Reducing the Inter-speaker Variance of CNN Acoustic Models Using Unsupervised Adversarial Multi-task Training

László Tóth[1(✉)] and Gábor Gosztolya[2]

[1] Institute of Informatics, University of Szeged, Szeged, Hungary
tothl@inf.u-szeged.hu
[2] MTA-SZTE Research Group on Artificial Intelligence, Szeged, Hungary
ggabor@inf.u-szeged.hu

Abstract. Although the Deep Neural Network (DNN) technology has brought significant improvements in automatic speech recognition, the technology is still vulnerable to changing environmental conditions. The adversarial multi-task training method was recently proposed to increase the domain and noise robustness of DNN acoustic models. Here, we apply this method to reduce the inter-speaker variance of a convolutional neural network-based speech recognition system. One drawback of the baseline method is that it requires speaker labels for the training dataset. Hence, we propose two modifications which allow the application of the method in the unsupervised scenarios; that is, when speaker annotation is not available. Our approach applies unsupervised speaker clustering, which is based on a standard feature set in the first case, while in the second case we modify the network structure to perform speaker discrimination in the manner of a Siamese DNN. In the supervised scenario we report a relative error rate reduction of 4%. The two unsupervised approaches achieve smaller, but consistent improvements of about 3% on average.

Keywords: Convolutional neural network · Siamese neural network · Multi-task · Adversarial training · Unsupervised training

1 Introduction

Since the introduction of Deep Neural Network-based technologies, the error rate of speech recognition systems has decreased significantly [8]. However, improving the robustness of the recognizers is still an active area of research, as even these DNN-based systems are sensitive to various adversarial environmental conditions such as background noise, reverberant environments, and different speaker accents. The sensitivity to these factors can partly be explained by the fact that neural networks are inclined to overfit the actual training data, and generalize poorly to cases that were not seen during training. Among other options, regularization methods are routinely applied to tackle this overfitting phenomenon [6].

A. A. Salah et al. (Eds.): SPECOM 2019, LNAI 11658, pp. 481–490, 2019.
https://doi.org/10.1007/978-3-030-26061-3_49

For example, it is known that presenting multiple tasks to the network at the same time – known as multi-task training [3] – also has a regularization effect. That is, having to solve two (or more) similar, but slightly different tasks at the same time forces the network to find a more general and more robust inner representation. Multi-task training has been shown to reduce the speech recognition error rate in several studies [2,14].

While multi-task training seeks to minimize the error of both tasks, there is a newer variant of the method known as adversarial multi-task training [5]. Here, we maximize the error of the secondary task. With this modification, we expect the network to prefer inner representations that are *invariant* with respect to the secondary task. In speech technology, adversarial multi-task training has mostly been applied to enhance the domain independence (i.e. noise robustness) of DNN acoustic models [4,15]. But we also found examples where it is used to make the system less sensitive to other factors like the accent of the user [16]. In this study, we seek to apply the adversarial multi-task training method to alleviate the sensitivity of speech recognizers to the identity of the actual speaker. Our starting point will be the recent study of Meng et al. [12]. The approach they described requires a training data set that contains speaker annotation. However, most of the current large training databases contain only transcripts of the text spoken without any speaker labels, which renders the method of Meng et al. inapplicable in practice. Here, we experiment with two possible extensions that do not require speaker annotation, and hence these methods are unsupervised in terms of the speakers. For the experimental evaluation we use the TIMIT database, which contains brief samples from significantly more speakers than the corpus used by Meng et al, so the task is presumably more difficult. Moreover, as TIMIT contains a speaker identifier for each file, we can directly compare the supervised approach with the proposed unsupervised methods.

2 Multi-task and Adversarial Multi-task Training

The typical Y-shaped architecture of a multi-task deep neural network is shown schematically in Fig. 1. The network has a dedicated output layer for both tasks (addressing more tasks is also possible, but here we shall assume there are just two tasks). Typically, the uppermost hidden layers are also arranged into task-specific counterparts. Both output layers have a corresponding error function, which are denoted in the figure by L_{CD} and L_S, while the corresponding parameters (weights) are denoted by θ_{CD} and θ_S. Although the network has two output layers, it has only one input layer, and the lower layers are also shared between the two tasks. This forces the network to find a hidden representation in these shared layers which is useful for both tasks. During error backpropagation, the errors coming from the two branches are combined by a simple linear combination. We can perform this using equal weights, but typically the accuracy of one of the tasks is more important for us than that of the other. We can express this importance using a λ weight in the combination formula (see Fig. 1). In our case, the more important main task will be speech recognition (the recognition of the

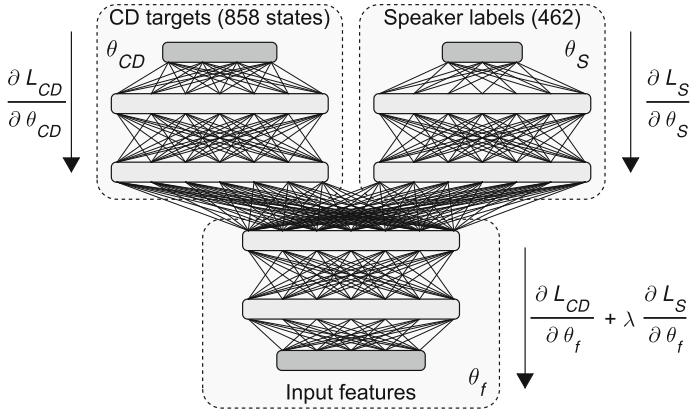

Fig. 1. Schematic structure of an (adversarial) multi-task neural network.

Hidden Markov Model states), while the secondary task will be the recognition of the actual speaker. Note that the secondary task is added only during training, as we expect it to help learn the main task. However, the actual accuracy score attained by this branch of the net is used only for verification purposes, and this branch is discarded in the evaluation phase.

Besides λ, another parameter of the model is the depth where the two branches should join. Intuitively, more different tasks require more task-specific and fewer shared layers (see, e.g. [18]), but the optimal configuration can be found only experimentally. Likewise, it is impossible to tell in advance whether a certain secondary task will help learn a given main task, but intuitively, the secondary task should be related, but slightly different from the main task.

To our knowledge, multi-task training was first applied in speech recognition in a study by Lu et al., where the secondary task was to clean the noisy speech features [11]. In the deep learning framework it was first applied by Seltzer and Droppo, who used the recognition of the phonetic context as a secondary task along with the main phone recognition task [14]. A similar solution was implemented by Bell and Renals, who combined the tasks of context-dependent and context-independent modeling [2].

Multi task-learning has a variant called *adversarial* multi-task learning [5]. Instead of preferring a hidden representation that helps handle both tasks, adversarial multi-task learning seeks to find a hidden representation that is invariant with respect to the secondary task, meaning that it contains no information that would allow the identification of the secondary targets. In adversarial training the Y-shaped network structure is the same as that for the standard multi-task model. However, we will try to *maximize* the error of the secondary task instead of minimizing it. Technically, it is realized by still minimizing the secondary error, but using a *negative* value for λ. This way, the task-specific secondary branch tries to solve the secondary task, but the shared layers will seek a representation that works against this (performing a sort of 'min-max' optimization [16]).

The adversarial multi-task training approach was first used in speech technology in 2016 [15]. Most authors mainly applied it to make the neural network 'domain-invariant' ([4]); that is, insensitive to the actual background noise, but we know of examples where the domain corresponds to speaker accent [16], or the identity of the actual speaker [12].

Shinohara recommends introducing adversarial training gradually, by slowly increasing the weight of the adversarial branch in each iteration [15]. Following his recommendation, we configured λ so as to attain its final value after 10 iterations, setting its absolute value in the kth iteration to

$$\lambda_k = min(\frac{k}{10}, 1) \cdot \lambda.$$

3 Experimental Set-Up

We used the English TIMIT speech dataset for our experiments. Though this dataset is now considered tiny for speech recognition purposes, we chose it because it also contains speaker annotations. Moreover, it is ideal in the sense that is contains samples from a lot of speakers in a uniform distribution. The train set consists of 8 sentences from 462 speakers, while the core test set comprises 24 other (independent) speakers. As the development set, we randomly separated 44 speakers from the train set, and we evaluated the models on the core test set.

For the recognition, we applied a standard Hidden Markov Model - Deep Neural Network (HMM/DNN) hybrid [8]. The neural network component was trained on a mel-spectrogram, and it contained convolutional neurons in its lowest layer (performing frequency-domain convolution) [1]. The convolutional layer was followed by two additional fully connected layers, which together formed the shared part of the network. The task-specific parts of the network consisted of 1-1 hidden layers, as this was found optimal in preliminary experiments. All the hidden layers contained 2000 rectified (ReLU) neurons. In the speech recognition (or main) branch, the output layer consisted of 858 softmax neurons, corresponding to the states of the HMM. In the speaker recognition (or secondary) branch, the 462 softmax neurons had to identify the speakers of the database. The network was trained using standard backpropagation, applying the frame-level cross-entropy function as the loss function for both output layers.

4 Results with Supervised Adversarial Training

In the first adversarial training experiment we trained the network in a supervised manner; that is, using the original speaker labels as training targets for the secondary task. To find the optimal value of λ, we varied it between -0.05 and -0.25 with a step size of 0.05. Table 1 shows the frame-level error rates got for both branches. It should be mentioned that for the secondary task we have scores only for the train set, and we listed these scores only to verify the behavior

Table 1. The frame-level and phone-level error rates obtained for various values of λ, using supervised training.

Parameter	Frame error rate		Phone error rate	
λ	(train, sec. task)	(dev, main task)	(dev. set)	(test set)
0 (baseline)	24.7%	35.4%	16.6%	18.8%
−0.05	70.8%	34.7%	16.3%	18.4%
−0.10	77.2%	34.8%	**16.0%**	**18.0%**
−0.15	86.4%	34.9%	16.2%	18.1%
−0.20	89.3%	35.1%	16.3%	18.2%

of the network, as this branch of the network is not used by the final recognition system. For the main task we listed the results on the independent development set, as this tells us more about the generalization ability of the model. The rightmost columns of the table show the phone recognition error rates on the development and test sets, obtained after performing HMM decoding using the state-level probabilities produced by the main branch of the network. The first row contains the baseline result, which is obtained with $\lambda = 0$. We will call this the 'passive' configuration, as in this case the secondary branch is allowed to learn, but it cannot influence the shared hidden representation. The table shows that by decreasing λ, the frame error rate of the secondary task rose consistently, just as one would expect. The frame error rate of the main task decreased in parallel until it reached its optimum point, then it started to rise again when λ became smaller. The phone recognition error rate attained its optimum both for the development and test sets at $\lambda = -0.1$, but the scores are consistently lower than those for the baseline system for all parameter values tested. In the best case the relative error rate reduction was 4.2% on the test set. As for Meng et al., they reported an error rate reduction of 5% [12].

It is well known that adding a small disturbance to either the weights or the input of the neural network can reduce overfitting, and bring a slight improvement in the scores [6]. We wanted to verify that the improvement was not simply due to this effect. Hence, we calculated the variance of the recognition scores with respect to the speakers, and we found that it decreased by about 10%, compared to the baseline model. This confirms that the adversarial training method indeed makes the model less speaker-sensitive – although the model is still far from being 'speaker-invariant'. Meng et al. themselves reported additional improvements by applying speaker adaptation after adversarial training [12]. We found earlier that the application of CNNs instead of fully connected DNNs reduces the inter-speaker variance by about 5.7% [17]. As we used CNNs here, the reduction of 10% was obtained in addition to this previously reported improvement.

Table 2. The error rates obtained using the conventional speaker clustering method.

No. of clusters	Parameter λ	Frame error rate (dev)	Phone error rate (dev)	Phone error rate (test)
–	0 (baseline)	35.4%	16.6%	18.8%
50	−0.10	34.8%	16.4%	18.6%
100	−0.15	34.6%	**16.0%**	**18.0%**
150	−0.10	34.8%	**16.0%**	18.3%
200	−0.10	34.8%	16.2%	18.4%
250	−0.10	34.6%	**16.0%**	18.3%

5 Unsupervised Training Without Speaker Labels

The approach we presented in the previous section has a big drawback, namely that it requires the annotation of the speakers. Though this is available in the case of the TIMIT dataset, nowadays we train our systems on much larger corpora, which are usually recorded under more natural conditions. For these databases a precise speaker-level segmentation and labelling is typically not available, which means that our adversarial training method cannot be directly applied. Here, we propose two approaches to overcome this limitation. These methods do not require speaker annotation, but they create the training targets for the speaker classifier branch of the network in an unsupervised manner. The only assumption is that there is no speaker change within a file, so each file belongs to exactly one speaker. This is a much weaker constraint in general than that of the availability of a speaker annotation.

5.1 Conventional Speaker Clustering

Not having speaker labels, we can apply a clustering method to group the files into clusters, according to the similarity of the speakers' voices. Many conventional algorithms exist for this, and we chose a hierarchical clustering method that was accessible to us. The original algorithm applies a bottom-up, agglomerative hierarchical cluster method, which merges clusters based on the generalized likelihood ratio of Gaussian models fit on standard acoustic features like mel-frequency cepstral coefficients (MFCCs) [7]. Various modifications of the algorithm were later suggested by Wang et al. [19] and Kaya et al. [9].

When using the clustering algorithm, the number of clusters becomes an additional parameter. We tried to vary this value between 50 and 250 with a step size of 50. We applied the adversarial training method just as before, but the speaker labels were replaced by the automatically found cluster identifiers. Table 2 lists the recognition results obtained in this case. For each cluster size we report only the λ value that gave the best score on the development set. On the development set we attained the same error rate (16.0%) as that with the supervised approach for several cluster sizes. However, the improvement did

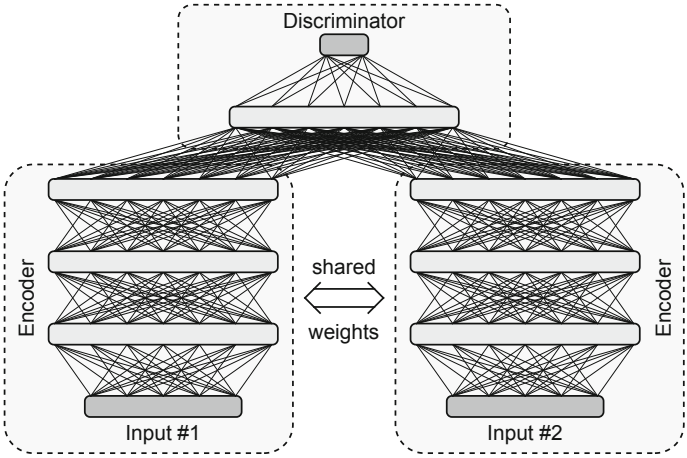

Fig. 2. Illustration of the architecture of a Siamese neural network.

not carry over to the test set, where the best supervised score (18.0%) was achieved only in one configuration. Disregarding the cluster size of 50 (which gave an inferior performance), the average score on the test set was 18.25%, which corresponds to a 3% relative error rate reduction over the baseline.

5.2 Clustering Using a Siamese Multi-task Network

The conventional speaker clustering method we applied is built on MFCCs and Gaussian modelling, but our acoustic model is a CNN that uses mel-frequency energy features. This means that we calculate two types of features and two types of models, which is a waste of resources. We could do better if we adjusted our network (more precisely, its speaker classifier branch) to the unsupervised task. The approach we applied is based on the method outlined by Ravanelli and Bengio [13], but it is also closely related to the concept of Siamese neural networks [20]. Siamese networks are usually applied to decide whether two images depict the same object or not, and they consist of two main parts (see Fig. 2). The upper, discriminator part is trained to discriminate *a pair* of input vectors. In our case, the discriminator consists of one hidden layer and an output layer of just two neurons, which try to decide whether two input speech frames belong to the same speaker or not. The lower, encoder part seeks to find the optimal representation for this discrimination. As we try to discriminate a pair of inputs, the encoder is present in two copies in the network, but these are practically identical (technically, this can be solved by weight sharing, for example).

In the case of our multi-task network, the lower, shared part will serve as the encoder, and the network branch corresponding to the speaker classification task has to be replaced by the discriminator. We had to solve two problems to achieve this. First, the discriminator required two input vectors instead of just one. Second, we had to create pairs that came from the same file (negative examples),

Algorithm 1. Constructing batches of data that allows the training of the speaker discriminator and the classifier network branches in a multi-task fashion

Let N denote the batch size
$f[k]$ (k = 1, ..., N) will store the batch of feature vectors
$l_c[k]$ (k = 1, ..., N) will store the training targets for state classification
$l_s[k]$ (k = 2, ..., N) will store the training targets for the speaker discriminator

$j \leftarrow$ a randomly selected file index from the training file list
$f[1] \leftarrow$ a randomly sampled frame from the jth file
$l_c[1] \leftarrow$ the state label of $f[1]$
$l_s[1] \leftarrow$ undefined ▷ Not used during training
for (k=2; $k \leq N$; k++) **do**
  if k is even **then**
    $f[k] \leftarrow$ a randomly sampled frame from the jth file
    $l_c[k] \leftarrow$ the state label of $f[k]$
    $l_s[k] \leftarrow 0$ ▷ $f[k-1]$ and $f[k]$ are from the *same* file
  if k is odd **then**
    $j \leftarrow$ a randomly selected new file index, different from the previous value of j
    $f[k] \leftarrow$ a randomly sampled frame from the jth file
    $l_c[k] \leftarrow$ the state label of $f[k]$
    $l_s[k] \leftarrow 1$ ▷ $f[k-1]$ and $f[k]$ are from *different* files

Train the main network branch using $f[k]$ and $l_c[k]$ $(k = 1, ..., N)$
Train the secondary branch using the $< f[k-1], f[k] >$ pairs and $l_s[k]$ $(k = 2, ..., N)$

and pairs that came from different files (positive examples) in turn. We created an algorithm that constructs the data batches in such a way that it allows one to train the speech recognition and the speaker discriminator branches in parallel, in a multi-task fashion (see Algorithm 1). That is, the N data vectors returned by the algorithm can be used to train the speaker recognition branch directly, while the $N-1$ pairs of neighboring vectors are alternating positive and negative examples for the 2-class speaker discriminator branch. We should add that while we want to discriminate the speakers, our implementation approximates this by discriminating the *files*, as we have no access to speaker labels. However, as long as the train set consists of many speakers, the chance of mislabelling a pair is actually quite low (in this case, 8 files out of 418 belong to the same speaker).

In our preliminary tests the Siamese speaker discriminator branch of the network attained an error rate of 18% in passive training mode. In multi-task mode the error decreased to about 2%, which is similar to that reported by Ravanelli and Bengio [13]. However, both in multi-task and in adversarial multi-task training the discriminator branch had only a negligible influence on the accuracy of the other branch. We think that deciding whether the speaker is the same or different is a much weaker constraint on the hidden representation than actually identifying the speakers.

As we were unable to apply adversarial training using the Siamese branch directly, we opted for a two-stage approach. After training the network, we performed a clustering on the training files, using the discriminator output as the

Table 3. The error rates obtained using the Siamese network-based clustering method.

No. of clusters	Parameter λ	Frame error rate (dev)	Phone error rate	
			(dev)	(test)
—	0 (baseline)	35.4%	16.6%	18.8%
50	−0.10	34.9%	16.4%	18.4%
100	−0.35	34.8%	**16.0%**	**18.3%**
150	−0.30	34.6%	16.1%	18.4%
200	−0.50	34.7%	16.3%	18.3%
250	−0.55	34.8%	16.2%	18.2%

distance function. We applied complete-linkage agglomerative hierarchical clustering [10], where the distance between two files was estimated in the following way. The speaker discriminator branch outputs posterior estimates (scores between 0 and 1) of whether two frames belong to the same file or not. We defined the distance between any two files as the average of these posterior values over ten randomly selected frame pairs. After we had performed the clustering, we repeated the training of the adversarial multi-task network using the cluster labels as training targets for the secondary branch.

The recognition error rates obtained with this clustering method are shown in Table 3. Similar to the standard clustering method, the score obtained with 50 clusters is just slightly better than the baseline score. For larger cluster sizes, on the development set the results are typically slightly worse than those got with the standard clustering method. However, on the test set the average improvement is not significantly different, corresponding to a relative error rate reduction of about 3% relative to the baseline.

6 Summary

Here, we examined the applicability of adversarial multi-task training to reduce the inter-speaker variance of CNN acoustic models. First, we investigated supervised training that requires speaker annotation, and then we proposed two unsupervised solutions to generate training targets when speaker labels are not available. In the supervised case we reported relative phone error rate reductions of 4%, and both unsupervised approaches performed slightly worse, giving an error rate reduction of about 3%. Currently both proposed methods require a clustering step, but in the future we intend to modify the Siamese network-based approach so that it can work in one training pass, without the need for clustering and re-training.

Acknowledgments. This research was supported by the Ministry of Human Capacities, Hungary by grant number TUDFO/47138-1/2019-ITM. László Tóth was supported by the Janos Bolyai Research Scholarship of the Hungarian Academy of Sciences. The GPU card used for the computations was donated by the NVIDIA Corporation.

References

1. Abdel-Hamid, O., Mohamed, A., Jiang, H., Penn, G.: Applying convolutional neural network concepts to hybrid NN-HMM model for speech recognition. In: Proceedings of ICASSP, pp. 4277–4280 (2012)
2. Bell, P., Renals, S.: Regularization of deep neural networks with context-independent multi-task training. In: Proceedings of ICASSP, pp. 4290–4294 (2015)
3. Caruana, R.: Multitask learning. J. Mach. Learn. Res. **17**(1), 41–75 (1997)
4. Denisov, P., Vu, N., Font, F.: Unsupervised domain adaptation by adversarial learning for robust speech recognition. In: Proceedings of ITG Conference of Speech Communication (2018)
5. Ganin, Y., et al.: Domain-adversarial training of neural networks. J. Mach. Learn. Res. **17**(59), 1–35 (2016)
6. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
7. Han, K.J., Kim, S., Narayanan, S.S.: Strategies to improve the robustness of Agglomerative Hierarchical Clustering under data source variation for speaker diarization. IEEE Trans. Audio Speech Lang. Process. **16**(8), 1590–1601 (2008)
8. Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A., Jaitly, N., et al.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Sig. Process. Mag. **29**(6), 82–97 (2012)
9. Kaya, H., Karpov, A., Salah, A.: Fisher vectors with cascaded normalization for paralinguistic analysis. In: Proceedings of Interspeech, pp. 909–913 (2015)
10. Krznaric, D., Levcopoulos, C.: Fast algorithms for complete linkage clustering. Discrete Comput. Geom. **19**(1), 131–145 (1998)
11. Lu, Y., et al.: Multitask learning in connectionist speech recognition. In: Proceedings of Australian International Conference on Speech Science and Technology (2004)
12. Meng, Z., et al.: Speaker-invariant training via adversarial learning. In: Proceedings of ICASSP, pp. 5969–5973 (2018)
13. Ravanelli, M., Bengio, Y.: Learning speaker representation with mutual information. Interspeech (2019). https://arxiv.org/abs/1812.00271
14. Seltzer, M., Droppo, J.: Multi-task learning in deep neural networks for improved phoneme recognition. In: Proceedings of ICASSP, pp. 6965–6969 (2013)
15. Shinohara, Y.: Adversarial multi-task learning of deep neural networks for robust speech recognition. In: Proceedings of Interspeech, pp. 2369–2372 (2016)
16. Sun, S., Yeh, C., Hwang, M., Ostendorf, M., Xie, L.: Domain-adversarial training for accented speech recognition. In: Proceedings of ICASSP, pp. 4854–4858 (2018)
17. Tóth, L.: Phone recognition with hierarchical convolutional deep maxout networks. EURASIP J. Audio Speech Music Process. 25 (2015)
18. Tóth, L., Grósz, T., Markó, A., Csapó, T.: Multi-task learning of speech recognition and speech synthesis parameters for ultrasound-based silent speech interfaces. In: Proceedings of Interspeech, pp. 3172–3176 (2018)
19. Wang, W., Lu, P., Yan, Y.: An improved hierarchical speaker clustering. Acta Acustica **33**(1), 9–14 (2008)
20. Zagoruyko, S., Komodakis, N.: Learning to compare image patches via convolutional neural networks. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2015