Multi-task Recurrent Model for Speech and Speaker Recognition

Zhiyuan Tang[1,2], *Lantian Li*[1], *Dong Wang*[1*]

[1]Center for Speech and Language Technologies (CSLT), Tsinghua University
[2]Chengdu Institute of Computer Applications, Chinese Academy of Sciences

{tangzy,lilt}@cslt.riit.tsinghua.edu.cn

[*]Corresponding author:wangdong99@mails.tsinghua.edu.cn

Abstract

Although highly correlated, speech and speaker recognition have been regarded as two independent tasks and studied by two communities. This is certainly not the way that people behave: we decipher both speech content and speaker traits at the same time.

This paper presents a unified model to perform speech and speaker recognition simultaneously and altogether. The model is based on a unified neural network where the output of one task is fed to the input of the other, leading to a multi-task recurrent network. Experiments show that the joint model outperforms the task-specific models on both the two tasks.

Index Terms: multi-task learning, recurrent neural network, speech recognition, speaker recognition

1. Introduction

Speech recognition (ASR) and speaker recognition (SRE) are two important research areas in speech processing. Traditionally, these two tasks are treated independently and studied by two independent communities, although some researchers indeed work on both areas. Unfortunately, this is not the way that human processes speech signals: we always decipher speech content and other meta information together and simultaneously, including languages, speaker characteristics, emotions, etc. This 'multi-task decoding' is based on two foundations: (1) all these human capabilities share the same signal processing pipeline in our aural system, and (2) they are mutually beneficial as the success of one task promotes others' in real life. Therefore, we believe that multiple tasks in speech processing should be performed by a unified artificial intelligence system. This paper focuses on speech and speaker recognition, and demonstrates that these two tasks can be solved by a single unified model.

In fact, the relevance of speech and speaker recognition has been recognized by researchers for a long time. On one hand, these two tasks share many common techniques, from the MFCC feature extraction to the HMM modeling; and on the other hand, researchers in both areas have been used to learning from each other. For instance, the success of deep neural networks (DNNs) in speech recognition [1, 2] has motivated the neural model in speaker recognition [3, 4]. Additionally, researchers also know for a long time that employing the knowledge provided by one area often helps improve the other. For instance, i-vectors produced by speaker recognition have been used to improve speech recognition [5], and phone posteriors derived from speech recognition have been utilized to improve speaker recognition [6, 7]. Moreover, the combination of these two systems has already gained attention. For instance, speech and speaker joint inference was proposed in [8], and an LSTM-based multi-task model was proposed in [9]. Although highly interesting, all the above research can not be considered as multi-task learning, and the speech and speaker recognition systems are designed, trained and executed independently.

The development of deep learning techniques in speech processing provides new hope for multi-task learning. Since 2011, deep recurrent neural networks (RNNs) have become the new state-of-the-art architectures in speech recognition [10, 11], and recently, the same architecture has gained much success in speaker recognition, at least in text-dependent conditions [12]. In both the two tasks, deep learning delivers two main advantages: first, the structural depth (multiple layers) extracts task-oriented features, and second, the temporal depth (recurrent connections) accumulates dynamic evidence. Due to the similarity in the model structure, a simple question rises that can we use a single model to perform the two tasks together?

Indeed, this 'multi-task learning' has been known working well to boost correlated tasks [13]. For example, in multilingual speech recognition, it has been known that sharing low-level layers of DNNs can improve performance on each language [14]. And in another experiment, phone and grapheme recognition were treated as two correlated tasks [15]. The central idea of multi-task learning in the deep learning era is that correlated tasks can share the same feature extraction, and so the low-level layers of DNNs for these tasks can be shared. However, this feature-sharing architecture does not apply to speech and speaker recognition. This is because these two tasks are actually 'negatively correlated': speech recognition requires features involving as much as content information, with speaker variance removed; while speaker recognition requires features involving as much as speaker information, with linguistic content removed. For these tasks, feature sharing is certainly not applicable. Unfortunately, many tasks are negatively correlated, e.g., language identification and speaker recognition, emotion recognition and speech recognition. Finding a multi-task learning approach that can deal with negatively-correlated tasks is therefore highly desirable.

This paper presents a novel recurrent architecture that can be used to learn negatively-correlated tasks simultaneously. The basic idea is to use the output of one task as part of the input of others. It would be ideal if the output of one task can provide information for others immediately, but this is not feasible in implementation. Therefore the output of one task at the previous time step is used to provide information for others at the current time step. This leads to an inter-task recurrent structure that is similar to conventional RNNs, though the recurrent connections link different tasks. We employed this multi-task recurrent learning to speech and speaker recognition and ob-

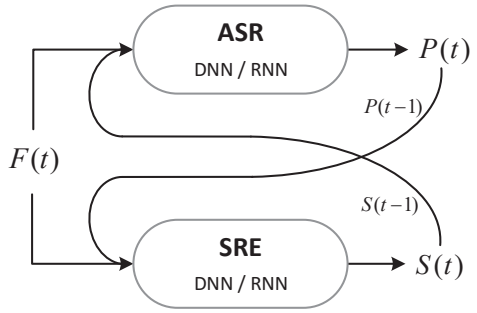

Figure 1: *Multi-task recurrent learning for ASR and SRE. $F(t)$ denotes primary features (e.g., Fbanks), $P(t)$ denotes phone identities (e.g., phone posteriors, high-level representations for phones), $S(t)$ denotes speaker identities (e.g., speaker posteriors, high-level representations for speakers).*

served promising results. The idea is illustrated in Figure 1. We note that a similar multi-task architecture was recently proposed in [9]. The difference is that they focus on speaker adaptation for ASR, while we demonstrated improvement on both ASR and SRE tasks with the joint learning.

The rest of the paper is organized as follows: Section 2 presents the model architecture, and Section 3 reports the experiments. The conclusions plus the future work are presented in Section 4.

2. Models

2.1. Basic single-task model

We start from the single-task models for ASR and SRE. As mentioned, the state-of-the-art architecture for ASR is the recurrent neural network, especially the long short-term memory (LSTM) [10]. This model also delivers good performance in SRE [12]. We therefore choose LSTM to build the baseline single-task systems. Particularly, the modified LSTM structure proposed in [11] is used. The network structure is shown in Figure 2.

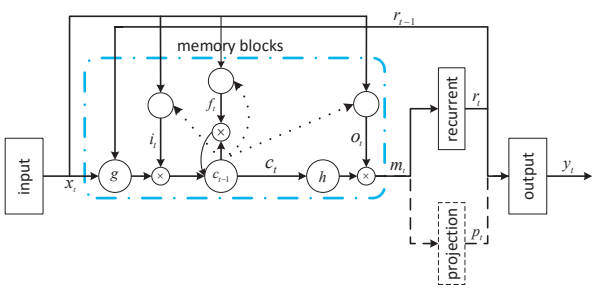

Figure 2: *Basic recurrent LSTM model for ASR and SRE single-task baselines. The picture is reproduced from [11].*

The associated computation is as follows:

$$
\begin{aligned}
i_t &= \sigma(W_{ix}x_t + W_{ir}r_{t-1} + W_{ic}c_{t-1} + b_i) \\
f_t &= \sigma(W_{fx}x_t + W_{fr}r_{t-1} + W_{fc}c_{t-1} + b_f) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g(W_{cx}x_t + W_{cr}r_{t-1} + b_c) \\
o_t &= \sigma(W_{ox}x_t + W_{or}r_{t-1} + W_{oc}c_t + b_o) \\
m_t &= o_t \odot h(c_t) \\
r_t &= W_{rm}m_t \\
p_t &= W_{pm}m_t \\
y_t &= W_{yr}r_t + W_{yp}p_t + b_y
\end{aligned}
$$

In the above equations, the W terms denote weight matrices and those associated with cells were set to be diagonal in our implementation. The b terms denote bias vectors. x_t and y_t are the input and output symbols respectively; i_t, f_t, o_t represent respectively the input, forget and output gates; c_t is the cell and m_t is the cell output. r_t and p_t are two output components derived from m_t, where r_t is recurrent and fed to the next time step, while p_t is not recurrent and contributes to the present output only. $\sigma(\cdot)$ is the logistic sigmoid function, and $g(\cdot)$ and $h(\cdot)$ are non-linear activation functions, often chosen to be hyperbolic. $\odot$ denotes the element-wise multiplication.

2.2. Multi-task recurrent model

The basic idea of the multi-task recurrent model, as shown in Figure 1, is to use the output of one task at the current frame as an auxiliary information to supervise other tasks when processing the next frame. When this idea is materialized as a computational model, there are many alternatives that need to be carefully investigated. In this study, we use the recurrent LSTM model shown in the previous section to build the ASR component and the SRE component, as shown in Figure 3. These two components are identical in structure and accept the same input signal. The only difference is that they are trained with different targets, one for phone discrimination and the other for speaker discrimination. Most importantly, there are some inter-task recurrent links that combine the two components as a single network, as shown by the dash lines in Figure 3.

Besides the model structure, a bunch of design options need to be chosen. The first question is where the recurrent information should be extracted. For example, it can be extracted from the cell c_t or cell output m_t, or from the output component r_t or p_t, or even from the output y_t. Another question is which computation block will receive the recurrent information. It can be simply the input variable x_t, but can also be the input gate i_t, the output gate o_t, the forget gate f_t or the non-linear function $g(\cdot)$. Actually, augmenting the recurrent information to x_t is equal to feed the information to i_t, o_t, f_t and $g(\cdot)$ simultaneously. Note that a weight matrix is introduced as an extra free parameter for each recurrent information feedback. Moreover, the component that the information is extracted from is not necessarily the same for different tasks, nor is the component that receives the information. However in this study, we simply consider the symmetric structure.

With all the above alternatives, the multi-task recurrent model is rather flexible. The structure shown in Figure 3 is just one simple example, where the recurrent information is extracted from both the recurrent projection r_t and the nonrecurrent projection p_t, and the information is applied to the non-linear function $g(\cdot)$. We use the superscript a and s to denote the ASR and SRE tasks respectively. The computation for ASR can be expressed as follows:

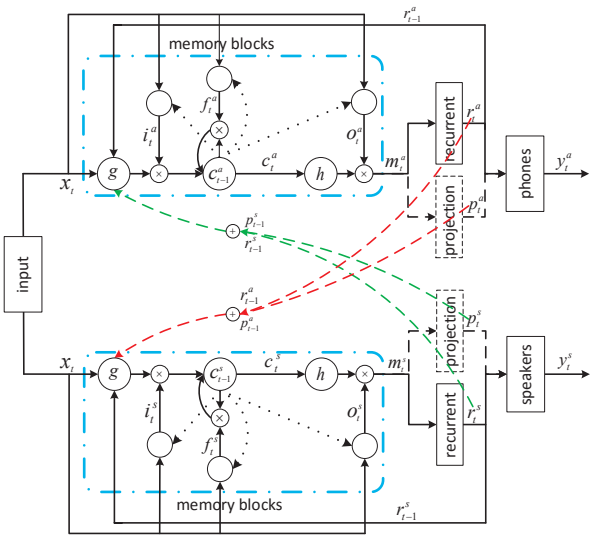

Figure 3: *Multi-task recurrent model for ASR and SRE, an example.*

$$
\begin{aligned}
i_t^a &= \sigma(W_{ix}^a x_t + W_{ir}^a r_{t-1}^a + W_{ic}^a c_{t-1}^a + b_i^a) \\
f_t^a &= \sigma(W_{fx}^a x_t + W_{fr}^a r_{t-1}^a + W_{fc}^a c_{t-1}^a + b_f^a) \\
g_t^a &= g(W_{cx}^a x_t^a + W_{cr}^a r_{t-1}^a + b_c^a + \underline{W_{cr}^{as} r_{t-1}^s + W_{cp}^{as} p_{t-1}^s}) \\
c_t^a &= f_t^a \odot c_{t-1}^a + i_t^a \odot g_t^a \\
o_t^a &= \sigma(W_{ox}^a x_t^a + W_{or}^a r_{t-1}^a + W_{oc}^a c_t^a + b_o^a) \\
m_t^a &= o_t^a \odot h(c_t^a) \\
r_t^a &= W_{rm}^a m_t^a \\
p_t^a &= W_{pm}^a m_t^a \\
y_t^a &= W_{yr}^a r_t^a + W_{yp}^a p_t^a + b_y^a
\end{aligned}
$$

and the computation for SRE is as follows:

$$
\begin{aligned}
i_t^s &= \sigma(W_{ix}^s x_t + W_{ir}^s r_{t-1}^s + W_{ic}^s c_{t-1}^s + b_i^s) \\
f_t^s &= \sigma(W_{fx}^s x_t + W_{fr}^s r_{t-1}^s + W_{fc}^s c_{t-1}^s + b_f^s) \\
g_t^s &= g(W_{cx}^s x_t^s + W_{cr}^s r_{t-1}^s + b_c^s + \underline{W_{cr}^{sa} r_{t-1}^a + W_{cp}^{sa} p_{t-1}^a}) \\
c_t^s &= f_t^s \odot c_{t-1}^s + i_t^s \odot g_t^s \\
o_t^s &= \sigma(W_{ox}^s x_t^s + W_{or}^s r_{t-1}^s + W_{oc}^s c_t^s + b_o^s) \\
m_t^s &= o_t^s \odot h(c_t^s) \\
r_t^s &= W_{rm}^s m_t^s \\
p_t^s &= W_{pm}^s m_t^s \\
y_t^s &= W_{yr}^s r_t^s + W_{yp}^s p_t^s + b_y^s
\end{aligned}
$$

3. Experiments

The proposed method was tested with the WSJ database, which has been labelled with both word transcripts and speaker identities. We first present the ASR and SRE baselines and then report the multi-task model. All the experiments were conducted with the Kaldi toolkit [16].

3.1. Data

- Training set: This set involves 90% of the speech data randomly selected from train_si284 (the other 10% used for speaker identification test whose results for almost all systems were perfect thus not presented). It consists of 282 speakers and 33,587 utterances, with 40-144 utterances per speaker. This set was used to train the two LSTM-based single-task systems, an i-vector SRE baseline, and the proposed multi-task system.

- Test set: This set involves three datasets (dev93, eval92 and eval93). It consists of 27 speakers and 1,049 utterances. This dataset was used to evaluate the performance of both ASR and SRE. For SRE, the evaluation consists of 21,350 target trials and 528,326 non-target trials, constructed based on the test set.

3.2. ASR baseline

The ASR system was built largely following the Kaldi WSJ s5 nnet3 recipe, except that we used a single LSTM layer for simplicity. The dimension of the cell was 1,024, and the dimensions of the recurrent and nonrecurrent projections were set to 256. The target delay was 5 frames. The natural stochastic gradient descent (NSGD) algorithm was employed to train the model [17]. The input feature was the 40-dimensional Fbanks, with a symmetric 2-frame window to splice neighboring frames. The output layer consisted of 3,419 units, equal to the total number of pdfs in the conventional GMM system that was trained to bootstrap the LSTM model. The baseline performance is reported in Table 1.

Table 1: *ASR baseline results.*

	dev92	eval92	eval93	Total
WER%	8.36	5.14	8.06	7.41

3.3. SRE baseline

We built two SRE baseline systems: one is an i-vector system and the other is an 'r-vector' system that is based on the recurrent LSTM model.

For the i-vector system, the acoustic feature was 39-dimensional MFCCs. The number of Gaussian components of the UBM was 1,024, and the dimension of i-vectors was 200. For the r-vector system, the architecture was similar to the one used by the LSTM-based ASR baseline, except that the dimension of the cell was 512, and the dimensions of the recurrent and nonrecurrent projections were set to 128. Additionally, there was no target delay. The input of the r-vector system was the same as ASR system, and the output was corresponding to the 282 speakers in the training set. Similar to the work in [3, 4], the speaker vector ('r-vector') was derived from the output of the recurrent and nonrecurrent projections, by averaging the output of all the frames. The dimension was 256.

The baseline performance is reported in Table 2. It can be observed that the i-vector system generally outperforms the r-vector system. Particularly, the discriminative methods (LDA and PLDA) offer much more significant improvement for the i-vector system than for the r-vector system. This observation is consistent with the results reported in [4], and can be attributed to the fact that the r-vector model has already been learned 'discriminatively' with the LSTM structure. For this reason, we only consider the simple cosine kernel when scoring r-vectors in the following experiments.

Table 2: *SRE baseline results.*

	EER%		
System	Cosine	LDA	PLDA
i-vector (200)	2.89	1.03	0.57
r-vector (256)	1.84	1.34	3.18

3.4. Multi-task joint training

Due to the flexibility of the multi-task recurrent LSTM structure, it is not possible to evaluate all the configurations. We chose some typical ones and report the results in Table 3. We just show the ASR results on the combined dataset mentioned before. Note that the last configure, where the recurrent information is fed to all the gates and the non-linear activation $g(\cdot)$, is equal to augmenting the information to the input variable x.

Table 3: *Joint training results.*

Feedback Info.		Feedback Input				ASR WER%	SRE EER%
r	p	i	f	o	g		
						7.41	1.84
√		√				7.05	0.62
√	√	√				6.97	0.64
√			√			7.12	0.66
√	√		√			7.24	0.65
√				√		7.26	0.65
√	√			√		7.28	0.59
√					√	7.11	0.62
√	√				√	7.11	0.67
√		√	√	√		7.06	0.66
√	√	√	√	√		7.23	0.71
√		√	√	√	√	7.05	0.55
√	√	√	√	√	√	7.23	0.62

From the results shown in Table 3, we first observe that the multi-task recurrent model consistently improves performance on both ASR and SRE, no matter where the recurrent information is extracted and where it applies. Most interestingly, on the SRE task, the multi-task system can obtain equal or even better performance than the i-vector/PLDA system. This is the first time that the two negatively-correlated tasks are learned jointly in a unified framework and boost each other.

For the recurrent information, it looks like the recurrent projection r_t is sufficient to provide valuable supervision for the partner task. Involving more information from the nonrecurrent projection does not offer consistent benefit. This observation, however, is only based on the present experiments. With more data, it is likely that more information leads to additional gains.

For the recurrent information 'receiver', i.e., the component that receives the recurrent information, it seems that for ASR the input gate and the activation function are equally effective, while the output gate seems not so appropriate. For SRE, all results seem good. Again, these observations are just based on a relative small database; with more data, the performance with different configurations may become distinguishable.

4. Conclusions

We report a novel multi-task recurrent learning architecture that can jointly train multiple negatively-correlated tasks. Primary results on the WSJ database demonstrated that the presented method can learn speech and speaker models simultaneously and improve the performance on both tasks. Future work involves analyzing more factors such as target delay, exploiting partially labelled data, and applying the approach to other negatively-correlated tasks.

5. References

[1] G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 1, pp. 30–42, 2012.

[2] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[3] E. Variani, X. Lei, E. McDermott, I. Lopez Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4052–4056.

[4] L. Li, Y. Lin, Z. Zhang, and D. Wang, "Improved deep speaker feature learning for text-dependent speaker recognition," in *Proceedings of APSIPA Annual Summit and Conference*. APSIPA, 2015.

[5] A. W. Senior and I. Lopez-Moreno, "Improving dnn speaker independence with i-vector inputs." in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 225–229.

[6] Y. Lei, L. Ferrer, M. McLaren *et al.*, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 1695–1699.

[7] P. Kenny, V. Gupta, T. Stafylakis, P. Ouellet, and J. Alam, "Deep neural networks for extracting Baum-Welch statistics for speaker recognition," in *Proceedings of Odyssey*, 2014, pp. 293–298.

[8] M. F. BenZeghiba and H. Bourlard, "On the combination of speech and speaker recognition," in *European Conference On Speech, Communication and Technology (EUROSPEECH)*, no. EPFL-CONF-82941, 2003, pp. 1361–1364.

[9] X. Li and X. Wu, "Modeling speaker variability using long short-term memory networks for speech recognition," in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2015.

[10] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1764–1772.

[11] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2014.

[12] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," *arXiv preprint arXiv:1509.08062*, 2015.

[13] D. Wang and T. F. Zheng, "Transfer learning for speech and language processing," in *Proceedings of APSIPA Annual Summit and Conference*. APSIPA, 2015.

[14] J.-T. Huang, J. Li, D. Yu, L. Deng, and Y. Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2013, pp. 7304–7308.

[15] N. Chen, Y. Qian, and K. Yu, "Multi-task learning for text-dependent speaker verification," in *Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH)*, 2015.

[16] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[17] D. Povey, X. Zhang, and S. Khudanpur, "Parallel training of deep neural networks with natural gradient and parameter averaging," *arXiv preprint arXiv:1410.7455*, 2014.