INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

LSTM Neural Network-based Speaker Segmentation using Acoustic and Language Modelling

Miquel India, José A.R. Fonollosa, Javier Hernando

TALP Research Center, Universitat Politecnica de Catalunya, Spain
miquel.india@tsc.upc.edu, {jose.fonollosa,javier.hernando}@upc.edu

Abstract

This paper presents a new speaker change detection system based on Long Short-Term Memory (LSTM) neural networks using acoustic data and linguistic content. Language modelling is combined with two different Joint Factor Analysis (JFA) acoustic approaches: i-vectors and speaker factors. Both of them are compared with a baseline algorithm that uses cosine distance to detect speaker turn changes. LSTM neural networks with both linguistic and acoustic features have been able to produce a robust speaker segmentation. The experimental results show that our proposal clearly outperforms the baseline system.

Index Terms: Speaker Segmentation, Neural Language Modelling, I-vectors, Speaker Factors, LSTM Neural Networks.

1. Introduction

Speaker segmentation is an essential subtask for some speaker recognition applications. In tasks such speaker diarization or tracking, it is needed to split the signal into speaker turns. Speaker segmentation aims to divide the signal into speaker segments, detecting the boundaries between speaker changes.

Speaker segmentation systems are generally based on a two step algorithm that includes an initial segmentation and a refinement stage. In the initial segmentation, a set of speaker change candidate points are detected. The second step performs a refinement that discard the candidate points which are false alarms. Two kind of approaches are distinguished to perform this second step. The first approach performs only one robust re-evaluation to discard false speaker change candidates such in [1]. On the other hand, the second approach refinement is implemented with an iterative processing of some way to converge into an optimum speaker segmentation output. In this second approach, segmentation is normally processed iteratively using clustering algorithms such in [2].

The most common strategy to detect a speaker change turn between two speech segments is to compute a score or distance. Depending on this score/distance, the boundary between these segments is assigned to a speaker change turn. We distinguish two kind of algorithms based on this strategy: metric-based and model-based approaches. Several metric-based algorithms like Bayesian Information Criterion (BIC), Generalized Likelihood Ratio (GLR) or Divergence Shape Distance (DSD) have shown good competitive results [3, 4]. On the other hand, model-based techniques have also shown a good performance. The first model-based approaches were based on Gaussian Mixture Models (GMM) such in [5, 6]. At the present, Joint Factor Analysis (JFA) segmentation algorithms have outperformed GMM modelling [7, 8].

This work has been developed in the framework of the Camomille project (PCIN-2013-067) and the Spanish Project DeepVoice (TEC2015-69266-P).

Linguistic content is the principal source of information used in several tasks such machine translation or language modelling. In these tasks, the main research topics are based on word embeddings. Word embeddings are word representations using vectors, which represent the state of the art in neural language modelling. These embeddings exhibit the property whereby semantically close words are likewise close in the induced vector space. Several models are known to produce these vectors, such as the word2vec approach presented in [9] or the character-level models proposed in [10]. Word embeddings have shown its best performance in both language modelling and machine translation tasks when they are used as inputs of Recurrent Neural Networks (RNN)[11]. Works like [12, 13, 10] exhibit the good performance of these embeddings with RNN architectures like Long-Short Term Memory (LSTM) or Gated Recurrent Unit (GRU) neural netorks.

In this work, an alternative algorithm to apply in speaker segmentation is presented. Speaker turn changes can be detected by analysing the sequential variability between speech utterances. This sequential analysis fits the recurrent architecture where language modelling approaches are normally implemented. At the same time, linguistic content is able to determine speaker change candidates i.e. the end of a sentence. Hence, we propose a recurrent algorithm approach based on LSTM networks, where acoustic data and linguistic content are merged. Two different acoustic approaches based on Joint Factor Analysis (JFA) are tested to model speech utterances. Furthermore, a character-level Convolutional Neural Network (CharCNN) is used to create word embeddings from the linguistic content. Acoustic speaker vectors and word embeddings are introduced as inputs of a LSTM, whose output assigns speaker change/non-change turns between words.

The outline of the paper is as follows. Factor analysis approaches are firstly described in Section 2. Section 3 explains in detail the architecture of the proposed system. The experiments and the results are given in Section 4. Section 5 concludes the paper with some future work remarks.

2. Front-End Factor Analysis

In Joint Factor Analysis (JFA), a speaker utterance can be represented by a supervector, which is composed by a sum of components from different speaker subspaces. A speaker-dependent supervector can be defined as:

$$M = m + Vy + Ux + Dz \tag{1}$$

where m is the speaker and session independent supervector, normally extracted from an Universal Background Model (UBM), V and D define the speaker subspace (eigenvoice matrix and diagonal residual, respectively), and U represents the session subspace (eigenchannel matrix). The vectors x, y and z are referred to the speaker- and session-dependent factors in

http://dx.doi.org/10.21437/Interspeech.2017-407

their respective subspaces. We assume that each vector corresponds to a random variable with a normal distribution $N(0, I)$. In this work two different speaker vectors based on JFA approaches are used: i-vectors (Total Variability) and speaker factors (Eigenvoice Modelling).

2.1. Total Variability

The Total Variability approach considers only one space to represent the speaker-dependent supervector. The concatenation of the session and speaker subspace defines a new space referred as "Total Variability space". Given an utterance, the new speaker- and channel-dependent supervector defined by [14] can be expressed as follows:

$$M = m + Tw \tag{2}$$

where m is the speaker- and channel-independent supervector (UBM), T is a low rank matrix and w is a normal distribution $N(0, I)$ random vector. We refer to the factor w as identity vector or *i-vector* and to T as the *Total Variability* matrix [15].

Total factor or i-vector w, can be defined by its posterior distribution conditioned to the Baum-Welch statistics from a given utterance. Let define a sequence of L frames $\{y_1, y_2, ..., y_L\}$ and an UBM Ω composed of C mixtures components defined in some feature space of dimension F. The Baum-Welch statistics needed to estimate the i-vector for a given speech utterance u are obtained by:

$$N_c = \sum_{t=1}^{L} P(c|y_t, \Omega) \tag{3}$$

$$\tilde{F}_c = \sum_{t=1}^{L} P(c|y_t, \Omega)(y_t - m_c) \tag{4}$$

where $c = 1, ..., C$ is the Gaussian index, $P(c|y_t, \Omega)$ corresponds to the posterior probability of mixture component c generating the vector y_t and m_c is the mean of UBM mixture component c. The i-vector for a given utterance u can be obtained using the following equation:

$$w = (I + T^t \Sigma^{-1} N(u) T)^{-1} T^t \Sigma^{-1} \tilde{F}(u) \tag{5}$$

We define $N(u)$ as a diagonal matrix of dimension $CF \times CF$ whose diagonal blocks are $N_c I$ $(c = 1, ..., C)$. $\tilde{F}(u)$ is a supervector of dimension of $CF \times 1$ obtained by concatenating all first-order Baum-Welch statistics for a given utterance u. Finally, Σ and T corresponds to the diagonal covariance matrix and total variability matrix respectively. Both Σ and T are estimated during factor analysis training (see equations in [16]).

2.2. Eigenvoice Modelling

The eigenvoice approach also defines only one space to represent the speaker-dependent supervector. Hence, the speaker and channel-dependent supervector is defined as:

$$M = m + Vx_s \tag{6}$$

where m is the speaker and channel-independent supervector, V is the low-rank eigenvoice matrix and x_s is a normal distribution $N(0, I)$ random vector referred as speaker factor. In comparison with the Total Variability approach, eigenvoice matrix is trained to reduce the channel and intra-speaker variability for each speech utterance. In the T matrix training, a given speaker's set of utterances are regarded as having been produced by different speakers. On the other hand, in eigenvoice training all the recordings of a given speaker are considered to belong to the same person [15].

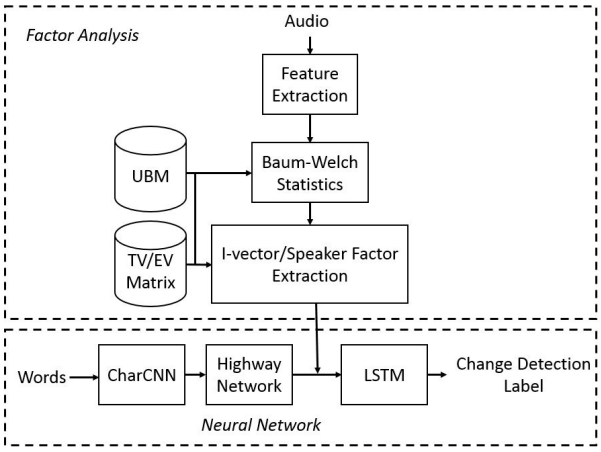

Figure 1: *System architecture*

3. System Architecture

One of the most common strategies to detect a speaker change in a point t, is to compute a score or distance comparing its left $[t-\tau, t]$ and right $[t, t+\tau]$ speech segments. Depending on the speaker turn lengths, the window length τ is tuned to optimize the performance of the system. This performance decreases as shorter is τ because from some window length the reduction of acoustic data highly decreases the reliability of acoustic models. The proposed algorithm is designed to work in telephone conversation datasets where speaker turns are very short. Hence, our proposal aims to combine linguistic content with speaker vectors extracted from short speech segments in order to detect speaker turn changes. Only the words extracted from the manual transcription and the signal are used as inputs of the system.

The architecture of the proposed system is based on a neural network that combines linguistic content and acoustic data (Figure 1). In this neural network, the concatenation of a word embedding and a speaker vectors is used as in input on a LSTM, whose output is a speaker change score. The proposed algorithm proceeds by the following steps:

1. The words extracted from the manual transcription are introduced in the neural network. These words are transformed into word embeddings (Section 3.1), which will be used as one of the inputs of the LSTM (Section 3.2).
2. For each word k_t, the left side segment from the boundary between k_t and k_{t+1} is used to extract the acoustic features. With these features a speaker vector (speaker factor, i-vector) is computed and concatenated with the word embedding extracted in the previous step. The concatenation of both features are introduced in the LSTM.
3. The LSTM outputs an score $p \in$ [0,1]. A score threshold λ is used to decide which boundaries are assigned to speaker turns changes.

The different parts form the neural network are described in detail in the following subsections. Word embeddings extraction is described in Section 3.1. LSTM implementation is explained in Section 3.2.

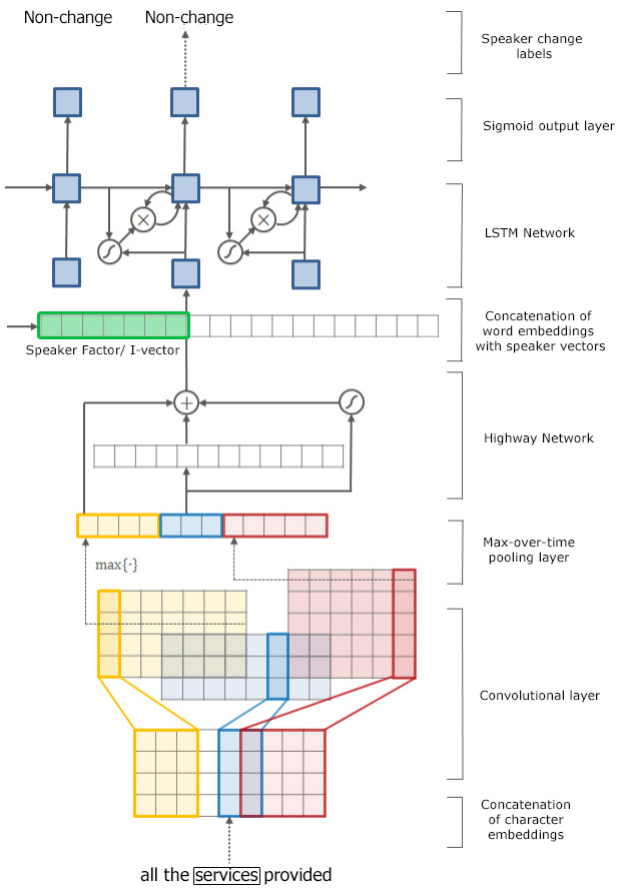

Figure 2: *Neural network scheme [10]*

3.1. Character-level Word Embedding

The architecture system proposed contains a LSTM neural network, whose one of its inputs is a word embedding. Word embeddings are word representations modelled as real value vectors mapped from its textual form. In this system, word embeddings are obtained from the output of a character-level convolutional neural network (CharCNN).

Let define C as the vocabulary of characters, d as the dimension of character embeddings, and $\mathrm{Q} \in \mathrm{R}^{d \times |\mathrm{C}|}$ as the character embedding matrix. Each word $k \in \mathrm{V}$ is composed by a sequence of characters $[c_1, ..., c_l]$, where l is the length of word k. Hence, the word k can be representated in character-level space as the matrix $\mathrm{C}^k \in \mathrm{R}^{d \times l}$. A narrow convolution is applied between C^k and a filter $\mathrm{H} \in \mathrm{R}^{d \times v}$ of width w, where a bias is added and a nonlinearity is applied to obtain a feature map $f^k \in \mathrm{R}^{l-w+1}$ (equations provided in [10]). We take the max-over-time y_k as the feature corresponding to the filter H (when applied to the word k). For many NLP tasks the number of filters h is used to be chosen between [100,1000].

Additionally to the CharCNN, one more network is implemented replacing y_k with x_k at each t in the LSTM structure. Instead of using a typical set of fully-connected layers, those are replaced by a Highway network [17, 18] . As is shown in [10], these networks show a better performance by modelling the interactions between the character n-grams extracted by the filters over y_k.

3.2. Recurrent Speaker Change Detection

LSTM networks are used in this work in order to score the speaker turn change likelihood between two words. The LSTM network is implemented using speaker vectors and word embeddings as inputs and applying some delay in the system. Hence, a speaker turn decision is based on past, present and future words.

The idea behind the use of the LSTM for this task is different in both feature cases. Language modelling is able to grammatically detect possible speaker turn candidates i.e. the end of a sentence. Acoustically, the variability of speech utterances can be used to detect speaker change points. Given a speaker change turn, the sequence of segments with overlapped speakers contain the speaker vector transition between the previous speaker and the new one. This sequential transition can be learned recursively by the LSTM network in order to detect speaker turn boundaries. Furthermore, the combination of both features in a recurrent neural network architecture circumvents the necessity of using post processing stages like speaker turn length filters.

4. Experiments and Results

4.1. Experimental Setup

This task has been evaluated using two different English corpora based on conversational telephone speech. These conversations are characterized by containing short speech turns and speaker overlapping. Total variability and eigenvoice matrix haven been trained on 433 hours of speech (5198 speakers) from the Fisher Training corpus [19] . Otherwise Neural Networks have been trained with both whole Fisher and SwitchBoard-1 Release 2 dataset [20, 21]. A set of 40 recordings from both corpus have been randomly discarded for the training to be used for the test. Evaluation data contains a total of 4648 speaker boundaries.

Acoustic modelling operates on cepstral features, extracted using a 30 ms Hamming window. Every 10 ms, 20 mel frequency cepstral coefficients (MFCCs) were calculated. We used a 64 Gaussian UBM for both JFA approaches. V matrix was 20 rank size such in [7] and T matrix rank was set to 400, which is a common value in the speaker identification/verification state of the art. Both speaker factors and i-vectors were normalized.

The neural network was trained by the truncated backpropagation trough time approach [22, 23]. Stochastic gradient was used with an initial 0.1 learning rate and the backpropagation was done for 35 steps. The learning rate was decayed by a 0.5 factor if validation perplexity did not improve by more than 1.0 after an epoch. Both networks were trainned for 14 epochs with 20 size minibatches. For regularization we used dropout [24] with probability 0.5. The dropout was applied on the LSTM input to hidden layers (except on the initial Highway to the LSTM layer) and the hidden-to-output sigmoid layer. Gradient updating was constrained to normalize gradient to 5. If the L_2 norm gradient was above 5, it was normalized again before the updating.

Neural network architecture was setup similar to the large model presented in [10]. The CharCNN was setup with a set of h = 500 filters. These filters had the next range of widhts w = [1,2,3,4,5,6] with its respective size [25,50,75,100,100,200]. Character embeddings had a d=15 size and $tanh$ was the nonlinear function applied in the convolutional step. The Highway network was set with only one hidden layer and Rectified Linear Units (ReLU) as activation functions. LSTM was composed by l=2 hidden layers, with m=150 nodes per layer. The output layer was based on only one sigmoid activation with 2 sequence

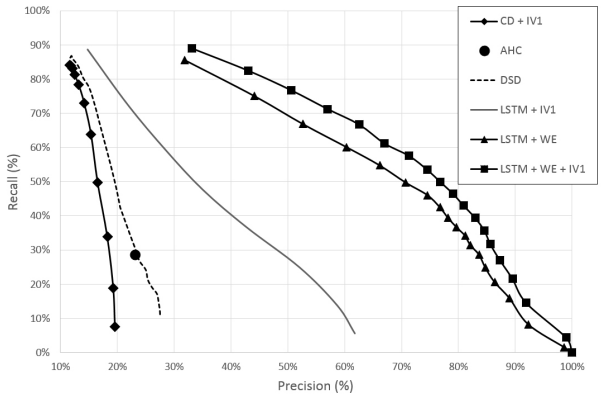

Figure 3: *Precision-recall curves. The CD and DSD approach curves are shown in terms of a distance threshold. The neural network approach is shown in function of* λ.

time delay.

Speaker segmentation has been evaluated comparing the boundaries created with the speaker turns from the manual transcriptions. A 0.5 second forgiveness collar is set to link the computed boundaries with the groundtruth speaker change points. The formed links determine the recall (percentage of correct boundaries respect the number of reference points) and precision (percentage of correct boundaries respect the computed ones). Additionally, the F1 measure is also computed.

The proposed system is evaluated with four different acoustics inputs and compared with a cosine distance (CD) approach. The behaviour of both speaker and speaker factors is analysed using two speaker segment lengths: 1 and 2 seconds. The baseline proposed computes the score of each speaker boundary using the cosine distance between the left and the right speech segment from that speaker turn candidate. A distance threshold is set to decide if that boundary corresponds to a speaker turn change. Furthermore, two speaker segmentation approaches are also proposed to compare with the proposed system. The first approach is based on the speaker change detector presented in [25]. In this work, Divergence Shape Distance (DSD) is applied to evaluate the speaker change turn between the left and right segments from each transcription word. The second approach corresponds to the diarization algorithm presented in [26]. This approach applies Viterbi decoding and BIC following an Agglomerative Hierarchical Clustering (AHC) strategy.

4.2. Results

Figure 3 shows the precision-recall curves of the CD, DSD and AHC approaches and the LSTM network system with speaker vectors and word embeddings both used separately and concatenated. For clarity, the CD and the LSTM results are only shown for the 1 second length i-vector (IV1) input. As it can be seen in Figure 3, LSTM results using only i-vectors are better than CD, DSD and AHC systems. Speech turns shorter than 1 second highly degrade model and metric-based algorithm performances like in the case of CD, AHC and DSD. However, LSTM approach is able to detect speaker turn changes outperforming these approaches with the same input. LSTM networks are able to retain speaker vectors information in order to decide when speaker changes are produced. On the other hand, we see how LSTM network shows better results using only word embeddings than acoustic vectors. The best results are obtained when LSTM network combines both acoustic features and linguistic content.

Table 1: *Speaker segmentation results. The experiments results shown for both systems correspond to the threshold that maximizes the F1 measure. CD is referred to cosine distance and LSTM corresponds to the neural network approach. DSD and Diarization are referred to the complementary systems proposed. JFA approaches are expressed as SF in case of speaker vectors and IV for the i-vectors. The number following to the JFA approach corresponds to the segment length* l *in seconds. WE is referred to the use of word embeddings.*

Speaker Segmentation Evaluation				
Features	System	Precision(%)	Recall(%)	F1(%)
SF1	CD	15.94	63.07	25.45
SF2	CD	17.23	52.47	25.97
WE+SF1	LSTM	**62.52**	**62.41**	**62.46**
WE+SF2	LSTM	63.24	60.26	61.71
IV1	CD	18.17	43.08	25.56
IV2	CD	17.05	55.58	26.09
WE+IV1	LSTM	**62.54**	**66.70**	**64.55**
WE+IV2	LSTM	62.71	62.28	62.49
IV1	LSTM	35.11	47.48	40.37
WE	LSTM	**60.35**	**60.08**	**60.21**
MFCC	DSD	19.80	48.17	28.06
MFCC	AHC	23.19	28.46	25.56

Table 1 shows the speaker segmentation results from all the experiments presented. The results clearly show how the neural network approach outperforms the baseline systems. Both i-vector and speaker factors results are better with the proposed system than with the cosine distance approach. With 1 second speaker factors, F1 measure with the LSTM approach is 62.46% compared to the 25.45% rate obtained with the baseline. With 2 seconds speaker factors the results are similar (61.71% compared to 25.97%).

I-vectors have shown a similar performance than speaker factors in both systems. With the baseline system, 2 seconds speaker factors F1 measure is 25.97% compared to 26.09% obtained with 2 seconds i-vector. With the LSTM approach the rates are likewise close (61.71% compared with 62.49%, respectively). In terms of segment length, 1 second vectors results are also similar compared to the 2 seconds vector ones.

5. Conclusion

In this paper we have investigated the combination of linguistic content and acoustic features for speaker segmentation. We tested neural language modelling architectures such as LSTM in order to merge acoustic and language modelling. LSTM networks are able to produce robust speaker segmentations with only the use of linguistic content. The combination of both features outperforms the cosine distance based baseline where only acoustic data are used. For future work, it would be interesting to analyse how these architectures could be combined with clustering techniques to perform speaker diarization. Furthermore, this research will be extended with the use of ASR system as word inputs, instead of using manual transcriptions.

6. References

[1] L. Lu and H.-J. Zhang, "Speaker change detection and tracking in real-time news broadcasting analysis," in *Proceedings of the tenth ACM international conference on Multimedia*. ACM, 2002, pp. 602–610.

[2] X. Anguera Miró, *Robust speaker diarization for meetings*. Universitat Politècnica de Catalunya, 2006.

[3] A. Tritschler and R. A. Gopinath, "Improved speaker segmentation and segments clustering using the bayesian information criterion." in *Eurospeech*, vol. 99, 1999, pp. 679–682.

[4] L. Lu and H.-J. Zhang, "Real-time unsupervised speaker change detection," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 2. IEEE, 2002, pp. 358–361.

[5] J. Ajmera and C. Wooters, "A robust speaker clustering algorithm," in *Automatic Speech Recognition and Understanding, 2003. ASRU'03. 2003 IEEE Workshop on*. IEEE, 2003, pp. 411–416.

[6] J. Ajmera, I. McCowan, and H. Bourlard, "Robust speaker change detection," *IEEE signal processing letters*, vol. 11, no. 8, pp. 649–651, 2004.

[7] B. Desplanques, K. Demuynck, and J.-P. Martens, "Factor analysis for speaker segmentation and improved speaker diarization," in *Interspeech*. Abstracts and Proceedings USB Productions, 2015, pp. 3081–3085.

[8] F. Castaldo, D. Colibro, E. Dalmasso, P. Laface, and C. Vair, "Stream-based speaker segmentation using speaker factors and eigenvoices," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE, 2008, pp. 4133–4136.

[9] Y. Goldberg and O. Levy, "word2vec explained: Deriving mikolov et al.'s negative-sampling word-embedding method," *arXiv preprint arXiv:1402.3722*, 2014.

[10] Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush, "Character-aware neural language models," *arXiv preprint arXiv:1508.06615*, 2015.

[11] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model." in *Interspeech*, vol. 2, 2010, p. 3.

[12] M. R. Costa-Jussa and J. A. Fonollosa, "Character-based neural machine translation," *arXiv preprint arXiv:1603.00810*, 2016.

[13] M. Sundermeyer, R. Schlüter, and H. Ney, "Lstm neural networks for language modeling." in *Interspeech*, 2012, pp. 194–197.

[14] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

[15] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[16] P. Kenny, G. Boulianne, and P. Dumouchel, "Eigenvoice modeling with sparse training data," *IEEE transactions on speech and audio processing*, vol. 13, no. 3, pp. 345–354, 2005.

[17] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," *arXiv preprint arXiv:1505.00387*, 2015.

[18] ——, "Training very deep networks," in *Advances in neural information processing systems*, 2015, pp. 2377–2385.

[19] C. Cieri, D. Miller, and K. Walker, "Fisher english training speech parts 1 and 2," *Philadelphia: Linguistic Data Consortium*, 2004.

[20] J. J. Godfrey and E. Holliman, "Switchboard-1 release 2," *Linguistic Data Consortium, Philadelphia*, vol. 926, p. 927, 1997.

[21] B. Wheatley, G. Doddington, C. Hemphill, J. Godfrey, E. Holliman, J. McDaniel, and D. Fisher, "Switchboard: A users manual," 1995.

[22] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[23] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[24] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[25] J. Luque and J. Hernando, "On the use of agglomerative and spectral clustering in speaker diarization of meetings." in *Odyssey*, 2012, pp. 130–137.

[26] J. Luque Serrano, "Speaker diarization and tracking in multiple-sensor environments," 2012.