Neural speech turn segmentation and affinity propagation for speaker diarization

Ruiqing Yin, Hervé Bredin, Claude Barras

LIMSI, CNRS, Univ. Paris-Sud, Université Paris-Saclay, F-91405 Orsay, France

firstname.lastname@limsi.fr

Abstract

Speaker diarization is the task of determining *"who speaks when"* in an audio stream. Most diarization systems rely on statistical models to address four sub-tasks: speech activity detection (SAD), speaker change detection (SCD), speech turn clustering, and re-segmentation. First, following the recent success of recurrent neural networks (RNN) for SAD and SCD, we propose to address re-segmentation with Long-Short Term Memory (LSTM) networks. Then, we propose to use affinity propagation on top of neural speaker embeddings for speech turn clustering, outperforming regular Hierarchical Agglomerative Clustering (HAC). Finally, all these modules are combined and jointly optimized to form a speaker diarization pipeline in which all but the clustering step are based on RNNs. We provide experimental results on the French Broadcast dataset ETAPE where we reach state-of-the-art performance.

Index Terms: speaker diarization, re-segmentation, LSTM, affinity propagation

1. Introduction

Speaker diarization is the task of determining *"who speaks when"* in an audio stream that usually contains an unknown amount of speech from an unknown number of speakers [1, 2]. While speech and speaker recognition systems have improved enormously thanks to deep learning approaches, speaker diarization systems have yet to fully take advantages of these new techniques. This may be explained by the fact that speaker diarization is an unsupervised classification task difficult to address with (mostly supervised) deep learning approaches.

Speaker diarization systems are usually built as the combination of four main stages. First, non-speech regions such as silence, music and noise are removed by speech activity detection (SAD). Next, speech regions are split into speaker-homogeneous segments by speaker change detection (SCD), later grouped according to the identity of the speaker thanks to unsupervised clustering approaches. Finally, speech turn boundaries and labels are (optionally) refined with a re-segmentation stage.

In conventional speaker diarization systems [3, 4], GMM-based SAD and sliding windows-based SCD are widely used. In GMM-based SAD, GMMs for speech class and non-speech class are used to remove non-speech regions with Viterbi decoding. In sliding window-based SCD, one will use two adjacent sliding windows on the audio data and compute a distance between them, then decide (usually by thresholding the distance) whether the two windows originate from the same speaker. Gaussian divergence [5] and Bayesian Information Criterion (BIC) [6] have been used extensively in the literature to compute such a distance.

Since SAD and SCD can be modeled as supervised binary classification task (speech vs. non-speech for SAD, change vs. non-change for SCD), we did manage to improve those two stages using deep learning approaches. Both were addressed as frame-wise sequence labeling tasks using bi-directional LSTM on top of MFCC features [7, 8], leading to much better performance than traditional methods. The first contribution of this paper (introduced in Section 2) is to adapt this LSTM-based sequence labeling framework to the case of unsupervised re-segmentation stage. Traditionally, a GMM is trained for each cluster. The audio is then re-segmented through a Viterbi decoding [9]. Several alternation of clustering and re-segmentation steps are usually performed. Post-processing segment boundaries using the output of a word or phone recogniser also improves the overall diarization output [9].

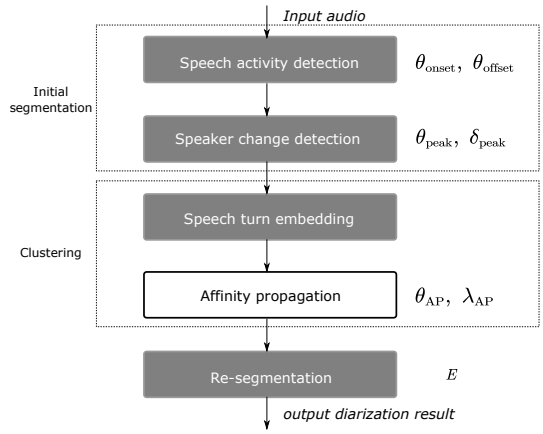

Figure 1: *Proposed diarization pipeline and hyper-parameters. Modules with gray background are based on neural approaches.*

Our second contribution concerns the clustering step and is described in Section 3. Most work rely on variations of Hierarchical Agglomerative Clustering (HAC) approaches [10] and use BIC, CLR or i-vector to compute similarity between clusters. In this paper, we propose to use affinity propagation clustering on top of neural speaker embeddings introduced in [11, 12]. While neural speaker embeddings have been used before with spectral clustering and K-means in [13], the authors only deal with telephone conversation with exactly 2 speakers. Similarly, an affinity propagation variant has been introduced in [14] for speaker diarization but it is also supervised by the number of speakers and relies on standard statistical models to compute speaker similarities.

Our third and final contribution is the combination and joint optimization of all these modules into a speaker diarization system where all but the clustering stage are based on RNNs: LSTM-based SAD, LSTM-based SCD, LSTM-based speaker embedding, and LSTM-based re-segmentation. The full pipeline is depicted in Figure 1. Experiments on the ETAPE

dataset are summarized in Section 4 and state-of-the-art results are discussed in Section 5.

2. Sequence labeling based on LSTM

In this section, we generalize the approach used for speech activity detection in [15] and speaker change detection in [7] and show how it can also be applied to re-segmentation.

2.1. Principle

Let $\mathbf{x} \in \mathcal{X}$ be a sequence of feature vectors extracted from an audio recording (e.g. MFCC features): $\mathbf{x} = (x_1, \dots, x_T)$ where T is the length of the sequence. Let $\mathbf{y} \in \mathcal{Y}$ be the corresponding sequence of labels: $\mathbf{y} = (y_1, \dots, y_T)$ and $y_i \in \{0, \dots, K-1\}$ where K is the number of classes.

The objective is to find a function $g : \mathcal{X} \to \mathcal{Y}$ that matches a feature sequence $\mathbf{x}$ to the corresponding label sequence $\mathbf{y}$. We propose to model this function g using a stacked LSTM neural architecture (further described in Section 4.2) trained with cross-entropy loss. Short fixed-length sub-sequences (a few seconds, typically) of (otherwise longer and with variable length) audio files are fed into the model. This allows to increase the number of training samples and augment their variability.

At test time, audio files are processed using overlapping sliding windows of the same length as above. For each time step i, this results in several overlapping sequences of K-dimensional (softmax-ed) scores, which are averaged to obtain the final score of each class.

2.2. Initial segmentation

The initial segmentation step aims at removing non-speech regions and splitting it into speaker-homogeneous segments. It is composed of two stages: speech activity detection (SAD) and speaker change detection (SCD).

SAD is the direct application of the above principle with $K = 2$ classes: $y_i = 1$ for speech, $y_i = 0$ for non-speech. At test time, the sequence of speech scores is post-processed using two (θ_{onset} and θ_{offset}) thresholds for the detection of the beginning and end of speech regions [8].

SCD can also be addressed using the same principle with $K = 2$ classes: $y_i = 1$ if there is a speaker change during the ith frame, $y_i = 0$ otherwise. To balance the number of positive and negative samples during training, the positive class is increased artificially by labeling as positive every frame in the direct neighborhood (e.g. less than 50 milliseconds apart) of the actual change point. At test time, the sequence of speaker change scores is post-processed so that all local maxima on a sliding window of duration δ_{peak} exceeding a threshold θ_{peak} are marked as speaker change points [7].

2.3. Re-segmentation

Given the output of the clustering step (later discussed in Section 3), re-segmentation aims at refining speech segments boundaries and labels and is usually solved with a combination of GMMs cluster modeling and Viterbi decoding. Assuming the output of the clustering step predicts k different speakers, we propose (and this is our first contribution) to use the same principle as above with $K = k + 1$ classes: $y_i = 0$ for non-speech and $y_i = k$ for speaker k.

At test time, using the (unsupervised) output of the clustering step as its unique training file, the neural network is trained for a (tunable) number of epochs E and applied on the very same test file it has been trained on. The resulting sequence of K-dimensional scores is post-processed by choosing the class with maximum score for each frame. To stabilize the choice of the hyper-parameter E and make the prediction scores smoother, scores from the $m = 3$ previous epochs are averaged when doing predictions at epoch E.

While this re-segmentation step does improve the labeling of speech regions, it also has the side effect of increasing false alarms (i.e. non-speech regions classified as speech). Therefore, its output is further post-processed to revert speech/non-speech regions back to the original SAD output.

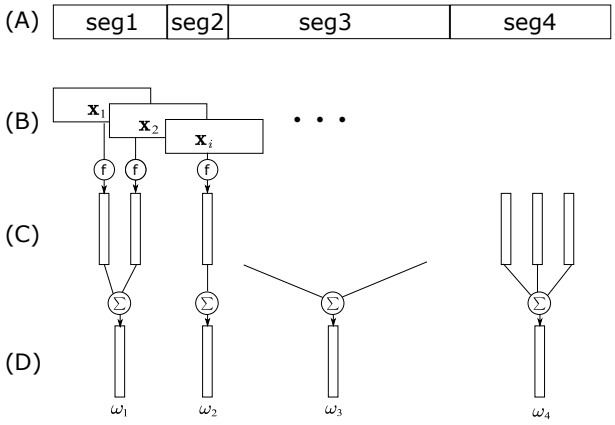

Figure 2: *Aggregation of fixed-length subsequence embeddings.*

3. Clustering

In this section, we introduce our second contribution: the combination of neural embeddings and affinity propagation for speech turn clustering.

3.1. Neural embedding

First, a neural embedding network $f : \mathcal{X} \to \mathbb{R}^D$ is trained using the triplet loss paradigm to embed speech sequences $\mathbf{x}$ into a D-dimensional space. The network architecture used is the one introduced in [11] and further improved in [15]. In the embedding space, two sequences $\mathbf{x}_i$ and $\mathbf{x}_j$ of the same speaker (resp. two different speakers) are expected to be close to (resp. far from) each other according to their angular distance:

$$\angle(\mathbf{x}_i, \mathbf{x}_j) = \arccos\left(\frac{f(\mathbf{x}_i) \cdot f(\mathbf{x}_j)}{\|f(\mathbf{x}_i)\| \cdot \|f(\mathbf{x}_j\|)}\right) \quad (1)$$

Figure 2 depicts how this network f – initially meant to process fixed-length (a few seconds, typically) speech sequences – can be used to embed variable-length speech segments coming from the initial segmentation step (A). The idea is to slide a fixed-length window (B) over the duration of the file, embed each of these subsequences (C), and then sum the embedding of all overlapping subsequences to obtain one embedding per initial segment (D). The embedding of segment i is denoted as ω_i in the next paragraph.

3.2. Clustering by Affinity propagation (AP)

The goal of SAD and SCD is to produce pure speaker segments containing a single speaker. The clustering stage is then responsible for grouping these segments based on speaker identities.

We chose the affinity propagation (AP) algorithm [16] for clustering. AP does not require a prior choice of the number of clusters contrary to other clustering methods [13]. All speech segments are potential cluster centers (exemplars). Taking as input the pair-wise similarities between all pairs of speech segments, AP will select the exemplars and associate all other speech segments to an exemplar. In our case, the similarity between i^{th} and j^{th} speech segments is the negative angular distance between their embeddings: $s(i,j) = -\angle(\omega_i, \omega_j)$

On the diagonal of the similarity matrix, $s(k,k)$ is set to the preference value θ_{AP}, a hyper parameter which influences the choice of ω_k as exemplar and thus the final number of clusters. AP clustering can be viewed as a "message passing" process between speech segments with two kinds of message: responsibility and availability. Responsibility $r(i,k)$ is a message sent from segment i to k that quantifies how well-suited x_k is to serve as the exemplar for x_i. Availability $a(i,k)$ is a message sent from segment k to i that represents how appropriate it would be for segment i to pick segment k as its exemplar. Responsibilities and availabilities are first initialized to 0 and then updated iteratively with a damping factor λ_{AP} which is introduced to avoid numerical oscillations. At each iteration, AP combines the responsibilities and availabilities to control the selection of exemplars. For segment i, the segment k which maximizes $r(i,k) + a(i,k)$ is the corresponding exemplar. The whole AP procedure terminates after a fixed number of iterations or after the exemplar stay unchanged for a chosen number of iterations. More details about AP can be found in [16]

4. Experiments

This section describes the experimental framework used in the rest of the paper. Open-source implementations of both the evaluation protocol and the proposed pipeline are available here: `github.com/yinruiqing/diarization_with_neural_approach`

4.1. Corpora and evaluation metric

Use	Dataset	Hours (speech)	nb. of speakers	
			Total	Per file
train	REPERE	59 (96%)	1758	9.6±6.1
dev.	ETAPE TV (dev.)	4 (93%)	93	8.0±4.4
test	ETAPE TV (test.)	4 (92%)	92	9.2±5.6

Table 1: *Datasets statistics with mean and standard deviation of speaker counts per file.*

Table 1 summarizes the data used for running the experiments. Both REPERE [17] and ETAPE TV [18] datasets contain recording of French TV broadcast with news, debates, and entertainment. The REPERE corpus was used for training the neural networks used in SAD, SCD, and embeddings. The ETAPE TV development subset was used for hyper-parameter tuning, while the ETAPE TV test subset was used for evaluation.

Speaker diarization systems are usually compared using diarization error rate (DER). In order to account for manual annotation imprecision, it is common practice to not evaluate short collars centered on each speech turn boundary (usually 250ms on both sides) and speech regions with more than one simultaneous speaker. In the case of the ETAPE TV dataset, these skipped regions (which are likely to be the most difficult to correctly classify) represent more than 20% of the total speech duration.

	Bi-LSTM	MLP	Output
SAD	16×2	16	2
SCD	32×2, 20×2	40, 10	2
Re-segm.	16×2, 16×2	16	K

Table 2: *Network architectures for (re-)segmentation models. Note: K depends on each file.*

Yet, precise "who speaks when" annotations are distributed alongside the ETAPE TV dataset. They were obtained using the following two-steps process: automatic forced alignment of the manual speech transcription followed by manual boundaries adjustment by trained phoneticians. Therefore, and unless otherwise stated, results reported in this paper do not use collars, nor do they skip overlapping regions. Practically, we use the open-source implementation of diarization error rate available in *pyannote.metrics* [19].

4.2. Implementation details

Feature extraction. Each part of the diarization pipeline shares the same set of input features extracted every 10ms on a 25ms window using Yaafe toolkit [20]: 19 mel-frequency cepstral coefficients (MFCC), their first and second derivatives, and the first and second derivatives of the energy (amounting to a total of 59 dimensions).

Sequence labeling. SAD, SCD and re-segmentation modules share a similar network architecture which stacks Bi-LSTM and a multi-layer perceptron. The architecture details are shown in Table 2. For example, the model for SCD is composed by two Bi-LSTM layers and 2 fully connected layers. *Bi-LSTM1* has 64 outputs (32 forward and 32 backward). *Bi-LSTM2* has 40 (20 each). The fully connected layers are 40- and 10-dimensional respectively. As described in Section 2, the final output layer dimension depends on the task.

Sequence embedding. Implementation details are identical to the ones used in [12]. Trained on REPERE dataset, 192-dimensional embeddings are extracted every 0.8s on subsequences of duration 3.2s.

4.3. Contrastive approaches

The proposed speaker diarization pipeline is compared to two alternative approaches. The first one is a variant of the proposed approach where the affinity propagation module is replaced by standard hierarchical agglomerative clustering. The second one, dubbed "S4D" in the rest of the paper, is a state-of-the-art system developed at LIUM using sidekit [21] and S4D [22] toolkits.

Hierarchical agglomerative clustering. This variant of the proposed pipeline relies on complete-link clustering [23] on top of the affinity matrix S based on angular distance defined in Section 3 and a distance threshold θ_{HAC} to stop merging clusters. Other linkages were tested (average, single) but found to lead to worse performance.

S4D system outputs were provided to us by LIUM and use the following pipeline. Segmentation based on Gaussian divergence first generates (short) pure segments. Adjacent segments from the same speaker are then fused based on the Bayesian Information Criterion (BIC), leading to (longer) speech turns. Hierachical clustering based on Cross-Likelihood Ratio then

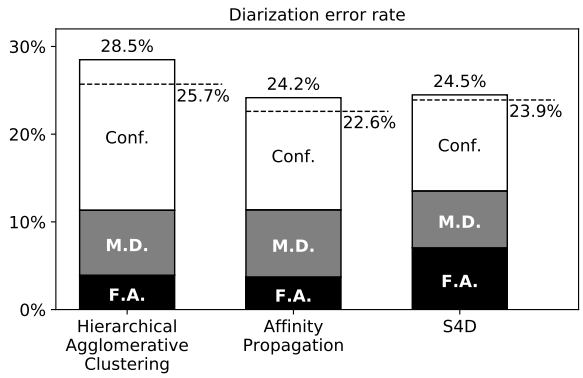

Figure 3: *Performance on ETAPE TV test set. Black bars, gray bars and white bars indicate false alarm, miss detection and confusion respectively. Dashed lines indicate performance when tuned on test set.*

groups them into pure clusters, further grouped into larger clusters using another i-vector-based clustering. Three clustering approaches were compared for this last step: regular HAC, graph-based clustering, and integer linear programing. Graph-based clustering was selected as the one leading to the best performance on the ETAPE TV development set. Finally, Viterbi decoding adjusts segment boundaries. Note that the trainable parts of this system were trained on a much larger dataset (ESTER [24], ETAPE [18], REPERE [17], and NIST RT03) than the one we used (REPERE only).

4.4. Joint optimization

While speaker diarization modules are usually tuned empirically and independently from each other, our third contribution consists in the joint global optimization of the whole diarization pipeline. More precisely, we use the Tree-structured Parzen Estimator hyper-parameter optimization approach [25] available in *hyperopt* toolkit [26] to automatically select the set of hyper-parameters that minimizes diarization error rate on ETAPE TV development set: θ_{onset} and θ_{offset} for SAD, θ_{peak} and δ_{peak} for SCD, θ_{AP} and λ_{AP} for affinity propagation (or θ_{HAC} for hierarchical agglomerative clustering). Note that re-segmentation hyper-parameter E was tuned separately as we wanted to carefully analyze its behavior, but it should ideally be optimized with the rest of the pipeline.

5. Results and discussions

Figure 3 summarizes the main experimental results. The proposed pipeline reaches state-of-the-art performance on the ETAPE TV dataset, though the difference with LIUM's S4D is not statistically significant (24.2% vs. 24.5%). However, switching from affinity propagation to hierarchical agglomerative clustering does degrade significantly the performance (24.2% vs. 28.5%). Comparison to results obtained when tuned directly on test set shows that there is room for improvement regarding the joint optimization of the proposed pipeline: LIUM's S4D tends to generalize better (ΔDER $=$ 1.6% vs. 0.6%). Nevertheless, affinity propagation beats hierarchical agglomerative clustering under this criterion (ΔDER $=$ 1.6% vs. 2.8%).

Table 3 shows the effect of the proposed re-segmentation step on the output of affinity propagation clustering: it improves both cluster purity and coverage, leading to an absolute decrease

	DER	Purity	Coverage
Before re-segm.	25.6%	81.8%	82.1%
After re-segm.	24.2%	83.4%	82.9%

Table 3: *Effect of re-segmentation on proposed pipeline*

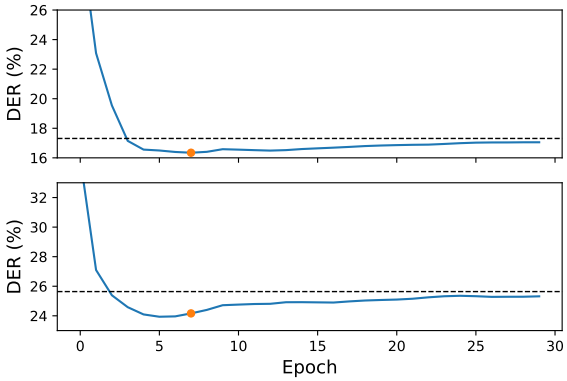

Figure 4: *Re-segmentation on development (top) and test sets (bottom). The best epoch on the development set is marked with an orange dot.*

of 1.4% in diarization error rate. Interestingly, the same conclusion holds when it is applied to the HAC-based pipeline or the LIUM S4D, even though the latter uses its own re-segmentation step in Figure 3.

Figure 4 is meant to analyze the behavior of the approach and to evaluate the robustness of its unique hyper-parameter E. The horizontal dashed line is the DER of the system before re-segmentation (i.e. the output of the clustering step). DER quickly decreases during the first few epochs, reaches an improved minimum value, then starts to over-fit to the original input and asymptotically converges to the original performance. This observation, combined with the fact that the optimal number of epochs on the test set is close to the one selected on the development set, leads us to the conclusion that the proposed LSTM-based re-segmentation is stable and very unlikely to degrade performance.

6. Conclusion

The proposed pipeline is a step toward an integrated end-to-end neural approach to speaker diarization. We show that both the initial segmentation and the final re-segmentation can be formulated as a set of sequence labeling problems, addressed using recurrent neural networks. However, in re-segmentation step, finding the best epoch E relies on a development set. We plan to investigate a way to automatically select the best epoch for each file. In addition, though neural networks can be used to embed and compare pairs of speech segments, it remains unclear how to do also cluster them in a differentiable manner. Our experiments also show that affinity propagation outperforms the standard agglomerative clustering with complete-link, when comparing speaker embeddings.

7. Acknowledgements

This work was partly supported by ANR through the ODESSA (ANR-15-CE39-0010) and PLUMCOT (ANR-16-CE92-0025) projects. We would like to thank Sylvain Meignier for providing us with the output of the LIUM's S4D system.

8. References

[1] S. E. Tranter and D. A. Reynolds, "An Overview of Automatic Speaker Diarization Systems," *IEEE Transactions on audio, speech, and language processing*, vol. 14, no. 5, pp. 1557–1565, 2006.

[2] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker diarization: A Review of Recent Research," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 356–370, 2012.

[3] C. Barras, X. Zhu, S. Meignier, and J. L. Gauvain, "Multi-Stage Speaker Diarization of Broadcast New," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 5, pp. 1505–1512, Sep. 2006.

[4] M. Rouvier, G. Dupuy, P. Gay, E. el Khoury, T. Merlin, and S. Meignier, "An Open-source State-of-the-art Toolbox for Broadcast News Diarization," in *Interspeech 2013, 14th Annual Conference of the International Speech Communication Association*, 2013.

[5] M. A. Siegler, U. Jain, B. Raj, and R. M. Stern, "Automatic Segmentation, Classification and Clustering of Broadcast News Audio," in *Proc. DARPA speech recognition workshop*, vol. 1997, 1997.

[6] S. Chen and P. Gopalakrishnan, "Speaker, Environment and Channel Change Detection and Clustering via the Bayesian Information Criterion," in *Proc. DARPA Broadcast News Transcription and Understanding Workshop*, vol. 8. Virginia, USA, 1998, pp. 127–132.

[7] R. Yin, H. Bredin, and C. Barras, "Speaker Change Detection in Broadcast TV using Bidirectional Long Short-Term Memory Networks," in *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association*, Stockholm, Sweden, August 2017.

[8] G. Gelly and J.-L. Gauvain, "Minimum Word Error Training of RNN-based Voice Activity Detection." in *Interspeech 2015, 16th Annual Conference of the International Speech Communication Association*, 2015, pp. 2650–2654.

[9] C. Barras, X. Zhu, S. Meignier, and J.-L. Gauvain, "Improving Speaker Diarization," in *RT-04F workshop*, 2004.

[10] S. Meignier and T. Merlin, "LIUM SpkDiarization: An Open Source Toolkit for Diarization," in *CMU SPUD Workshop*, 2010.

[11] H. Bredin, "TristouNet: Triplet Loss for Speaker Turn Embedding," in *ICASSP 2017, IEEE International Conference on Acoustics, Speech, and Signal Processing*, New Orleans, USA, March 2017.

[12] G. Wisniewski, H. Bredin, G. Gelly, and C. Barras, "Combining Speaker Turn Embedding and Incremental Structure Prediction for Low-Latency Speaker Diarization," in *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association*, August 2017.

[13] Q. Wang, C. Downey, L. Wan, P. A. Mansfield, and I. L. Moreno, "Speaker Diarization with LSTM," in *ICASSP 2018, IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2018.

[14] X. Zhang, J. Gao, P. Lu, and Y. Yan, "A Novel Speaker Clustering Algorithm via Supervised Affinity Propagation," *ICASSP 2008, IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4369–4372, 2008.

[15] G. Gelly and J.-L. Gauvain, "Spoken Language Identification using LSTM-based Angular Proximity," in *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association*, August 2017.

[16] B. J. Frey and D. Dueck, "Clustering by Passing Messages Between Data Points," *science*, vol. 315, no. 5814, pp. 972–976, 2007.

[17] A. Giraudel, M. Carré, V. Mapelli, J. Kahn, O. Galibert, and L. Quintard, "The REPERE Corpus: A Multimodal Corpus for Person Recognition." in *LREC*, 2012, pp. 1102–1107.

[18] G. Gravier, G. Adda, N. Paulson, M. Carré, A. Giraudel, and O. Galibert, "The ETAPE Corpus for the Evaluation of Speech-based TV Content Processing in the French Language," in *LREC - Eighth international conference on Language Resources and Evaluation*, Turkey, 2012, p. na. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00712591

[19] H. Bredin, "pyannote.metrics: A Toolkit for Reproducible Evaluation, Diagnostic, and Error Analysis of Speaker Diarization Systems," in *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association*, Stockholm, Sweden, August 2017. [Online]. Available: http://pyannote.github.io/pyannote-metrics

[20] B. Mathieu, S. Essid, T. Fillon, J. Prado, and G. Richard, "YAAFE, an Easy to Use and Efficient Audio Feature Extraction Software." in *ISMIR 2010, 11th International Society for Music Information Retrieval Conference*, 2010, pp. 441–446.

[21] A. Larcher, K. A. Lee, and S. Meignier, "An Extensible Speaker Identification Sidekit in Python," in *ICASSP 2016, IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 2016, pp. 5095–5099.

[22] P.-A. Broux, F. Desnous, A. Larcher, S. Petitrenaud, J. Carrive, and S. Meignier, "S4D: Speaker Diarization Toolkit in Python," in *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association*, 2018.

[23] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. John Wiley & Sons, 2012.

[24] S. Galliano, E. Geoffrois, D. Mostefa, K. Choukri, J.-F. Bonastre, and G. Gravier, "The ESTER Phase II Evaluation Campaign for the Rich Transcription of French Broadcast News," in *Ninth European Conference on Speech Communication and Technology*, 2005.

[25] J. Bergstra, D. Yamins, and D. Cox, "Making A Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures," in *International Conference on Machine Learning*, 2013, pp. 115–123.

[26] J. Bergstra, D. Yamins, and D. D. Cox, "Hyperopt: A Python Library for Optimizing the Hyperparameters of Machine Learning Algorithms," in *Proceedings of the 12th Python in Science Conference*. Citeseer, 2013, pp. 13–20.