A DISTANCE MEASURE BETWEEN COLLECTIONS OF DISTRIBUTIONS AND ITS APPLICATION TO SPEAKER RECOGNITION

Homayoon S. M. Beigi, Stéphane H. Maes and Jeffrey S. Sorensen

Human Language Technologies Group
IBM Research, T.J. Watson Center
P.O. Box 218, Yorktown Heights, NY 10598
EMail: beigi@watson.ibm.com

ABSTRACT

This paper presents a distance measure for evaluating the closeness of two sets of distributions. The job of finding the distance between two distributions has been addressed with many solutions present in the literature. To cluster speakers using the pre-computed models of their speech, a need arises for computing a distance between these models which are normally built of a collection of distributions such as Gaussians (e.g., comparison between two HMM models). The definition of this distance measure creates many possibilities for speaker verification, speaker adaptation, speaker segmentation and many other related applications. A distance measure is presented for evaluating the closeness of a collection of distributions with centralized atoms such as Gaussians (but not limited to Gaussians). Several applications including some in speaker recognition with some results are presented using this distance measure.

1. INTRODUCTION

A practical solution to the problem of computing a meaningful distance between two collections of statistical distributions, although very useful, is not available in the literature. Let us consider a possible model for speech as being a collection of distributions (e.g., Gaussian distributions). To compute the distance between two speakers, we should be able to compute the distance between two such models. Normally, in speech related applications, here are two problem to deal with. One is when a speech model is computed based on many frames of speech and stored in terms of its parameters. Then, in a comparison of new speech (test data) with this model (Prototype), each frame of speech is evaluated by computing the distance or likelihood of the frame to the model. The distance (likelihood) scores are then acted upon in relation to other such distances (likelihoods) to come up with a decision.

A second problem is one of ranking different models which have been created from frames of speech. In this problem, many models of speech are created and we would like to know the distance between these models without having access to the original frames from which the models were built. This problem arises in many speech applications such as *speaker recognition*, *speech segmentation*, *speaker classification*, etc. One notion that makes this job a difficult task is the fact that the number of distributions in each model may be different with another model. It is, therefore, not trivially apparent how to compute such a distance.

The idea behind the distance measure presented in this paper is the computation of the total distance measure between models based on the cumulative distances of the closest distributions between the two models. A provision is also given for treating two models with different number of distributions. The next section introduces the distance measure and an algorithm for computing it between two models. Then, a set of applications are discussed in *speaker classification*, *speech segmentation* and *speaker verification*.

Then, a set of results are given for these applications with the use of the proposed distance measure. Finally, a conclusion is drawn on the pros and cons of the proposed distance measure.

2. DISTANCE MEASURE

Consider models 1 and 2 represented in 1 with 3 and 4 distributions respectively. As we mentioned in the introduction, the distributions from the two models are paired up as well as possible. With no loss of generality, let us consider the specific case where these distributions are multi-dimensional and Gaussian in nature and were derived from a set of multi-dimensional feature vectors. For each distribution, the counts are written as c_1 and c_2 based on models 1 and 2 respectively. Please note the three distributions on the lower left corner of figure 1. In this case, one distribution from model 1 has been paired up with two distributions from model 2. This is because of the fact that model 2 has one more distribution present than model 1. For the sake of generality, let us allow directional distance measures between any two distributions. These distance measure are labeled d_1 and d_2 depending on if they measure the distance from a distribution in model 1 to one in model 2 or from model 2 to model 1.

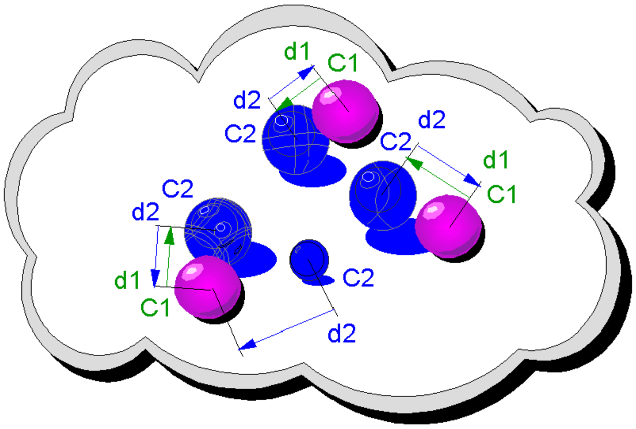

Figure 1: Distance Computation between two Collections of Distributions

The choice of the distance measure used between two distributions is left to the user and should include considerations for the nature of the problem. Three typical distance measure which may be used with this algorithm are given in equations 1- 3, Euclidean, Mahalanobis and Kullback-Leibler, respectively.

Basically, the proposed distance measure would be the sum of all the d_1 distances shown in figure 1 multiplied by the corresponding counts, c_1 plus all the d_2 distances multiplied by the counts, c_2 divided by the sum of all the counts used in the computation. Note that the small distribution in the lower left corner of figure 1 is only counted one (going from model 2 to model 1). This is done to reduce the effect of outliers if present. Imagine that model 2 may be created from the same data as model 1 was created from, with the addition of some noise. Given the new data and also because of some random effects such as those associated with initializing the seeds of a bottom-up clusterer (e.g., k-means), model 2 may end up with slightly different distributions with one of its clusters being split up into two. Then, the distance measure should show some robustness to the fact that a new cluster has been generated. For this reason, it is desirable to consider the clusters which paired up more than the ones left out. This is done since the counts from the small cluster in the figure are used only once from the abandoned cluster to the closest cluster in the other model. Euclidean:

$$d_E = \|\mu^{(i)} - \mu^{(j)}\|_{\mathcal{E}} \tag{1}$$

Mahalanobis:

$$d_M = (\mu^{(i)} - \mu^{(j)})^T \Sigma^{-1} (\mu^{(i)} - \mu^{(j)}) \tag{2}$$

Kullback-Leibler:

$$\xi_l = ({\mu_l}^{(i)} - {\mu_l}^{(j)})^2$$
$$d_{kl} = \frac{{\sigma_l}^{(i)}}{{\sigma_l}^{(j)}} + \frac{{\sigma_l}^{(j)}}{{\sigma_l}^{(i)}} + \frac{\xi_l}{{\sigma_l}^{(i)}} + \frac{\xi_l}{{\sigma_l}^{(j)}} \tag{3}$$

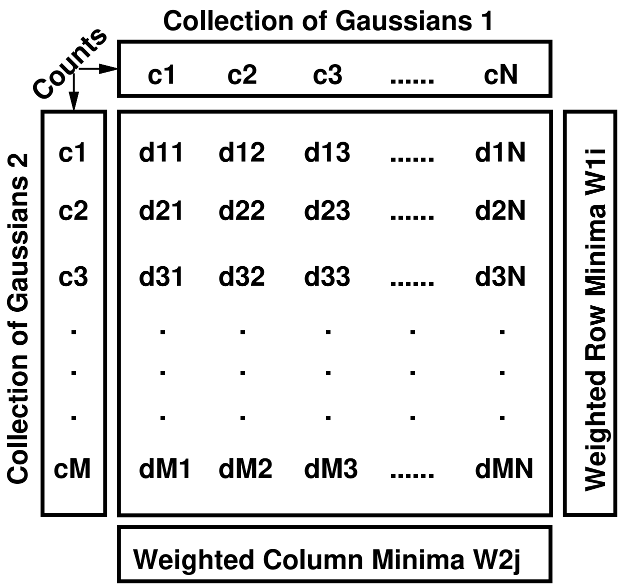

Figure 2: Distance Computation between two Collections of Gaussian Distributions

Figure 2 shows a picture for a systematic computation of the proposed distance measure between models. Let $c_i^2 \quad (i = 1, \cdots, M)$ denote the counts for the M clusters in model 2. Likewise, $c_j^1 \quad (j = 1, \cdots, N)$ are the counts for the N clusters in model 1. d_{ij} would be the distance between cluster i in model 2 and cluster j in model 1. d_{ij} may be obtained from any of equations 1 to 3. Once all the d_{ij} are computed, the minima of the row elements in the matrix given by figure 2 are multiplied by the corresponding counts, c_i^1, for those rows and the results are stored in W_1^i. The same things is done for the columns in the matrix and the results are stored in W_2^j.

Then, equation 4 is used to compute the distance D which is the proposed distance between two collections of distributions, model 1 and model 2.

$$D = \frac{\sum_{i=1}^{M} W_i^1 + \sum_{j=1}^{N} W_j^2}{\sum_{i=1}^{M} c_i + \sum_{j=1}^{N} c_j} \tag{4}$$

3. APPLICATIONS

Once the distance measure is established (equation 4), quite a number of applications may be considered. The main application which enables many other experiments is speaker classification.

3.1. Speaker Classification

Speakers in a pool may be clustered and their models stored in a hierarchical fashion. This is made possible by the creation of the proposed distance measure. A bottom-up clustering may be done on speakers using their models and with no need for the original data from which those models were derived. Once such clustering is done, a few bins are created with a collection of speaker models in each. This is useful in an array of different applications. The first application of this classification is as follows. For each of the generated classes of speaker models, a recognition prototype may be trained and stored. Once a new speaker comes in, his/her first few seconds of speech may be used to find the class of speakers which is closest to that speaker and the corresponding prototype may be used. [3]

3.2. Speaker Segmentation

Speaker clustering is also used in speaker segmentation. Reference [1] describes in detail a speaker segmentation problem in which a stream of speech and non-speech is segmented into pieces corresponding to music, noise, and different speakers. The segmentation algorithm described in [1], first segments the stream into small pieces based on some other distance measure. Then, these over-segments pieces are combined to create fewer classes. These merging is done based on the distance measure given by equation 4. The idea behind this clustering is to pull together all the speech segments from the same speaker into one or just a few clusters so that a training model would be built for each speaker. Later, in a decoding session, an action similar to that described in the previous section is taken to decode the incoming speech with a more suitable model. [1, 4, 5, 6]

3.3. Speaker Verification

One field of speaker recognition is *speaker verification* in which a speaker makes a claim on his/her identity and a sample of his speech is analyzed by the computer to evaluate the nature of the claim. If his/her speech accurately matches the claim, he/she is authenticated and otherwise rejected. Figure 3 shows a block diagram of this procedure. In the diagram, *Cohort* means the collection of speakers whose speech productions are closest to the speech of the claimed person.

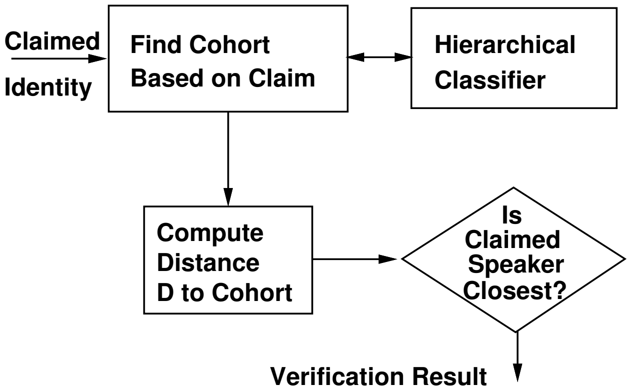

Figure 3: Speaker Verification

One way this problem may addressed is to start with a collection of speech samples for all the individuals in the database and build a hierarchical database using the distance measure given by this paper. The speakers are clustered in a hierarchical fashion and the parameters of the new structure are stored. In performing the authentication task, a claimant produces his name which is used to come up with the $\mathcal{P}$ closest speakers to the speaker with the claimed identity. Please note that the proposed distance measure of equation 4 was used to create the hierarchy which includes the information about the *cohort* of $\mathcal{P}$ speakers. The claimant's speech segment is then used to create a model and that model is compared against the $\mathcal{P}$ models in the *cohort*. If the closest speaker in the set has the same identity as claimed, the speaker is claimant is authenticated and otherwise rejected.

The same distance measure may also be used to do *speaker identification* which is considered to be another division of *speaker recognition*. [2, 7]

4. RESULTS

Figure 4 shows a table of results obtained using the proposed distance measure in performing speaker recognition. The first column shows the name of the data used in performing the test. On the average, in all the

Corpus	# of Speakers	False Rejection	Identific. Error
BOS	10	0.00 %	0.00 %
WSJ	283	0.09 %	1.5 %
IBM	60	1.67 %	2.92 %
IBMN	60	5.06 %	6.75 %

Figure 4: Speaker Verification Results

cases, about 30 seconds of speech was used for enrollment purposes and an average of another 10 seconds was used for testing. **IBMN** denotes the same data as **IBM** with the addition of a noise with a ratio of $19dB$. This was done to test robustness to noise. Depending on the total number of speakers in the database, a cohort of size 8 or 16 was chosen to perform the speaker verification test. In speaker identification, the test speech was compared to all the speakers in the database. The identification was also performed using a frame-by-frame distance measure based on voting techniques [2, 7]. Results show that the accuracy of the frame-by-frame system starts out higher when an amount equivalent to 1 second of test data is used. However, as the amount of test data is increased to near 10 seconds, the method described in this paper surpasses the frame-by-frame performance by considerable amounts. for a more quantitative comparison see [7].

5. CONCLUSION

In this paper we have proposed a new distance measure for measuring the similarity between speakers by using the statistical models built from their speech. We have compared this approach to using a frame-by-frame approach where the frames of speech from one stream are compared to the model created from another stream. Like anything else in nature, there are pros and cons to this distance measure. This distance measure is very useful for comparing models such as two HMMs which are usually made of a collection of Gaussian or other centralized distributions. This allows speaker clustering and classification which may be used for a quick speaker adaptation, speech segmentation and speaker recognition. However, the problems with this distance measure are the following. One would need to receive the whole utterance to be able to do a distance measurement. This is due to the fact that a model should be build of the entire or a significant portion of the speech to be able to compare it to other models. This problem does not exist with a frame-by-frame technique since the distance between each frame with the stored models may be computed as the frames are received. In addition, to be able to have considerable results, more data would have to be gathered to insure adequate modeling of the stream (i.e., a good estimate of the means and variances or any other related parameters). However, once enough data is available, this distance measure shows equivalent and sometimes better results when compared to a frame-by-frame distance measure. It has shown to perform better under more noisy environments. Also, since it is a parametric system, it is often faster for comparing speech segments especially when this comparison is repeated with the same data.

6. REFERENCES

[1] Homayoon S. M. Beigi and Stéphane H. Maes, "Speaker, Channel and Environment Change Detection", *World Automation Congress, ISSCI98*, Anchorage, Alaska, May 18-22, 1998.

[2] Stéphane H. Maes and Homayoon S. M. Beigi, "Open SESAME! Speech, Password or Key to Secure Your Door?", *Asian Conference on Computer Vision*, Hong Kong, Jan. 8-11, 1998.

[3] Yuqing Gao, Mukund Padmanabhan and Michael Picheny, "Speaker Adaptation Based on Pre-Clustering Training Speakers", *EuroSpeech*, Rhodes, Greece, , Sep. 22-25, 1997, Vol. 4, pp. 2095-2098.

[4] L. Heck and A. Sankar, "Acoustic clustering and adaptation for improved speaker recognition", *Proceedings of Speech Recognition Workshop, ARPA*, "Chantilly, VA, Feb. 1997.

[5] H. Jin and F. Kubala and R. Scwartz, "Automatic speaker clustering", *Proceedings of Speech Recognition Workshop, ARPA*, "Chantilly, VA, Feb. 1997.

[6] M. A. Sigler and U. Jain and B. Raj and M. Stern, "Automatic segmentation, classification and clustering of broadcast news audio ", *Proceedings of Speech Recognition Workshop, ARPA*, "Chantilly, VA, Feb. 1997.

[7] Homayoon S. M. Beigi, Stéphane H. Maes and Jeffrey S. Sorensen, "IBM Frame-by-Frame Speaker Recognition Technology", *Speaker Recognition and its Commercial and Forensic Applications*, Avignon, France, Apr. 20-23, 1998.