Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 147 (2019) 36–43

Procedia Computer Science

www.elsevier.com/locate/procedia

2018 International Conference on Identification, Information and Knowledge in the Internet of Things, IIKI 2018

I-vector features and deep neural network modeling for language recognition

Wei Wang*, Wenjie Song, Chen Chen, Zhaoxin Zhang, Yi Xin

School of Computer Science and Technology, Harbin Institute of Technology, 92 West Dazhi Street, Nan Gang District, Harbin, Heilongjiang province, Harbin, 150001, China

Abstract

We combine Total Variability algorithm with Deep Learning theory to complete the language recognition task. The Total Variability algorithm can compensate for the influence of differences in channels and speakers among various languages, while deep learning methods have a stronger ability of nonlinear modeling compared with traditional statistical models. In this paper, I-vector feature is extracted using Total Variability algorithm, and model training is established using fully connected neural network. Meanwhile, the dropout strategy is also used to suppress overfitting. The experimental results show that the new system outperforms the baseline system on the NIST LRE 2007 corpus.

Keywords: language recognition, deep neural network, feature extraction

1. Introduction

With the improvement of the accuracy of speech recognition, speech recognition technology will further promote the revolution of internet of things, which provides great convenience for automobiles, household appliances and wearable products. Language Recognition (LR) is branch of Speech Recognition, which is a technology to determine which language the input speech belongs to. The language recognition system consists of three phases

* Corresponding author. Tel.: +86-0631-5687506; fax: +86-0631-5687506.
E-mail address: wangwei_hitwh@126.com

10.1016/j.procs.2019.01.181

including feature extraction, model training and model testing. In the feature extraction phase, i-vector feature is extracted using Total Variability (TV) algorithm [1-3], which is one of the mainstream methods in the field of language recognition for its good performance [4-6]. It is based on the task-independent redundant information contained in underlying acoustic features, using a low-dimensional vector, which is called I-vector, to represent the difference between Gaussian mean supervectors of each speech segment. Then the discriminative dimensionality reduction method is used to compensate for the influence caused by differences in channels, speakers among various languages, and thereby the low-dimensional discriminative representation of each speech segment is obtained. I-vector has many advantages, for example, the dimension of I-vector is a fixed value, without considering the original speech information with variable length. Meanwhile, the use of I-vector facilitates the modeling and testing process in language recognition, so that Nuisance Attribute Projection (NAP)[7],Within-Class Covariance Normalization (WCCN)[8], Probabilistic Linear Discriminant Analysis (PLDA)[9], Softmax Regression[10] and other technologies can be combined into the I-vector system. In the model training phase, Deep Learning[11,12] is a hot orientation in machine learning field in recent years[13]. It is a collection of modeling techniques that use multilayer nonlinear transformation to extract and generalize high-level information implicit in data. Deep Neural Network (DNN)[14-16] is a network system composed of a large number of neurons through extensive interconnection. It has the basic characteristics of biological nervous system. As a kind of simulation of biological systems, it has many advantages like massive parallel, distributed processing, self-organization, self-learning, etc. Therefore, it has been widely used in many fields such as speech analysis, pattern recognition and computer vision, and has achieved many outstanding results. In several neural network structures, Recurrent Neural Network (RNN), Convolutional Neural Network (CNN) and Fully Connected Neural Network have better performance. RNN is suitable for proceed the temporal sequence data, and CNN is suitable for deal with high dimensionality data. In this paper, we use fully connected neural network, which has significant performance advantages over traditional linear classification models, because i-vector includes only a few hundred dimensions. In addition, I-vector has a very low dimension compared with acoustic features such as Mel-Frequency Cepstral Coefficient (MFCC) and Shifted Delta Cepstral (SDC)[17], which may lead to overfitting. Thus we use the dropout strategy to suppress this phenomenon[18,19].

2. I-vector feature extraction

Classic factor analysis theory is based on two factors, language and channel factors, and their corresponding two independent spaces: language space and channel space. However, TV model uses only a single total variability space to replace these two separate spaces. Total variability space is defined by a matrix consists of those eigenvectors corresponding to the largest eigenvalues in the covariance matrix. For a given speech segment, the GMM mean supervector based on total variability space can be expressed by the following formula:

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

where M is the GMM supervector for each segment, m is a supervector which independent of language and channel, represented by UBM mean supervector, T is the total variability matrix whose dimension is $CF \times R_i$, where C is the number of single-peak Gaussian in GMM, F is the acoustic feature dimension, and R_i is the number of eigenvectors included in T. In addition, w is the vector whose every element represents a one-dimensional total variability factor. This approach can be seen as a simple factor analysis technique that maps a speech segment into a low-dimensional total variability space.

The specific I-vector extraction process is shown in Fig. 1. MFCC features are extracted from the preprocessed speech segments, further shifted and calculated delta to obtain SDC features. Then, the Gaussian Mixture Model-Universal Background Model (GMM-UBM) and the total variability matrix T are trained using SDC features. Finally, the I-vector of each speech segment can be calculated according to the formula (1). The following is a brief introduction to the training process of GMM-UBM and matrix T.

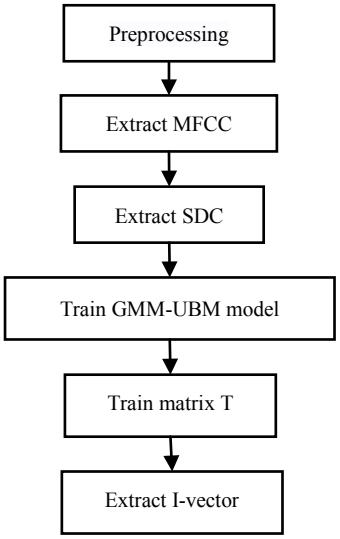

Fig. 1. I-vector extraction process.

2.1. GMM-UBM training process

GMM is a multivariate distributed general probability model widely used in language recognition system. This characteristic makes GMM very suitable for speaker-independent and text-independent language recognition task.

For an F-dimensional feature vector x, its likelihood probability can be described by a mixed density distribution function which is a weighted linear combination of C single-peak Gaussian functions:

$$p(x|\lambda) = \sum_{i=1}^{C} w_i p_i(x) \tag{2}$$

where $\sum_{i=1}^{C} w_i = 1$, $p_i(x)$ is a normal distribution function determined by a F-dimensional mean vector μ and the covariance matrix Σ whose dimension is $F \times F$:

$$p_i(x) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_i)^{\intercal}\Sigma_i^{-1}(x-\mu_i)\right\} \tag{3}$$

In general, a GMM model can be represented by the parameter λ,

$$\lambda = \{w_i, \mu_i, \Sigma_i\}, i = 1, \cdots, C.$$

For a given set of training vectors, Exception Maximization (EM) algorithm can be used to estimate their model parameters. It can adjust the GMM parameters iteratively so that the likelihood probability can be monotonically increasing, that is $p\left(X\middle|\lambda^{(k+1)}\right) > p\left(X\middle|\lambda^{(k)}\right)$. Generally speaking, about 5 iterations can make the parameters converge.

EM algorithm is as follows:

1) E step: For every training vector x_i, calculate the probability of which is generated by the k-th Gaussian component of GMM:

$$\beta(i,k) = \frac{w_k p_k(x_i)}{p(x_i|\lambda)} = \frac{w_k p_k(x_i)}{\sum_{j=1}^{C} w_j p_j(x_i)} \quad k = 1, \cdots, C \tag{4}$$

2) M step: Update the model parameters according to formula (5) to (7):

$$\mu_k = \frac{1}{N_k}\sum_{i=1}^{N} \beta(i,k) x_i \tag{5}$$

$$\Sigma_k = \frac{1}{N_k}\sum_{i=1}^{N} \beta(i,k)(x_i-\mu_k)(x_i-\mu_k)^{\intercal} \tag{6}$$

$$w_k = N_k/N \tag{7}$$

3) Repeat the previous two steps until the model converges to get the GMM-UBM model containing all the language information.

2.2. Matrix T training process

When estimating the matrix *T*, the GMM mean supervector is obtained by calculating the zeroth order, first order, and second order statistics of the UBM mean supervector. The specific calculation process is as follows:

1) Initialization: Assuming that the phonetic feature is $x_{s,t}$ (s is the language and t is the time series), then the zeroth order statistic $N_{c,s}$, the first order statistic $F_{c,s}$, and the second order statistic $S_{c,s}$ can be calculated according to formula (8), (9) and (10):

$$N_{c,s} = \sum_t \gamma_{c,s,t} \tag{8}$$

$$F_{c,s} = \sum_t \gamma_{c,s,t} (x_{s,t} - m_c) \tag{9}$$

$$S_{c,s} = \mathrm{diag}\{\sum_t \gamma_{c,s,t} (x_{s,t} - m_c)(x_{s,t} - m_c)^{\intercal}\} \tag{10}$$

Where m_c means the *c*-th Gaussian component of the UBM mean supervector *m* and $\gamma_{c,s,t}$ is its posterior probability. If the dimension of m_c is *F*, then the dimension of F_s is *FC,* which splicing *C* UBM mean supervectors into one column Meanwhile, N_s is a *FC*×*FC* matrix in which $N_{c,s}$ is expanded into a column and placed in the diagonal of a zero matrix. Similarly, S_s is such a *FC*×*FC* matrix. Then randomly initialize the matrix *T* and use EM algorithm to complete the following calculations process.

2) E step: After fixing T, calculate the expectation of the first and second order statistics of the total variability factor vector w:

$$L_s = I + T^{\intercal}\Sigma^{-1} N_s T \tag{11}$$

$$\mathrm{E}[w_s] = L_s^{-1} T^{\intercal}\Sigma^{-1} F_s \tag{12}$$

$$\mathrm{E}[w_s w_s^{\intercal}] = \mathrm{E}[w_s]\mathrm{E}[w_s^{\intercal}] + L_s^{-1} \tag{13}$$

where L_s is a temporary variable and Σ is the covariance matrix of UBM.

3) M step:Update matrix T:

$$\Sigma_s N_s T E[w_s w_s^{\intercal}] = \Sigma_s F_s \mathrm{E}[w_s] \tag{14}$$

And Update Σ:

$$\Sigma = N^{-1}\Sigma_s S_s - N^{-1}\mathrm{diag}\{\Sigma_s F_s \mathrm{E}[w_s^{\intercal}] T^{\intercal}\} \tag{15}$$

where $N = \sum N_s$ represents the sum of all zero order statistics

4) Repeat steps (2) and (3) for about 5 iterations, and *T* and Σ can be seen as approximately convergent.

Finally, I-vectors can be extracted from all speech segments according to formula (1) and total variability matrix *T*.

3. Fully Connected Neural Network modeling

The neural network used in this paper is a fully connected neural network with Rectified Linear Unit (ReLU) [9] as its activation function. ReLU is essentially a piecewise function, which turns all negative values into 0 and the positive values remain unchanged. This practice is called unilateral inhibition and makes neurons sparsely activated. Its advantage is that pre-training process can be omitted without the problem of gradient vanishing. In addition, ReLU only requires addition and multiplication, thus it is faster and more efficient than other functions like sigmoid and tanh.

The basic unit of the neural network is neuron, and data flow direction in a neuron using ReLU is shown in Fig. 2.

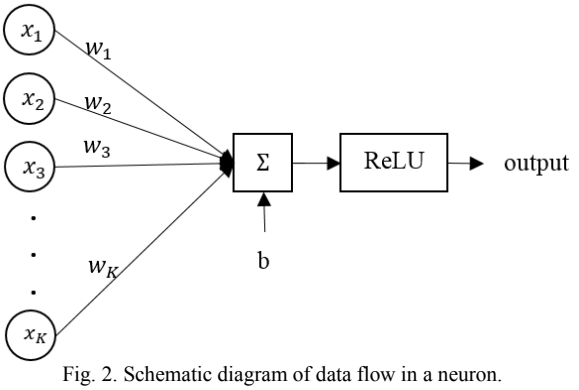

Fig. 2. Schematic diagram of data flow in a neuron.

The calculation formula corresponding to Figure 2 is:

$$y = \mathrm{ReLU}(x) = \max(0, x) \tag{16}$$

$$x = b + \sum_K w_k x_k \tag{17}$$

A large number of neurons form a neural network through extensive interconnection, and its structure is shown in Fig. 3.

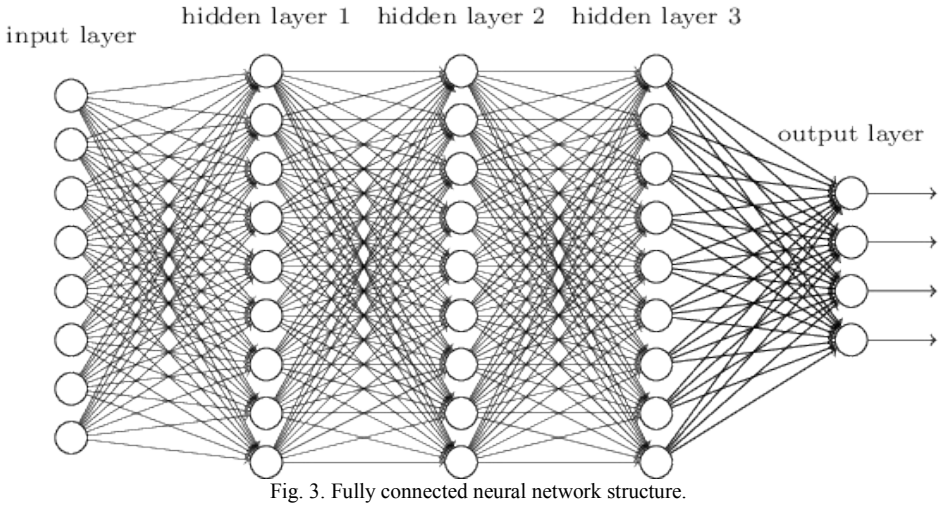

Fig. 3. Fully connected neural network structure.

As the input of the neural network, I-vector feature is best suitable for the linear classifier. However, for neural networks, more number of hidden layers means the weaker degree of linearity. Therefore, we use the fewer hidden layers in this paper. At the same time, when there are too many network parameters while too few training samples, overfitting is very likely to occur. Overfitting means the model has a small loss and a high prediction accuracy on the training data, but a large loss and a low prediction accuracy on the test data. Currently, the dropout strategy is an effective method of suppressing overfitting. Dropout means that when training the neural network makes the activation values of the neurons change to 0 in a certain ratio v, that is, invalidate a part of the hidden layer nodes according to the ratio v. When testing, the output values of the hidden layer nodes need to be reduced to (1-v) times, for example, if the normal output is a, it needs to be reduced to a(1-v).

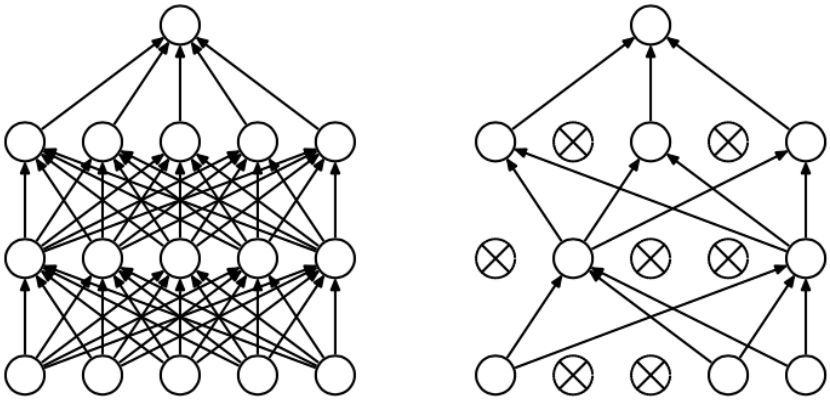

Fig. 4. The strategy of Dropout.

Dropout can be seen as a model averaging method that averages estimates or predictions from different models by a certain weight, which is also referred to as model combination in some literature. In each training process, since the nodes are randomly ignored, the network obtained is different and can be regarded as a "new" model. In addition, nodes are randomly activated with a certain probability, so there is no guarantee that every 2 nodes will be valid together each time, then the update of weights no longer depends on the joint action of nodes with fixed relationships, which prevents the situation where some features are only effective when other features are present.

4. Experimental results

In order to evaluate the performance of the proposed method, experiments are carried out on the NIST LRE 2007 database, which only includes two parts, training set and test set, excluding validation set or development set according to NIST standard. It is a language recognition database with 8 types language, where all utterances are recorded in single channel with 16-bit streams at 8000Hz sampling rate. Since the dialect is not researched in this paper, experimental evaluation are fixed on 4 types standard language which include Arabic, Bengali, Russian and Thai. Each type of language consists of 400 minutes training speech, and three kinds of test set including average durations of 3 second, 10second and 30 second where each class has 80 utterances.

4.1. Experimental setup

Baseline. The experiments operated on cepstral features which are extracted using a 20-ms frames with 10ms overlaps. 56-dimensional SDC(optimal configuration 7-1-3-7) are calculated corresponding to each recording. We use the training data of NIST LRE 2007 database to train the UBM with 1024 Gaussians, the total variability matrix composed of 400 total factors and the i-vecot of 400 dimensionality by Kaldi Identity Toolbox [http://www.kaldi-asr.org/doc/]. Softmax regression is then used as classification.

Proposed method. We extract the same 400-dimensional I-vectors from the TV model whose parameter settings is the same with the baseline system. The fully connected neural network we use in this paper has only one hidden layer with 2048 nodes, and with ReLU as its activation function to better adapt to the special i-vector input, and we set learning rate to 0.001 and batch size to 32 to train the network. In addition, when the ratio of dropout is set to 0.5, the system gets its best classification effect.

4.2. Results and discussions

I-vector which inputted in fully connected neural network is the same as that inputted in baseline system. The experiments are running on 10-second test set. Firstly, The performance is evaluated based on the number of hidden layers on NIST LRE 2007 database. In Table 1, fixed on 128 hidden nodes in each hidden layer, it shows when the number of hidden layers increases, the accuracy decreases. The best performance is obtained with 1 hidden layer, so we use the network with 1 hidden layer. It may be explained that more number of hidden layers does not mean a better performance.

Table 1. Comparison of accuracy about the number of different hidden layers on NIST LRE 2007 database.

Number of layer (*s*)	Accuracy (*%*)
1	79.69
2	63.44
3	56.25
4	52.19

In Table 2, fixed on 128 hidden nodes and 1 hidden layer, the performance is evaluated based on the number of different activation functions on NIST LRE 2007 database. It shows that the ReLU activation function has best performance compared with other activation functions.

Table 2. Comparison of accuracy about the different activation functions on NIST LRE 2007 database.

Activation function	Accuracy (*%*)
ReLU	79.69
Sigmoid	78.44
tanh	64.69
elu	75.00
ReLU6	62.19
SeLU	77.19

Furthermore, the performance of the language recognition is evaluated based on the number of hidden nodes on NIST LRE 2007 database. In Table 3, the experiments show clearly that different numbers of hidden nodes lead to different results. When the number of hidden nodes increases, the accuracy increases. At the number of 2048 hidden nodes, the accuracy gets the best performance. It may be explained that more number of hidden nodes does not mean a better performance.

Table 3. Comparison of accuracy about the number of different hidden nodes on NIST LRE 2007 database.

Number of node	Accuracy (%)
128	79.69
256	79.37
512	83.44
1024	84.06
2048	85.00
4096	82.19

In addition, the performance of the language recognition is evaluated based on the number of Dropout ratios on NIST LRE 2007 database. Several results with various Dropout ratios are tested on the test set as shown in Table 4. The experiments show clearly that different Dropout ratios lead to different results, where accuracy decreases as the Dropout ratio increase, and the performance tend to stabilize at the 0.5 Dropout ratio**.**

Table 4. Comparison of accuracy about the different Dropout ratios on NIST LRE 2007 database.

Dropout ratio	Accuracy (%)
1.0	87.81
0.9	89.06
0.8	90.00
0.7	90.00
0.6	90.31
0.5	92.19

Based on the upper result, we select a single hidden layer neural network with 2048 nodes as the fully connected neural network, use the ReLU activation function and set 0.5 Dropout ratio. The results are shown in Fig. 5, where the lines are used to describe the EER of baseline system (16.67%) and the new system(16.25%) on 3s test data in NIST LRE 2007 database. On 10s test data, the results are shown that the EER of the baseline system is 6.51% and that of the new system is 5.31%. On 30s test data, the results are shown that the EER of the baseline system is 5.36% and that of the new system is 3.54%. It shows the new system significantly outperforms the baseline system.

5. Conclusion

In this paper, we extract I-vector features using TV algorithm. Then, we use deep neural network to replace the linear classifier to classify the language. Finally, we input I-vectors into the fully connected neural network with ReLU as its activation function for classification, at the same time, the strategy of dropout is used to suppress overfitting. The experimental results prove that the new system is able to achieve the better performances than the baseline system on NIST LRE 2007 database.

Acknowledgements

This research is supported by the National key research and development plan (No.2017YFB0803001), National Science Foundation of China (No.61571144), CERNET Innovation Project(No.NGII20170412) and Co construction of Universities(No.ITEAZMZ001705).

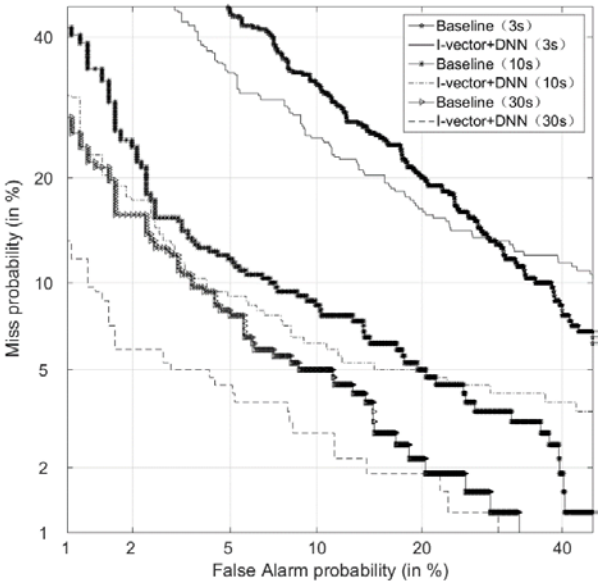

Fig. 5. Performance comparsion between I-vector+DNN and baseline system on NIST LRE 2007 database.

References

[1] Macková L, Čižmár A. (2015) "Emotional speaker verification based on i-vectors." *Cognitive Infocommunications* : 533-536.

[2] Dat T T, Jin Y K, Kim H G, et al. (2015) "Robust Speaker Verification Using Low-Rank Recovery under Total Variability Space." *International Conference on It Convergence and Security* : 1-4.

[3] Aronowitz H, Barkan O. (2011) "New Developments in Joint Factor Analysis for Speaker Verification." *International Conference on Speech Communication Association* : 129-132.

[4] D'Haro L F, Cordoba R, Salamea C, et al. (2014) "Extended phone log-likelihood ratio features and acoustic-based i-vectors for language recognition." *International Conference on Acoustics, Speech and Signal Processing* : 5342-5346.

[5] Tong A, Greenberg C, Martin A, et al. (2016) "Summary of the 2015 NIST Language Recognition i-Vector Machine Learning Challenge." *Odyssey* : 297-302.

[6] Kinnunen T, Kinnunen T, Kinnunen T, et al. (2017) "Direct Optimization of the Detection Cost for I-Vector-Based Spoken Language Recognition." *IEEE/ACM Transactions on Audio Speech & Language Processing* **25** (**3**):588-597.

[7] Richardson F S, Campbell W M. (2011) "NAP for high level language identification." *International Conference on Acoustics, Speech and Signal Processing* : 4392-4395.

[8] Cummins N, Epps J, Sethu V, et al. (2014) "Variability compensation in small data: Oversampled extraction of i-vectors for the classification of depressed speech." *IEEE International Conference on Acoustics, Speech and Signal Processing* : 970-974.

[9] Rahman M H, Himawan I, Dean D, et al. (2017) "Domain Mismatch Modeling of Out-Domain i-Vectors for PLDA Speaker Verification." *International Conference on Speech Communication Association* : 1581-1585.

[10] Zhang R, Sun J S. (2016) "Research on logistic regression spoken language understanding method." *Information Technology* **4** (**23**): 92-104.

[11] Lecun Y, Bengio Y, Hinton G. (2015) "Deep learning.". *Nature* **521** (**7553**): 436-444.

[12] Schmidhuber J. (2015) "Deep learning in neural networks: An overview." *Neural Networks* **61**: 85-117.

[13] Xiong J. Tutorial-1: Machine learning and deep learning." *Design Automation Conference* : 19-25.

[14] Lopez-Moreno I, Gonzalez-Dominguez J, Plchot O, et al. (2014) "Automatic language identification using deep neural networks." *International Conference on Acoustics, Speech and Signal Processing* : 5337-5341.

[15] Gonzalezdominguez J, Lopezmoreno I, Moreno P J, et al. (2015) "Frame-by-frame language identification in short utterances using deep neural networks." *Neural Networks* **64** (**C**): 49-58.

[16] Lopez-Moreno I, Gonzalez-Dominguez J, Martinez D, et al. (2016) "On the use of deep feedforward neural networks for automatic language identification." *Computer Speech & Language* **40** (**C**): 46-59.

[17] Babu K S, Yarramalle S, Penumatsa S V. (2012) "Text Independent Speaker Recognition Model Based on Gamma Distribution Using Delta, Shifted Delta Cepstrals." *Advances in Computer Science, Engineering & Applications. Springer Berlin Heidelberg* : 25-29.

[18] Srivastava N, Hinton G, Krizhevsky A, et al. (2014) "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research* **15** (**1**): 1929-1958.

[19] Liang J, Liu R. (2016) "Stacked denoising autoencoder and dropout together to prevent overfitting in deep neural network." *International Congress on Image and Signal Processing*: 697-701.