ISSN 1346-5597

National Institute of Informatics

NII Technical Report

A simple RNN-plus-highway network for statistical parametric speech synthesis

Xin Wang, Shinji Takaki, Junichi Yamagishi

NII-2017-003E
Apr. 2017

URL: http://www.nii.ac.jp/ Tokyo 101-8430, JAPAN +81-3-4212-2000 FAX: +81-3-3556-1916

A simple RNN-plus-highway network for statistical parametric speech synthesis

Xin Wang, Shinji Takaki, Junichi Yamagishi

Abstract

In this report, we proposes a neural network structure that combines a recurrent neural network (RNN) and a deep highway network. Compared with the highway RNN structures proposed in other studies, the one proposed in this study is simpler since it only concatenates a highway network after a pre-trained RNN. The main idea is to use the 'iterative unrolled estimation' of a highway network to finely change the output from the RNN. The experiments on the proposed network structure with a baseline RNN and 7 highway blocks demonstrated that this network performed relatively better than a deep RNN network with a similar mode size. Furthermore, it took less than half the training time of the deep RNN.

1 Introduction

Statistical parametric speech synthesis is widely used in text-to-speech (TTS) synthesis. We assume a TTS system using a pipeline structure in which a front-end derives the information on the pronunciation and prosody of the input text, then the back-end generates the acoustic features based on the output of the front-end module. While various approaches can be used for the back-end acoustic modeling, we focus on the neural-network (NN)-based acoustic models.

Suppose the output from the front-end is a sequence of textual feature vectors $\boldsymbol{x}_{1:T} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T]$ in T frames, where each $\boldsymbol{x}_t$ may include the phoneme identity, pitch accent, and other binary or continuous-valued linguistic features. The task of the NN-based acoustic model is to convert $\boldsymbol{x}_{1:T}$ into a sequence of acoustic features $\boldsymbol{o}_{1:T} = [\boldsymbol{o}_1, \cdots, \boldsymbol{o}_T]$. The vector $\boldsymbol{o}_t$ can encode various types of acoustic features, but this report is only focused on the commonly used low-dimensional spectral and F0 features. The task of the NN-based acoustic model is to convert $\boldsymbol{x}_{1:T}$ to $\boldsymbol{o}_{1:T}$. Various types of networks, including conventional feedforward and recurrent neural networks (RNN), can be used for this task.

Neural networks with a sufficient number of hidden layers are assumed to be powerful tools for regression tasks [Ben09]. For acoustic modeling, recent studies have explored feedforward networks with up to 7 hidden layers [ZSS13, ZS14]. Our recent study based on highway networks investigated deeper networks with

up to 40 hidden layers [1]. However, these networks contain no recurrent layers.

This study proposed a structure that combines the highway network and RNN. Although there have been related studies on this topic [ZSKS16, ZCY+16], the structures of RNN highway networks in those studies are very complex. Based on a recent theoretical analysis on the highway network [GSS16], which argues that each highway block inside the network learns through iterative estimation, we believe a simple structure can be used to combine a highway network and RNN without significantly increasing the computational complexity. The idea is just to simply attach a normal highway network after the pre-trained RNN. The experiments demonstrated that this simple combination performed relatively better than a deep RNN with a similar number of model parameters. Furthermore, the training time of the combined network was less than half that for the deep RNN.

2 Methods

2.1 Highway block

A highway network consists of one or multiple highway blocks. Suppose an input linguistic feature vector $\boldsymbol{x}_t$ is fed as the input to one highway block. This block first processes $\boldsymbol{x}_t$ as a conventional feedforward layer

$$\mathcal{H}(\boldsymbol{x}_t) = f(\boldsymbol{W}_H \boldsymbol{x}_t + \boldsymbol{b}_H). \tag{1}$$

Here, $f(\cdot)$ is the non-linear activation function, $\boldsymbol{b}_H$ is the bias vector, and $\boldsymbol{W}_H$ is the transformation matrix. Then, the highway block uses a *highway gate* to compute a control vector

$$\mathcal{T}(\boldsymbol{x}_t) = \sigma(\boldsymbol{W}_T \boldsymbol{x}_t + \boldsymbol{b}_T), \tag{2}$$

and merges the first feature vector $\mathcal{H}(\boldsymbol{x}_t)$ with the input $\boldsymbol{x}_t$ as the output of this highway block:

$$\boldsymbol{h}_t = \mathcal{T}(\boldsymbol{x}_t) \odot \mathcal{H}(\boldsymbol{x}_t) + [\boldsymbol{1} - \mathcal{T}(\boldsymbol{x}_t)] \odot \boldsymbol{x}_t. \tag{3}$$

Here, $\odot$ denotes element-wise multiplication, and the sigmoid function $\sigma(x) = \frac{1}{1+e^{(-x)}}$ is used in the highway gate. The output vector $\boldsymbol{h}_t$ can be further processed by any type of hidden layer to approximate the target vector $\boldsymbol{o}_t$.

Note that parameters $\boldsymbol{W}_T$ and $\boldsymbol{b}_T$ in the highway gate are also trainable. When the output of the gate $\mathcal{T}(\boldsymbol{x})$ is approximately zero, the input $\boldsymbol{x}$ can be directly propagated forwards, i.e., $\boldsymbol{y} \approx \boldsymbol{x}$. In this case, the gradient can also be propagated backwards without being attenuated by the feedforward transformation layer in the highway block. Thus, a very deep network based on highway blocks can be trained using the standard gradient-descent back-propagation algorithm. Note that $\mathcal{H}(\boldsymbol{x})$ can be a transformation conducted by multiple feedforward layers. In other words, one highway block can contain more than one feedforward transformation layer.

[1] Please find the manuscript on http://tonywangx.github.io

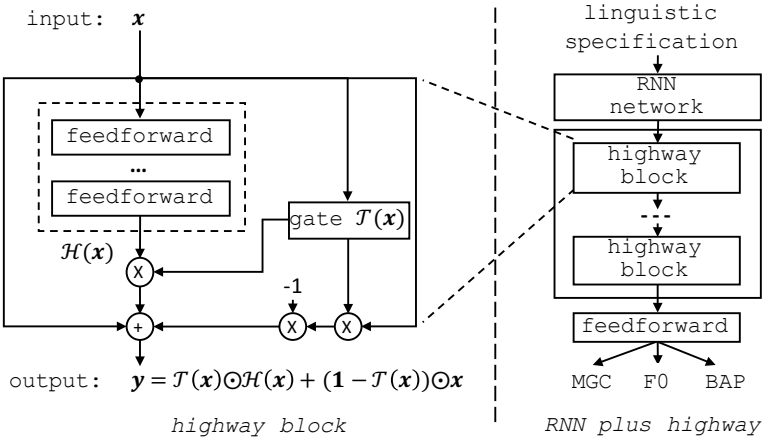

Figure 1: Structure of the RNN-plus-highway network.

2.2 Recurrent layer

Different from the feedforward layer and highway block, an RNN layer is used to transform the input $\boldsymbol{x}_t$ and previously extracted hidden state $\boldsymbol{h}_{t-1}$ into a new vector $\boldsymbol{h}_t$:

$$\boldsymbol{h}_t = f(\boldsymbol{W}_I \boldsymbol{x}_t + \boldsymbol{W}_{HH} \boldsymbol{h}_{t-1} + \boldsymbol{b}_{HH}). \tag{4}$$

The generated feature vectors $\{\boldsymbol{h}_1, \cdots, \boldsymbol{h}_t, \cdots\}$ can be further transformed using another RNN or feedforward layer. At the output layer, the hidden feature vectors are transformed into the target acoustic feature vectors.

The network parameters, including $\boldsymbol{W}_I, \boldsymbol{W}_{HH}$, and $\boldsymbol{b}_{HH}$, are learned from the training data by using the back-propagation algorithm. However, the vanilla RNN shown above is difficult to train because of the gradient vanishing and exploding problem. As a solution, the long short term memory (LSTM) unit, in which trainable gates control the input, output, and state of the memory cell [Gra08], has been proposed to replace the simple function $f(.)$.

2.3 Combination of highway network and RNN

The highway block targets the gradient-vanishing problem across layers while an RNN with LSTM units focuses on the gradient propagation across time. It seems natural to combine these two so that the gradient can be well propagated both across layers and time. Some studies have taken into account this idea by designing more complex network structures that facilitate the gradient flow across layers and time [ZSKS16, ZCY+16].

However, a recent study provided a different view on the highway network [GSS16]. The basic idea is that, a highway block does not extract completely different structural hidden representations. Instead, it finely changes the output from the previous layer. Based on this explanation, we think a simple way to combine RNN and highway network is to attach highway blocks after a normal RNN. The role played by an RNN is as a rough feature transformer. Then, the

highway blocks finely tune the output from the RNN. This combined structure is shown in Figure 1.

Other structures, such as putting the highway network before RNN or adding highway networks to both sides of the RNN, suffer from the gradient-vanishing problem since the gradients must pass through the RNN part before arriving at the highway network on the network's input side. Our experiments failed to train these structures with acceptable performance.

3 Experiments

3.1 Corpus and i/o features

In the experiments, the Blizzard Challenge 2011 Nancy corpus that has 12072 English utterances [KK11] was used. Both the test and validation set contained 500 randomly selected utterances. Mel-generalized cepstral coefficients (MGCs) of order 60, continuous F0 trajectory, voiced/unvoiced (V/U) condition, and band aperiodicity (BAP) of order 25 were extracted for each speech frame by using the STRAIGHT vocoder [KMKC99]. The Flite toolkit [HTS14] was used to conduct the text-analysis for the entire corpus. The outputs of Flite were converted into a vector of order 382 as the input to the neural network ($\boldsymbol{x}_t$). This vector encodes common textual features similar to those used in the HMM-based framework [TZB02].

The experiments were conducted on the three types of network listed in Table 1. The toolkit for training the networks was modified on the basis of the CURRENNT library [WBS15].

3.2 Networks and configurations

Similar to the configuration in [FQXS14], the network `R-B` used 2 feedforward layers with 512 nodes, 2 bi-directional LSTM layers with 256 nodes, and a linear output layer. For deeper networks, `R-FF` was created by attaching 21 tanh-based single-stream feedforward layers after the last LSTM layer of `R-B`, where each attached layer had 256 hidden nodes; `R-FF` was constructed in a similar way by attaching 7 single-stream highway blocks with a layer size of 256. Each of the highway block contained 2 tanh-based feedforward layers. The `R-FF` and `R-HS` had the number of model parameters (listed in Table 2). For reference, a deep RNN `R-D` was included. This network was similar to `R-B` but included 3 additional bi-directional LSTM layers. The number of parameters in `R-D` was roughly the same as that in `R-HS` and `R-FF`.

The `R-B` was trained first using stochastic gradient descent with early stopping. The other deep networks were first initialized for the 4 layers near the input side by using the weights of `R-B`. The rest of the parameters were then initialized using the layer-size-dependent uniform distribution [GB10]. The training method for these deep networks was the same as that for `R-B`. [2]

[2]The toolkit modified based on CURRENNT [WBS15], implementation details of `AR-RMDN`,

Table 1: Experimental networks.

	Definition
R-B	Baseline RNN
R-D	Baseline RNN + 3 RNN layers
R-FF	Baseline RNN + 21 tanh feedforward layers
R-HS	Baseline RNN + 7 highway blocks (2 tanh layers in each block)

Table 2: Objective evaluation results. #.Para denotes the number of model parameters and the last column is the average training time per epoch (hour). Note that the Nvidia K80 with CUDA8.0 was used to train the networks.

	F0 RMSE (Hz)	F0 CORR	F0 U/V	MGC RMSE	#.Para	Ave.T per epoch
R-B	40.0	0.766	4.75%	1.004	1577475	0.88h
R-D	39.5	0.776	4.65%	0.998	3025155	2.50h
R-FF	40.9	0.756	4.67%	1.004	2959107	1.00h
R-HS	38.9	0.783	4.55%	0.978	2959107	1.00h

3.3 Results and analysis

The gross errors over all types of acoustic features for each training epoch are plotted in Figure 2. The objective measures were calculated and are shown in Table 2. Note that the test utterances were synthesized given the natural alignment. The MLPG algorithm [KT$^+$00] was not used.

Figure 2 shows that the three deep networks performed better than the baseline on the validation set. The R-HS achieved the lowest error curve. Furthermore, the training curve of R-HS converged first. Compared with the curve of R-HS, R-FF's curve showed a similar shape but was higher than that of R-HS. The deep RNN R-D was the slowest to converge. Although the final gross error of R-D was close to that of R-HS, the total training time for R-D was much larger (see average training time per epoch for each network in Table 2).

The objective measure on the test set demonstrated that R-HS achieved the best performance. The R-D performed similarly to R-HS. Although R-FF had a better learning curve on the validation set, its objective performance was worse than R-B. Sample trajectories of the generated MGC and F0 features are plotted in Figure 3. The synthesized speech samples can be found at http://tonywangx.github.io. Please search RNNHighway on the webpage and then download the package of sample waveforms from the link.

and speech samples can be found at http://tonywangx.github.io

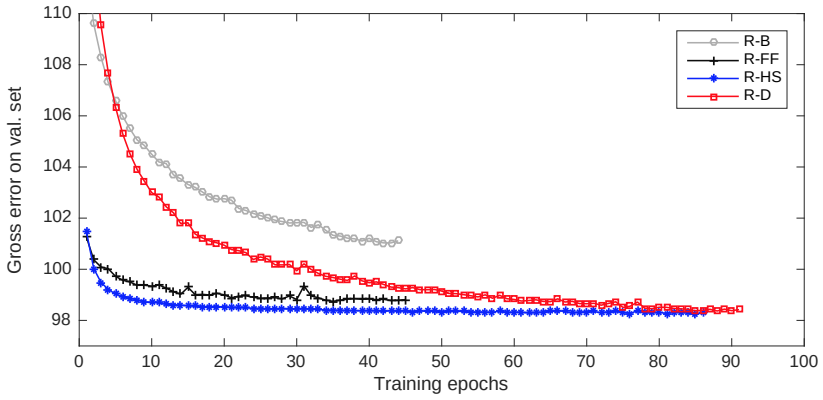

Figure 2: Gross error of networks on the validation set.

4 Conclusion

A simple network structure was proposed to combine the highway network and RNN. Specifically, the highway network was concatenated after the pre-trained RNN. Then, the whole network can be trained using the standard back-propagation algorithm. The experiments demonstrated that this structure exhibited better objective performance than a deep RNN with roughly the same model size. Furthermore, the training time for the proposed RNN-plus-highway network was less than half that for training the deep RNN.

References

[Ben09] Yoshua Bengio. Learning deep architectures for AI. *Found. Trends Mach. Learn.*, 2(1):1–127, January 2009.

[FQXS14] Yuchen Fan, Yap Qian, Feilong Xie, and Frank K. Soong. TTS synthesis with bidirectional LSTM based recurrent neural networks. In *Proc. Interspeech*, pages 1964–1968, 2014.

[GB10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proc. AISTATS*, pages 249–256, 2010.

[Gra08] Alex Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*. PhD thesis, Technische Universität München, 2008.

[GSS16] Klaus Greff, Rupesh K Srivastava, and Jürgen Schmidhuber. Highway and residual networks learn unrolled iterative estimation. *arXiv preprint arXiv:1612.07771*, 2016.

[HTS14] HTS Working Group. The English TTS system Flite+HTS_engine, 2014.

[KK11] Simon King and Vasilis Karaiskos. The Blizzard Challenge 2011. In *Proc. Blizzard Challenge Workshop*, pages 1–10, 2011.

[KMKC99] Hideki Kawahara, Ikuyo Masuda-Katsuse, and Alain de Cheveigne. Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds. *Speech Communication*, 27:187–207, 1999.

[KT+00] Tokuda Keiichi, , Yoshimura Takayoshi, Masuko Takashi, Kobayashi Takao, and Kitamura Tadashi. Speech parameter generation algorithms for HMM-based speech synthesis. In *Proc. ICASSP*, pages 936–939, 2000.

[TZB02] Keiichi Tokuda, Heiga Zen, and Alan W. Black. An HMM-based speech synthesis system applied to english. In *Proc. SSW*, pages 227–230, Sept 2002.

[WBS15] Felix Weninger, Johannes Bergmann, and Björn Schuller. Introducing CURRENT: The Munich open-source CUDA recurrent neural network toolkit. *The Journal of Machine Learning Research*, 16(1):547–551, 2015.

[ZCY+16] Yu Zhang, Guoguo Chen, Dong Yu, Kaisheng Yaco, Sanjeev Khudanpur, and James Glass. Highway long short-term memory rnns for distant speech recognition. In *Proc. ICASSP*, pages 5755–5759, 2016.

[ZS14] Heiga Zen and Andrew Senior. Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis. In *Proc. ICASSP*, pages 3844–3848, 2014.

[ZSKS16] Julian Georg Zilly, Rupesh Kumar Srivastava, Jan Koutník, and Jürgen Schmidhuber. Recurrent highway networks. *arXiv preprint arXiv:1607.03474*, 2016.

[ZSS13] Heiga Zen, Alan Senior, and Martin Schuster. Statistical parametric speech synthesis using deep neural networks. In *Proc. ICASSP*, pages 7962–7966, 2013.

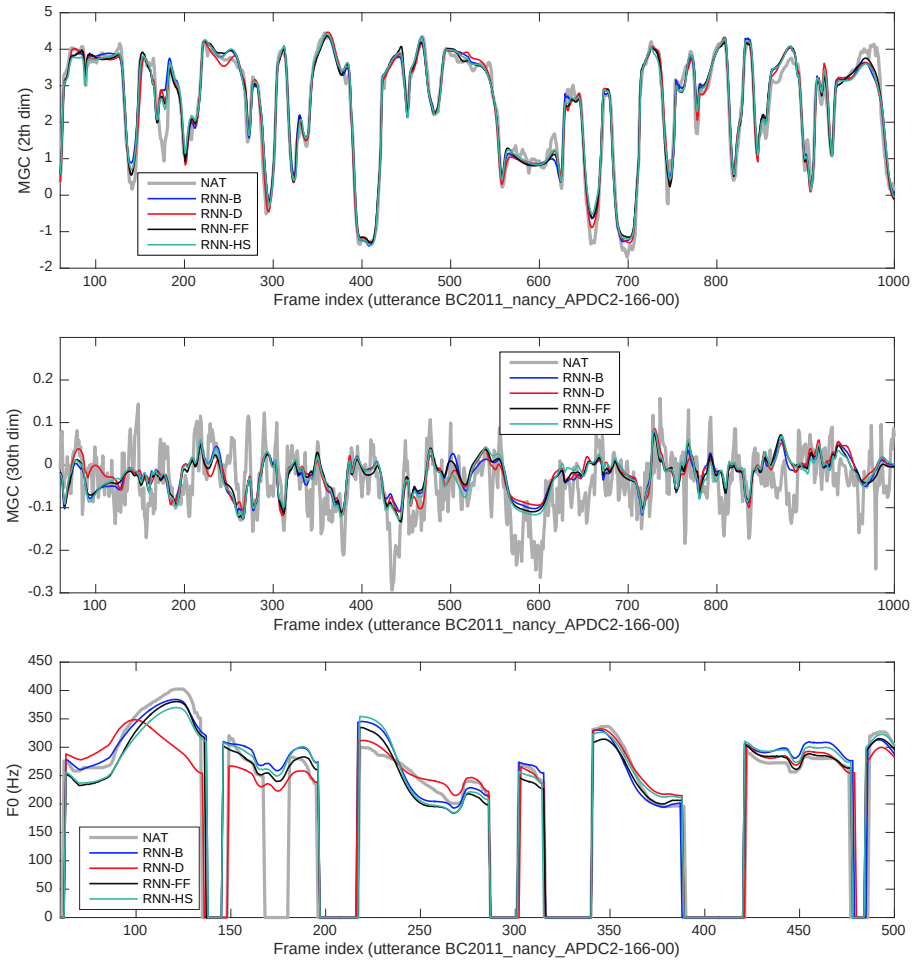

Figure 3: Predicted feature trajectories of MGC (2nd and 30th order) and F0 of utterance BC2011_nancy_APDC2-166-00. Frame shift is 5ms.