Study of speaker recognition system based on Feed Forward Deep Neural Networks exploring text-dependent mode

Ben Jdira Makrem, Jemâa Imen and Ouni Kaïs
Research Unit: Signals and Mechatronic Systems, SMS, UR13ES49
National Engineering School of Carthage, ENICarthage, University of Carthage
CharguiaII, Tunisia
makrembjdira@gmail.com, imen.jemaa@yahoo.fr and kais.ouni@enicarthage.rnu.tn

Abstract—**We aim by this work to follow the significant progress in speaker recognition systems getting the benefits of the advancement in the artificial intelligence (AI). Indeed, the deep learning algorithms have proved a real performance in the recognition and classification data. In this contest, we present a study of three different speaker recognition system based in Feed Forward neural networks. The first one is the logic regression, the second one is the Multilayer Perceptron (MLP) and the third one is the Stacked Denoising Autoencodeurs (SDA). We evaluated these recognition rates using the parameterization technique Mel Frequency Cepstral Coefficients (MFCC). To find the best results and to better optimize automatic recognition algorithms, we tested our speaker recognition system under the text-dependent database RSR2015. We studied the recognition rates by varying the values of neural networks parameters, number of neurons and number of hidden layers…etc. We discussed the different results obtained and we selected best parameter values which lead the minimum rate error of recognition.**

Keywords— *Speaker recognition; text-dependent mode; feed forward networks; logistic regression; multilayers perceptron; stacked denoising autoencoders.*

I. Introduction

The main task of speaker recognition systems is to identify or to confirm the identity of an unknown user from the extracted features of his voice. In speaker recognition we have achieved in recent years a large performance. This progress was attainted despite the presence of many difficulties. The new speaker recognition technologies were applied in many fields such as politic, police search, university research, biometric authentication and various applications on Internet and remote access data bases. They were also integrated in different software such as security systems, portable devices and smart phones. This research field benefits from advances and progress of artificial intelligence (AI) in development of increasingly efficient software and systems.

The deep Learning is a new area of Machine Learning research, which has been introduced with the objective of moving Machine Learning closer to one of its original goals: Artificial Intelligence [1]. It allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state-of-the-art in speech and speaker recognition, visual object recognition, object detection and many other domains [2]. Deep learning discovers intricate structure in large data sets by using the back propagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer.

An example of much known deep learning method is artificial neural networks algorithm. Indeed, it seeks to replicate the human brain function and rely today on several layers of artificial neurons, or calculation units organized hierarchically. Neural networks approaches have gone a long way in their evolution. Until now there are a very large number of neural networks approaches varying in layers, in connections, in structures and training methods. It caused significant advances in speech and speaker recognition.

We present a study of text dependant speaker recognition system. We adopted three approaches of feed forward neural network: the logistic regression, the Multi Layer Perceptron (MLP) and the Stacked Denoising Autoencoders (SDA) . We studied the recognition rate by varying the learning rate (the learning level for the gradient descent) and the number and the size of layers. The adopted parameterization technique is Mel Frequency Cepstral Coefficients (MFCC). We used the recognition rate to compare the approaches.

In this paper, we will define in the section.2 the Feed Forward Neural Network, we will present in the section.3 three main approaches of Feed Forward neural network and we will describe the implementation of the algorithm and the packages in the section.4. In the final section, we will present several tests of our ASR system and we will discuss the obtained results.

II. Feed Forward Neural Network

The application of artificial neural networks for speakers recognition has been especially developed in the early of 1990s as shown in [3]. The neural networks are an approach for addressing the problems of learning and reasoning which are sometimes qualified complexes. They are proving promising alternative to solve these problems.

According to the work of Nigrin [4,5], a neural network is a circuit composed of a very large number of simple processing elements that are neurally based [5]. Each element operates only on local information. Furthermore each element operates asynchronously; thus there is no overall system clock. A neural network is a massively parallel distributed processor [5,6] that has a natural propensity for storing experiential knowledge and making it available for use [4,5,6].

When a neural network is without a recurrent connection it is called feed forward neural network. A simple example of a Feed Forward neural network is presented in Fig.1.

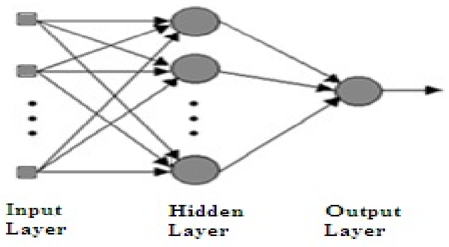

Fig. 1. Basic configuration of a Feed Forward Neural Network

The main drawback of the basic configuration of neural network is that it roughly consists of only three layers of neurons, an input layer, which received the information, the decision neurons or the output layer activated by the first layer [7] and only one intermediate hidden layer. The failure of the neural network blocked a moment the research on this type of model. But neural networks are back in force, thanks in part to the work of Geoffrey Hinton thanks to the addition of other intermediate layers of neurons, between perception 'the input layer and the decision' the output layer [7]. This technique is known as deep Feed Forward Neural Network presented in Fig.2.The template is used to format your paper and style the text.

The use of unsupervised learning techniques is linked to another major change in recent years which it is the use of data. Therefore machine learning, deep learning and big data are interrelated. Another key factor is the democratization of computing power, thanks to parallel architectures using graphics processors (GPU), capable of running machine learning algorithms in reasonable time.

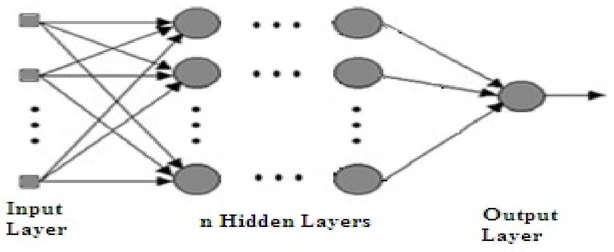

Fig. 2. Feed Forward Neural Network

We will discuss in the next section three main architecture of neural network used in our ASR system to test the best of them in recognition performance.

A. Logistic regression

Logistic Regression is a very popular and widely used technique. It is a probabilistic linear classifier parametrized by a weight matrix W and a bias vector b ($Y=x^T.W+b$) where is applied a logistic function (1/1+ez) to convert it into a more categorical thing (binary if there's only two classes). Essentially, logistic regression is considered as a one layer neural network as shown below in fig 3.

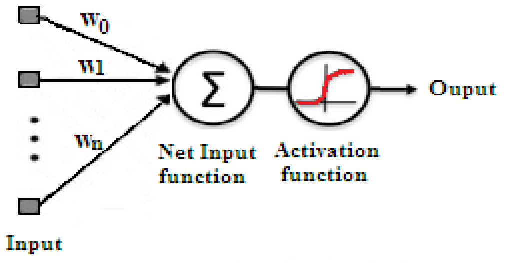

Fig. 3. Logistic regression

In particular, when making logistic regression, the hypothesis is put that the log of the probabilities is a logistic regression defined by Eq.1 below:

$$\log(\frac{P}{1-P}) = X^T .W \longleftrightarrow P = \frac{\exp(X^T .W)}{1+\exp(X^T .W)} \quad (1)$$

The logistic function is represented in fig. 4:

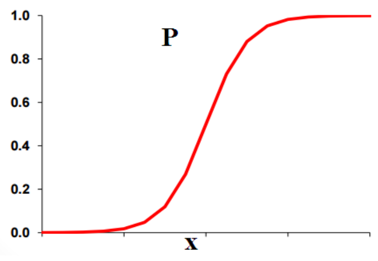

Fig. 4. Logistic function

The classification is made by the projection of the input vector x to a set of hyperplanes, each corresponding to a class. The distance between the input and the hyperplane reflects the probability that the input is a member of the corresponding class. This probability is written as follows:

$$P(Y = i \mid x, W, b) = \mathrm{softmax}_i(Wx+b) = \frac{e^{W_i x+b_i}}{\sum_j e^{W_i x+b_i}} \quad (2)$$

The prediction of the model is ypred which is the class which has the highest probability.

$$y_{pred} = \arg\max P(Y = i \mid x, W, b) \quad (3)$$

We want to maximize the likelihood, thinking in terms of loss function which measures the cost of y_{pred} when the current response is y, the negative log-likelihood is minimized. In the general case, this is equivalent to maximize the log-likelihood of the whole data D according to the model parameterized by Θ:

$$l(\theta = \{W, b\}, D) = \mathrm{NLL}(\theta = \{W, b\}, D) = -\sum_{i=0}^{|D|} \log(P(Y = y^{(i)} \mid x^{(i)}, W, b)) \quad (4)$$

To implement the model we use an optimization algorithm the Gradient Descent (GD) to update the weights incrementally after each epoch (iteration) as shown in fig. 5 below:

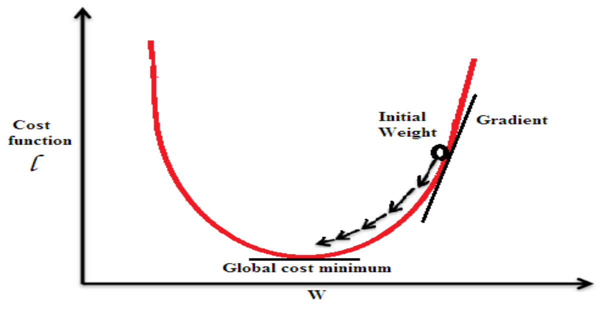

Fig. 5. Optimization algorithm Gradient Descent

But In our case we have a large data, the using of Gradient Descent can be quite costly since the slower the algorithm updates the weights, the slower it converges to the global cost minimum. So we use the Stochastic Gradient Descent (SGD) who don't accumulate the weight updates as the GD [16,17]. It updates the weights after every training sample [16,17]:

```
• For 1 to nbre_of_epochs || global_cost_minimum:
    o For training_sample(i):
        o For weight(j):
            ▪ $W_j := w + \Delta w_j$
  where $\Delta w_j = Lr \cdot (target^{(I)} - output^{(i)}) \cdot x_j^{(i)}$
  where Lr = learning rate
```

The learning rate Lr is called also the step size.

B. The Multi Layer Perceptron

The most common neural network model is the multilayer perceptron (MLP). This type of neural network is known as a supervised network because it requires a desired output in order to learn. The goal of this type of network is to create a model that correctly maps the input to the output using historical data so that the model can then be used to produce the output when the desired output is unknown.

This is a network structured in input layer, output layer and in multiple intermediate layers where each layer consists of several neurons without connection with each other, each layer performs a vector processing, it receives an input vector and converts it into an output vector. The input and output dimensions can be different[8]. Each layer can only use the outputs of previous layers is called feed-forward structure.

1. The neuron of the MLP is defined by:
2. An activation function:
    - Hyperbolic tangent T(x)=tanh(h),
    - Logistic function T(x) = 1/1+exp(-x).

A parameter n where is the number of inputs associated to: n+1 parameters, n synaptic weights W and a bias. The neuron with n inputs is computed by the following function:

$$f(x_1, .., x_n) = T(\sum_{i=1}^{n} W_i x_i + t) \quad (5)$$

Instead of feeding the registers directly to the logistic regression an intermediate layers called hidden layers are added, which have a non-linear activation function (tanh or logistic function (sigmoid)). The use of several layers performs a deep architecture. The MLP and many other neural networks learn using an algorithm called back propagation [8,9]. With back propagation algorithm, the input data is repeatedly presented to the neural network. With each presentation the output of the neural network is compared to the desired output and an error is computed. This error is then fed back (back propagated) to the neural network and used to adjust the weights such that the error decreases with each iteration and the neural model gets closer and closer to producing the desired output. This process is known as "training"[8,9].

C. Stacked denoising Autoencoders

An auto-encoder is a feed-forward neural network, non-recurring similar to MLP, which aims to learn a distributed and compressed representation for a set of data for dimension reduction. The difference with the MLP is that, the auto-encoder output layer has many nodes as the input layer, instead of train to predict the value y given x, an auto-encoder is trained to rebuild its own entered [1].

A denoising auto encoder is like an Auto encoder except that during learning, the input seen by the auto encoder is not crude input but a corrupted version stochastically input. The denoising auto encoder is thus trained to rebuild the original input from the noisy version and the DAE can have as many hidden units as needed to capture the distribution. We can use the average of the reconstruction error [1].

The denoising autoencoder can be stacked to form a deep network. The pre-unsupervised learning is via a layer at a time. Each layer is trained as a denoising auto encoder by minimizing error in the reconstruction of its inputs. Once the first k layers are trained, we can train the k + 1 layer because we can now calculate latent representation of the lower layer [1].

After the pre-training of all layers, the network proceeds to the second step of training called fine-tuning. Fine tuning is considered supervised when the prediction error is minimized on a supervised task. For this, we firstly added a layer of logistic regression at the top of the network (network output). Thus, the training of the entire network goes as training a multilayer perceptron [1].

III. Implementation

In our ASR system we used two new releases packages: SIDEKIT package and THEANO package. By using these two libraries, we benefited from their multiple advantages [10, 11].

A. *Sidekit package*

SIDEKIT is a free and open source package and toolkit developed by the team of LST of the LIUM laboratory. It released under LGPL License and is intended for use by language, speech and speaker recognition researchers. It is written in Python that is simple to modify and to update and it is tested under Python 3.4 [15].

One of its advantages that it uses a limited number of classes to make the reusability and the readability of the code. SIDEKIT is composed of several tools and algorithms for whole chain of treatment. It includes tools for acoustic features extraction, for modeling and classification [15].

B. *Theano package*

THEANO is a library written in Python and was written at the LISA lab to support rapid development of efficient machine learning algorithm. Its name comes from a Greek mathematician. This package was realized under BSD license [10, 11].

THEANO evaluate complex and multidimensional mathematical expressions. It can handle with great speed problems involving large amount of data. This combination with optimizing compilation is particularly useful for tasks with iterative complicated mathematical expressions [10, 11].

The using of Theano package facilitates the implementation of the algorithm, by using the class previously developed and defined for the Stacked Denoising Auto encoder. We can have Stacked Denoising Auto encoder as having two sides or more as two facades: The first is a list of auto encoders and the second is a Multi Layers Perceptron. During the pre-training, the first facade is used, our model is treated as a list of auto encoders, and each auto encoder is trained separately. In the second step of the training, we use the second facade. Both facades are related because:

-The Auto encoders and the sigmoid layers of MLP are sharing parameters.

-The latent representations calculated by intermediate layers of the MLP are feed as inputs to the Auto encoders.

IV. Result

In order to assess the efficacy of these algorithms and to get best measurement we tested our system under the newest and the largest text dependent database RSR2015 [12,13] (a text dependent speaker recognition demand that speakers utter the same phrase or password). RSR2015 database is described in the sub section below.

We run our tests using a laptop (notebook computer) with these characteristics and features: Intel Core i5 dual core with 1,6 GHz (Turbo Boost until 2,7 GHz) and 3 Mo cache memory, 4 Go RAM. Dependant on this laptop we note the time taken by the algorithm called run time.

The adopted parameterization technique is Mel Frequency Cepstral Coefficients (MFCC) with log energy and derivative and second derivative.

A. *RSR2015 Database*

The RSR2015 database, designed to evaluate text-dependent speaker verification systems under different durations and lexical constraints has been collected and released by the Human Language Technology (HLT) department at Institute for Infocomm Research (I2 R) in Singapore [12]. This database contains audio recordings from 300 persons, 143 female and 157 male speakers. Participants were selected to be representative of the ethnic distribution of Singaporean population. Selected speakers were between 17 to 42 years old. Each of the participants recorded nine sessions using three portable devices [13]. For each session, a speaker read thirty short sentences. Sentences were selected from TIMIT database by making sure that its cover all English phones [13]. The number of words per sentence varies from four to eight.

The database is randomly split into three non-overlapping groups of speakers, one for background training, one for development and one for evaluation. Each group contains a balanced number of male and female speakers: 50/47 for the background set, 50/47 for the development set and 57/49 for the evaluation set.

The RSR2015 database is one of the largest publicly available database which targets to text-dependent speaker recognition [12, 13].

B. *Result of Logistic Regression*

In order to see the system performance, we tested by 30 male speakers from RSR_2015, we varied in every time the values of learning rate parameter (Lr).

We present in the curve in fig. 6 below the results of the error test classification obtained using logistic regression.

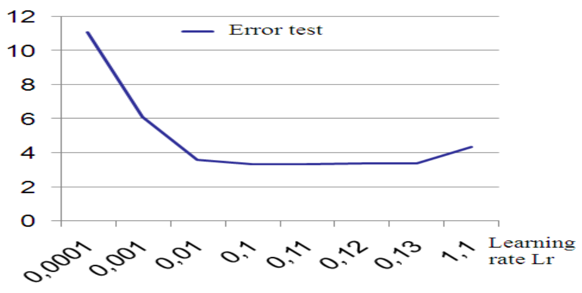

Fig. 6. Error classification test obtaining in varying learning rate

By varying the learning rate logarithmically yielded the best result by using the learning rate Lr = 0.1 and this is achieved in the shorter run time and we try to vary the learning rate values around 0.1. We note that if Lr is in the interval

[0.10, 0.13], the minimum classification error rate is around 3.5% is obtained.

Therefore, in the next tests, we fix the learning rate at 0.13.

C. Result of Multi Layer Perceptron

To assess the efficacy of MLP based ASR, we tested also with 30 male speakers and we use 512 neurons for each layer in order to seek the best numbers hidden layers. We obtained the results presented by the table 1 below.

As deduced from the previous tests, we fixed the learning rate at Lr=0.13.

TABLE I. RESULTS OF MLP OBTAINED IN VARIYING THE NUMBERS OF HIDDEN LAYERS

Size of hidden layer	Nbre of Hidden layers	Test classification error	Run time
512 Neurons/ layers	2	5.257%	35.12 min
	3	**5.242%**	**122.9 min**
	4	6.164%	213.96 min
	5	6.378%	341.26 min
	6	6.678%	519.34 min

The best results are obtained using three hidden layers. Adding an extra layer doubles the running time of the program. The classification error rate increases each time we add a hidden layer.

The run time is the time taken by the algorithm. It depend of the note book computer who his features is mentioned above.

Using only 3 hidden layers and varying in each case their sizes, we presented results in the following table 2:

TABLE II. RESULTS OF MLP_3LAYERS OBTAING IN VARYING NUMBER OF NEURONS IN EACH HIDDEN LAYER

Nbre of hidden layers	Size of hidden layers	Test classification error	Run time
3	64	5.328%	6.58 min
	128	5.464%	13.84 min
	256	**4.900%**	**37.92 min**
	512	5.242%	122.9 min
	1024	5.514%	414.65 min

Using 256 neurons for each layer gave the best results, we take into consideration that the use of sizes of layers equal to the input size gives the best results according to minimum error test and shorter running time. But when we use a number of neurons greater than 256 neurons this leads the decrease of our ASR system performance.

D. Results of stacked denoising autoencoder

Also those tests are performed on the text dependant data base RSR2015. With 30 classes (30 male speakers), we firstly tested the system using two hidden layers by varying each time its number of neurons and we obtained the following results shown in table 3 below:

TABLE III. RESULTS OBTAING IN IN VARYING NUMBER OF NEURONS IN EACH HIDDEN LAYER (HIDDEN LAYERS SIZE)

Nbre of hidden layer	Hidden layer size	Test classification error	Run time
2	64	14.535%	3.2 min
	128	12.135%	6.41 min
	200	10.171%	8.97 min
	512	**3.985%**	**31.96 min**
	1024	**3.421%**	349.37 min

We obtained the best results by increasing the size of the layers around more than 500 neurons and we get 3.421% error test using 1024 neurons in each hidden layer but it took a long running time. The best error test/running time ratio is obtained by assigning the number of neurons of each hidden layer equal to 512.

In this second test, we added a third hidden layer and we varied the number of neurons of each hidden layer. We noted the following results in table 4:

TABLE IV. RESULTS OBTAING IN VARYING THE THREE HIDDEN LAYERS SIZE

Nbre of hidden layer	Size of hidden layer	Test classification error	Run time
3	64	10.907%	3.73 min
	128	7.042%	9.44 min
	256	5.050%	26.24 min
	512	**3.971%**	**82.18 min**
	1024	**3.657%**	**261.66 min**

Using a number of neurons equal to 512 , the result obtained by adding a third hidden layer is better than the results obtained in the other tests in all cases except that using 1024 neurons to the hidden layer size we got better results using 2 hidden layers but unfortunately we spent a lot of running time. As we have seen, the best error test/ running time ratio is obtained by assigning the hidden layers size equal to 512 using two hidden layers.

Finally, we bring together the best results obtained in table 5 presented below:

TABLE V. MINIMUM ERROR TEST CLASSIFICATION OBTAINED UNDER RSR2015_30MALE_SPEAKERS

Reg_logistic	MLP_3layer_256units	SDA_2layers_1024units
3.500%	**4.900%**	**3.421%**

The minimum error rate classification is acquired with the Stached Denoising Algorithm.

V. CONCLUSION & FUTURE WORK

In this work, we implemented three algorithms of deep learning. The first one is the logic regression, the second one is the Multilayer Perceptron and the third one is the Stacked Denoising Autoencoders. To find the best recognition performance and to better optimize automatic recognition algorithms, we tested our speaker recognition system under the RSR2015 database in text-dependent mode. During our test we varied in every time, values of neural networks parameters, number of neurons and number of hidden layers. We tested the first logic regression algorithm by varying the learning rate logarithmically and we obtained the best result with minimum error test by using the learning rate around Lr = 0.1 and this is achieved in the shorter running time. Then we fixed Lr=.0.13 and we testes the second MLP algorithm, and we varied the number of hidden layers and the neuron number of each layer. We used 128 neurons in each layer and only three hidden layers. We tested the third Stacked Denoising Autoencoder algorithm, we have seen, the best error test/running time ratio is obtained by assigning the hidden layers size equal to 512 using two hidden layers.

Despite the performance and the powerful that are proved by MLP algorithm, we have obtained the best results error test/running time ratio performed by the third algorithm, Stacked Denoising Autoencoder algorithm, which it proved good performance through the results obtained.

We will advance in work and we will proceed to the next step. We will implement a speaker recognition system based on Deep Belief Networks approach (DBN) and also the Convolutional Neural Network (CNN).

The results obtained and acquired in this work will remain and will help us in the next work to assess and to evaluate the different algorithms settled. It will be very useful to compare the performance between them and to benefit from their advantages to create an efficiently hybrid ASR system.

REFERENCES

[1] Y. Bengio, "Learning deep architecture for AI", Foundation and Trends in Machine Learning, 2009.

[2] Y. LeCun, Y. Bengio and G. Hinton" Deep learning", Review Nature, Vol 521, pp.436-444,2015.

[3] J. Hennebert, “Hidden Markov models and artificiel neural Networks for speech and speaker recognition",PHD Thesis no.1860,EPFL , 1998.

[4] C.M. Bishop(1995),“Neural networks for pattern recognition",Oxford: Oxford University Press,1995.

[5] DARPA Neural Network Study (U.S.), “DARPA Neural Network Stdy", Widrow, Morrow, and Gschwendtner (Eds.), AFCEA Intl,1988.

[6] S. Haykin, “Neural networks:A Comprehensive Foundation" Prentice Hall PTR Upper Saddle River, NJ, USA, 1998.

[7] Y. Lecun, L., G. Marcus, M. Minsky, G. Hinton, G. Marcus, and S. Pinker, “Le ‘ deep learning ’ pour tous ?,” pp. 10–13,2015. [Online].Available:http://www.internetactu.net/2014/10/02/le-deep-learning-pour-tous/

[8] “Neural network,” 2015. [Online]. Available: http://www.mu-sigma.com /analytics/thought_leadership/cafe-cerebral-neural-network.html.

[9] E.. Hervet, “Reseaux de neurones”, Moncton University, 2009.

[10] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley, and Y. Bengio, “Theano: new features and speed improvements,” pp. 1–10, 2012.

[11] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, “Theano: a CPU and GPU math compiler in Python,” 9th Python Sci. Conf., no. Scipy, pp. 1–7, 2010.

[12] A. Larcher, K. A. Lee, B. Ma, and H. Li, “Text-dependent speaker verification: Classifiers, databases and RSR2015,” Speech Commun., vol. 60, pp. 56–77, May 2014.

[13] A. Larcher, K. A. Lee, B. Ma, and H. Li, “The RSR2015:Database for Text Dependent Speaker Verification using Multiple Pass-Phrases,” In. Interspeech 2012, 13th Annual Conference of the International Speech Communication Association, Portland, Oregon, USA, September 9-13, 2012.

[14] M. A. Nielsen, “Neural Networks and Deep Learning”, Determination Press, 2015.

[15] A. Larcher, K. A. Lee and S. Meignier, “An extensible speaker identification SIDEKIT in Python”, in International Conference on Audio Speech and Signal Processing (ICASSP), 2016.

[16] L. Bottou “Large-scale machine learning with SGD.” Proceedings of COMPSTAT’2010. Physica-Verlag HD, 2010. 177-186.

[17] S. Raschka, “Python Machine Learning”, Packt Publishing Ltd. September 24th , 2015 . [online] Available: http://sebastianraschka.com /faq/index.html

[18] L. Bouafif and K. Ouni, "A speech tool software for signal processing applications," 2012 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Sousse, 2012, pp. 788-791.

[19] K. Jayech, M. A. Mahjoub and N. Ghanmi, "Application of Bayesian networks for pattern recognition: Character recognition case," 2012 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Sousse, 2012, pp. 748-757.

[20] A. Hmich, A. Badri, A. Sahel and M. Moughit, "Discriminating coding applied to the Automatic Speaker Identification," 2012 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Sousse, 2012, pp. 44-49

[21] H. Chihi and N. Arous, "Recurrent Neural Network learning by adaptive genetic operators: Case study: Phonemes recognition," 2012 6th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT), Sousse, 2012, pp. 832-834.