Capturing Voice Prosody with RNNs for Improved Speaker Clustering

Thilo Stadelmann[1], Sebastian Glinski-Haefeli[1], Patrick Gerber[1], and Oliver Dürr[1,2]

[1] ZHAW Datalab, Zurich University of Applied Sciences, Winterthur, Switzerland
[2] Institute for Optical Systems, Konstanz University of Applied Sciences, Germany
stdm@zhaw.ch, sebastian.glinski@gmail.com, gerber.pat@gmail.com, oliver.duerr@gmail.com

Abstract. Deep neural networks have become a veritable alternative to classic speaker recognition and -clustering methods in recent years. However, while the speech signal clearly is a time series, and despite the body of literature on the benefits of prosodic (time-dependent) features, identifying voices has usually not been approached with sequence learning methods. Only recently has a recurrent neural network (RNN) been successfully applied to this task, while the use of convolutional neural networks (CNNs) (that is not able to capture arbitrary time dependencies, unlike RNNs) still prevails. In this paper, we show the effectiveness of RNNs for speaker recognition by improving state of the art speaker clustering performance and robustness on the classic TIMIT benchmark. We provide arguments why RNNs are superior by experimentally showing a "sweet spot" of the segment length for successfully capturing prosodic information that has been theoretically predicted in previous work.

Keywords: speaker clustering · speaker recognition · recurrent neural network

1 Introduction

Automatic speaker recognition comes in many flavors, of which speaker clustering is the most unconstrained and hence the most difficult one [3]. As it is essentially the task of judging if two short utterances come from the same (previously unknown) speaker, it is a suitable benchmark for the general ability of a system to capture what makes up a voice: speaker clustering can only be solved satisfactory by regarding all available cues in the utterances themselves. This distinguishes speaker clustering from a more complex experimental setup like e.g. speaker diarization, where engineering a complex system of many components has a not negligible influence on the final result besides the pure voice modeling [2]; and for example from speaker identification, where more available data enables the creation of models that work well just because of the sheer amount of collected training statistics [33]. Previous work [40] hence suggests that the bottleneck for speaker clustering performance lies in exploiting the supra-frame information

present in the audio signal's evolution in time. This information of how single audio frames depend on each other in a speaker-characteristic way can be identified with the prosodic features of a voice—with its "sound".

In recent years, deep neural networks have been successfully applied to various speaker recognition tasks [21,7,44,36,27,23], reaching and exceeding state of the art results of classic GMM- [34] or i-vector-based [8] systems. With few exceptions, systems based on a convolutional neural network (CNN) architecture have been used on spectrogram input for their known unprecedented performance on visual recognition tasks [20]. While spectrograms encode the time information of the audio signal explicitly on the image's horizontal axis, and CNNs can in principal learn temporal patterns by having filters with a certain extend in the horizontal direction, CNNs are per se not sequence learning models. Recurrent neural networks (RNN) [37,35] instead are explicitly built with temporal modeling in mind [12] and have shown exceptional performance on other audio recognition tasks [4,13,19,17]. Despite this natural fit, RNNs have only very recently been successfully applied to the task of speaker recognition for the first time (see Section 2.1). This discrepancy is potentially due to the reported inherent difficulty to successfully fit recurrent models [31].

In this paper, we present a quantitative and qualitative analysis of RNNs for speaker clustering on the TIMIT database. We demonstrate results that slightly improve state of the art on the evaluation set of 40 speakers, while being more robust to hyperparameter choice and model initializations than previously used CNN models. More importantly, extensive experiments allow the conclusion that the RNN model achieves this performance through an ability to model voice prosody effectively. This contributes a strong rationale with empirical evidence to the recently published first experiments with RNNs for speaker recognition, and provides valuable insights into the workings of deep learning approaches on audio data. Specifically, we empirically confirm the "sweet spot" of the segment length to capture time-dependent (prosodic) information that has been predicted in earlier work [40]. Section 2 provides an introduction and the background to our approach, including related work. Section 3 reports on our experimental setup and results, before findings are discussed in Section 4. Section 5 finally provides conclusions and suggestions for future work.

2 Speaker clustering with RNNs

2.1 Related work

Learning speaker embeddings has been approached with different neural network architectures such as Siamese [6], fully connected [42], and CNN [23,24,10]. Only very recently have RNNs been used successfully [32,22,43].

While for standard classification problems, where the network learns to distinguish different classes, the cross entropy is the natural choice for the loss function [11], there is no natural choice for learning embeddings and hence there exist many different approaches to choose a suitable loss. *Garcia-Romera et*

al. [10] learn embeddings through the training of a deep neural network. They consider a learnable distance function defined in the spirit of probabilistic linear discriminant analysis (PLDA) [5]. This distance function is then used as the input to a binary cross entropy to classify if two segments come from the same speaker or not. To account for the fact that there are more pairs of segments between different speakers than between identical speakers, they introduce a weighting in the loss function. Using the distance function in an agglomerative hierarchical clustering and further refining the result with variational Bayes re-segmentation [39], diarization error rates (DER) between 11.2% and 9.9% on the CALLHOME corpus are achieved. While the network training and in particular the loss function of this approach is comparable to ours, the temporal information is still extracted by a classical feed-forward neural net instead of a sequence-learning RNN.

Cyrta et al. [32] suggest to learn the embeddings by training a recurrent convolutional neural network for the task of speaker classification. Although they utilize recurrent layers to retrieve temporal information, the feature extraction is still done by convolutional layers. Furthermore, the training is done using a surrogate classification task with the standard cross-entropy as a loss function, and no clustering specific loss is defined. They reportedly outperform the DER of a GMM-based baseline [26] with 30% relative improvement, and a CNN [23] with 12% relative improvement on a novel dataset.

Li et al. [22] experiment with two architectures to extract features: a residual CNN [14] and a RNN using gated recurrent units. Two training stages are performed successively, initially with cross entropy loss, followed by using triplet loss [38] to minimize intra-speaker distances while maximizing inter-speaker distances. Best results are achieved by the residual CNN with both cross entropy- and triplet loss. They report an equal error rate (EER) of 1.13% and accuracy (ACC) of 96.83% for text-independent speaker identification on the Mandarin and English UIDs dataset, resulting in relative improvements of EER and ACC by 50% and 60%, respectively. Notably, the final model trained on Mandarin speech is able to improve accuracy for English speaker recognition, too. However, in our work, we achieve state of the art speaker clustering performance without using an additional convolutional network in front of the RNN or dual training, showing the sufficiency of a recurrent architecture alone for speaker modeling.

Finally, *Wang et al.* [43] use long short-term memory (LSTM) cells indirectly to convert MFCC feature vectors to embeddings that are fed into a subsequent off-line spectral clustering [30] process. They reach an absolute DER of 12.0% on the CALLHOME dataset, where comparable studies achieved DERs between 14.9% and 12.1%. The approach is similar to ours in using only a RNN to extract speaker embeddings, but differs in training (network architecture, loss function) as well as front-end (features) and post-processing (clustering approach).

2.2 Overview of our approach

Generally, speaker embeddings are fixed-size vectorial representations of a voice, formed by the activations of neurons at higher-level layers of a neural network

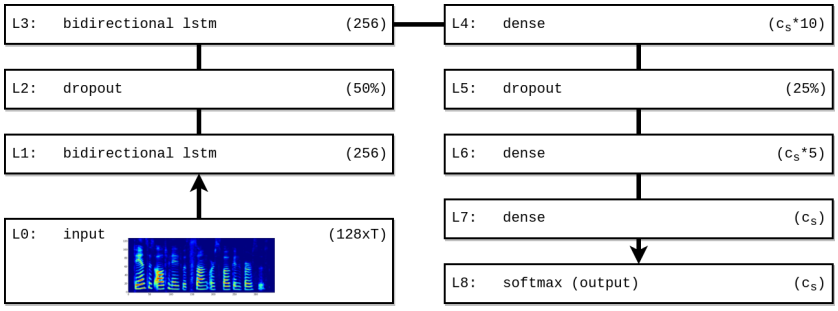


Fig. 1: Neural network architecture used for all experiments, including important hyperparameters of each layer like input size, number of neurons and dropout factor. The width T (in pixels) of each input segment is varied in our experiments.

during the forward pass of a respective speech utterance of that voice. Our neural network (cp. Figure 1) uses a combination of bidirectional LSTM (BLSTM) layers (L1 and L3) with additional fully connected layers (L4, L6 and L7) right in front of the output layer L8 with c_s neurons. We chose c_s to equal the number of speakers in the training set and input the audio in form of an image (spectrogram). We chose BLSTM layers because their awareness of previous and upcoming sequence steps, i.e., past and future time steps in the signal. They are thus able to relate current information to its semantically relevant temporal context. It is shown in [23] that it is not advisable to use the final layer trained for a surrogate task for extracting the embeddings; we thus add three dense layers between the last BLSTM and the output to later experiment with from where to actually extract our embeddings.

Training We treat the c_s-dimensional output vector from layer L8 as a distribution that should be similar for all spectrograms from the same speaker, and dissimilar for spectrograms from different speakers. We train the network to give this output by using a loss function based on the pairwise Kullback-Leibler divergence (PKLD), as described in [15]: it enforces said within-speaker similarity / between-speaker dissimilarity between all possible pairs of spectrograms in each mini batch. This loss function helps the neural network to be fit for clustering previously unseen speakers, as it is not specifically forcing the network to learn a one-hot encoding of the speaker identity as would be the case when using cross entropy. Rather, it ensures a voice-specific arbitrary discrete distribution.

For each mini batch, the loss is computed as follows: first, the mini batch is created by randomly selecting c_m segments of length $T \cdot 10$ms from the training set (a mini batch thus does not contain a fixed number of speakers). Then, each segment within a mini batch is passed forward through the network, resulting in an output distribution at layer L8. Finally, the loss function is calculated for all possible pairs of output distributions (p, q) within a mini batch as follows:

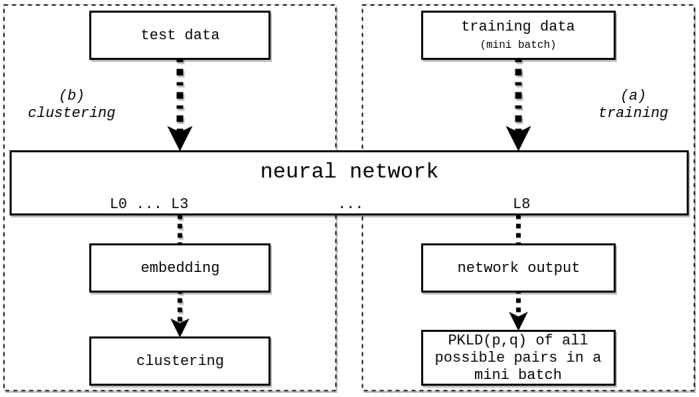


Fig. 2: Schematic of the training and clustering process with our neural network: (a) training is executed by computing the loss between all possible pairs of outputs from L8 within a mini batch; (b) AHC clustering is fed with embeddings extracted from a lower layer during a forward pass on the trained neural network.

If the two outputs p and q are from the same speaker, the Kullback-Leibler (KL) divergence is calculated:

$$\mathrm{KL}(\mathbf{p} \parallel \mathbf{q}) = \sum_{i}^{c_s} p_i \log \frac{p_i}{q_i}. \tag{1}$$

Otherwise, if a paired output is from different speakers, the hinge loss is calculated:

$$\mathrm{HL}(\mathbf{p} \parallel \mathbf{q}) = \max(0, \mathrm{margin} - \mathrm{KL}(\mathbf{p} \parallel \mathbf{q})) \tag{2}$$

where the margin hyperparameter defines the maximum distance between two elements of a pair. Both loss terms are combined as follows:

$$\mathrm{loss}(\mathbf{p} \parallel \mathbf{q}) = I_s \cdot \mathrm{KL}(\mathbf{p} \parallel \mathbf{q}) + I_{ds} \cdot \mathrm{HL}(\mathbf{p} \parallel \mathbf{q}) \tag{3}$$

where I_s equals 1 for pairs from the same speaker and 0 for pairs from different speakers. Inversely, I_{ds} equals 1 for pairs from different speakers and 0 for pairs from the same speaker. Finally, we symmetrize the loss function via:

$$\mathrm{L}(\mathbf{p}, \mathbf{q}) = \mathrm{loss}(\mathbf{p} \parallel \mathbf{q}) + \mathrm{loss}(\mathbf{q} \parallel \mathbf{p}) \tag{4}$$

Clustering To perform speaker clustering on a completely disjunct set of test speakers, the trained neural network is applied to their utterances (chopped into $T \cdot 10$ms long segments), and the respective embeddings are extracted during a forward pass. We experiment with different layers as potential sources to extract the embedding vectors from, ranging from L3 to L8. Then, hierarchical agglomerative clustering [28] is used off-line to perform the actual clustering of these vectors (see Figure 2).

3 Experimental evaluation

The goal of these experiments is to provide support for the assumption that RNNs capture prosodic features of a voice. The experiments are evaluated on the TIMIT [9] corpus, which has been chosen for two reasons: (a) its cleanness, to prevent detraction from the pure voice modeling aspect; and (b) to compare directly with related work on speaker clustering [40,23,24]. Despite speaker recognition progress in more challenging environments like meeting diarization [1] or less constrained speaker identification [29], speaker clustering of even just 40 speakers from TIMIT only recently gave reasonable results [23], while attempting to cluster all 630 speakers failed altogether [40]. Studying pure voice modeling capability in isolation thus still seems appropriate, despite the clean studio recordings of sufficient length in TIMIT. The code for our experiments can be found online[1].

3.1 Experimental setup

In accordance with [24], we perform all training of neural networks on the `speaker_100_50w_50m_not_reynolds` subset of 100 TIMIT speakers[2]. Of these speakers, randomly selected 80% are used as training data and 20% for validation during the training procedure. All audio is converted to mel spectrograms of 128 pixels height (frequency resolution). Unlike [23,24], who used a fixed input width of 100 pixels (1,000ms) for the network, we experiment with different segment length below. We chose a batch size of $c_m = 100$ in conjunction with the Adam optimizer [16] (and unchanged standard parameters learningrate $= 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e^{-8}$, and decay $= 0.0$). All trainings run for $10,000$ mini batches. For the margin parameter of the PKLD, Hsu et al. [15] suggested a value of 2; we determined margin $= 3$ to work best after grid search within $\{1.5, 2, 2.5, 3\}$ for speaker clustering tasks with $N = \{40, 60, 80\}$ speakers.

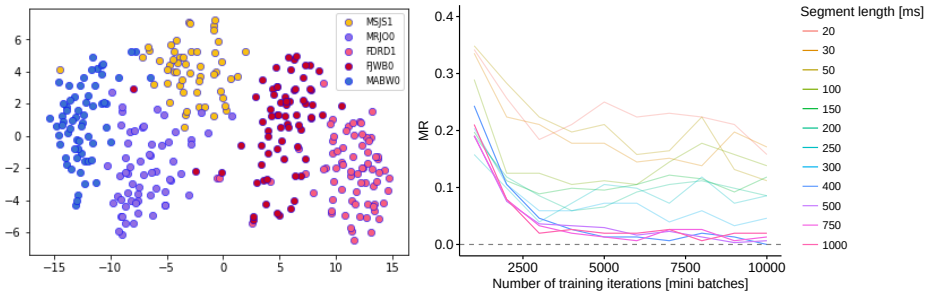
Evaluation is based on the `speakers_40_clustering_vs_reynolds` list from [40] in two stages: intermediate experiments are performed on a 38 speaker subset of this list, where the 10 sentences per speaker are randomly split into 2 utterances, 8 and 2 sentences long, respectively[3]. Final evaluations are done in accordance with [40] on the complete list of 40 speakers, using the first 8 sentences (lexicographically ordered by filename) of each speaker to form utterance 1 and the remaining 2 sentences for the second utterance[4]. As in [23,24], we finally use agglomerative hierarchical clustering (AHC) with complete-linkage and the cosine distance between embeddings, and average multiple embeddings per utterance prior to entering AHC for utterances longer than the segment length. Fig. 3a visually confirms this practice of averaging.

[1] See https://github.com/stdm/ZHAW_deep_voice.

[2] See /common/data/speaker_lists on GitHub.

[3] Both changes to the setup of [40] are due to unintentional anomalies in the data loading process that got corrected later; arguably, they reduce the chance of overfitting the intermediate results to the final evaluation set. The missing speakers are the well-clustering `FPKT0` and `FAKS0` (see Figure 5).

[4] Evidence in the source code suggests that [23,24] used random allocation here, too.



(a) (b)

Fig. 3: (a) A t-SNE visualization [25] of all single embeddings extracted from all available data of 5 TIMIT speakers, colored by speaker identity. (b) MR as a function of training iterations for all evaluated segment lengths, evaluated every 1,000 mini batches on the 38 speaker evaluation set. For visual clarity, segment lengths of 300ms and below are faded.

We evaluate each clustering result using the misclassification rate (MR) as introduced by Kotti et al. [18]: MR $= \frac{1}{N}\sum_{j=1}^{N_{cl}} e_j$, where N is the overall number of embeddings to cluster, N_{cl} the number of found clusters, and e_j is defined as the number of embeddings in cluster j that are not assigned to the *correct* cluster. The unique correct cluster for any speaker is arguably the one that fulfills the following two conditions: (a) it is the cluster with the largest number of embeddings from this speaker; and (b) if there are also embeddings from other speakers in this cluster, their number is smaller. However, previous work [23,24] used a slightly more conservative definition, adding two more necessary conditions: (c) clusters holding only one embedding cannot be correct; and (d) all embeddings in a cluster with mixed speakers are incorrect. In this paper, we additionally report MR using the more reasonable (and likely more popular) interpretation of correct clusters with only conditions (a)–(b).

3.2 Results

Our goal in the following experiments is to vary the input segment length available for the RNN to learn the temporal dependencies, in order to verify if a "sweet spot" for prosodic feature modeling as shown by [40] is found. To use an optimal number of training steps, we first evaluate the MR against the number of training iterations as the first intermediate experiment.

Figure 3b shows how different networks for the varying segment lengths perform on the clustering task, depending on how much they have been trained in terms of number of mini batches. Two things can be seen: first, the networks that perform well/average/poorly after full training do so as well after shorter training. Second, training (at least for models trained with 400ms segments or longer) seems to stabilize somewhat after ca. 5,000 mini batches; it does not

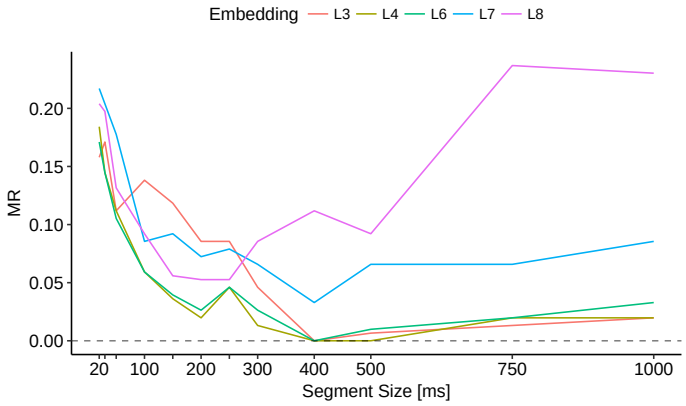


Fig. 4: MR on the 38 speaker evaluation set as a function of the input segment length used for different embedding layers (L3/4/6/7/8), averaged over 2 runs. All layers show a "sweet spot" for the segment length; the best performing layers L3–L6 have this "sweet spot" at around 400ms with MR = 0.0.

appear as if longer training would have significantly altered the results presented above nor do we observe significant overfitting. In the following, we therefore set the number of training steps to "full" 10,000 mini batches for all experiments, also to avoid cherry picking of particularly well-behaving training snapshots.

Next, we study the effect of varying segment sizes for the different embedding layers (see Figure 4) as the second intermediate experiment. The results are averages of two runs with independently trained networks of different random weight initializations to account for potential instabilities in the training. With the layers L3, L4, and L6, we achieve a perfect clustering, while the layers L7 and L8 perform much worse. For all layers, we observe the following universal behavior: with increasing segment size, the MR first decreases and reaches a minimum before it rises again. The minimum or "sweet spot" is at about 400ms for the lower layers L3–L6 and at about 150–250ms for the final layer L8.

Finally, we evaluate one of our best-performing network configurations (segment length 400ms, embedding extraction from L3) on the full 40 speakers test set to compare with previous results. Table 1 shows that the pure clustering performance is slightly better in terms of the more strict "legacy" MR than the work of Lukic et al. [23,24]. It is computed as the average over 4 independently trained networks with identical parameter settings, showing the independence of the result from random fluctuations in the optimization. This has to be seen in contrast to the reported dependence of the results in [24] from optimizer hyperparameters and large number of training rounds. Figure 5 shows a dendrogram of one of the best performing clusterings (MR = 0.0125) for reference.

Table 1: Comparison of model performances on the TIMIT 40 speaker evaluation set, given in terms of MR and the more strict legacy variant.

Method	MR	MR (legacy)
RNN /w PKLD	**2.19**% $\left(\frac{1.25\%+2.5\%+1.25\%+3.75\%}{4}\right)$	**4.38**% (average of 4 runs)
CNN /w PKLD [24]	-	**5**%
CNN /w cross entropy [23]	-	**5**%
ν-SVM [40]	6.25%	-
GMM/MFCC [40]	12.5%	-

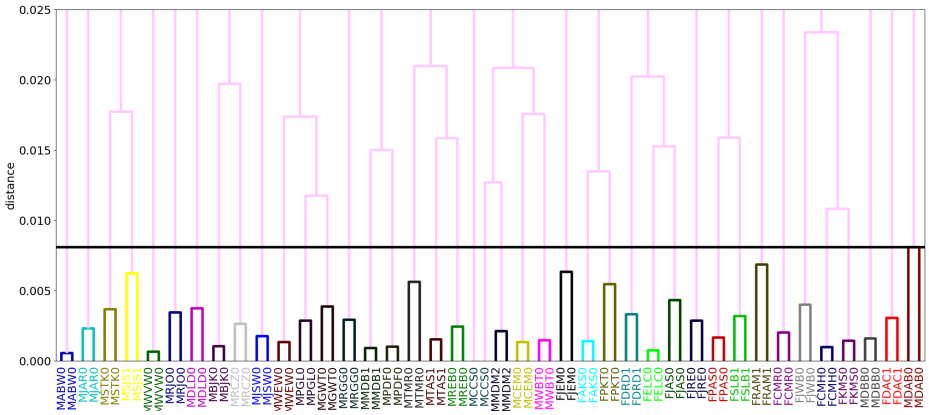


Fig. 5: The (lower part of a) dendrogram of all 40 speakers from the evaluation set, created with one of our best performing models (segment length 400ms, embeddings extracted from L3). Misclassifications arise only for speaker `MCCS0`.

4 Discussion

Stadelmann and Freisleben [40] suggested that using prosodic features of a voice is likely to improve speaker clustering performance by an order of magnitude. They considered a temporal context lengths of 32–496ms length as potentially reasonable and demonstrated that an implementation using MFCCs and one-class SVMs was able to realize parts of this potential using a context length of 130ms, reaching MR = 0.0625 on the 40 speaker subset of TIMIT.

Our experimental results above suggest that for our RNN implementation, a temporal context length around 400ms is optimal, slightly increasing the state of the art in speaker clustering performance on the 40 TIMIT speakers. Is this due to the RNN capturing voice prosody? Perceptual experiments as in [41] would be necessary to ultimately confirm it, but two facts suggest this conclusion: (a) RNNs are sequence learning models, specifically designed to learn temporal context; and (b) the detected "sweet spot" of 400ms lies within the interval suggested in the prior work. That the MR does not return to the high value of segment lengths below 300ms could be due to the fact that below the sweet spot, the network is

basically missing the necessary prosodic information in the signal; above 400ms, the signal might be dominated by speech- rather than voice-specific information, but the network is able to learn what it shall by virtue of the properties of RNNs.

We consistently achieve top results over several independent runs without specifically tuning the hyperparameters of the Adam optimizer or the network architecture; we thus regard our approach as generally robust against random influences (weight initialization, mini batch constitution, architectural choices). We observe however a dependency on the data: some particularly well clustering speakers like e.g. `MREB0` (according to low distances in Figure 5) where already mentioned by [40] to be easily clustered by humans, while on the other hand the few misclassifications consistently involve speaker `MCSS0`. Finally, results fluctuate depending on which sentences are grouped to utterances—thus, for comparability it is important to have returned to the original experimental setup of [40] in this paper with respect to MR definition and utterance assembly.

The results with respect to which layer we use for extracting embeddings are interesting: we originally chose the PKLD loss function to train on a task as close as possible to our actual goal of clustering. Related work [23] had shown that a network learned purely for speaker identification (a surrogate task) is not ideal for later determining the similarity of unknown voices. This could be mitigated if an earlier than the final layer is used. The intuition behind this is that at the final layer is too well adapted to the specific voices seen during training, whereas the lower embeddings are more abstract. A continuation of said work [24] suggested to use the PKLD to improve this issue. It is thus interesting that we still need to extract embeddings from a lower layer—a layer farther away from the trained task—in order to achieve optimal results for speaker clustering. This suggests that even training for a pairwise similarity or dissimilarity of distributions is still a surrogate task far away from the actual task of speaker clustering; and that learning speaker clustering end-to-end with the actual clustering task incorporated in the loss function and network output could improve this.

5 Conclusions and future work

In this work, executed simultaneously to first published results on RNNs for speaker embeddings, we demonstrated that recurrent neural networks can be used to model prosodic aspects of a voice. We were able to show that specifically bidirectional LSTMs in combination with the PKLD loss function perform better than any other machine learning approach tested so far for speaker clustering on the TIMIT corpus—i.e., are fit for voice modeling per se. Furthermore, our results show a "sweet spot" for extracting temporal context information with this kind of RNN at around 400ms for a range of embedding extraction layers without extensive tuning of the optimizer and other hyperparameters.

Future work will include more challenging data as VoxCeleb [29]. Additionally, the following aspects in the presented approach offer room for further analysis: how to better inform the clustering stage about the common bond of multiple embeddings from the same utterance, beyond averaging; how to use deeper

RNN architectures to exploit more speaker data during training; how to scale to considerable more than 40 speakers/80 utterances; how to formulate a more suitable surrogate task for training, towards end-to-end neural clustering.

Acknowledgements: *We thank Timon Gygax and Jörg Egli, Benjamin Heusser and Savin Niederer, and Niclas Simmler and Amin Trabi for discussions & code.*

References

1. Anguera, X., Bozonnet, S., Evans, N., Fredouille, C., Friedland, G., Vinyals, O.: Speaker diarization: A review of recent research. TASLP (2012)
2. Barras, C., Zhu, X., Meignier, S., Gauvain, J.L.: Multistage speaker diarization of broadcast news. TASLP (2006)
3. Beigi, H.: Fundamentals of speaker recognition. Springer (2011)
4. Boulanger-Lewandowski, N., Bengio, Y., Vincent, P.: Audio chord recognition with recurrent neural networks. In: ISMIR (2013)
5. Burget, L., Plchot, O., Cumani, S., Glembek, O., Matějka, P., Brümmer, N.: Discriminatively trained probabilistic linear discriminant analysis for speaker verification. In: ICASSP (2011)
6. Chen, K., Salman, A.: Extracting speaker-specific information with a regularized siamese deep network. In: NIPS (2011)
7. Chen, K., Salman, A.: Learning speaker-specific characteristics with a deep neural architecture. Trans. Neural Networks (2011)
8. Dehak, N., Kenny, P.J., Dehak, R., Dumouchel, P., Ouellet, P.: Front-end factor analysis for speaker verification. TASLP (2011)
9. Fisher, W., Doddington, G., Marshall, G.K.: The DARPA speech recognition research database: Specification and status. In: Proceedings of the DARPA Speech Recognition Workshop. pp. 93–100 (1986)
10. Garcia-Romero, D., Snyder, D., Sell, G., Povey, D., McCree, A.: Speaker diarization using deep neural network embeddings. In: ICASSP (2017)
11. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
12. Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: ICML (2006)
13. Graves, A., Mohamed, A.r., Hinton, G.: Speech recognition with deep recurrent neural networks. In: ICASSP. IEEE (2013)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
15. Hsu, Y.C., Kira, Z.: Neural network-based clustering using pairwise constraints. arXiv:1511.06321 (2015)
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv:1412.6980 (2014)
17. Kolbæk, M., Yu, D., Tan, Z.H., Jensen, J.: Multi-talker speech separation and tracing with permutation invariant training of deep recurrent neural networks. arXiv:1703.06284 (2017)
18. Kotti, M., Moschou, V., Kotropoulos, C.: Speaker segmentation and clustering. Signal Processing (2008)
19. Koutnik, J., Greff, K., Gomez, F., Schmidhuber, J.: A clockwork rnn. arXiv:1402.3511 (2014)

20. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature (2015)
21. Lee, H., Pham, P., Largman, Y., Ng, A.Y.: Unsupervised feature learning for audio classification using convolutional deep belief networks. In: NIPS (2009)
22. Li, C., Ma, X., Jiang, B., Li, X., Zhang, X., Liu, X., Cao, Y., Kannan, A., Zhu, Z.: Deep speaker: an end-to-end neural speaker embedding system. arXiv:1705.02304 (2017)
23. Lukic, Y., Vogt, C., Dürr, O., Stadelmann, T.: Speaker identification and clustering using convolutional neural networks. In: MLSP (2016)
24. Lukic, Y., Vogt, C., Dürr, O., Stadelmann, T.: Learning embeddings for speaker clustering based on voice equality. In: MLSP (2017)
25. van der Maaten, L., Hinton, G.: Visualizing data using t-sne. JMLR (2008)
26. Meignier, S., Merlin, T.: Lium spkdiarization: an open source toolkit for diarization. In: CMU SPUD Workshop (2010)
27. Milner, R., Hain, T.: Dnn-based speaker clustering for speaker diarisation. In: Interspeech (2016)
28. Murtagh, F.: A survey of recent advances in hierarchical clustering algorithms. The Computer Journal (1983)
29. Nagrani, A., Chung, J.S., Zisserman, A.: Voxceleb: a large-scale speaker identification dataset. arXiv:1706.08612 (2017)
30. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: NIPS (2002)
31. Pascanu, R., Mikolov, T., Bengio, Y.: On the difficulty of training recurrent neural networks. In: ICML (2013)
32. Pawel Cyrta, Tomasz Trzciński, W.S.: Speaker diarization using deep recurrent convolutional neural networks for speaker embeddings. arXiv:1708.02840 (2017)
33. Reynolds, D.A.: Speaker identification and verification using gaussian mixture speaker models. Speech Communication (1995)
34. Reynolds, D.A., Quatieri, T.F., Dunn, R.B.: Speaker verification using adapted gaussian mixture models. Digital signal processing (2000)
35. Robinson, A., Fallside, F.: The utility driven dynamic error propagation network. University of Cambridge Department of Engineering (1987)
36. Rouvier, M., Bousquet, P.M., Favre, B.: Speaker diarization through speaker embeddings. In: EUSIPCO. IEEE (2015)
37. Schmidhuber, J.: Deep learning in neural networks: An overview. Neural Networks (2015)
38. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: CVPR (2015)
39. Sell, G., Garcia-Romero, D.: Diarization resegmentation in the factor analysis subspace. In: ICASSP (2015)
40. Stadelmann, T., Freisleben, B.: Unfolding speaker clustering potential: a biomimetic approach. In: ACM Multimedia (2009)
41. Stadelmann, T., Wang, Y., Smith, M., Ewerth, R., Freisleben, B.: Rethinking algorithm design and development in speech processing. In: ICPR (2010)
42. Variani, E., Lei, X., McDermott, E., Moreno, I.L., Gonzalez-Dominguez, J.: Deep neural networks for small footprint text-dependent speaker verification. In: ICASSP (2014)
43. Wang, Q., Downey, C., Wan, L., Mansfield, P.A., Moreno, I.L.: Speaker diarization with lstm. arXiv:1710.10468 (2017)
44. Yella, S.H., Stolcke, A., Slaney, M.: Artificial neural network features for speaker diarization. In: SLT Workshop. IEEE (2014)