INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Conditional Generative Adversarial Nets Classifier for Spoken Language Identification

Peng Shen, Xugang Lu, Sheng Li, Hisashi Kawai

Institute of Information and Communications Technology, Japan

peng.shen@nict.go.jp

Abstract

The i-vector technique using deep neural network has been successfully applied in spoken language identification systems. Neural network modeling showed its effectiveness as both discriminant feature transformation and classification in many tasks, in particular with a large training data set. However, on a small data set, neural networks suffer from the overfitting problem which degrades the performance. Many strategies have been investigated and used to improve the regularization for deep neural networks, for example, weigh decay, dropout, data augmentation. In this paper, we study and use conditional generative adversarial nets as a classifier for the spoken language identification task. Unlike the previous works on GAN for image generation, our purpose is to focus on improving regularization of the neural network by jointly optimizing the "Real/Fake" objective function and the categorical objective function. Compared with dropout and data augmentation methods, the proposed method obtained 29.7% and 31.8% relative improvement on NIST 2015 i-vector challenge data set for spoken language identification.

Index Terms: Conditional generative adversarial nets, cGAN-classifier, i-vector, spoken language identification.

1. Introduction

Spoken language identification (LID) is a task to determine which language is being spoken within a speech utterance [1, 2]. Recently, the i-vector-based method which has been widely used in speaker recognition, has been introduced to LID tasks [3]. I-vector is a fixed-length low-dimensional vector which is convenient for classification modeling. I-vector-based method obtained state-of-the-art performance in many systems for both speaker recognition and language identification tasks [3, 4, 5, 6, 7, 8].

I-vector compresses Gaussian mixture model (GMM) supervectors by confining all sorts of variabilities (language, speaker, channel, session, etc) to a low-dimensional vector. Since i-vector includes total variability factors, discriminant analysis methods, such as linear discriminant analysis (LDA) and local fisher discriminant analysis, have been applied to the i-vectors to obtain discriminative features [9, 7], i.e., for different tasks, the discriminative transform need to emphasize the feature variations correlated to the tasks. Then, the transformed feature can be modeled with classifiers for classification, e.g., support vector machine (SVM), probabilistic linear discriminant analysis (PLDA) [3, 10].

Deep neural network (DNN) has shown its effectiveness for feature learning and classification for many tasks, such as image processing and speech recognition [11, 12]. It has also been used in speaker recognition and spoken language recognition [13, 14]. For LID tasks, DNN can be used as a discriminative classifier which can directly map the features (e.g., i-vector features) to their language IDs [14, 15], or as a front-end processor for feature learning with conventional classifiers for classification [6]. In order to achieve a better performance, a large quantity of labeled training data are often provided, for example, the training data of ASR task includes hundreds of hours of labeled data. However, the DNN model suffers from the overfitting problem with smaller data set which results in poor performance on a test data set. To improve the generalization of the model, regularization methods, such as weight decay [16], dropout [17], data augmentation, have been proposed. For i-vector-based LID tasks, previous works have already investigated DNN with dropout [15] and distance metric learning [18, 19], with limited training data.

Recently, generative adversarial nets (GAN) have attracted a lot of attention for their abilities on image generation [20]. GAN can learn generative models using a two-player zero-sum game borrowed from the game theory. The goal of GAN is to train a generator network that produces samples from real data distribution, by feeding vectors of noise as inputs. To direct the data generation processing, a conditional GAN was proposed and successfully used on image generation and image-to-image translation [21, 22, 23, 24, 25]. Previous works have conditioned GANs on discrete labels [21, 24], text [26], and images [23].

In this work, unlike most of the previous works on GAN for image processing which focus on producing excellent samples, we propose a conditional GAN-based classifier (cGAN-classifier) for the i-vector-based LID task. Compared with the conditional GAN, the cGAN-classifier uses real samples as conditional information to the generator network of GAN, and adds categorical output related to class labels as output to the discriminator network, in addition to "Real/Fake" output. The generator and discriminator networks are optimized by jointly minimizing the "Real/Fake" objective function and categorical classification objective function. The generator can be considered as a feature transformation that supplies additional information to improve the generalization of the discriminator network for classification. Different from the other works which use the generated data to train the additional classifier, the well-trained networks can be directly used for the classification task.

The use of the discriminator network of GAN as classifier has already been investigated for image classification tasks [27, 28]. A categorical GAN (CatGAN) has been investigated for robust unsupervised and semi-supervised learning [27]. The CatGAN combines the neural network classifier with an adversarial generative model that regularizes a discriminatively trained classifier for image classification. Odena [28] and Salimans et al. [25] also investigated to extend GAN to the semi-supervised context by forcing the discriminator network to output class labels. To the best of our knowledge, directly using the conditional GAN for classification has not yet been studied, and there are no previous works using GAN for LID tasks. Exper-

http://dx.doi.org/10.21437/Interspeech.2017-553

iment results indicate that the proposed method is surprisingly effective for the i-vector-based LID task.

The remainder of the paper is organized as follows. Sections 2, 3 present the basic generative adversarial nets and conditional generative adversarial nets, respectively. The proposed methods, i.e., cGAN-classifier is described in Section 4. In Section 5, the results of experiments and investigation are given to evaluate the performance of cGAN-classifier. Discussion and conclusion are given in Section 6.

2. Generative Adversarial Nets

Generative adversarial net (GAN) was introduced by Goodfellow et al. [20], which is a framework for training deep generative models based on the two-player zero-sum game theory. The goal of GAN is to train a generator network $G(\mathbf{z};\theta_g)$ that produces samples from the data distribution $P(\mathbf{x})$, by transforming noise variables $\mathbf{z}$ into fake samples $G(\mathbf{z})$. The generator is trained by playing against an adversarial discriminator network D that aims to distinguish between samples from the true data distribution $P(\mathbf{x})$ and the generator distribution. More formally, D and G play the two-player minimax game by the following expression:

$$\min_G \max_D V = \mathbb{E}_{\mathbf{x}\sim P(\mathbf{x})}[log(D(\mathbf{x}))] + \mathbb{E}_{\mathbf{z}\sim P(\mathbf{z})}[log(1 - D(G(\mathbf{z})))], \quad (1)$$

where $D(\mathbf{x})$ represents the probability that $\mathbf{x}$ comes from the real data rather than the fake data. The D is trained to maximize the probability of assigning the "Real" label to real training examples and "Fake" label to fake samples produced by G, and simultaneously G is trained to minimize $log(1 - D(G(\mathbf{z})))$.

3. Conditional Generative Adversarial Nets

The original GAN is an unconditioned generative model, i.e., there is no control to direct the data generation processing. A conditional GAN [21, 24] was introduced to direct the data generation processing by conditioning the model on additional conditional information, such as class labels or data from other modalities. The conditional GAN is an extension of GAN created by feeding conditional information into both the generator and the discriminator networks to produce samples. Previous works showed that using class conditional synthesis can significantly improve the quality of generated samples [29].

As shown in Fig. 1.(a), the input to the generator network is a combination of noise $\mathbf{z}$, and conditional information, such as class labels. The real or fake data with conditional information are presented as inputs to the discriminator network. Mathematically, we define symbol $\mathbf{c}$ as the conditional information, and equation 1 can be rewritten as:

$$\min_G \max_D V = \mathbb{E}_{\mathbf{x}\sim P(\mathbf{x})}[log(D(\mathbf{x},\mathbf{c}))] + \mathbb{E}_{\mathbf{z}\sim P(\mathbf{z})}[log(1 - D(G(\mathbf{z},\mathbf{c}),\mathbf{c}))]. \quad (2)$$

Conditional GAN was successfully used for face generation [24], image-to-image translation [23], and image synthesis [30].

4. Conditional Generative Adversarial Nets-based Classifier

Previous works on conditional GAN used discrete labels [21, 24], text [26], and images [23] as conditional information for image generation and image-to-image translation. These works focused on producing excellent images or certain images given the conditional information. For classification tasks, normally, additional classifiers are used with the generated images, or as additional images to the original images. Because of GAN models the data distribution explicitly through the reconstruction of input samples, using the generated images is expected to improve model generalization for the classification task, in particular, with limited training data. Previous works showed that, the well-trained generator could generate "very good" images, for example the generated number on the MNIST data set. Nevertheless, for the classification task, the challenges are not "normal" samples but "abnormal" samples which often lie in the boundary of a class. Using the "very good" generated images cannot help improve the regularization of the model for the classification task. Therefore, how to effectively select the generated images for classification model training will be a challenging task. In this work, unlike many works focusing on image/feature generation, we investigate conditional GAN as a classifier (cGAN-classifier) to improve regularization of the neural network.

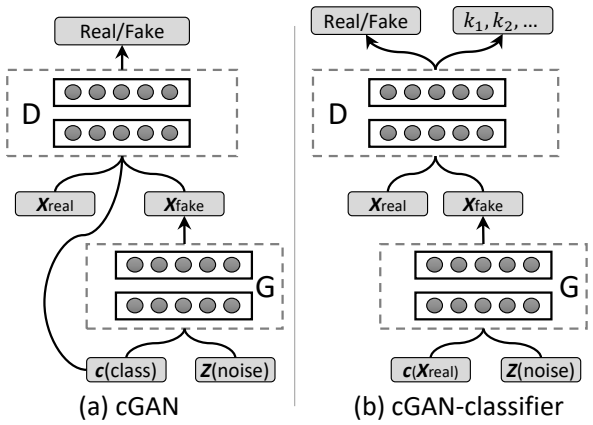

Figure 1: *cGAN (a): Conditional GAN and cGAN-classifier (b): conditional GAN-based classifier.*

Figure 1.(b) shows the framework of the proposed cGAN-classifier. The cGAN-classifier includes both a discriminator and a generator network which are similar to the conditional GAN. The inputs of the generator network are noise, for example Gaussian noise, and conditional information, i.e., real samples. The generator here can be considered as a feature transformation which can transfer real input samples to generated features. Then, the real samples and the generated samples are used for discriminator network optimization. The discriminator network has a source output, i.e., "Real/Fake", same with the basic GAN, and a categorical output corresponding to the class labels of the real input samples.

Mathematically, given an input noise $\mathbf{z}$ and a conditional information $\mathbf{x}_{real}$, i.e., a real sample. The generator network transfers the real sample $\mathbf{x}_{real}$ and the noise vector $\mathbf{z}$ to a fake sample $\mathbf{x}_{fake}$. Then $\mathbf{x}_{real}$ and $\mathbf{x}_{fake}$ are fed into the discriminator network. Aside from the "Real/Fake" source output, K classes labels $\{k_1, k_2, ...k_K\}$ corresponding to the class labels are used for classification optimization. The parameters of the discriminator network are shared between the source output objective function and the classification objective function. We define V_D as binary source output objective function and V_C as K-class classification objective function, V_D and V_C are de-

fined as:

$$\min_G \max_D V_D = \mathbb{E}_{\mathbf{x}\sim P(\mathbf{x})}[log(D(\mathbf{x}))] + \mathbb{E}_{\mathbf{z}\sim P(\mathbf{z});\mathbf{c}\sim P(\mathbf{x})}[log(1 - D(G(\mathbf{z}, \mathbf{c})))], \quad (3)$$

$$\max_G \max_D V_C = \mathbb{E}_{\mathbf{x}\sim P(\mathbf{x})}[log(D(k|\mathbf{x}))] + \mathbb{E}_{\mathbf{z}\sim P(\mathbf{z});\mathbf{c}\sim P(\mathbf{x})}[log(D(k|G(\mathbf{z}, \mathbf{c})))], \quad (4)$$

where $D(\mathbf{x})$ represents the probability that $\mathbf{x}$ comes from the real data $\mathbf{x}_{real}$ rather than $\mathbf{x}_{fake}$, $D(k|\mathbf{x})$ represents the probability that $\mathbf{x}$ belong to class k, and $G(\mathbf{z}, \mathbf{c})$ is the output of the generator network, i.e., $\mathbf{x}_{fake} = G(\mathbf{z}, \mathbf{c})$. The network for V_D and V_C can be same or different, in this work, the discriminator network is shared between V_D and V_C. The optimization procedure of V_D and V_C are same with the basic GAN described in Section 2. The cGAN-classifier objective function is defined as:

$$V = V_D + \alpha V_C, \quad (5)$$

where α is a trade-off between the discriminator objective function and the classifier objective function. In this work, the trade-off α was set to 1.

The proposed framework has the following characteristics: 1) Unlike the conditional GAN using class labels as conditional information, the framework can be directly used for the classification task using real samples as conditional information. 2) The generated features can be optimized and selected automatically to improve regularization of the model for the classification task.

5. Experiments

In this section, experiments were conducted to evaluate the effectiveness of the proposed method. The i-vector data based on NIST 2015 language recognition i-vector machine learning challenge were used for examination. In NIST 2015 i-vector challenge data, there are fifty languages in training data set, and each language has 300 400-dimensional i-vector samples. In this study, we selected 250 i-vectors from the training data set for each language as training data and the remaining 50 i-vectors were used for validation. In total, there are 12,500 samples in our training data set and 2,500 samples in the validation data set. There are 6,500 test samples in the originally test set. We removed the out-of-set class samples because we are not focusing on the solution to the out-of-set problem. Finally, the test data set includes 5,000 samples (100 samples for each language). Identification error rate was used as the evaluation criterion.

5.1. Implementation of baseline systems

We built the baseline systems using a DNN model with L2, dropout and data augmentation methods. The configuration of the network is 400 (RAW features) or 49 (LDA features) neurons in the input layer, 2 hidden layers with 512 neurons for each layer, and 50 neurons in the output layer corresponding to language IDs. The mini-batch size was set to 128, and learning rate was set to 0.001. The maximum learning epoch was set to 500, and early stopping strategy was used with the validation data set. We conducted each experiment three times using stochastic gradient descent (SGD) optimizer and selected the best result. Another baseline method is data augmentation; the previous DNN models (with and without dropout) were used. The generated data were created by adding random uniformly distributed noise over the interval [-1, 1] to the original i-vector data. We also compared the proposed method with conventional methods, i.e., cosine distance, support vector machine (SVM) with linear and radial basis function (RBF) kernels [19]. The optimal model parameters of SVMs were obtained based on a grid search with cross-validation. The result of DNN with distance metric learning method was also listed for comparison [19].

Table 1: *The discriminator and generator networks used for cGAN-classifier; conv. means convolutional layer, FC means fully connected layer.*

Discriminator D	**Generator** G
Input: $\mathbf{x}_{real}, \mathbf{x}_{fake}$	Input: $\mathbf{z}$, $\mathbf{x}_{real}$
FC_1 ($\mathbf{x}_{real}$), tanh	FC_1 ($\mathbf{z}$), tanh
FC_2 ($\mathbf{x}_{fake}$ or $\mathbf{x}_{real}$), tanh	FC_2 ($\mathbf{x}_{real}$), tanh
Merge FC_1 and FC_2, tanh	Merge FC_1 and FC_2, tanh
3×3 conv. 128, tanh	Batch normalization
FC (1024), tanh	2×2 perforated up-sampling
	5×5 conv. 64, tanh
	2×2 perforated up-sampling
	5×5 conv. 1, tanh
Output: (51), softmax	Output: (49), tanh

5.2. Implementation of the proposed method

Previous works showed that the configuration of GAN is very important for stable training. Table 1 illustrates the configuration of cGAN-classifier networks used in this work which refer to the previous work [27]. For the generator network, there are two input layers corresponding to noise $\mathbf{z}$ and conditional information $\mathbf{x}_{real}$. The two input layers were merged into a fully-connected (FC) neural network layer. The output of the generator network is a 49-dimensional vector $\mathbf{x}_{fake}$. Similar to the generator network, there are two input layers in the discriminator network corresponding to $\mathbf{x}_{real}$ and $\mathbf{x}_{fake}$. Batch normalization was used before the convolutional layers. In our experiments, $\mathbf{x}_{real}$ were the dimension-reduced 49-dimensional features via LDA (LDA features). The networks were trained using SGD with momentum 0.9 and adaptive subgradient method (Adagrad) optimizer, and the mini-batch size was set to 128, the learning rate was set to 0.0005. The maximum learning epoch was set to 500, and early stopping strategy was used with the validation data set.

5.3. Results of baseline systems

Table 2 shows the results of the baseline systems. As described in subsection 5.1, the DNN model has two fully-connected hidden layers with dropout 0.3 for the input layer and 0.5 for the hidden layers. This configuration of DNN was selected by comparing the different numbers of hidden layers and different dropout settings. Data augmentation was done by adding uniformly distributed noise to the original i-vector features. Aside from the original 12500 training samples, 12500 training samples were created for data augmentation experiments. From the results, we can see that DNN with dropout obtained better results than the cosine distance method and SVM with linear or RBF kernel methods. Compared with L2 regularization, the data augmentation method achieved a better result. Although data augmentation on speech data can improve the final performance [8], for this i-vector data, the data augmentation method could not outperform the dropout method. Our previous work,

Table 2: *Results of baseline systems: DNN with L2, dropout, and data augmentation; COSINE, and SVM with Linear and RBF kernels. RAW means the original 400-dimensional i-vector, LDA means the dimension-reduced 49-dimensional features via LDA; DP means dropout; DM means distance metric; Valid. means results on validation data set. (Identification error rate %)*

Baseline methods	Train	Valid.	Test
DNN L2 (RAW)	1.31	17.92	18.48
DNN DP (RAW)	0.88	13.84	**15.70**
DNN DP (LDA)	8.83	14.12	16.12
DNN (augmentation)	0.00	17.16	17.76
DNN DP(augmentation)	5.30	14.44	16.62
COSINE (RAW) [19]	10.93	17.24	17.86
LINSVM (LDA) [19]	8.61	16.04	16.44
RBFSVM (LDA) [19]	2.91	15.78	16.40
DNN+DM+DP [19]	3.39	14.45	**15.14**

Table 3: *Results of proposed methods (cGAN-classifier (SGD) and cGAN-classifier (Adagrad)) and investigation experiments. LDA means the dimension-reduced 49-dimensional features via LDA; DP means dropout; Valid. means results on validation data set.(Identification error rate %)*

Methods	Train	Valid.	Test
Network-D (LDA, SGD)	9.21	14.40	16.34
Network-D (LDA, Adagrad)	1.56	15.16	16.76
DNN DP ($\mathbf{x}_{fake}$, SGD)	12.70	18.08	20.16
DNN DP (LDA, $\mathbf{x}_{fake}$, SGD)	10.30	14.84	17.84
cGAN-classifier (SGD)	7.49	12.68	**14.60**
cGAN-classifier (Adagrad)	5.41	10.12	**11.34**

DNN with dropout and distance metric learning was the best baseline system with an identification error rate of 15.14%.

5.4. Results of the proposed method

Before examining the proposed method, we evaluated the performance of the discriminator network as a classifier for the classification task. The experiments were done by removing the "Real/Fake" output and the $\mathbf{x}_{fake}$ input. The network was trained with LDA features with both SGD and Adagrad optimizers. The learning rate was set to 0.0005. The results were listed in Table 3. From the results, we can see that the discriminator network with SGD method, i.e., "Network-D (LDA, SGD)" obtained 16.34%, could not outperform the DNN model with dropout method, i.e., "DNN DP (LDA)". We also evaluated this discriminator network with dropout regularization. However, it could not obtain a better result than the baseline method with dropout. Comparing SGD and Adagrad optimizer, the result using SGD optimizer is better than the result using Adagrad optimizer.

We evaluated the proposed method with both SGD and Adagrad optimizers. The cGAN-classifier with SGD optimizer, i.e., "cGAN-classifier (SGD)", obtained 14.60% identification error rate. Compared to the dropout method with LDA features, i.e., "DNN DP (LDA)", it achieved 9.4% relative improvement. The results of "Network-D (LDA, SGD)" and "Network-D (LDA, Adagrad)" show that using Adagrad optimizer could not improve the performance. However, the cGAN-classifier with Adagrad, i.e., "cGAN-classifier (Adagrad)", achieved 29.7% relative improvement than DNN with dropout method, i.e., "DNN DP (LDA)". The Adagrad optimizer is more effective on GAN optimization than SGD with the setting of our experiments. We already know that the performance using the discriminator network for classification can not outperform the performance using the baseline DNN network for the classification task. Therefore, the improvements of the proposed method were due to the generated features produced by the generator network. As the progress of the model optimization, the generator network can supply useful information to improve the generalization of the model which results in high performance on test data set.

Two additional experiments were done to investigate the contribution of the generated features. One of the experiments was on DNN network with dropout using the generated 49-dimensional features as input, and this experiment was termed as "DNN DP ($\mathbf{x}_{fake}$)". The other experiment was termed "DNN DP (LDA, $\mathbf{x}_{fake}$)". In this experiment, we combined the LDA features and the generated features as the input to train the DNN network with dropout setting. The generated features used in these two experiments were created using the "best" generator network which was selected based on the performance on validation data set. The configuration of "DNN DP" is the same with the baseline system. The results were listed in Table 3. From the results, we can see that using only the generated features obtained 20.16%. The experiment with both LDA features and the generated features, obtained 17.84%. These two experiments obtained worse results than the corresponding baseline system, i.e., "DNN DP (LDA)". From these results, we can see that it is hard to improve the generalization of the model with the generated features, although it is reasonable via designing an effective evaluation metric to select better generated samples. However, the proposed method can automatically select useful generated features to improve the regularization of the model for classification and the well-trained model can be directly used for the classification task.

For LID tasks, the experiment results showed that the proposed method is a very effective method with i-vector features. The proposed method achieved 29.7%, 31.8% relative improvements than DNN with dropout method and data augmentation method, respectively. Compared with the best baseline system, i.e., DNN with distance metric and dropout (DNN+DM+DP), the proposed method obtained 25.1% relative improvement.

6. Discussion and Conclusions

In this paper, we investigated conditional generative adversarial nets and proposed a conditional generative adversarial nets-based classifier (cGAN-classifier). cGAN-classifier directly uses both the discriminator and generator networks as a classifier by using real samples as conditional information. This framework is easy to implement because it can automatically optimize and select the generated features to improve regularization of the model for the classification task. Experiments on the i-vector-base LID task showed that the proposed method is a very effective method.

There are still many challenges in the cGAN-classifier. Previous works discussed the importance of the configuration of the networks of GAN. For the proposed cGAN-classifier, how to design effective discriminator and generator networks will be one of our future works. In the proposed framework, there is a trade-off between the two outputs of D, and how to balance these two objective functions will be another future work.

7. References

[1] H. Li, B. Ma and K. A. Lee, "Spoken language recognition: From fundamentalsto practice," in Proc. of *The IEEE*, vol. 101, no. 5, pp. 1136-1159, 2013.

[2] C.-H. Lee, "Principles of spoken language recognition," in *Springer Handbook of Speech Processing and Speech Communication*, 2008.

[3] N. Dehak, P. Torres-Carrasquillo, D. Reynolds and R. Dehak, "Language recognition via ivectors and dimensionality reduction," in Proc. of *Interspeech*, pp. 857-860, 2011.

[4] S. Novoselov, T. Pekhovsky and K. Simonchik, "STC speaker recognition system for the NIST i-vector challenge," in Proc. of *Odyssey 2014*, 2014.

[5] S. O. Sadjadi, J. W. Pelecanos and S. Ganapathy, "Nearest neighbor discriminant analysis for language recognition," in Proc. of *ICASSP*, pp.4205-4209, 2015.

[6] Y. Song, B. Jiang, Y. Bao, S. Wei and L.-R. Dai, "I-vector representation based on bottleneck features for language identification," in *Electronics Letters*, vol. 49, no. 24, pp. 1569-1570, 2013.

[7] P. Shen, X. Lu, L. Liu, H. Kawai, "Local Fisher discrimiant analysis for spoken language identification," in Proc. of *ICASSP*, 2016.

[8] M. Najafian, S. Safavi, P. Weber, M. Russell, "Augmented Data Training of Joint Acoustic/Phonotactic DNN i-vectors for NIST LRE15,", in Proc. of *Odyssey 2016*, June, 2016.

[9] R. A. Fisher, "The use of multiple measurements in taxonomic problems," in *Annals of Eugenics*, vol. 7, no. 2, pp. 179-188, 1936.

[10] S. Prince, and J. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *IEEE International Conference on Computer Vision*, pp. 1-8, 2007.

[11] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82-97, 2012.

[12] Y. LeCun, Y. Bengio, G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436-444, 2015.

[13] R. Richardson, D. Reynolds, and N. Dehak, "Deep Neural Network Approaches to Speaker and Language Recognition," *IEEE Signal Processing Letters*, Vol. 22, No. 10, pp. 1671-1675, 2015.

[14] G. Montavon, "Deep Learning for Spoken Language Identification," *NIPS workshop on Deep Learning for Speech Recognition and Related Applications*, 2009.

[15] S. Ranjan, C. Yu, C. Zhang, F. Kelly, J. Hansen, "Language recognition using deep neural networks with very limited training data," In Proc. of *ICASSP*, 5830-5834, 2016.

[16] C. M. Bishop, "Pattern Recognition and Machine Learning (Information Science and Statistics)," Springer-Verlag New York, 2006

[17] G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detector," arXiv preprint arXiv:1207.0580, 2012.

[18] X. Lu, P. Shen, Y. Tsao, H. Kawai, "Pair-wise Distance Metric Learning of Neural Network Model for Spoken Language Identification," in Proc. of *Interspeech*, Sep. 2016.

[19] X. Lu, P. Shen, Y. Tsao, H. Kawai, "Regularization of neural network model with distance metric learning for i-vector based spoken language identification," in *Computer Speech & Language*, 2017.

[20] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," ArXiv e-prints, June 2014.

[21] M. Mirza and S. Osindero, "Conditional Generative Adversarial Nets," arXiv:1411.1784, Nov. 2014.

[22] A. Radford, L. Metz, S. Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," in *ICLR*, 2016.

[23] P. Isola, J. Zhu, T. Zhou, A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," arXiv:1611.07004, Nov. 2016.

[24] J. Gauthier, "Conditional generative adversarial nets for convolutional face generation," Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester, 2014.

[25] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, "Improved Techniques for Training GANs," arXiv:1606.03498, 2016

[26] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," arXiv preprint arXiv:1605.05396, 2016.

[27] J. T. Springenberg, "Unsupervised and semi-supervised learning with categorical generative adversarial networks," arXiv:1511.06390, 2016.

[28] A. Odena, "Semi-Supervised Learning with Generative Adversarial Networks,", arXiv:1606.01583, 2016.

[29] A. van den Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, and K. Kavukcuoglu, "Conditional image generation with pixelcnn decoders," CoRR, abs/1606.05328, 2016.

[30] A. Odena, C. Olah, J. Shlens, "Conditional Image Synthesis With Auxiliary Classifier GANs,", arXiv:1610.09585, Jan. 2017.