INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Pair-wise Distance Metric Learning of Neural Network Model for Spoken Language Identification

Xugang Lu[1], *Peng Shen*[1], *Yu Tsao*[2], *Hisashi Kawai*[1]

[1] National Institute of Information and Communications Technology, Japan
[2] Research Center for Information Technology Innovation, Academic Sinica, Taiwan

xugang.lu@nict.go.jp

Abstract

The i-vector representation and modeling technique has been successfully applied in spoken language identification (SLI). In modeling, a discriminative transform or classifier must be applied to emphasize variations correlated to language identity since the i-vector representation encodes most of the acoustic variations (e.g., speaker variation, transmission channel variation, etc.). Due to the strong nonlinear discriminative power of neural network (NN) modeling (including its deep form DNN), the NN has been directly used to learn the mapping function between the i-vector representation and language identity labels. In most studies, only the point-wise feature-label information is feeded to NN for parameter learning which may result in model overfitting, particularly when with limited training data. In this study, we propose to integrate pair-wise distance metric learning in NN parameter optimization. In the representation space of nonlinear transforms of hidden layers, a distance metric learning is explicitly designed for minimizing the pair-wise intra-class variation and maximizing the inter-class variation. With the distance metric as a constraint in the point-wise learning, the i-vectors are transformed to a new feature space which are much more discriminative for samples belonging to different languages while are much more similar for samples belonging to the same language. We tested the algorithm on a SLI task, encouraging results were obtained with more than 20% relative improvement on identification error rate.

Index Terms: Deep neural network, cross-entropy, pair-wise distance metric learning, spoken language identification.

1. Introduction

The i-vector representation and modeling technique has been successfully applied in speaker recognition and spoken language recognition [1]. One of the advantages of using the i-vector representation is that speech utterances with various durations can be represented as fixed-length feature vectors. It can be regarded as a middle level representation between Gaussian mixture model (GMM) based super-vector and MFCC feature representation. The i-vector representation encodes most of the acoustic variations which is convenient for classification modeling. Since the acoustic variations include many factors, e.g., speaker and transmission channel variations, a discriminative transform must be applied to remove uncorrelated variations while emphasizing discriminative variations for different task [2][3]. For speaker recognition task, the discriminative transform must emphasize the feature variations correlated to speaker identity. For spoken language identification (SLI) task, the discriminative transform must emphasize the feature variations correlated to language identity. In this study, the later task is focused on.

Conventionally, a linear discriminant analysis (LDA) based transform is applied on the i-vectors to obtain discriminative features. The transformed feature can be modeled with many types of classifiers for SLI, e.g., Gaussian mixture model (GMM), support vector machine (SVM), probabilistic linear discriminant analysis (PLDA) [4][5]. Algorithm of artificial neural network (NN) (including its deep form, i.e., deep neural network (DNN)), has showed its dominant power for feature learning and classification in image processing and speech recognition [6][7][8]. It has also been used in speaker recognition and spoken language recognition [9][10][11]. The DNN modeling can automatically explore the nonlinear feature variations related to the classification task. In most studies for using DNN for SLI, two modeling methods are adopted [10]. One is using DNN as a discriminative classifier to directly map the i-vector representation (or other acoustic feature representations) to their language IDs, i.e., "direct method". The other is using DNN as a front-end processing for feature learning (e.g., bottleneck feature and i-vector extraction), then modeling the extracted feature with another classifier, i.e., "indirect method". In this study, we focus on the "direct method" of using DNN.

In conventional algorithms, the DNN model parameters are optimized with point-wise training based on the principle of minimizing an objective function measuring the difference between predicted language labels and true target labels (e.g., cross-entropy) [11]. This idea is directly inspired by the DNN acoustic modeling in automatic speech recognition (ASR). In DNN acoustic modeling, a large quantity of labeled training data must be provided, for example, hundreds of hours of acoustic data is labeled in training, i.e., a large quantity of samples for each class label is provided. In SLI for a small data task, only a limited number of samples (e.g., number of hundreds) for each language is provided in training. In this case, the model is easily over-fitted since it is optimized on the training data set. The over-fitted DNN model will lose its strong capacity in classification task which results in bad performance for a testing data set (i.e., weak generalization). In this study, rather than only using point-wise feature-label information in DNN parameter training, we investigate the possibility of feeding much more information of training data set to improve the DNN model generalization ability.

In a training data set, besides point-wise feature-label information which is widely used in supervised learning, other information may provide auxiliary information for robust modeling, for example, training data geometric structure, pattern clustering property etc. In this study, besides using feature-label information for supervised training of DNN, the similarity or distance measure of pair-wise samples is taken into con-

Copyright © 2016 ISCA

http://dx.doi.org/10.21437/Interspeech.2016-722

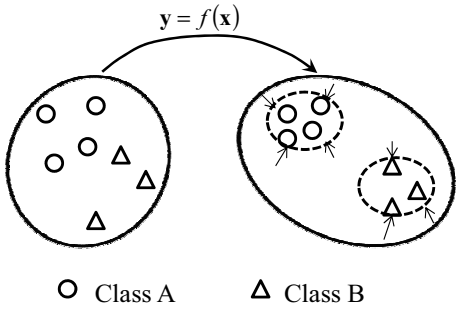

Figure 1: Distance metric learning with local-push transform to reduce intra-class variation, and local-pull transform to increase inter-class variation.

sideration. Learning with consideration of pair-wise distance or similarity structure belongs to a large category of machine learning, i.e., metric learning [12][13][14]. For example, in linear metric learning (either distance metric or similarity metric) [12], the Mahalanobis distance metric is learned on the input feature space in order to measure the similarity of a pair of input samples. With the DNN framework, nonlinear distance metric learning has been proposed in face recognition and re-identification [15][16]. In most metric learning studies, the discriminative transform is explicitly optimized based on an objective function which is supposed to reduce intra-class variation while increasing iter-class variation. Fig. 1 gives an illustration of this process. As shown in this figure, a distance (or similarity) metric transform $f(\mathbf{x})$ should be learned to "push" samples belonging to the same class to neighboring space while "pull" samples belong to different class to some distance. In most studies, the distance metric is learned with a conventional classifier for classification task, e.g., K-nearest neighbor classifier, support vector machine, etc. In this study, we integrate this metric learning with conventional DNN learning framework for SLI.

The reminder of the is paper is organized as follows. Section 2 introduces the DNN model framework which explicitly integrates pair-wise distance metric learning in model parameter optimization. SLI experiments were carried out in Section 3. Discussion and conclusion were given in Section 4.

2. DNN with pair-wise distance metric learning

In DNN modeling for classification, a softmax layer is often stacked as a classifier layer with normalized probability output. The hidden layers with nonlinear transforms provide discriminative features as input to the classifier. In this sense, the function of a series of nonlinear transforms in those hidden layers can be regarded as a feature metric learning process, i.e., learning a distance metric (or similarity metric) which is suitable for classification. Fig. 2 shows the two stages of DNN framework as distance metric and classification learning. In Fig. 2 (b), as most widely used in DNN learning, the feature-label map is directly learned by minimizing a cross entropy based objective function. In parameter optimization, there is no explicit constraint on how the hidden layer features are learned. In metric learning (as in Fig.2 (a), the transform functions via hidden layers are constrained with a pair-wise loss function. As shown in the figure, two representations from the last hidden layers are obtained from a pair of input vectors. The two input vectors share the same DNN model parameters which is similar as used in Siamese network [17]. In the followings, the point-wise feature-label learning and pair-wise distance metric learning are introduced.

2.1. Learning point-wise feature-label mapping

For classification, the DNN framework can be directly used for learning feature-label mapping as conventionally used. For a DNN with $K-1$ hidden layers, the output of a hidden layer is represented as

$$\mathbf{h}^k = \mathbf{f}^k\left(\mathbf{W}^k\mathbf{h}^{k-1} + \mathbf{b}^k\right), \tag{1}$$

where $k = 1, ..., K-1$, $\mathbf{h}^0 = \mathbf{x}$ is the input layer with feature vector $\mathbf{x}$ (i-vector as used in this paper). $\mathbf{W}^k$ and $\mathbf{b}^k$ is the neural weight matrix and bias of the k-th hidden layer, respectively. $\mathbf{f}^k(.)$ is a nonlinear active function (element-wise transform), e.g., sigmoid function, tanh function, Rectified Linear Units (ReLU) [18], etc. In this study, a tanh function was used as

$$f^k(z) = \tanh(z) = \frac{\exp(z) - \exp(-z)}{\exp(z) + \exp(-z)} \tag{2}$$

For an input feature vector $\mathbf{x}$, a predicted label is obtained from the final output layer which is defined as a softmax layer as

$$\begin{aligned}\hat{y}_j &= p(y_j = 1|\mathbf{x}, \mathbf{W}, \mathbf{b}) \\ &= \frac{\exp\left(\mathbf{W}_j^K\mathbf{h}_j^{K-1} + \mathbf{b}_j^{K-1}\right)}{\sum_{i=1}^{\#Class}\exp\left(\mathbf{W}_i^K\mathbf{h}_i^{K-1} + \mathbf{b}_i^{K-1}\right)}\end{aligned} \tag{3}$$

where y_j is the output of the j-th neuron in the softmax layer, "$\#Class$" is the total number of classes. For learning the model parameters, an objective function defined as cross entropy (CE) between the predicted and true target labels is used as

$$l(\Theta) = \sum_{i=1}^{\#Sample} CE(\mathbf{y}_i, \hat{\mathbf{y}}_i) = -\sum_{i=1}^{\#Samples}\sum_{j=1}^{\#Class} y_{i,j}\log\hat{y}_{i,j} \tag{4}$$

where $\hat{y}_{i,j}$ and $y_{i,j}$ are the elements of predicted target and true target vector of $\hat{\mathbf{y}}_i$ and $\mathbf{y}_i$, respectively. i and j are the index of sample and class number, respectively. "$\#Samples$" is the total number of training samples.

The learning of the DNN parameters is based on minimizing an objective function defined on the cross entropy of a training data set as

$$\begin{aligned}\Theta^* &= \arg\min_{\Theta} C(\Theta) \\ C(\Theta) &= l(\Theta) + \lambda R(\Theta)\end{aligned} \tag{5}$$

where $\Theta = \left\{\mathbf{W}^k, \mathbf{b}^k, k = 1, 2, ..., K\right\}$ is the DNN parameter set. λ is a regularization coefficient to control the tradeoff between the cross entropy based loss and parameter regularization $R(\Theta)$. In most studies, parameter regularization is defined as smoothness or sparseness of the model parameter space (e.g., either L_1 or L_2 regularization) which is proved to improve the generalization ability of the DNN modelling.

In learning, the stochastic gradient descent (SGD) algorithm is used. From the transforms and objective function Eqs. (1)-(5), we can see that the learning tries to find a local optimal solution to approximate the feature-label mapping on the training data set. In order to find a better solution, constraints must be given on the learned transform functions. In this study, a pair-wise distance metric on the representation space obtained from

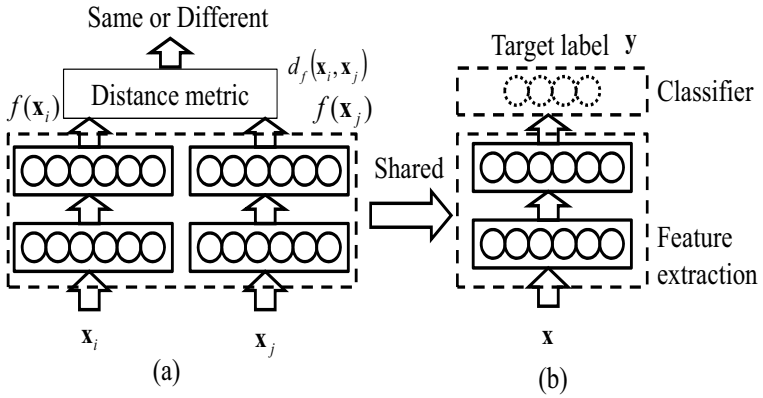

Figure 2: Deep neural network learning (a) pair-wise distance metric learning in hidden layers, (b) point-wise classifier learning by softmax layer.

the last hidden layer output is added as a constraint in DNN parameter learning.

2.2. Pair-wise nonlinear distance metric learning

The basic principle for metric learning is to find a transform function by which the distance of two samples from the same class to be close while far for samples from different classes, i.e., reducing intra-class variation and increasing inter-class variation [12]. In our DNN framework, we explicitly add this property to control the nonlinear transform function realized by hidden layer transforms.

For a pair of samples $\mathbf{x}_i$ and $\mathbf{x}_j$, in a transformed space from a hidden layer output, their distance can be defined as Euclidean distance or cosine distance. As in most studies for i-vector based spoken language recognition, the cosine distance metric is widely used. In this study, the cosine distance metric is also used. In a transformed space with $\mathbf{f}^k(\cdot)$, the cosine distance metric is defined as

$$d_{\mathbf{f}^k}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{f}^k(\mathbf{x}_i)^T \mathbf{f}^k(\mathbf{x}_j)}{\|\mathbf{f}^k(\mathbf{x}_i)\|_2^1 * \|\mathbf{f}^k(\mathbf{x}_j)\|_2^1} \tag{6}$$

This metric measures the angles of two vectors which is a similarity measure as widely used in vector space modeling (VSM) [19]. It has maximum 1 (angle 0) and minimum values -1 (angle π). Therefore, the values of pair-wise distance are distributed between $[-1, 1]$.

For convenience of analysis, for a training data set with feature vector $\mathbf{x}_i$ and label vector $\mathbf{y}_i$ (one-hot encoding vector), $i = 1, 2, ...$, we define two data sets S and D of pair-wise samples as follows:

$$\begin{array}{l} S = \{(\mathbf{x}_i, \mathbf{x}_j) \,|\forall \mathbf{y}_i = \mathbf{y}_j\} \\ D = \{(\mathbf{x}_i, \mathbf{x}_j) \,|\forall \mathbf{y}_i \neq \mathbf{y}_j\} \end{array} \tag{7}$$

i.e., data sets S and D consist of pair-wise data samples with labels belonging to the same and different classes, respectively.

Based on the basic principle of metric learning, two loss functions are defined on the pair-wise data sets as:

$$\begin{array}{l} J_{\text{Intra}}(\Theta) = \frac{1}{\#S} \sum\limits_{(\mathbf{x}_i, \mathbf{x}_j) \in S} (d_{\mathbf{f}^k}(\mathbf{x}_i, \mathbf{x}_j) - 1)^2 \\ J_{\text{Inter}}(\Theta) = \frac{1}{\#D} \sum\limits_{(\mathbf{x}_i, \mathbf{x}_j) \in D} (d_{\mathbf{f}^k}(\mathbf{x}_i, \mathbf{x}_j) + 1)^2 \end{array} \tag{8}$$

"$\#S$" and "$\#D$" are the number of sample pairs in sets S and D, respectively. In these equations, minimizing $J_{\text{Intra}}(\Theta)$ could decrease the pair-wise intra-class variation, and minimizing $J_{\text{Inter}}(\Theta)$ could increase the pair-wise inter-class variation.

Considering the tradeoff between the robustness and discrimination, the objective function for pair-wise metric learning is formulated as follows:

$$J(\Theta) = J_{\text{Intra}}(\Theta) + \alpha J_{\text{Inter}}(\Theta) \tag{9}$$

where α controls the tradeoff between these two loss functions. For equal weighting of each pair-wise loss, the metric learning is based on minimizing the following objective function as

$$J(\Theta) = \frac{1}{\#\{S \cup D\}} \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in \{S \cup D\}} (d_{\mathbf{f}^k}(\mathbf{x}_i, \mathbf{x}_j) - t_{i,j})^2, \tag{10}$$

where "$\#\{S \cup D\}$" is number of sample pairs in sets S and D, and the pair-wise label $t_{i,j}$ is defined as:

$$t_{i,j} = \begin{cases} 1, \forall (\mathbf{x}_i, \mathbf{x}_j) \in S \\ -1, \forall (\mathbf{x}_i, \mathbf{x}_j) \in D \end{cases} \tag{11}$$

2.3. Minimizing cross entropy with pair-wise distance metric learning

In our proposed DNN modeling, pair-wise distance metric learning was integrated with the point-wise cross-entropy learning. The pair-wise distance metric can be learned from each hidden layer of DNN. In our study, we only consider the distance metric learning from the last hidden layer. The objective function in DNN parameter learning is formulated as

$$\begin{array}{l} \Theta^* = \arg\min\limits_{\Theta} M(\Theta) \\ \text{with } M(\Theta) = l(\Theta) + \gamma J(\Theta) + \lambda R(\Theta) \end{array} \tag{12}$$

In this objective function, the first term $l(\Theta)$ is the point-wise cross entropy between predicted and true labels, the second term $J(\theta)$ is the objective function for distance metric learning.

In learning the model parameters, we suppose the parameter set related to distance metric learning and classifier (softmax layer) as $\Theta = \{\Theta_F, \Theta_C\}$, then the gradients of them are calculated as follows:

$$\begin{array}{l} \nabla\Theta_F = \frac{\partial M}{\partial \Theta_F} = \frac{\partial l}{\partial \Theta_F} + \gamma \frac{\partial J}{\partial \Theta_F} + \lambda \frac{\partial R}{\partial \Theta_F} \\ \nabla\Theta_C = \frac{\partial M}{\partial \Theta_C} = \frac{\partial l}{\partial \Theta_C} + \lambda \frac{\partial R}{\partial \Theta_C} \end{array} \tag{13}$$

In this formulation, Θ_F represents the DNN parameter set (neural connection weights and bias) except the softmax layer, and Θ_C is the DNN parameter set only associated to softmax layer. From these equations, we can see that in feature metric parameter Θ_F learning, besides the gradient estimated from loss function of cross-entropy, the gradient is explicitly regularized with the pair-wise loss function $J(\Theta)$, while the classifier layer parameter Θ_C learning, the gradient is calculated from the derivative of the cross entropy of all training samples.

3. Experiments

In this study, rather than exploring all possible ways ("direct" and "indirect" methods as introduced in section one) to obtain the best results for SLI, we examine whether integrating pair-wise metric learning in a unified DNN model could improve the performance or not. In this section, we test the algorithm on spoken language identification task.

For simply integrating utterance based acoustic variations in modeling, i-vector feature is used as input of the DNN modeling. A data set from NIST i-vector challenge for SLI is used in this paper [20]. 50 types of languages are included in the data set, and each utterance in the data set is represented as a 400-dimension i-vector. The data set was reorganized into three subsets for training, validation and test. 10k samples (200 samples for each language) and all their pair-wise combinations (10k*(10k-1)/2) are used in training. 2.5k samples were used in validation, and another 2.5k samples are used in testing (50 samples for each language).

In building the baseline DNN models, two types of architectures were used as $400 - L * 512 - 50, L = 1, 2$, i.e., 400 dimension of input i-vector, L hidden layers with 512 neurons for each layer, and 50 neurons in output layer corresponding to 50 language IDs. The network was first layer-wised pre-trained as restricted Boltzmann machine (RBM) with contrastive divergence algorithm [21]. In fine tuning using stochastic gradient descendent algorithm, mini-batch size was 128, and learning rate was 0.0002. In the implementation, neural connection weights were penalized with L_2 regularization (with coefficient 0.001). The result for test set was obtained when the performance on validation data set reached the best in a total of 500 epoches. The results are shown in Table 1. In this table, "M1*512" and "M2*512" denote the NN model with one and two hidden layers, respectively. "LDA+SVM" represents the conventional linear modeling technique, i.e., LDA for feature transform and SVM for classification. From this table, we can see that on this small task, the performance of linear model is much better than the plain neural network model. For neural network model, with adding more hidden layers, there is no significant increase in performance on testing data set although the training error continuously decreased. But in using the back propagation for finding a local solution, if the network is regularized with a proper constraint, it is possible to use a deep network for finding a solution with good performance for both training and testing data sets. In the following, we implemented the proposed pair-wise metric learning on the two models. In the implementation, Eqs. (10) and (11) were used. As shown in Eq. 12, with different regularization parameter γ, the tradeoff between feature metric learning and classifier learning is controlled. We first show the performance when varying the regularization parameter in metric learning, the results are shown in Table 2 (for M2*512). From the results, we can see that with increasing of the regularization of pair-wise loss, the error for training data set was increased, but the performance for validation and test data sets were improved significantly. When γ is around 0.01, we obtained the best performance on the testing data set. These results suggest that with the pair-wise metric learning as constraint, the performance on testing data set was improved.

Table 1: Performance of point-wise training for DNN baseline systems and LDA+SVM system (identification error rate in %)

Model	Train	Validation	Test
M1*512	0.53	18.59	**18.34**
M2*512	0.10	18.87	**18.32**
LDA+SVM [2]	-	-	**16.28**

Table 2: Performance of DNN system with pair-wise metric learning with varying of regularization coefficient (identification error rate in %)

Coef γ	Train	Validation	Test
0	0.10	18.87	**18.32**
0.001	0.85	17.43	**16.32**
0.005	1.13	16.92	**15.75**
0.01	2.89	15.50	**14.42**
0.03	2.60	15.42	**15.12**
0.05	3.55	16.38	**16.00**

By adjusting the regularization parameter γ for the two NN model architectures, we obtained the best performance of each model, and show the results in Table 3. In this table, the column with "Rel." shows the relative improvement compared with their baseline models (in table 1), respectively. From these results, we can find that all the models benefit from the pair-wise loss constraint with significant improvements.

Table 3: Performance of DNN systems with pair-wise metric learning (identification error rate in %)

Model	Train	Validation	Test	Rel.
M1*512	4.18	16.00	14.67	**20.01**
M2*512	2.89	15.50	14.42	**22.64**

4. Discussion and conclusion

The pair-wise metric learning gives constraint on the DNN representation space with a certain structure as similarly used in Siamese neural network (SNN). In conventional DNN parameter learning, all the parameters are learned based on an objective function measuring the loss between the predicted and true target labels. Our parameter learning algorithm explicitly takes pair-wise distance metric learning as a constraint which results in a model with better generalization. We have tested the pair-wise loss as regularization on cross-entropy training of DNN for a SLI task, and showed encouraging improvement.

In this study, only the results of the regularization framework with metric learning on the last hidden layer was showed. Further improvement was obtained with adding distance metric learning on all hidden layers. In addition, we have also tested with the metric learning as a pretraining step for DNN without a softmax layer using pair-wise data, then fine tuned the DNN with stacking a softmax layer with point-wise training. On the SLI task as used in this study, there was improvement (but not large) compared with the regularization framework as in section 2.3. In the future, we will further investigate different integration strategy in our task.

5. 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, pp. 788-798, 2011.

[2] P. Shen, X. Lu, L. Liu, H. Kawai, "Local Fisher discriminant analysis for spoken language identification," *ICASSP*, 2016.

[3] M. Sugiyama, "Local Fisher discriminant analysis for supervised dimensionality reduction," in *Int. Conf. ICML*, pp. 905-912, 2006.

[4] N. Dehak, P. Torres-Carrasquillo, D. Reynolds, R. Dehak, "Language Recognition via I-vectors and Dimensionality Reduction," *Interspeech*, pp. 857-560, 2011.

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

[6] 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.

[7] D. Yu, L. Deng, L., Automatic Speech Recognition A Deep Learning Approach, Springer-Verlag London, 2015

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

[9] R. Richardson, D. Reynolds, and N. Dehak, "A Unified Deep Neural Network for Speaker and Language Recognition," in *Proc. Interspeech*, pp. 1146-1150, 2015.

[10] 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.

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

[12] E. Xing, A. Ng, M. Jordan, and R. Russell, "Distance Metric Learning, with application to Clustering with side-information," *Advances in Neural Information Processing Systems* 16, MIT Press, pp. 521-528, 2002.

[13] K. Weinberger, J. Blitzer, L. Saul, "Distance Metric Learning for Large Margin Nearest Neighbor Classification," *Advances in Neural Information Processing Systems* 18, pp. 1473-1480, 2006.

[14] K. Weinberger, L. Saul, "Distance Metric Learning for Large Margin Classification," *Journal of Machine Learning Research* 10, pp. 207-244, 2009.

[15] M. Guillaumin, J. Verbeek, C. Schmid, "Is that you? Metric learning approaches for face identification," *the IEEE 12th International Conference on Computer Vision*, pp. 498-505, 2009.

[16] J. Hu, J. Lu, Y. Tan, "Deep Transfer Metric Learning," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.325-333, 2015.

[17] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively with application to face verification," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. 539-546, 2005.

[18] V. Nair, and G. Hinton, "Rectified Linear Units Improve Restricted Boltzmann Machines," *the 27th International Conference on Machine Learning*, pp. 807-814, 2010.

[19] G. Sidorov, A. Gelbukh, H. Gomez-Adorno, D. Pinto, "Soft Similarity and Soft Cosine Measure: Similarity of Features in Vector Space Model," *Computacion y Sistemas*, vol. 18, no. 3, pp. 491-504, 2014.

[20] https:lre.nist.gov

[21] G. Hinton, "A Practical Guide to Training Restricted Boltzmann Machines," *UTML TR 2010-003*, University of Toronto.