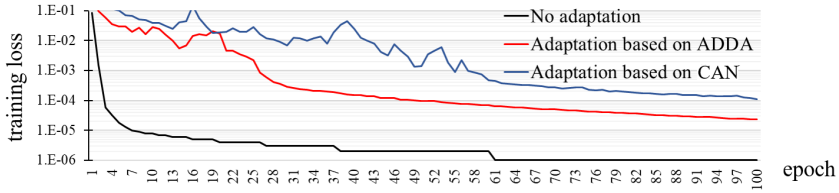


Figure 6: Experimental results for domain recognition using codes from E

6 Conclusions

In this study, we proposed an CAN framework by applying cosine similarity-based adversarial process. Our framework consists of an encoder E, primary model M, and subsidiary model S; the objective of our proposed framework is to eliminate subsidiary information. In order to do so, our framework repeats the process of training the weight parameters of model S, thus rendering the output of the encoder E almost orthogonal to the weight parameters of S. In particular, the proposed framework eliminates the dependence of the subsidiary class label y^{sub} on codes from the encoder. Therefore, the proposed loss function based on cosine similarity can eliminate the subsidiary information. Several improvements were applied to make the proposed CAN framework more efficient, including using leaky-ReLU activation and L2 normalization to the code of E as well as having multiple hidden layers in model S. The performance of our CAN framework was evaluated and compared with other techniques based on the adversarial process through experiments for speaker identification and digit recognition tasks. Experimental results showed that using the proposed CAN framework improved the performance of the primary task. In addition, the output of encoder E was visualized to confirm that the subsidiary information was actually eliminated.

It is required to validate the generalization performance of the subsidiary task for future study. In particular, we treated only the cases where the subsidiary classes are known. For example, we considered only known channels (or known recording devices) in experiments related to reduce channel information. Therefore, it is necessary to design and conduct experiments to confirm the effect of unknown channel information on the encoder E trained using utterances of known channels.

REFERENCES

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

François Chollet et al. Keras. `https://github.com/keras-team/keras`, 2015.

Najim Dehak, Patrick J Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, 2011.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pp. 630–645. Springer, 2016.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Anthony Larcher, Kong Aik Lee, Bin Ma, and Haizhou Li. Text-dependent speaker verification: Classifiers, databases and rsr2015. *Speech Communication*, 60:56–77, 2014.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL `http://yann.lecun.com/exdb/mnist/`.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint*, 2016.

Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *Advances in neural information processing systems*, pp. 469–477, 2016.

Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, pp. 3, 2013.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 5, 2011.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, number EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

Yusuke Shinohara. Adversarial multi-task learning of deep neural networks for robust speech recognition. In *INTERSPEECH*, pp. 2369–2372, 2016.

Alex Solomonoff, Carl Quillen, and William M Campbell. Channel compensation for svm speaker recognition. In *Odyssey*, volume 4, pp. 219–226. Citeseer, 2004.

Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1521–1528. IEEE, 2011.

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, pp. 4, 2017.

Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. Generalized end-to-end loss for speaker verification. *arXiv preprint arXiv:1710.10467*, 2017.

Xiang Wu, Ran He, Zhenan Sun, and Tieniu Tan. A light cnn for deep face representation with noisy labels. *arXiv preprint arXiv:1511.02683*, 2015.

Hong Yu, Zheng-Hua Tan, Zhanyu Ma, and Jun Guo. Adversarial network bottleneck features for noise robust speaker verification. In *Interspeech*. ISCA, 2017.

Appendices

A Detail about the network for speaker identification

The entire network, including E, M, and S, is depicted in Fig. 7-(a). The encoder E comprises of several residual blocks (He et al. (2016)) (Fig. 7-(b)) and a fully-connected layer for code output. In order to reduce the channel information and train the model for speaker identification, a process with one epoch long training of the model S, one epoch long training of the model M, and 5 epochs long training of the model M and encoder E for speaker identification was repeated. We used the Adam optimizer (Kingma & Ba (2014)).

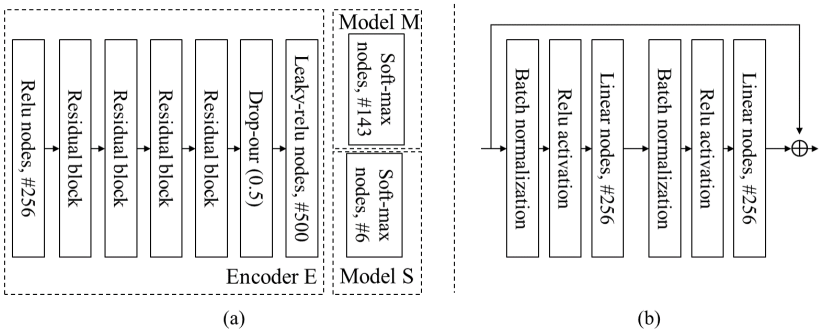


Figure 7: (a) Entire network for speaker and channel identification based on residual blocks. (b) Detailed structure of a residual block

B DETAIL ABOUT THE NETWORK FOR UNSUPERVISED DOMAIN ADAPTATION

The encoder E receives the 16 $\times$16 image (adaptation to USPS from MNIST) or 32 $\times$32 image (adaptation to MNIST from SVHN) as input and outputs the code. The two models (M and S) receive the code and perform digit recognition and domain identification. The entire network, including E, M, and S, for this task is depicted in Fig. 8. In order to train the model for digit recognition and model adaptation, a process of one epoch long training of the model S, one epoch long training of the model M, and 5 epoch long training of the model M and encoder E for digit recognition was repeated. The training process was conducted using the Adam optimizer (Kingma & Ba (2014)). In our study, we report the average accuracy over 5 networks that were randomly selected after sufficient training (Liu & Tuzel (2016)).

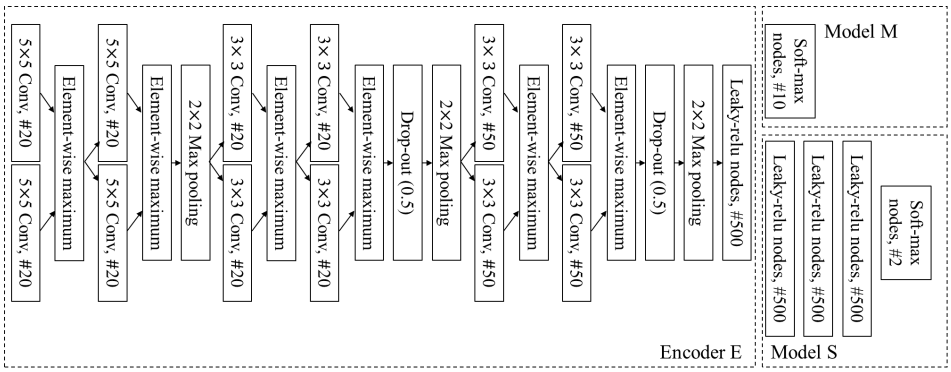


Figure 8: The entire network for digit recognition and domain identification.