The UVigo-GTM Language Verification Systems for the Albayzin 2010 Evaluation

Paula Lopez-Otero, Laura Docio-Fernandez, Carmen Garcia-Mateo

Department of Signal Theory and Communications, Universidade de Vigo
plopez@gts.tsc.uvigo.es, ldocio@gts.tsc.uvigo.es, carmen@gts.tsc.uvigo.es

Abstract

This paper describes the two systems submitted by the UVigo-GTM group for the Albayzin 2010 language verification evaluation. They were initially thought to perform speaker recognition and verification, so they use language-independent information to apply the algorithms to language verification treating the problem as a pattern recognition task. The principal system consists in a dimensionality reduction approach that transforms the data into a lower dimensionality subspace by performing a two-stage process to reduce the dimensionality and extract a discriminative subspace. The alternative system uses the Nonnegative Matrix Factorization to obtain a representation of the data in terms of a set of basis functions, obtaining the utterances represented as a feature vector of lower dimensionality.
Index Terms: fishervoices, language identification.

1. Introduction

Language recognition is a task that may take into account different kinds of information: linguistic and phonetic information, which is language-dependent, and acoustic information, which is language-independent. The use of language-dependent information implies the training of language models and, sometimes, an in-depth analysis of the target language. On the other hand, the language-independent information does not require any prior knowledge of the target languages, they are all modeled in the same way without taking into account their own characteristics.

The systems developed for the Albayzin 2010 Language Verification Evaluation do not require language-dependent information for some reasons. On the one hand, the less information the system requires to decide among the target languages, the faster the decision. On the other hand, the language recognition task can be thought as a pattern recognition problem, where a sample of data has to be classified into one of the possible classes. This brings the possibility of using speaker recognition algorithms to perform language recognition because, from a pattern recognition perspective, both problems are the same: there are several classes (speakers, languages), and a series of utterances that have to be classified into these classes.

The main system that is described in this paper is based on a face recognition approach [2], where a transformation of the data is performed in order to reduce the dimensionality and to find a discriminative subspace. This Fisherface reduction technique was also applied in speech processing to perform speaker clustering [3]. In previous work [1], this technique was modified and employed to perform speaker identification, and this representation of the speech utterances was named after Fishervoice. As the Fishervoice approach consists in a transformation of the data in order to classify it into different classes (speakers), its application in language recognition is straightforward, and is described in detail in section 4.

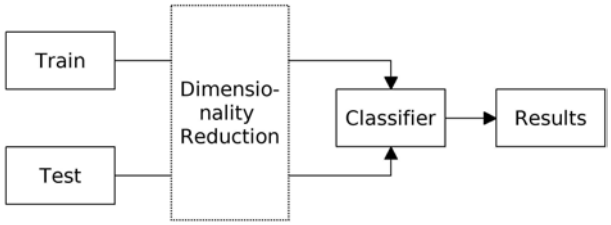

Figure 1: *Fishervoice Language Identification System.*

The alternative system presented in section 5 describes an approach called NMFvoices. It is based in the Nonnegative Matrix Factorization (NMF) [1], a technique to decompose a matrix into two new ones, where one represents a set of basis vectors and the other one the corresponding weights to obtain the original matrix with these basis vectors. NMF is used in this work to represent the speech utterances as the weights obtained with this factorization, obtaining a representation of the data using feature vectors of lower dimensionality.

2. The Language Verification System

The main structure of the language verification system is shown in Fig. 1. The system receives two inputs: a dataset to train the system and to build the models for each different language, and another one to test the performance of the system. A dimensionality reduction step is performed, where the approaches described here are applied, and after that a classifier decides if an utterance is spoken in the target language or not. This decision is taken by doing language identification, the utterance is assigned to the most likely language.

3. Datasets

Three different datasets are necessary to perform speaker verification with the proposed algorithm:

- A train dataset to train a GMM-UBM ($A_{GMM-UBM}$).
- A train dataset (A_{train}) to build a model of the different classes (languages).
- A test dataset (A_{test}) to test the performance of the system: each of its utterances has to be assigned to a class.

This three datasets are composed of several utterances of speech in the different target languages spoken by different speakers. Each utterance is represented as a matrix of dimension $m \times n$ that consists in the means obtained by performing a Maximum a Posteriori (MAP) adaptation of the GMM-UBM

to the utterance. Thus, m is the number of gaussians of the GMM, and n is the dimension of the feature space. In this case, the acoustic features that represent the speech utterances are 12 Mel-frequency Cepstral Coefficients (MFCC), extracted using a 25 ms Hamming window at a rate of 10 ms per frame, and augmented with the normalized log-energy and their delta and acceleration coefficients. Thus, the dimension of the feature space (n) used in these experiments is 39. This features are normalized in order to fit a zero mean and a unit variance distribution.

The whole datasets A_{train} and A_{test} are represented as tridimensional matrices of dimension $m \times n \times L_{train}$ and $m \times n \times L_{test}$ respectively, where L_{train} is the number of utterances in A_{train}, and L_{test} is the number of utterances in A_{test}.

4. Fishervoices

4.1. The Fishervoice technique

The Fishervoices technique transforms a dataset A into a set C which represents the same information but with less features, i.e. it reduces the dimensionality of the subspace. To carry out this transformation, two transformation matrices X and Y have to be computed.

Matrix X is obtained by performing 2D-PCA in the A_{train} dataset. Three scatter matrices (between-class D_b, within-class D_w and total D_t) are defined:

$$D_b = \sum_{i=1}^{c} P_i (M_i - M)^T (M_i - M) \quad (1)$$

$$D_w = \sum_{i=1}^{c} \sum_{j,j \in i} (A_{train_j} - M_i)^T (A_{train_j} - M_i) \quad (2)$$

$$D_t = D_b + D_w \quad (3)$$

where c is the number of different languages (classes) in A_{train}, P_i is the a priori probability of the ith class, M_i is the mean matrix of the ith class ($i = 1, 2, \cdots, c$), M is the total mean matrix of A_{train}, and A_{train_j} is the $m \times n$ matrix of the j^{th} segment in A_{train}. Thus, M represents the mean matrix of the whole set, and M_i is the mean matrix of language i.

The eigenvectors and eigenvalues of the total scatter matrix D_t are obtained, finding a matrix X that maximizes $J(X) = X^T D_t X$. The dimensionality reduction is achieved by dropping some of the eigenvectors in X. This is done by keeping only a percentage e_1 of the energy of the subspace E_X:

$$E_X = \sum_{i=1}^{n} \lambda_i \quad (4)$$

where λ_i is the i^{th} greatest eigenvalue of X. Finally, a matrix $X \in \Re^{n \times u}$ is obtained, where u is the number of eigenvectors needed to absorb a percentage e_1 of the energy of the subspace.

The matrix X is employed to transform the set A_{train} into a lower dimensionality subspace by doing $B_{train} = A_{train} X$. Then, a LDA discriminative subspace is computed, obtaining the transformation matrix Y. New between-class and within-class scatter matrices (R_b and R_w, respectively) are computed:

$$R_b = \sum_{i=1}^{c} P_i (L_i - L)(L_i - L)^T \quad (5)$$

$$R_w = \sum_{i=1}^{c} \sum_{j,j \in i} (B_{train_j} - L_i)(B_{train_j} - L_i)^T \quad (6)$$

where L is the total mean matrix of the set B_{train}, and L_i is the mean matrix of the ith class in that set.

The Fisher criterion is applied, thus, a matrix Y that maximizes $J(Y) = \frac{Y^T R_b Y}{Y^T R_w Y}$ is obtained. As before, only a percentage e_2 of the energy of the subspace E_Y is kept, obtaining a matrix $Y \in \Re^{m \times v}$. Then, the dataset B_{train} is transformed into the final subspace $C_{train} = Y^T B_{train}$, where $C_{train} \in \Re^{v \times u}$.

Once X and Y are obtained, they are used to project A_{test} to the new subspace by doing

$$C_{test} = Y^T B_{test} = Y^T A_{test} X \quad (7)$$

4.2. Classifier

After performing the transformation of the datasets, the initial utterances are obtained, but represented in a space of lower dimensionality: the initial number of features to describe an utterance was $m \cdot n$, while after the transformation of the feature space it is reduced to $v \cdot u$, where $v \leq m$ and $u \leq n$.

The classification of each utterance in C_{test} is done by comparing these utterances with the models of the different languages C_{train}. An utterance S is compared to the different models by measuring the spatial distance between the utterance and the model:

$$T = \min_i d(C'_{test_S}, C'_{train_i}) \quad (8)$$

$d(.,.)$ is the euclidean distance between an utterance and a model (which is another utterance), C'_{test_S} is a vector that represents the utterance S in terms of a supervector obtained by concatenating the rows of C_{test_S}, and the same for C'_{train_i}.

The classifier in Eq. (8) decides which of the utterances in the model is spatially closer to the test utterance, assigning the language of the model to this test utterance. This is not an actual language verification system, it performs verification by performing language identification.

4.3. Processing speed

Three different processes had to be executed to run the experiments with the test dataset provided for the Albayzin 2010 Language Verification Evaluation. The CPU time necessary to run each process is shown below:

- Parameterization (with HTK-3.4): 722.83 s
- Normalization of the features (C code): 32.97 s
- Computing and generation of the results file (MatLab): 1464.96 s
- **TOTAL: 37 min 0.76 s**

$$\left.\begin{array}{l} 4992\ recognitions \\ \\ 2220.76\ s \end{array}\right\} 2.25\ recognitions/s$$

This processes were executed in a server with a processor Intel Xeon E5620 2.4 GHz and 18 GB of memory.

5. NMFvoices

5.1. The NMFvoices Technique

Nonnegative matrix factorization (NMF) is a dimensionality reduction technique employed over nonnegative data. Given a data matrix $V \in \Re^{\geq 0, F \times N}$, NMF finds a factorization

$$V \approx WH \quad (9)$$

where $W \in \Re^{\geq 0, F \times R}$ and $H \in \Re^{\geq 0, R \times N}$. R is the value that performs the dimensionality reduction, and it is usually chosen in a way that $FR + RN \ll FN$.

NMF is an iterative algorithm whose target is to reduce the euclidean distance between V and WH, its divergence, etc. In this paper, a multiplicative algorithm is employed, which is fast and easy to implement. Its update rules are:

$$H \leftarrow H \frac{W^T V}{W^T W H} \tag{10}$$

$$W \leftarrow W \frac{V H^T}{W H H^T} \tag{11}$$

The target of this iterative algorithm will be to minimize the euclidean distance $\|V - WH\|$.

The F row vectors of W can be interpreted as basis vectors, and the N column vectors of H would be the corresponding weights needed to obtain each of the vectors in V by combining the basis vectors.

5.1.1. Nonnegativity

A requirement of NMF is that matrices V, W and H have no negative or zero values, so a little adjustment of the data is performed in order to transform a matrix into a non-negative one. Given a matrix $M \in \Re^{I \times J}$:

$$M_+ = M \cdot \min_{i,j} m_{ij},\ i \leq I,\ j \leq J \tag{12}$$

where m_{ij} is the element (i, j) in matrix M, and M_+ is the obtained non-negative matrix. This adjustment will be applied if V or the initialization of W and H have negative or zero values.

5.1.2. Initialization of matrices W and H

The most usual way to initialize W y H is by doing it randomly. Nevertheless, as the matrices will be different for any trial of an experiment, it would be interesting to use a deterministic manner to initialize them. In this paper an initialization algorithm is proposed: given a matrix $V \in \Re^{F \times N}$, the R most different rows from this matrix will be selected, For each row f_i, the following distance measure is computed:

$$D(f_i) = \sum_{j \neq i, j=1}^{F} d(f_i, f_j) \tag{13}$$

where $d(f_i, f_j)$ is the euclidean distance between the rows f_i and f_j. Then, the initial matrix H will be composed by the R rows which obtained the higher values for $D(f_i)$.

Once H is obtained, W is computed by doing:

$$W = V H^T \tag{14}$$

5.2. Using W as a basis

A matrix V' can be represented by using the basis vectors obtained by performing NMF in another matrix V. V is decomposed into the two matrices W and H. Then, for matrix V', W' is initialized as $W' = W$ and $H' = W^T V'$. While executing NMF, the update rule (11) is not applied, so H' will be the only updated matrix, while W' will remain the same. Thus, as the weight matrices H' and H were obtained using the same basis vectors, they can be compared to each other.

5.2.1. NMF and its application in language recognition

In 5.1, it was explained that NMF makes it possible to decompose a matrix V into two matrices W, representing a set of basis vectors, and H, representing the corresponding weights. The algorithm to obtain the representation of the speech utterances in NMFvoices is as follows:

- Decompose the matrix A_{train} into two matrices W_{train} and H_{train}.
- Decompose the matrix A_{test} into two matrices W_{test} and H_{test}, but restricting the algorithm so that $W_{test} = W_{train}$ as explained in 5.2.
- The columns of H_{train} are the models for the possible languages, while the columns of H_{test} represent the test utterances that have to be assigned to the target languages.

5.3. Classifier

The procedure to perform language recognition is simple: each column of H_{test} will be compared to all the columns in H_{train}. The language of the column in H_{train} that is closer to the column in H_{test} is the one that will be assigned to that speech utterance. The classifier employed to compare the utterance is the same as in Eq. (8):

$$T = \min_i d(H_{test_S}, H_{train_i}) \tag{15}$$

where H_{test_S} is the column S of the matrix of the test weights, and H_{train_i} is the i^{th} column of the matrix of the train weights.

5.4. Processing speed

As in the Fishervoice system, three processes had to be executed to run the experiments with the test data. The CPU time necessary to run each process was:

- Parameterization (with HTK-3.4): 722.83 s
- Normalization of the features (C code): 32.97 s
- Computing and generation of the results file (MatLab): 3352.52 s
- **TOTAL: 1 h 8 min 28.32 s**

$$\left.\begin{array}{l} 4992\ recognitions \\ \\ 4108.32\ s \end{array}\right\} 1.215\ recognitions/s$$

This processes were executed in a server with a processor Intel Xeon E5620 2.4 GHz and 18 GB of memory.

6. Train and development data

Initially, the Fishervoice algorithm was thought to perform speaker identification, as in [1]. The BANCA database [7] [8] was used for this experiments. When it was first developed to perform language recognition, the database that was employed is the COST278 Pan-European Broadcast Database [4]. This database includes broadcast news programs in 9 different European languages (Belgian Dutch, Portuguese, Galician, Czech, Slovenian, Slovak, Greek, Hungarian and Croatian) featuring clean speech and noisy speech. Nevertheless, in some of the experiments a GMM-UBM trained with data extracted from the Transcrigal Database [5] was employed.

In the case of the NMFvoices approach, it was initially developed to perform speaker verification, and was successfully

tested using the BANCA database. After this, it was decided to transform it into a recognition system by changing the classifier, and it was tested with the development set of the KALAKA-2 database.

The only data employed to train the systems for the Albayzin 2010 Language Verification Evaluation was the database provided for that purpose, the KALAKA-2 database. This data is divided into three groups, namely *GMM*, *Devel* and *Test*. Attending to the datasets described in 3, the given data was distributed as follows:

- *GMM*: This is the only labeled data available, and includes clean and noisy utterances. Only the clean speech was employed, and it was divided into two groups of the same size: one employed to train the $GMM - UBM$ (corresponding to the dataset $A_{GMM-UBM}$) and the other one used to train the transformation matrices and, therefore, to build the models for the different languages (corresponding to the dataset A_{train}).
- *Devel*: All the data in this group was employed to tune the free parameters of the system, and to decide the best features for the algorithm. Thus, this corresponds to the dataset A_{test} in a first stage. The algorithm was run using different values for m, e_1 and e_2 in the case of Fishervoices, and different values of M and R in the case of NMFvoices. Different types of features were also tested(39 MFCCs, 39 MFCCs normalized to fit a zero mean and a unit variance distribution, and 39 MFCCs applying the M-norm after the MAP adaptation of the GMM-UBM). The parameters and features that obtained the best average cost were chosen:
  - Fishervoices: the best values were $M = 64$; $e_1 = 1.0$ and $e_2 = 0.5$ for clean speech, and $e_1 = 0.6$ and $e_2 = 0.8$ for noisy speech.
  - NMFvoices: the best values were $M = 32$; $R = 50$ for clean speech and $R = 100$ for noisy speech.
- *Test*: This data was employed to run the test that was submitted for the evaluation. Thus, in a stage after the tuning, this data corresponds to the dataset A_{test}.

7. References

[1] P. Lopez-Otero, L. Docio-Fernandez, C. Garcia-Mateo, "A Fishervoice-based Speaker Identification System", Proceedings of FALA 2010: VI Jornadas en Tecnología del Habla and II Iberian SLTech Workshop.

[2] X.Y. Jing, H.S. Wong and D. Zhang, "Face Recognition Based on 2D Fisherface Approach", Pattern Recognition, vol. 39, n. 4, pp. 707–710, 2006.

[3] S.M. Chu, H. Tang, T.S. Huang, "Fishervoice and Semi-supervised Speaker Clustering", IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 4089–4092, Los Alamitos, CA, USA, 2009.

[1] Daniel D. Lee and H. Sebastian Seung, "Algorithms for Non-negative Matrix Factorization", Advances in Neural Information Processing Systems (NIPS) vol. 13, 556–562, 2000. Online: `http://citeseer.ist.psu.edu/lee01algorithms.html`

[4] A. Vandecatseye, J.-P. Martens, J. Neto, H. Meinedo, C. Garcia-Mateo, J. Dieguez, F. Mihelic, J. Zibert, J. Nouza, P. David, M. Pleva, A. Cizmar, H. Papageorgiou, C. Alexandris, "The COST278 pan-European Broadcast News Database", Fourth International Conference on Language Resources and Evaluation (LREC '04), Lisbon, Portugal.

[5] C. Garcia-Mateo, J. Dieguez-Tirado, A. Cardenal-Lopez, L. Docio-Fernandez, "Transcrigal: a bilingual system for automatic indexing of broadcast news", Proc. International Conference on Language Resources and Evaluation, Lisbon, Portugal, May 2004.

[7] E. Bailly-Baillire, S. Bengio, F. Bimbot, M. Hamouz, J. Marithoz, J. Matas, K. Messer, F. Poree, B. Ruiz, "The BANCA Database and Evaluation Protocol", 2003.

[8] The BANCA Database Website, Online: `http://www.ee.surrey.ac.uk/CVSSP/banca/`