Journal of Engineering and Applied Sciences

MEDWELL PUBLICATIONS SINCE 2000

Unsupervised Speaker Retrieval and Identification in Large Scale Environment

[1]Rami Ammar, [2]Assef Jaffar and [2]Kadan Aljoumaa
[1]Higher Institute for Applied Science and Technology, Damascus, Syria
[2]Department of Computer Science, Higher Institute for Applied Science and Technology, Damascus, Syria

Key words: I-Vector technique, speaker identification, k-means++, large-scale environment, deep autoencoder, SideKit, VoxCeleb

Corresponding Author:
Rami Ammar
Higher Institute for Applied Science and Technology, Damascus, Syria

Page No.: 2457-2463
Volume: 15, Issue 11, 2020
ISSN: 1816-949x
Journal of Engineering and Applied Sciences
Copy Right: Medwell Publications

Abstract: The identity vector is one of the state-of-the-art techniques for building speaker identification and retrieval systems. These systems are used in many crucial applications. Recently, mainly due to the facilities in audio content acquisition, the need to analyzing unlabeled datasets has become a vital advantage. Our contribution is to enhance the identity vector approach by using k-means++ instead of using the random initial state of the universal background model "UBM", this randomness may lead to a local minimum. This enhancement increased the accuracy of the system and decreased the needed number of epochs, thus, decreased the training time. In addition, we presented a study of the effect of changing the voice information extraction and the UBM parameters also we enhanced the performance of the system by using dimensionality reduction for identity vectors through using a deep autoencoder. Finally, we enhanced the well-known "SideKit" toolkit to work on large datasets in batches. We used a large dataset obtained under different conditions "VoxCeleb1". VoxCeleb1 is a free and well-known dataset was recorded in real-world conditions.

INTRODUCTION

Nowadays, the need of speaker recognition systems is very high. They can be used in different applications such as authentication for high- confidential applications (in order to identify a person through his voice), monitoring applications for security agencies and forensic speaker recognition tests. In addition, it has recently been used in personalized user interface where the system can adapt its functionalities to match the identified user's needs[1]. All of these tasks require a high performance to recognize the speakers under real-world conditions. Speaker recognition is very difficult task due to both extrinsic and intrinsic variations. Extrinsic variations include the effects of background noise, music, echo, channel effect and other sounds. On the other hand, the intrinsic variations are factors related to the speaker himself such as age, accent, emotion and manner of speaking[2].

Speaker recognition could be categorized into two main tasks: speaker identification (matching with a multiple speakers "1: N") or speaker verification (matching with single speaker "1: 1"). The task of identifying the speaker is considered more complicated than the task of verifying him as the reliability of the speaker identification system is affected by the number of the registered speakers in the system due to the increased probability of making the wrong decision (wrong identity) while the reliability of speaker verification system is not affected by the number of speakers because the matching occurs between two speakers only.

Most of speaker identification approaches depend on considerable amounts of labeled training speech data to produce robust models and achieve better accuracy in recognition. Such methods often use a heavily trained Gaussian Mixture Models (GMMs) or a deep neural network to compute sufficient statistics for extraction of the low dimensional embedding vectors. I-Vector approach produces an embedding vector for each speech segment without any prior knowledge of the identity of speaker. Although, effective, a shortcoming intrinsic to this approach consists in using the randomness to define the initial state of the Universal Background Model (UBM). This randomness makes the results of the model sometime inaccurate has dependency on this initial state and needs to increase the needed number of epochs to train the model, thus, increasing the training time. In other hand, the accuracy of all recognition models is contingent on choosing the best values for the parameters of the model and features extraction. Finally, in large scale environment which contains a big number of segments, the system's performance is vital issue and should be considerable.

In this study, we are interested in building a speaker identification and retrieval system based on identity vector "i-vector" approach using a free, well-known dataset which was recorded in real-world conditions "VoxCeleb1". The basic objective is to enhance the i-vector approach by proposing a novel approach to define the initial state of UBM by using K-means++ algorithm, which is faster and more robust than k- mean algorithm, and by selecting the best values for voice features extraction, UBM and total variability parameters. The other objective is to enhance the performance of the retrieval system by unsupervised reduction of the i-vector dimensions by building deep autoencoder. Finally, the SideKit toolkit is enhanced to work in mini-batches, which makes it available in large scale domain.

Speaker identification and retrieval: The unsupervised speaker identification task builds on speaker retrieval task as final step.

Speaker identification: The speaker doesn't need to provide his identity, he only has to provide a speech segment to the system and then his identity is determined by the result of the comparison[3].

suppose x is the latent vector of the query speech segment) to be identified(and its class is C_x, the search space is $D = \{w_1, w_2, ..., w_s\}$ where S is the number of speakers w_n is the latent vector from speech segment for speaker s y_s is the score between x and w_s. For closed-set speaker identification which the query speech segment is assumed to be spoken by one of the registered speakers in the system, the system identifies the query using the scores $\{yn|n\in 1 ,... ,S\}$ the best matching speaker as $C_x = C_{y*}$ where $Y* = \max_{s = 1} sY_s$.

Speaker retrieval: For speaker retrieval, the speaker also doesn't need to provide his identity, he provides a speech segment to the system and then the comparisons are computed between the speech segment and all development segments. Based on the distance measures, ranked lists are computed to represent the most similar segments for a given query.

For the latent vector of the query speech segment (x), the search space is $D = \{w_1, w_2, ..., w_N\}$ where N is the total number of segments to search w_n is the latent vector from speech segment n, y_n is the score between x and w_n. The ranked List (L) contains the top-k most similar speech segments to query speech.

Literature review: For a long time, Gaussian Mixture Models (GMMs) were used in speaker recognition systems to model each speaker by a GMM that represents speaker discrimination information. GMM parameters (covariances, means and weights) estimate the formant bandwidths, magnitudes and location of speech signals[3]. This model has some limitations, first of all the accuracy of the model is relatively weak compared to other model in large scale environment in addition, the model requires acceptable number of speech segments for each speaker. Finally, this model cannot be applied on an unlabeled data set. Bhattacharjee *et al*.[4] presented a speaker verification system relying on GMM-UBM where GMM-UBM is based on the idea of a single large GMM which is trained to represent the speaker-independent distribution of the speech features for all speakers. However, Bhattacharjee *et al*.[4] found that the relation between the speaker model and the background model provides better performance than that of the independently trained GMMs. GMM-UBM based speaker verification system degrades considerably with change in training and testing language and it cannot be applied on an unlabeled data. P. Kenny *et al*.[5] presented the Joint Factor Analysis (JFA) which assumes that both speaker and channel variability lie in low dimensional subspace of GMM supervector space. Dehak *et al*.[6] show that the channel factors in JFA contain speaker-dependent information, so the Total Variability (TV) space (including speaker and channel) is modeled instead of modeling the channel-space and speaker-space separately and this is an identity vector "i-vector" approach. Identification system in[2] and retrieval system in Shon *et al*.[7] rely on the total variability of the Vox Celeb dataset to produce i- vectors for each speech segment. Additionally, M. Adiban *et al*. in[8] presented a heart sound classification system that relies on an i-vector/autoencoder system to detect heart diseases where human heart sounds can be considered as the physiological traits of a person and only irregular events like illnesses or aging can alter these traits and they found that the i-vector of a heart sound is a more suitable feature to describe the characteristics of heart sound than other variable length features.

MATERIALS AND METHODS

The main motivation of the proposed methodology consists in providing general unsupervised speaker retrieval and identification system. In fact, the methodology is completely independent of training data. In this way, it can be maintained completely unsupervised for speaker retrieval and only a final step can be included for speaker identification. This methodology can be useful in scenarios where the speaker labels could be not known at a given moment when the retrieval is being processed. Once the labels become available, the system does not require any re-processing such that the identification can be computed as a final step by exploiting the retrieval results.

Figure1 shows the basic steps of the approach which starts with extracting the features from raw speech segment to model them by embedding vectors, then retrieval task is done to find a ranked list for each query speech in test phase. Finally, identification task will be done by exploiting the result of retrieval task.

Features extraction: The features refer to the parameters which are extracted from the short speech frames where Mel-Frequency Cepstral Coefficients (MFCCs)[9] and Perceptual Linear Prediction (PLP)[10] are the most popular features that represent speech segments in low-dimensional space.

First, the audio segment is divided into short overlapping frames (usually 20-25 milliseconds) and then the speech frames are detected. This detection necessitates for Voice Activity Detection (VAD) algorithms. The most popular VAD algorithms are the Signal to Noise Ratio (SNR) which compares the desired signal level to noise level. the Energy thresholding is used to compare the energy of the frame with a threshold.

Finally, feature normalization techniques such as Cepstral Mean and Variance Normalization (CMVN) can be applied to obtain meaningful speaker-dependent information[11].

Speaker modeling: Supported by the features extraction step, the speaker modeling advances to finding patterns and modeling the speech segment information.

The identity vector “i-vector” method is based on the basic idea of the JFA “Joint Factor Analysis” method. JFA method is based on modeling the channel and the speaker variability to restrict speaker model parameters to lie in a low dimensional subspace. However, the i-vector approach models the Total Variability (TV) (including channel and speaker) instead of modeling channel and speaker separately[12]. For speech segment of a speaker (s), the channel and speaker dependent GMM-supervector is written as in (Eq. 1:)

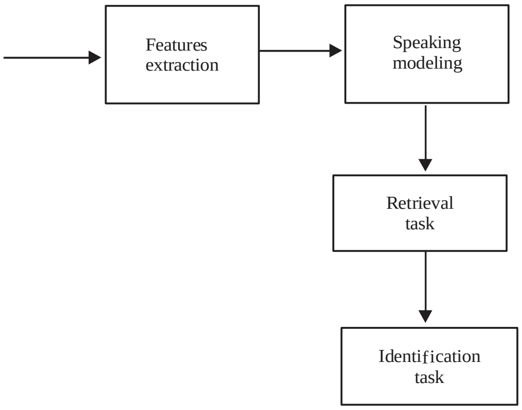

Fig. 1: The basic steps of the approach

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

Where:

m = The UBM-supervector and it is independent of speaker and channel

T = A low-dimensional total variability matrix (TV) (usually between 400-800)[13]

w = A standard normal random vector w~ N (0,1)

The posterior mean of w is a low-dimensional vector called i-vector. The baseline I-Vector approach has some disadvantages, like slowness and inaccuracy. These disadvantages due to the reliance of the Universal Background Model (UBM) on the random initial state. In this paper, we have improved the I-Vector approach by using k-mean++ algorithm[14] (standard k-mean algorithm with an initialization algorithm for selecting initial values for centroid) in order to choose initial values for UBM (Fig. 2). The initialization algorithm:

- Randomly, select the first centroid from the data points
- For each data point, compute its distance from the nearest, previously chosen centroid
- Choose one new data point at random as a new center using a weighted probability distribution where a point x is chosen with probability proportional to D(x)2
- Repeat Steps 2 and 3 until k centers have been chosen

Retrieval task: The main method for speaker retrieval task is calculating similarity between tested segment i-+vector and development segments i-vectors, then

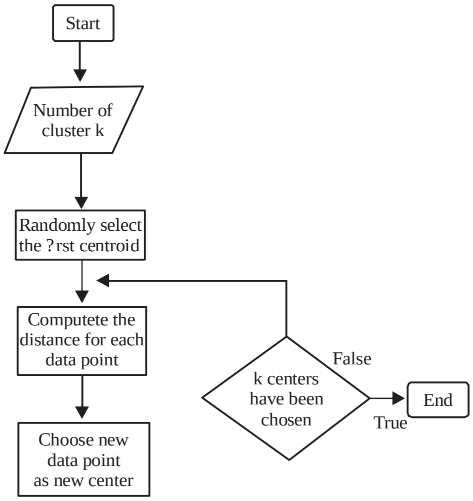
Start
Number of cluster k
Randomly select the ?rst centroid
Computete the distance for each data point
Choose new data point as new center
k centers have been chosen
False
True
End

Fig. 2: k-means++ initialization algorithm

retrieve a k-ranked list (L) which contains the top-K most similar speech segments to the tested segment[15]. In this study, we rely on the cosine similarity to calculate the similarity between i-vectors. Cosine similarity is given by:

$$\cos(A,B) = \frac{A.B}{|A\|B|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{2}$$

Identification task: In this step, we exploit the retrieval task results (ranked list L) to identify the speaker. The identity of the speech segment is identified by the speaker who has the max frequency segments in the ranked list.

The last two step (retrieval and identification steps) are implemented using K-Nearest Neighbors (KNN)[2] which makes the classification error as minimal as possible by choosing the class that has the most points within the neighborhood.

Deep autoencoder: The most important property in the large-scale environment is the performance of the system where the system should response to an input in acceptable time. To reduce the computation time; thus, increase the performance, we use a deep autoencoder[16] for dimensionality reduction of i-vectors. The autoencoder is an unsupervised data compression algorithm where the compression and decompression functions are data-specific and learned automatically from the data. Additionally, in almost all contexts where the term "autoencoder" is used the compression and decompression functions are implemented with neural networks (Fig. 3).

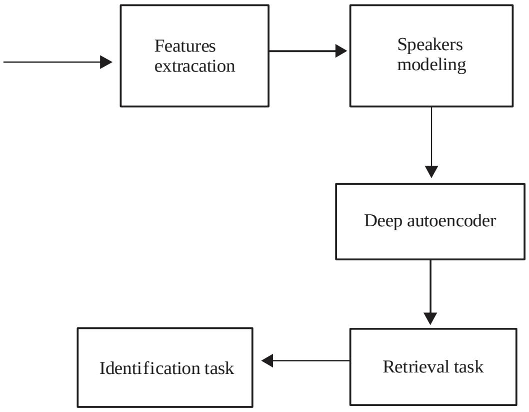

Fig. 3: The basic steps of the approach with autoencoder

Sidekit toolkit: The SideKit is a new open source tool that includes a large collection of state-of-the-art components written in Python. This toolkit allows the development of speaker recognition systems quickly[17]. The down side of the tool that some of its components (such as GMM training or i-vectors extraction) rely on transferring the complete data to the main memory (RAM) which makes using it impossible in a large-scale environment because of the limited resources. So, we enhanced the tool to work on batches of data. this means it would transfer part of the data to the memory sequentially. Our enhancement made the tool usable for limited resources and large datasets. Batch GMM Training Algorithm:

- Read batch of training data form hard disk 2-Execute E-step on this batch
- Repeat 1 and 2 until read all data
- Execute M-step
- Repeat 1-4 until likelihood converge

Algorithm 1: Batch GMM Training algorithm

```
1: procedure EM_Batch (feature_file, batch_size)
training GMM for features in the file feature_file
2:  batch ← read(feature_file, batch_size)
3:  llk_gain ← 0.01 likelihood convergce
4:  llk ← 10                          last convergce
5:  while llk > llk_gain do likelihood don't converge
6:  while batch ≠ empty do We don't read all data
7:  E_step(batch)
8:  batch ← read(feature_file, batch_size)
9:  end while
10:    llk ← M_step() - llk 11:
end while
12: return llk                        last likelihood value
13: end procedure
```

RESULTS AND DISCUSSION

Experiments: Before applying the approach to the entire dataset (1251 speakers), we initially apply it to a sample containing a specific number of speakers in order to

Table 1: Dataset statistics

No. of speakers	1521
No. of male speakers	690
No. of videos per speakers	36/18/8
No. of segments per speakers	250/123/45
Length of segments	145/8.2/4

Table 2: Development and test set statistics for the sample

Set	No. of speakers	No. of segments
Train	25	3,096
Test	25	269
Total	25	3,365

Table 3: Results of SNR value tests on identification accuracy

SNR (dB)	1-NN (%)	3-NN (%)	5-NN (%)
45	83.27	86.24	86.24
42	85.87	88.11	88.85
40	81.41	83.27	86.24
38	84.01	84.39	85.87
36	84.39	84.39	85.87
34	83.64	84.01	84.76
32	85.5	83.27	83.64
30	83.64	86.24	86.62
28	84.24	84.39	86.24
26	83.27	86.24	87.36
20	84.75	86.24	85.87

define the best value for the parameters and ensure the effectiveness of the SideKit toolkit after enhancing it to work on mini-batches of data. This task can be done by comparing the phased and the final results that we get from pre-developed and the batch-enabled SideKit.

Dataset: The VoxCeleb dataset[2] is a free, well-known dataset which contains more than 150,000 speech segments for 1251 speakers. These segments are extracted of videos uploaded to YouTube. It is gender balanced and the speakers span a wide range of ethnicities, accents and ages.

It was taken in real-world conditions where background chatter, laughter and overlapping talk exist, as well as a wide range of voice qualities. Table 1 shows the data set statistics. There are three entries in a field. The numbers refer to the maximum/average/minimum values (Adapted from[2]).

Experimental setup: For each speaker, we take the speech segments from one video for testing, provided that the number of speech segments for each speaker isn't >3 segments. The rest of the speech segments are kept for training. For retrieval and identification tasks, we rely on the similarity approach between the testing vector and the training vectors, then we apply K-NN where k = {1, 3, 5}.

Sample experiments: A sample of 25 speakers is taken using ID attribute to represent the speakers with identifiers [10001-10025]. Table 2 shows the statistics of training and testing set.

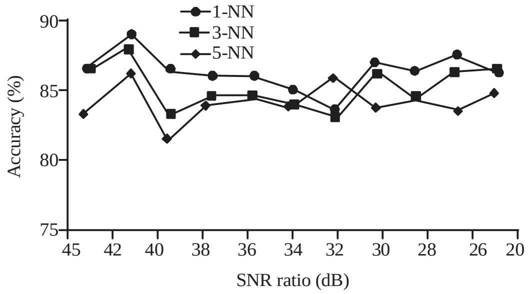

Fig. 4: SNR tests

GMM-UBM setup: We use feature vectors of dimension 72 (24 MFCC coefficients plus delta and double delta). Cepstral Mean and Variance Normalization (CMVN) is applied to normalize the features. VAD algorithms (SNR and Energy thresholding) are tested to detect speech frames. UBM of 64 mixture components (double of speaker's number) is trained for 10 iterations (as used the research[2]) with k-mean++ to define initial state.

i-vector setup: The identification system is trained to extract 15-dimensional i-vectors[18] concluding that training the Total Variability (TV) matrix for 5 iterations is enough to obtain near to optimal performance.

Results: The results of SNR algorithm test (represented by Table 3 and Fig. 4) show that the appropriate value is the ratio 42 dB where the accuracy is the highest value (88.85%) regardless of the chosen value for k in K-NN algorithm. While the results of energry thresoulding algorithm (Table 4) show that the SNR algorithm is more accurate for Voxceleb dataset.

Entire dataset experiments: After defining the best value for the system parameters, we apply our approach to the entire dataset (1251 speakers whose identifiers are [10001-11251]. Table 4 and 5 for statistics of training and testing set).

GMM-UBM setup: The same mechanism is used for features extraction (24 MFCC coefficients plus delta and double delta), CMVN is applied for features normalization. SNR algorithm with the best value (42 dB) is used to detect speech frames. We also tested the effect of the number of UBM distributions and the number of UBM training in identifying accuracy.

I-vector setup: Gender i-vector extractors in[7] are trained to produce 400-dimensional i-vectors and[20] trained the TV matrix for 5 iterations.

Table 4: Results of VAD algorithms tests on identification accuracy

Variables	1-NN (%)	3-NN (%)	5-NN (%)
SNR = 42dB	85.87	88.11	88.85
Energy threshold = 5.5	81.04	81.41	78.07

Table 5: Development and test set statistics for the entire dataset

Set	Speakers (#)	Segments (#)
Train	1251	140,082
Test	1251	13,431
Total	1251	153,513

Table 6: Test results for the number of UBM distributions and the number of training iterations on Identification accuracy

Mixtures (#)	No. of training iterations for UBM	1-NN (%)	3-NN (%)	5-NN (%)
1024	10	68.42	70.26	72.52
	0	65.77	67.55	69.71
2048	10	70.61	72.36	77.43
	0	66.55	68.40	70.67

Table 7: Trained autoencoder accuracy

Variables	Accuracy (%)
200-dimensional i-vectors	99.82
250-dimensional i-vectors	99.91

Autoencoder: the encoder and the decoder functions are implemented using neural networks to reduce the dimensions of the i-vectors (Fig. 5). The autoencoder network has 400 neurons (i-vector dimension) as input and output layers and three hidden layers, two of them have 300 neurons with linear activation function and the third layer which produces the low-dimensional i-vector has a number of neurons equals to the reduced size with Tanh activation function (400→300→reduced size→300→400).

We tested two values for the reduced size (200 and 250) with Adam optimizer, the means square error loss function and a batch size of 512. Finally, the trained autoencoder is tested using i-vectors test dataset by supplying the autoencoder with an i-vector from the dataset and calculating the nearest vector from the same set to the autoencoder's output vector to report the accuracy (Table 7).

Results: The results (Table 6) shows that the sample tests to define the best value for the system parameters and using K-mean++ algorithm to find the initial state of UBM provides a superior performance compared to the results in[2] which reached an accuracy of 56.6% with random initial state and 1024 distributions for UBM. The results also show that the use of k-means++ with 10 training iterates for 2048-UBM model gives a clear improvement. This superior improvement can be explained by two main points. The first one is using SNR algorithm with the best value to detect the speech frames dataset and calculating the nearest vector from the same set to the autoencoder's output vector to report the accuracy (Table 7).

Table 8: Autoencoder results

Accuracy	1-NN (%)	3-NN (%)	5-NN (%)
Basic i-vector (400)	70.61	72.36	77.43
i-vector + autoencoder (200)	70.8	73.2	78.9
i-vector + autoencoder (250)	71.01	73.72	79.2

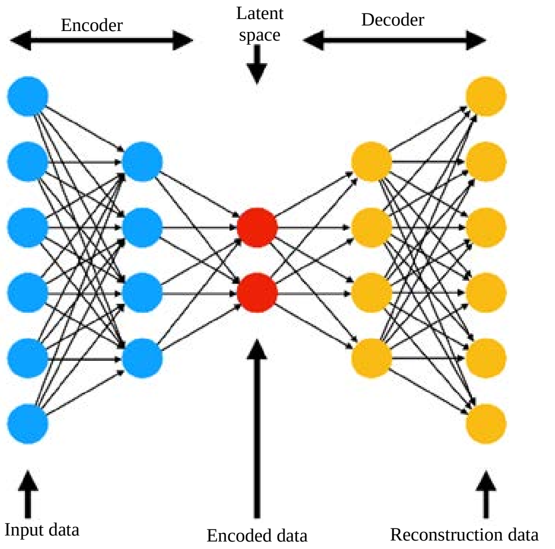

Fig. 8: Autoencoder architecture

Results: The results (Table 6) shows that the sample tests to define the best value for the system parameters and using K-mean++ algorithm to find the initial state of UBM provides a superior performance compared to the results in[2] which reached an accuracy of 56.6% with random initial state and 1024 distributions for UBM. The results also show that the use of k-means++ with 10 training iterates for 2048-UBM model gives a clear improvement. This superior improvement can be explained by two main points. The first one is using SNR algorithm with the best value to detect the speech frames well. This detection reduces the number of non-speech frames which give more useful information about the voice segment. The second point is using the k-means++ algorithm, instead of using random state for UBM which may lead us to local minimum, in addition to the massive time and massive number of training iteration which may reach hundreds.

We noticed that the performance of the identification system increased by using the autoencoder since the computation for 400-dimensional i-vector is bigger than (200 or 250) dimensional i-vector, plus the accuracy of the system increased a bit (Table 8). The reason of the increasing in the accuracy was a little bit is that the autoencoder spread the low dimensional produced vectors without respect to minimizing the within-speaker variance and maximizing the between-speakers variance which require labeled data.

CONCLUSION

We provide a practical implementation for speaker's i-vectors then the similarity measure is done between the tested i-vector and the cluster's representative vector, which would increase the performance. Finally, the deep autoencoder can be enhance in case the speaker labels were available before building the autoencoder by using triplet loss function plus mean square error.

AKNOWLEDGMENTS

This research is supported by a grant from the Higher Institute for Applied Sciences and Technology, Damascus, Syria. The authors appreciate the valuable comments provided by the anonymous referees.

REFERENCES

01. Singh, N., R.A. Khan and R. Shree, 2012. Applications of speaker recognition. Procedia Eng., 38: 3122-3126.
02. Nagrani, A., J.S. Chung and A. Zisserman, 2017. VoxCeleb: A large-scale speaker identification dataset. Proceedings of the 18th Annual Conference of the International Speech Communication Association (INTERSPEECH, 2017), August 20-24, 2017, International Speech Communication Association, Stockholm, Sweden, pp: 2616-2620.
03. Aroon, A. and S.B. Dhonde, 2015. Speaker recognition system using Gaussian mixture model. Int. J. Comput. Appl., 130: 38-40.
04. Bhattacharjee, U. and K. Sarmah, 2012. GMM-UBM based speaker verification in multilingual environments. Int. J. Comput. Sci. Issues (IJCSI.), 9: 373-380.
05. Kenny, P., G. Boulianne, P. Ouellet and P. Dumouchel, 2007. Joint factor analysis versus eigenchannels in speaker recognition. IEEE. Trans. Audio Speech Lang. Process., 15: 1435-1447.
06. Dehak, N., P. Kenny, R. Dehak, P. Dumouchel and P. Ouellet, 2010. Front-end factor analysis for speaker verification. IEEE Trans. Audio, Speech Language Process., 19: 788-798.
07. Shon, S., Y. Lee and T. Kim, 2018. Large-scale speaker retrieval on random speaker variability subspace. Audio Speech Process., Vol. 2.
08. Adiban, M., B.B. Ali and S. Shehnepoor, 2019. I-vector based features embedding for heart sound classification. Mach. Learn., Vol. 2,
09. Ahmad, K.S., A.S. Thosar, J.H. Nirmal and V.S. Pande, 2015. A unique approach in text independent speaker recognition using MFCC feature sets and probabilistic neural network. Proceedings of the 2015 8th International Conference on Advances in Pattern Recognition (ICAPR), January 4-7, 2015, IEEE, Kolkata, India, pp: 1-6.
10. Kalia, A., S. Sharma, S.K. Pandey, V.K. Jadoun and M. Das, 2020. Comparative Analysis of Speaker Recognition System Based on Voice Activity Detection Technique, MFCC and PLP Features. In: Intelligent Computing Techniques for Smart Energy Systems, Kalam, A., K. Niazi, A. Soni, S. Siddiqui and A. Mundra (Eds.). Springer, Singapore, pp: 781-787.
11. Tiwari, V., 2010. MFCC and its applications in speaker recognition. Int. J. Emerging Technol., 1: 19-22.
12. Rao, W., M.W. Mak and K.A. Lee, 2015. Normalization of total variability matrix for i-vector/PLDA speaker verification. Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), April 19-24, 2015, IEEE, Brisbane, Australia, pp: 4180-4184.
13. Hansen, J.H. and T. Hasan, 2015. Speaker recognition by machines and humans: A tutorial review. IEEE Signal Proc. Mag., 32: 74-99.
14. Arthur, D. and S. Vassilvitskii, 2007. K-means++: The advantages of careful seeding. Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, January 2007, ACM, New Orleans, Louisiana, pp: 1027-1035.
15. Campos, V.D.A. and D.C.G. Pedronette, 2019. A framework for speaker retrieval and identification through unsupervised learning. Comput. Speech Lang., 58: 153-174.
16. Zhao, R., R. Yan, Z. Chen, K. Mao, P. Wang and R.X. Gao, 2018. Deep learning and its applications to machine health monitoring. Mech. Syst. Signal Process., 115: 213-237.
17. Larcher, A., K.A. Lee and S. Meignier, 2016. An extensible speaker identification sidekit in python. Proceedings of the 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), March 20-25, 2016, IEEE, Shanghai, China, pp: 5095-5099.
18. Vestman, V. and T. Kinnunen, 2018. Supervector compression strategies to speed up i-vector system development. Audio Speech Process., Vol. 1.