Rouhe, Aku; Kaseva, Tuomas; Kurimo, Mikko

Speaker-Aware Training of Attention-Based End-to-End Speech Recognition Using Neural Speaker Embeddings

Published in:
2020 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2020 - Proceedings

DOI:
10.1109/ICASSP40776.2020.9053998

Published: 01/05/2020

Document Version
Peer reviewed version

SPEAKER-AWARE TRAINING OF ATTENTION-BASED END-TO-END SPEECH RECOGNITION USING NEURAL SPEAKER EMBEDDINGS

Aku Rouhe *Tuomas Kaseva* *Mikko Kurimo*

Aalto University, Department of Signal processing and acoustics

ABSTRACT

In speaker-aware training, a speaker embedding is appended to DNN input features. This allows the DNN to effectively learn representations, which are robust to speaker variability.

We apply speaker-aware training to attention-based end-to-end speech recognition. We show that it can improve over a purely end-to-end baseline. We also propose speaker-aware training as a viable method to leverage untranscribed, speaker annotated data.

We apply state-of-the-art embedding approaches, both i-vectors and neural embeddings, such as x-vectors. We experiment with embeddings trained in two conditions: on the fixed ASR data, and on a large untranscribed dataset. We run our experiments on the TED-LIUM and Wall Street Journal datasets. No embedding consistently outperforms all others, but in many settings neural embeddings outperform i-vectors.

Index Terms— end-to-end speech recognition, speaker-adaptation, speaker-aware training, speaker embedding

1. INTRODUCTION

Speaker independent speech recognition models attempt to find a suitable compromise for all speakers. Speaker adaptation lets the speaker independent models readjust to each speaker, by leveraging some speaker specific information. Fine-tuning the parameters of a DNN for each speaker separately would be computationally expensive and difficult, because the models are black boxes with a very large number of parameters. In hybrid HMM-DNN speech recognition, an effective speaker adaptation method is appending speaker embeddings to the input features, and having the DNN learn to use this information [1]. We call this speaker-aware training [2].

In the attention-based encoder-decoder end-to-end (AED) speech recognition [3, 4], fine-tuning the parameters for each speaker's acoustic characteristics is even more complicated, since the DNN also implements an implicit language model. In AED ASR, only a few speaker adaptation approaches have been proposed so far [5, 6], and to the best of our knowledge, speaker-aware training has not been applied to AED ASR.

In this work, we investigate speaker-aware training of AED ASR. We compare three different speaker embedding types: *i-vectors* [7], and two neural methods: *x-vectors* [8] and a *thin-ResNet* neural network architecture [9]. We present three main contributions.

Firstly, we show competitive word error rate improvements on the TED-LIUM [10] and Wall Street Journal (WSJ) [11] corpora. In our experiments, speaker-aware training outperforms an additional, end-to-end trained sequence summary network component [5].

Secondly, we propose speaker-aware training as a viable strategy to incorporate untranscribed data into the AED paradigm. Similar to the popular method of incorporating an external language model in shallow fusion [12], speaker-aware training is not purely end-to-end. The speaker embeddings are trained separately. This is beneficial, since the speaker embeddings can then be trained on untranscribed speech data, which only needs speaker annotations. We exploit state-of-the-art speaker embeddings trained on the large VoxCeleb datasets [13, 14]. We also compare these VoxCeleb embeddings with speaker embeddings trained only on the smaller fixed ASR datasets.

Thirdly, we show that neural embeddings outperform i-vectors in some settings, although no embedding consistently outperforms all others. In concurrent work, Rownicka et al. [15] explore neural embeddings for speaker-aware training of HMM-DNN ASR, but do not find improvements over i-vectors.

As a part of this work we present our findings in applying typical post-processing methods to the speaker embeddings: mean subtraction, dimensionality reduction and length normalization. Particularly, in our experiments, we find L2-normalization to be crucial. We show that neural embeddings may not need any other post-processing steps.

2. RELATED WORK

Only a few speaker adaptation methods have been proposed in AED ASR. In [5], a sequence summary network is added to the model architecture, and in [6], additional learning objectives are used to regularize the output of a speaker-dependent network.

Speaker-aware training has been applied to connectionist temporal classification models (CTC) [16, 17], which are trained with an end-to-end criterion. However, CTC models

are an implicit HMM [18], and in practice they are typically applied similarly to hybrid HMM-DNN models [19].

Rownicka et al. [15] are the first to present results where neural embeddings are used in speaker-aware training. In their work, i-vectors still outperform neural embeddings in speaker-aware ASR. The authors argue that compared to neural embeddings, i-vectors capture more additional information, other than speaker identity. In speech recognition, this other information, such as channel effects, are beneficial. However, also in concurrent work, Raj et al. [20] use probing tasks to show that x-vectors also encode channel information.

3. SPEAKER EMBEDDINGS

In speaker verification, the task is to distinguish whether two speech segments are spoken by the same speaker or not. Typically a speaker embedding extractor is trained separately, and then the embeddings are used as features for a binary classifier (such as cosine distance scoring or probabilistic linear discriminant analysis). [21] In this work, we use three embedding methods: i-vectors, x-vectors, and *thin-ResNet* embeddings.

I-vectors are based on factor analysis of Gaussian Mixture Model (GMM) supervectors. Thorough overviews of the method can be found in the related literature [22], but we omit it here for brevity.

3.1. Neural speaker embeddings

In the context of all-neural AED ASR, neural speaker embeddings could enable further work in fine-tuning the embeddings in the end-to-end ASR task. Furthermore, recently in speaker verification, neural speaker embeddings have been shown to outperform i-vectors [13, 8, 9].

X-vectors are a popular neural speaker embedding type. They use TDNN-layers, and are trained in speaker classification. After training, the embedding is extracted as the output of the second to last layer before the softmax. For details, see the original paper [8].

Thin-Resnet **embeddings** are also trained in speaker classification. Unlike the x-vectors, the model is a (2D) convolutional neural network, which operates directly on spectrograms, and includes an L2-normalization layer, after which the embedding is extracted. More details can be found in the original paper [9].

4. ATTENTION-BASED END-TO-END NEURAL SPEECH RECOGNITION

Attention-based encoder-decoder end-to-end neural speech recognition models [3, 4] have become a popular alternative to conventional HMM-based systems. These models directly transcribe speech to text. No language model or external lexicon is needed, but they are learned implicitly.

	VoxCeleb 1	VoxCeleb 2
Training hours	352	2442
Training speakers	1251	6112

Table 1. Details of the speaker embedding training datasets

	TED-LIUM	WSJ
Training hours	207	82
Dev hours	1.6	1.1
Test hours	2.6	0.7
Training speakers	1242	283
Dev speakers	8	10
Test speakers	10	8

Table 2. Details of the speech recognition corpora used in this paper

4.1. ESPnet encoder

The encoder in our model is slightly different from the standard approach above. Our implementation comes from the ESPnet toolkit[23]. The encoder is trained in a multi-task setting, by adding a CTC-decoder in parallel. The CTC-decoder is also used in inference, by interpolating the likelihoods from both decoders. [24]

ESPnet also implements a hybrid convolution and BLSTM-based encoder. However the convolution operation does not make sense for the appended speaker embeddings, because the embedding dimensions do not have any ordered structure. Therefore, in our experiments, we do not use the convolutional front-end.

5. EXPERIMENTS

5.1. Data

The untrascribed data x-vector and i-vector embedding extractors are trained on VoxCeleb [13] and VoxCeleb2 [14]. Furthermore, for the x-vectors, a large amount of data augmentation is applied [8]. The *thin-ResNet* model is only trained on VoxCeleb2. Table 1 lists the salient dataset details.

We run the speech recognition experiments on the TED-LIUM (release 2) [10] and Wall Street Journal (si-284 training set) [11] datasets. Table 2 shows the dataset characteristics. The fixed data speaker embeddings are trained on these ASR datasets' respective training sets.

5.2. Embedding models

For the untranscribed VoxCeleb data embeddings, we use pretrained models available online [25, 26]. Table 3 compares these embeddings in a speaker verification task. Neural embeddings outperform i-vectors.

For the embeddings trained on the ASR data (we call this the fixed data scenario), we adjust the embedding model hy-

perparameters to better suit these datasets, which are smaller than the VoxCeleb datasets. For the i-vector model, we choose 512 full-covariance Gaussians in the universal background model, and 100-dimensional i-vectors, without LDA. For x-vectors, the configuration is otherwise the same as the original VoxCeleb x-vector model [8], but the embedding size is halved to 256, and the number of training epochs doubled to 6. We arrived at these values using a heuristic: we pick the values which yield the highest adjusted rand index (ARI) [27], when clustered using spherical K-means. Spherical K-means is L2-normalized, which was shown to be important in earlier experiments. High ARI should reflect consistent embeddings, which we believe should help in ASR, and the procedure is computationally inexpensive. We first optimized the values on the TED-LIUM data. On WSJ, the TED-LIUM i-vector configuration resulted in a perfect 1.0 ARI, so we decided to simply reuse the TED-LIUM-tuned configurations without further optimization.

Furthermore, in the fixed data setting, we do not test the *thin-ResNet* embeddings, because the implementation was not readily available in the Kaldi toolkit.

	EER
i-vector [25]	5.3
x-vector [25]	3.1
thin-ResNet [9]	3.22

Table 3. The pretrained VoxCeleb speaker embeddings compared in speaker verification, on the VoxCeleb 1 test set. In speaker verification, the common performance metric is equal error rate (EER). It is the error rate at which there are equally many false acceptances and false rejections.

5.3. Post-processing the embeddings

For the i-vector and x-vector embeddings, we test standard post-processing procedures: subtracting the training set mean, dimensionality reduction by LDA, and using L2-normalization. The LDA transform is trained on the speech recognition training set; we reduce the dimensionality to 200, which is the x-vector default. The *thin-ResNet* output is already L2-normalized, which would be undone by any post-processing, so therefore we use the *thin-ResNet* outputs as they are.

Table 4 shows x-vector and i-vector results without LDA, and either subtracting the global mean or not. These experiments indicate that with x-vectors the mean subtraction hurts performance and with i-vectors it helps. We keep these choices for all x-vector and i-vector experiments.

In the Kaldi toolkit [28] (which we use for feature dumping), the default is to normalize to $\sqrt{d}$, where d is the dimensionality of the embedding. In preliminary experiments, we found that it is crucial to normalize to length 1. Otherwise, the embeddings only hurt performance. Thus in all of reported results, we have applied L2-normalization to unit length.

TED-LIUM	Test		Dev	
	No LM	+LM	No LM	+LM
x-vector	**20.1**	**17.2**	**20.9**	**18.1**
x-vector subtract mean	20.5	**17.2**	21.0	18.2
i-vector	20.7	17.8	21.5	18.7
i-vector subtract mean	**20.4**	**17.2**	**21.0**	**18.3**

Table 4. WER results with and without mean subtraction, for the VoxCeleb i-vector and x-vector embeddings without LDA.

5.4. ASR model configurations

With all of our models, we follow the same ESPnet recipes as Delcroix et al. for their sequence summary network (SeqSum) approach [5], except we do not use convolutional layers in the encoder for speaker-aware models as explained in section 4.1. We also train standard character level RNNLMs similar to Delcroix et al. [5], on the datasets' respective text resources, although note that Delcroix et al. do not present LM results on TED-LIUM. Details are omitted here for brevity. We achieve very similar baseline results, and therefore we present some of their results in comparison with ours.

In all models, the encoder consists of six 320-unit BLSTM layers, which subsample the input in time by a factor of four. The decoder has one 300-unit LSTM layer, and uses location-based attention, followed by a softmax layer, which outputs a distribution over characters (32 in TED-LIUM, 50 in WSJ). The models are trained for 15 epochs with the adadelta optimizer, with a batchsize of 30. In decoding use beam search with a beamsize of 20 for TED-LIUM and 30 for WSJ.

The encoders are trained with the multitask CTC-loss of ESPNet, and this is incorporated in decoding [23]. We also train some models on the WSJ task without the CTC-loss. Without the CTC-loss we retune the language model weight for the baseline model on the Dev93 set and use that same weight in all other non-CTC-loss experiments.

Our input features are mean and variance normalized 80-dimensional Mel-filterbank energies, and pitch information, which might not contribute in English, but we retain it for conformity. We extract one speaker embedding for the whole utterance, and append it to each feature vector.

5.5. ASR Results

Table 5 shows the main results of our experiments. The models without the CTC multitask loss are not directly comparable, so their results are presented separately, in Table 6.

On the WSJ dataset, when not using an LM we get a better baseline without the CTC-loss than with it. This is likely due

	TED-LIUM	Test		Dev	
		No LM	+LM	No LM	+LM
Fixed	Baseline	21.7	18.6	22.6	20.0
	SeqSum [5]	21.1	-	21.7	-
	i-vector$_{100}$	**20.9**	**17.9**	**21.4**	**18.9**
	x-vector$_{256}$	21.5	18.4	23.0	20.0
+VoxCeleb	i-vector$_{200\text{-LDA}}$	20.2	17.4	**20.7**	18.2
	i-vector$_{400}$	20.4	**17.2**	21.0	18.3
	x-vector$_{200\text{-LDA}}$	20.9	17.4	21.6	18.6
	x-vector$_{512}$	**20.1**	**17.2**	20.9	**18.1**
	thin-ResNet$_{512}$	20.7	**17.2**	21.0	18.3

	WSJ	Eval92		Dev93	
		No LM	+LM	No LM	+LM
Fixed	Baseline	17.5	9.3	22.1	13.2
	SeqSum [5]	16.3	8.7	21.3	13.2
	i-vector$_{100}$	17.6	**8.5**	22.3	**11.3**
	x-vector$_{256}$	**16.2**	8.6	**20.3**	11.6
+VoxCeleb	i-vector$_{200\text{-LDA}}$	17.2	9.1	21.2	11.9
	i-vector$_{400}$	**15.3**	**8.0**	20.5	11.7
	x-vector$_{200\text{-LDA}}$	18.8	9.5	25.0	13.5
	x-vector$_{512}$	16.2	8.7	20.5	**11.2**
	thin-ResNet$_{512}$	16.7	8.7	**20.4**	11.6

Table 5. WER results of the main speech recognition experiments. SeqSum refers to the sequence summary network approach of Delcroix et al. The embedding methods denote dimension, and whether LDA was used, in subscript. The +VoxCeleb sections present the results with the pretrained VoxCeleb embeddings. The Fixed sections present results with embeddings trained on the fixed ASR data.

to the original recipe being tuned for the performance with a language model.

6. DISCUSSION

We achieve around 7% relative WER improvements with the VoxCeleb speaker embeddings. The VoxCeleb embeddings consistently perform better than the fixed ASR data embeddings, which obtain around around 4% relative improvements. The fixed data embeddings still consistently outperform the end-to-end sequence summary method. We see speaker-aware training as a useful competitive evaluation baseline when developing true end-to-end methods, such as the sequence summary network.

No single embedding type consistently outperforms others. However, when embeddings are trained on the larger VoxCeleb dataset, the neural embeddings often outperform i-vectors. We suspect the neural embeddings are better able to leverage very large training sets. The *thin-ResNet* model is, without any modification, a good candidate for end-to-end finetuning in future work. For the x-vector approach, it seems an L2-normalization layer is needed.

	WSJ	Eval92		Dev93	
		No LM	+LM	No LM	+LM
	Baseline	14.9	10.7	18.7	**13.7**
+VoxCeleb	i-vector$_{200\text{-LDA}}$	16.0	12.9	19.8	15.4
	i-vector$_{400}$	13.2	10.9	17.5	14.5
	x-vector$_{200\text{-LDA}}$	16.0	12.4	20.1	15.5
	x-vector$_{512}$	13.5	**10.4**	**16.9**	15.0
	thin-ResNet$_{512}$	**12.9**	10.6	17.2	14.1

Table 6. Results without the CTC task, i.e. a purely attentional model. Again, the embedding methods denote dimension, and whether LDA was used, in subscript. All of the speaker-aware methods used the Voxceleb embeddings.

Our hyperparameter tuning procedure for the fixed ASR data embeddings was quite ad-hoc. The ARI metric is probably closer to the speaker verification metric than ASR. However, the VoxCeleb embeddings are also originally tuned for speaker verification. Good, sound criteria, which could be used to separately optimize speaker embeddings for speaker-aware ASR training, are an open research question.

Of the embedding post-processing steps, we see that L2-normalization is crucial. We suspect the additional sensitivity of the normalization to unit length might not be universal, but rather particular to our implementation. Mean subtraction seems to work for i-vectors, but not for x-vectors. In most experiments, LDA did not help, with the exception of the VoxCeleb i-vector embeddings on TED-LIUM. However, we do not investigate different LDA dimension sizes.

7. CONCLUSIONS

We have shown that speaker-aware training is a competitive speaker adaptation approach in attention-based end-to-end ASR. We propose speaker-aware training as a viable strategy to incorporate untranscribed, speaker annotated data. When trained on large speaker annotated data, we find that neural embeddings can outperform i-vectors in speaker-aware ASR.

8. ACKNOWLEDGEMENTS

This work was supported by the European Unions Horizon 2020 research andinnovation programme via the project MeMAD (GA780069). We acknowledge the computational resources provided by the Aalto Science-IT project.

9. REFERENCES

[1] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, Dec 2013, pp. 55–59.

[2] Xiaodong Cui, Vaibhava Goel, and George Saon, "Embedding-based speaker adaptive training of deep neural networks," in *Proc. Interspeech 2017*, 2017, pp. 122–126.

[3] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 4960–4964.

[4] D. Bahdanau, J. Chorowski, D. Serdyuk, P. Brakel, and Y. Bengio, "End-to-end attention-based large vocabulary speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 4945–4949.

[5] Marc Delcroix, Shinji Watanabe, Atsunori Ogawa, Shigeki Karita, and Tomohiro Nakatani, "Auxiliary feature based adaptation of end-to-end asr systems," in *Proc. Interspeech 2018*, 2018, pp. 2444–2448.

[6] Zhong Meng, Yashesh Gaur, Jinyu Li, and Yifan Gong, "Speaker Adaptation for Attention-Based End-to-End Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 241–245.

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

[8] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[9] Weidi Xie, Arsha Nagrani, Joon Son Chung, and Andrew Zisserman, "Utterance-level aggregation for speaker recognition in the wild," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5791–5795.

[10] Anthony Rousseau, Paul Deléglise, and Yannick Esteve, "Enhancing the ted-lium corpus with selected data for language modeling and more ted talks.," in *LREC*, 2014, pp. 3935–3939.

[11] Douglas B Paul and Janet M Baker, "The design for the wall street journal-based csr corpus," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

[12] Anjuli Kannan, Yonghui Wu, Patrick Nguyen, Tara N Sainath, ZhiJeng Chen, and Rohit Prabhavalkar, "An analysis of incorporating an external language model into a sequence-to-sequence model," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 1–5828.

[13] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *INTERSPEECH*, 2017.

[14] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *INTERSPEECH*, 2018.

[15] Joanna Rownicka, Peter Bell, and Steve Renals, "Embeddings for dnn speaker adaptive training," in *2019 IEEE Workshop on Automatic Speech Recognition and Understanding*, 2019, Accepted for publication, preprint accessed online 15.10.2019: https://arxiv.org/pdf/1909.13537.pdf.

[16] Kartik Audhkhasi, Bhuvana Ramabhadran, George Saon, Michael Picheny, and David Nahamoo, "Direct acoustics-to-word models for english conversational speech recognition," in *Proc. Interspeech 2017*, 2017, pp. 959–963.

[17] Natalia Tomashenko and Yannick Estève, "Evaluation of feature-space speaker adaptation for end-to-end acoustic models," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, 2018.

[18] Hossein Hadian, Hossein Sameti, Daniel Povey, and Sanjeev Khudanpur, "End-to-end speech recognition using lattice-free mmi," in *Proc. Interspeech 2018*, 2018, pp. 12–16.

[19] Yajie Miao, Mohammad Gowayyed, and Florian Metze, "Eesen: End-to-end speech recognition using deep rnn models and wfst-based decoding," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 167–174.

[20] Desh Raj, David Snyder, Daniel Povey, and Sanjeev Khudanpur, "Probing the information encoded in x-vectors," in *2019 IEEE Workshop on Automatic Speech Recognition and Understanding*, 2019, Accepted for publication, preprint accessed online 15.10.2019: https://arxiv.org/pdf/1909.06351.pdf.

[21] John HL Hansen and Taufiq Hasan, "Speaker recognition by machines and humans: A tutorial review," *IEEE Signal processing magazine*, vol. 32, no. 6, pp. 74–99, 2015.

[22] O. Glembek, L. Burget, P. Matějka, M. Karafiát, and P. Kenny, "Simplification and optimization of i-vector extraction," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2011, pp. 4516–4519.

[23] Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai, "Espnet: End-to-end speech processing toolkit," in *Proc. Interspeech 2018*, 2018, pp. 2207–2211.

[24] S. Kim, T. Hori, and S. Watanabe, "Joint ctc-attention based end-to-end speech recognition using multi-task learning," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017, pp. 4835–4839.

[25] David Snyder, "x-vector and i-vector pretrained models download page," Online, http://kaldi-asr.org/models/m7, accessed 8.7.2019.

[26] Weidi Xie, "thin-resnet vlad pretrained models," Online, https://github.com/WeidiXie/VGG-Speaker-Recognition, accessed 8.7.2019.

[27] Lawrence Hubert and Phipps Arabie, "Comparing partitions," *Journal of classification*, vol. 2, no. 1, pp. 193–218, 1985.

[28] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. Dec. 2011, IEEE Signal Processing Society, IEEE Catalog No.: CFP11SRW-USB.