Interspeech 2018
2-6 September 2018, Hyderabad

I-vector Transformation Using Conditional Generative Adversarial Networks for Short Utterance Speaker Verification

Jiacen Zhang, Nakamasa Inoue, Koichi Shinoda

Tokyo Institute of Technology
{jiacen, inoue}@ks.c.titech.ac.jp, shinoda@c.titech.ac.jp

Abstract

I-vector based text-independent speaker verification (SV) systems often have poor performance with short utterances, as the biased phonetic distribution in a short utterance makes the extracted i-vector unreliable. This paper proposes an i-vector compensation method using a generative adversarial network (GAN), where its generator network is trained to generate a compensated i-vector from a short-utterance i-vector and its discriminator network is trained to determine whether an i-vector is generated by the generator or the one extracted from a long utterance. Additionally, we assign two other learning tasks to the GAN to stabilize its training and to make the generated i-vector more speaker-specific. Speaker verification experiments on the NIST SRE 2008 “10sec-10sec” condition show that after applying our method, the equal error rate reduced by 11.3% from the conventional i-vector and PLDA system.

Index Terms: speaker verification, short utterance, i-vector transformation, generative adversarial networks, multi-task learning

1. Introduction

Recent years have seen a great improvement in text-independent speaker verification. The speaker verification system extracts speaker characteristic information from a given utterance and then verify the speaker ID. In the state-of-the-art methods of speaker verification, i-vector [1] is used to represent speaker characteristics, and probabilistic linear discriminant analysis (PLDA) [2, 3, 4] is used as a verifier. While this system performs well on long utterances, the performance degrades drastically when only short utterances are available [5]. The main cause of this problem is the biased phonetic distribution of short utterances, which makes the estimated speaker features become statistically unreliable. However, in many real world scenarios, users may be reluctant to provide several-minute-long utterances.

Significant efforts have been made to remedy the performance degradation in short utterance speaker verification. In [6][7][8], the variance of i-vectors for short utterances are modeled and used for i-vector normalization. [9] and [10] proposed to utilize duration information in PLDA model. [11] uses phonetic information to reconstruct reliable i-vectors.

In the past years, deep learning has become very popular in the speaker verification field. Many approaches use deep neural networks to process i-vectors. For example, [12] proposed a variational autoencoder as a back-end for i-vector based speaker recognition, [13] used denoising autoencoders to compensate for noisy speech. However, a large amount of data is required for training deep neural networks [14], while the amount of data available for speaker verification are usually very small. This has been one of the biggest obstacles for building an end-to-end speaker verification system using deep learning. Hence, it may be better to improve the i-vector and PLDA framework by using deep learning. Recently, a novel structure called generative adversarial network (GAN) [15] has become extremely popular. GAN can learn a mapping from random noise to target domain, by playing a zero-sum game with two networks, a generator G and a discriminator D: G tries to generate “real” samples which can fool D, while D tries to determine whether a given sample is from real data distribution or from G.

This paper describes an i-vector transformation method using conditional GAN for improving i-vector based short utterance speaker verification. The method uses GAN to estimate a generative model which can generate a reliable i-vector from an unreliable i-vector, in which we assume an i-vector from a long utterance is reliable, and an i-vector from a short utterance is unreliable. Specifically, we used the conditional version of GAN, where both the generator and the discriminator have an i-vector from a short utterance as the conditional input. The generator G tries to generate a reliable i-vector from an unreliable one, and the discriminator D tries to decide whether a given reliable i-vector is a real one extracted from a long utterance or a fake one generated by G. In order to stabilize GAN training, numerical difference (cosine distance) between generated i-vectors and target reliable i-vectors are used in the training stage. Moreover, inspired by [13], we tried to improve the speaker discriminative ability of generated i-vectors by adding an extra speaker label predicting task to G. This multi-task learning framework can better guide the training of GAN. In the testing stage, G is used to generate reliable i-vectors from those extracted from short utterances, and then the generated i-vectors would be used in PLDA scoring.

This paper is organized as follows: Section 2 briefly introduces related works of our methods. Section 3 presents the proposed GAN-based structure for i-vector restoration. Section 4 describes experimental evaluations for speaker verification in two NIST SRE tasks. Section 5 summarizes this paper.

2. Related Works

2.1. I-vector and PLDA

I-vector and PLDA have been widely used in the state-of-the-art systems for text-independent speaker verification. The i-vector approach aims to extract a fixed and low dimension representation from a given utterance based on a factor analysis model. As described in [1], an utterance is projected onto a low-dimensional total variability space which contains both channel- and speaker-dependent information, as an i-vector. Given an utterance, the channel- and speaker-dependent GMM supervector M can be written as:

$$M = m + Tw, \tag{1}$$

where m is the speaker- and channel-independent supervector taken from the universal background model (UBM), T is the

10.21437/Interspeech.2018-1680

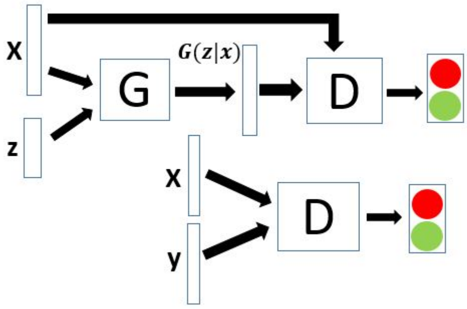

Figure 1: *Conditional Generative Adversarial Networks.*

total variability matrix (TVM) and w is the i-vector.

Probability linear discriminant analysis (PLDA) [4] is applied as a generative model for i-vectors, which can be written as follows,

$$w = \bar{w} + Ux + Vy + \epsilon \tag{2}$$

where $\bar{w}$ is the global mean of i-vectors, U and V are an eigenvoice and an eigenchannel matrix respectively, x and y are speaker- and channel-factors, and ϵ is residual noise.

Given two i-vectors, the log-likelihood ratio of the same-speaker and different-speaker hypotheses is computed by the PLDA model as the measure of their similarity.

2.2. Generative Adversarial Networks Family

Generative adversarial networks (GANs) were introduced in [15] to estimate a generative model by an adversarial process, in which a generator G tries to generate a sample using a random noise vector z and a discriminator D tries to compute the probability that a given sample is from real data y rather than generated by G. Training of GAN is equivalent to optimizing the following min-max function,

$$\begin{aligned}\min_G \max_D V_{\text{GAN}}(D, G) = E_y[\log D(y)]\\ + E_z[\log(1 - D(G(z)))].\end{aligned} \tag{3}$$

As our target is about transformation, we used GAN's conditional version (CGAN) [16] in our approach. The adversarial training procedure is almost the same as the original GAN, and the only difference is both the generator and the discriminator have a conditional input x, as in Figure 1. The min-max function is:

$$\begin{aligned}\min_G \max_D V_{\text{CGAN}}(D, G) = E_{x,y}[\log D(y|x)]\\ + E_{x,z}[\log(1 - D(G(z|x)))].\end{aligned} \tag{4}$$

There have already been several successful applications of CGAN in similar tasks. [17] uses it to convert image styles. [18] applies it to enhance speech. Inspired by their success, we apply CGAN in i-vector space to improve the performance of i-vector based short utterance speaker verification.

3. Proposed Method

Figure 2 shows the framework of our proposed method. At first, acoustic features (MFCC) are extracted from a short utterance, then an unreliable i-vector is extracted from them. Next, an i-vector transformation function is applied to the unreliable i-vector, and finally the transformed i-vector is fed into the PLDA model.

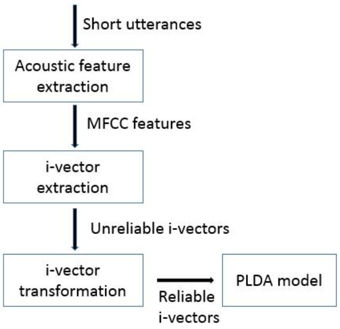

Figure 2: *Framework of our method.*

3.1. GAN for i-vector Transformation

Our target is to estimate a transformation function which can restore a reliable i-vector (extracted from a long utterance), from a short-utterance i-vector. We use a CGAN-based structure to estimate this function. Overall architecture of the proposed GAN is the one shown in Figure 1, where the conditional input x is an i-vector extracted from a short utterance, the real sample y is an i-vector from a long utterance. In the training stage, G is optimized to generate a reliable i-vector using the one extracted from a short utterance, and D is optimized to determine whether the given reliable i-vector is fake (generated by G) or real (extracted from a long utterance). In testing, G is used as the transformation function for an i-vector extracted from a short utterance in the testing set.

In order to prevent several problems such as unstable gradient and model collapse in GAN training, we use a special GAN structure Wasserstein GAN (WGAN) [19]. Denoting x as an unreliable i-vector, y as a reliable i-vector and z as random noise, the min-max function is represented as:

$$\begin{aligned}\min_G \max_D V_{\text{WCGAN}}(D, G) = E_{x,y}D(y|x)\\ - E_{x,z}D(G(z|x)),\end{aligned} \tag{5}$$

Then the objective function related to GAN for G is

$$\min \mathrm{G} = -E_{x,z}D(G(z|x)), \tag{6}$$

and for D, objective function is

$$\max \mathrm{D} = E_{x,y}D(y|x) - E_{x,z}D\left(G\left(z|x\right)\right). \tag{7}$$

Regarding the training data for GAN, i-vectors extracted from short and long utterances are required. While only long utterances are present in the training dataset, we obtained short utterances by segmenting a long utterance into short utterances. I-vectors are extracted from both long and short utterances using the same extractor. Through this process we can obtain an i-vector pair consisting two i-vectors from the same speaker and session, but one is from a short utterance and the other is from a long utterance. The i-vector pairs are utilized in the next section.

3.2. Speaker Verification-oriented Objective Functions

To better guide the training of GAN for our task and make the best use of the training data, two additional learning tasks are added to the GAN framework.

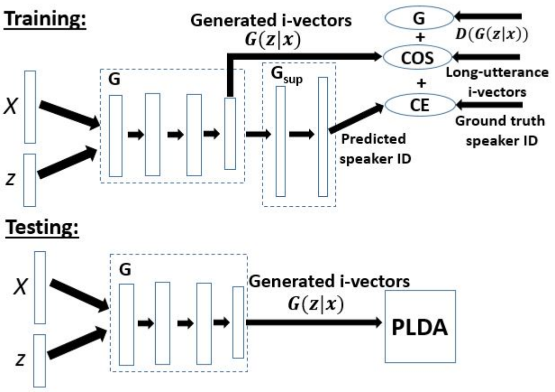

Figure 3: *Training of the generator network G and its application in the testing stage.*

3.2.1. Numerical difference

The most straight-forward approach to measure the performance of transformation is computing the numerical difference between the generated i-vector and the target. We compute this objective function using i-vector pairs mentioned above. In many other similar tasks, mean squared error (MSE) is used to measure such a numerical difference. However, for i-vectors, we believe cosine distance is more suitable. The objective function related to this task can be written as:

$$\min \mathrm{COS} = \frac{1}{m}\sum_{i=1}^{m}\left[\frac{1}{n_i}\sum_{j=1}^{n_i}\left(1 - \frac{G(z|x_{ij})\cdot y_i}{\|G(z|x_{ij})\|\,\|y_i\|}\right)\right], \tag{8}$$

where m is the number of long utterances in the training set, y_i refers to the i-vector extracted from the i-th long utterance in the training set, n_i is the number of short utterances extracted from the i-th long utterance, x_{ij} means the i-vector extracted from the j-th segment of the i-th long utterance and z is random noise.

3.2.2. Speaker discrimination

The training objectives explained above only compensate the variance brought by the biased phonetic distribution of short utterances, and the speaker labels provided by the training set are not used yet. Clearly, improving the speaker discriminative ability of generated i-vectors can enlarge the inter-speaker differences among i-vectors, which would improve the verification performance in the PLDA scoring stage. As shown in Figure 3, in the training stage, a supplementary section, G_{sup}, is concatenated after the generator G, which takes the generated i-vector as an input and predicts its speaker label. We minimize cross entropy between the prediction result and the ground truth:

$$\min \mathrm{CE} = \frac{1}{m}\sum_{i=1}^{m}\left[\frac{1}{n_i}\sum_{j=1}^{n_i} l_{ij}^{k}\left(\log o_{ij}^{k}\right)\right], \tag{9}$$

where l_{ij}^{k} is the empirical probability observed in the ground truth that the target i-vector belongs to the k-th class, and o_{ij}^{k} is the predicted probability that the generated i-vector belongs to the k-th class. In summary, for training G, our goal is to minimize

$$a\,\mathrm{G} + b\,\mathrm{COS} + c\,\mathrm{CE}, \tag{10}$$

where a, b, c are weight parameters for these three targets, respectively.

After training, as shown in Figure 3, only G is used to generate a reliable i-vector, which is fed into a PLDA model for the next scoring step.

4. Evaluation

4.1. Experimental setup

We evaluated the performance of our method in the speaker verification tasks of the NIST SRE 2008 [20]. We used the "short2-10sec" and "10sec-10sec" conditions as our trial sets, where each session is an excerpt of telephone speech, and "short2" refers to five-minute-long speech while "10sec" means that the active voice part in the sample is about 10 seconds. There are three sub-conditions in the trail sets: Condition 6 covers all the speech segments, Condition 7 involves only those spoken in English, and Condition 8 only has those spoken in English by native U.S. English speakers [20]. Performance measures for the evaluation were the equal error rate (EER) and the minimum detection cost function (minDCF) of NIST SRE 2008 [20] on the trails calculated with DETware provided by NIST [21].

We compared our method, which is named as "D-WCGAN" (Discriminative WCGAN) in the experiments with a baseline i-vector and PLDA system that does not apply any short-utterance compensation techniques. To demonstrate the contribution of GAN to the performance improvement, we made an extra system, which shared almost the same structure with the proposed GAN but did not contain a discriminator and did not use GAN-related objective function. This system is named as "Single G" in the following part.

4.1.1. Baseline system

The baseline system is the i-vector and PLDA system shown in Section 2. In this system, the input speech segment was first converted to a time series of 60 dimensional feature vectors of Mel-frequency cepstral coefficients (20 dimensional features followed by their first and second derivatives) extracted from a frame of 20ms long and 10ms shift. An i-vector of 400 dimensions was then extracted from the acoustic features using a Gaussian mixture model with 2048 mixture components as a universal background model (UBM) and a total variability matrix (TVM). Length normalization was applied to i-vectors as a preprocessing step before being sent to the PLDA model. Kaldi speech recognition toolkit [22] was used to run these steps.

The UBM, the TVM, and PLDA models were all gender-dependent and trained with SRE08's development data, which contains the NIST SRE2004-2006 data, Switchboard, and Fisher corpus. This dataset as a whole consistes 34,925 utterances from 7,275 male speakers.

4.1.2. Proposed GAN

The training data of GAN is a subset of SRE08's development set mentioned above and SRE08's training set, which contains 1,986 male speakers in total. To make the short and long utterance pairs mentioned in Section 4, we used a sliding window of 20s long and 10s shift to cut one long utterance into short utterances. The UBM, TVM for extracting i-vectors are the same as the one used in the baseline system. Finally, we got 331,675 i-vector pairs for GAN training. The activation function of hidden layers in the proposed GAN, if not specified, is a leaky ReLU [23] with an alpha value set to 0.3. As

Table 1: *The speaker verification results in terms of EER (%) on all the three conditions of the SRE08 "short2-10sec" male trail list.*

	EER (%)			
System	**Cond. 6**	**Cond. 7**	**Cond. 8**	**Average**
a) Baseline	7.28	6.15	6.06	6.50
b) Single G	10.04	8.85	8.33	9.07
c) a + b	7.28	5.77	6.06	6.37
d) D-WCGAN	9.45	8.08	8.33	8.62
e) a + d	**6.89**	**5.77**	**5.30**	**5.99**

Table 2: *The speaker verification results in terms of EER (%) on all the three conditions of the SRE08 "10sec-10sec" male trail list.*

	EER (%)			
System	**Cond. 6**	**Cond. 7**	**Cond. 8**	**Average**
a) Baseline	11.97	10.32	9.60	10.63
b) Single G	15.32	13.89	12.00	13.77
c) a + b	11.16	10.71	9.60	10.49
d) D-WCGAN	15.42	13.89	13.60	14.30
e) a + d	**10.75**	**8.73**	**8.80**	**9.43**

mentioned above, G generates an i-vector and G_{sup} predicts its speaker label. The input layer of G contains 450 nodes to accept the 400-dimension i-vectors and random noise vectors of 50 dimensions, followed by three hidden layers with 512 nodes. G's output layer has 400 nodes, which holds the generated i-vector. The activation function for the output layer of G is tanh. G_{sup} has one hidden layer, which contains 1,986 nodes. Output layer of G_{sup} also have 1,986 nodes and the activation function of each node is softmax.The random noise vectors were sampled from a Gaussian distribution with zero mean and standard deviation 0.5. D has four hidden layers and its input layer has 800 nodes, which accepts two concatenated i-vectors. Output layer of D has only one node with a linear activation function. As we used the WGAN structure, weight clipping is done on D, where the clipping range is -0.01 to 0.01.

We used the Tensorflow library [24] for our neural networks implementation. The networks were optimized using RMSProp [25] with a mini-batch of 64 samples. The learning rate was set to 0.0001. For G training, we set the value of a, b, c as 4, 7, 1, respectively.

In the testing phase, for the "short2-10sec" condition, an i-vector extracted from an utterance in the testing set are transformed by G, then PLDA scoring is done on the i-vector extracted from the enrollment set and the transformed i-vectors. At last, score-wise fusion is done between the baseline system and the proposed method. For the "10sec-10sec" case, almost all the steps are the same as the former one, but the i-vectors from both the enrollment and the testing set are transformed by G. The i-vector extractor and PLDA model are the same as those used in the baseline system.

4.2. Results

Table 1 shows the EERs of the "short2-10sec" condition of NIST SRE 2008. The average EER of our proposed method was 5.99%, and it outperformed that of the baseline i-vector PLDA system, 6.50%. The reduction of average EER is 7.85%. Table 2 shows the EERs of "10sec-10sec" condition of NIST SRE 2008. The average EER of our proposed method was 9.43%, and it outperformed the 10.63% of baseline, and the reduction of average EER is 11.29%. Although our method alone did not outperform the baseline system, it achieved better results when the score-wise fusion was done with the baseline method. We found that the best results was achieved when the score weight ratio of baseline system and our method is 7:3. Table 3 shows the minDCF of the Condition 6 of "short2-10sec" and "10sec-10sec" sets. The minDCF of our method is 1.33% worse than the baseline's in "short2-10sec", but 5.61% better in "10sec-10sec". These results showed that our proposed method can make i-vectors more reliable in most cases. However, in current stage, the amount of training data for the GAN is not enough, even smaller than the amount of PLDA's training data. If we have more training data for the GAN, the performance of the proposed methods may become much better.

Table 3: *The speaker verification results in terms of minDCF on Condition 6 of the SRE08 "short2-10sec" and "10sec-10sec" male trail lists.*

	minDCF	
System	**short2-10sec**	**10sec-10sec**
a) Baseline	**0.370**	0.553
b) Single G	0.494	0.717
c) a + b	0.391	0.540
d) D-WCGAN	0.454	0.678
e) a + d	0.375	**0.522**

Regarding the importance of GAN, our results (b, c in Table 1, 2 and 3) showed that performance became worse, but slightly better than the baseline system in EER, when D was absent. This fact demonstrates the contribution of GAN.

5. Conclusions

This paper has proposed a GAN-based speaker feature restoration method for speaker verification using short utterances. The generator is trained to transform an unreliable i-vector extracted from a short utterance to a reliable i-vector which can be extracted from a long utterance. Speaker labels are also used in the training of GAN to improve the speaker discriminative ability of generated i-vectors. The evaluation results on NIST SRE 2008 task show that our proposed method improved the performance, especially when only short utterances are available for enrollment and testing.

Our future work includes collecting more data for GAN training, as well as applying the GAN-based framework to other cases when i-vectors become unreliable, for example, noise exists in utterances. In addition, we plan to make the discriminator network able to determine whether two given i-vectors are from one speaker or not, so that we can use the GAN model as a back-end for the text-independent speaker verification system.

6. Acknowledgment

This work was supported by JSPS KAKENHI 16H02845 and by JST CREST Grant Number JPMJCR1687, Japan.

7. References

[1] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[2] S. J. D. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *Proc. of ICCV*, 2007, pp. 1–8.

[3] S. Ioffe, "Probabilistic linear discriminant analysis," in *Proc. of ECCV*, 2006, pp. 531–542.

[4] P. Kenny, "Bayesian speaker verification with heavy-tailed priors." in *Proc. of Odyssey Speaker and Language Recognition Workshop*, 2010, pp. 28–33.

[5] A. Kanagasundaram, R. Vogt, D. Dean, S. Sridharan, and M. Mason, "I-vector based speaker recognition on short utterances," in *Proc. of Interspeech*, 2011, pp. 2341–2344.

[6] A. Kanagasundaram, D. Dean, S. Sridharan, J. Gonzalez-Dominguez, J. Gonzalez-Rodriguez, and D. Ramos, "Improving short utterance i-vector speaker verification using utterance variance modelling and compensation techniques," *Speech Communication*, vol. 59, pp. 69–82, 2014.

[7] I. H. Yang, H. S. Heo, S. H. Yoon, and H. J. Yu, "Applying compensation techniques on i-vectors extracted from short-test utterances for speaker verification using deep neural network," in *Proc. of ICASSP*, 2017, pp. 5490–5494.

[8] B. Vesnicer, J. Gros, S. Dobrisek, and V. truc, "Incorporating duration information into i-vector-based speaker-recognition systems," in *Proc. of Odyssey Speaker and Language Recognition Workshop*, 2014, pp. 241–248.

[9] P. Kenny, T. Stafylakis, P. Ouellet, M. J. Alam, and P. Dumouchel, "PLDA for speaker verification with utterances of arbitrary duration," in *Proc. of ICASSP*, 2013, pp. 7649–7653.

[10] W. W. Lin, M. W. Mak, and J. T. Chien, "Fast scoring for PLDA with uncertainty propagation via i-vector grouping," *Computer Speech & Language*, vol. 45, pp. 503–515, 2017.

[11] H. Yamamoto and T. Koshinaka, "Denoising autoencoder-based speaker feature restoration for utterances of short duration," in *Proc. of Interspeech*, 2015, pp. 1052–1056.

[12] J. Villalba, N. Brümmer, and N. Dehak, "Tied variational autoencoder backends for i-vector speaker recognition," in *Proc. of Interspeech*, 2017, pp. 1004–1008.

[13] S. Mahto, H. Yamamoto, and T. Koshinaka, "I-vector transformation using a novel discriminative denoising autoencoder for noise-robust speaker recognition," in *Proc. of Interspeech*, 2017, pp. 3722–3726.

[14] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *IEEE Spoken Language Technology Workshop (SLT)*, 2016, pp. 165–170.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. of NIPS*, 2014, pp. 2672–2680.

[16] M. Mirza and S. Osindero, "Conditional generative adversarial nets," arXiv preprint arXiv:1411.1784, 2014.

[17] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. of CVPR*, 2017, pp. 1125–1134.

[18] S. Pascual, A. Bonafonte, and J. Serrà, "Segan: Speech enhancement generative adversarial network," in *Proc. of Interspeech*, 2017, pp. 3642–3646.

[19] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," arXiv preprint arXiv:1701.07875, 2017.

[20] "The NIST year 2008 speaker recognition evaluation plan," 2018. [Online]. Available: https://www.itl.nist.gov/iad/mig/tests/spk/2008/index.html

[21] "Detware v2.1." [Online]. Available: https://www.nist.gov/file/65996

[22] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *IEEE workshop on automatic speech recognition and understanding*, 2011.

[23] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. of ICML*, vol. 30, no. 1, 2013, pp. 3–8.

[24] "Tensorflow." [Online]. Available: https://www.tensorflow.org

[25] T. Tieleman and G. Hinton, "RMSProp: Divide the gradient by a running average of its recent magnitude." COURSERA: Neural Networks for Machine Learning, 2012.