Interspeech 2018
2-6 September 2018, Hyderabad

Semi-tied Units for Efficient Gating in LSTM and Highway Networks

C. Zhang & P. C. Woodland

Cambridge University Engineering Dept., Trumpington St., Cambridge, CB2 1PZ U.K.
{cz277,pcw}@eng.cam.ac.uk

Abstract

Gating is a key technique used for integrating information from multiple sources by long short-term memory (LSTM) models and has recently also been applied to other models such as the highway network. Although gating is powerful, it is rather expensive in terms of both computation and storage as each gating unit uses a separate full weight matrix. This issue can be severe since several gates can be used together in e.g. an LSTM cell. This paper proposes a semi-tied unit (STU) approach to solve this efficiency issue, which uses one shared weight matrix to replace those in all the units in the same layer. The approach is termed "semi-tied" since extra parameters are used to separately scale each of the shared output values. These extra scaling factors are associated with the network activation functions and result in the use of parameterised sigmoid, hyperbolic tangent, and rectified linear unit functions. Speech recognition experiments using British English multi-genre broadcast data showed that using STUs can reduce the calculation and storage cost by a factor of three for highway networks and four for LSTMs, while giving similar word error rates to the original models.
Index Terms: LSTM, highway network, gating, parameterised activation function, speech recognition

1. Introduction

Gating units have become a key component in many types of artificial neural network (ANN) models. These units yield soft 0-1 valued outputs that are used to scale signals from other parts of the network. In recurrent neural networks (RNNs) a key issue is solving the vanishing gradient problem [1] in training which causes standard RNNs to find it difficult to learn long-term information. Gated RNNs such as the long short-term memory (LSTM) model [2] define explicit memory cells where updates to the memory cell values are controlled by two gating units and updating the hidden state value uses a further gate. The gated recurrent unit (GRU) is an RNN that uses two gates [3].

Recently, highway connections have been proposed to enable a feed-forward or a recurrent layer to have an extra nonlinearity by combining its input and output values via gating units [4–6]. The highway idea has also been applied to connect the memory cells of neighbouring LSTM layers [7]. Furthermore, gating is also useful for convolutional layers [8, 9]. A quasi-RNN uses gates to integrate different time step outputs from a layer shared across time, which can be viewed as temporal convolutional model as the shared layer serves as a time-invariant filter [9]. All models discussed above have been applied to acoustic modelling for speech recognition [10–19].

Normally a gating unit is defined as a sublayer that outputs a "gating vector" of soft 0-1 values by operating on e.g. current input values or those from previous layers, with full weight matrices. This gating vector is often applied to a "candidate vector" that would, for instance in the case of an LSTM, be used to update the memory cell values. Since the calculation of the gating vectors often has a similar functional form to that used to find the candidate vectors, the overall number of parameters and computational complexity of gated models is high [2, 4, 8]. This can be severe when models use several different gating units.

In this paper we propose an alternative type of unit for gating termed a semi-tied unit (STU), which aims at implementing a similar function to the traditional gating unit in a more efficient way. As its name suggests, the key idea in STU is to share parameters to save computation while also adding some untied parameters so that gating units can learn distinct functions. This paper studies the most commonly used gated models, LSTMs and highway networks, which have each of their units implemented based on full weight matrices. In order to reduce the number of matrix multiplications, the STUs share the weights and biases among all the gating and candidate units in the same layer. Meanwhile, additional untied parameter vectors are introduced as component-wise adaptive scaling factors through parameterised activation functions [20, 21], which allows the STUs to generate distinct gating and candidate vectors. Experimental results found using STUs in both LSTM and highway network resulted in similar WERs to those based on traditional gating units, while being significantly more efficient.

The rest of the paper is organised as follows. Section 2 reviews the gating mechanism along with LSTMs and highway networks. STUs based on parameterised activation functions are described in Section 3. The experimental setup and results are given in Sections 4 and 5, which is followed by conclusions.

2. Gating Mechanism

Analogous to an array of logic gate in electronics, an ANN gating unit converts its vector input into a 0-1 valued gating vector. For an LSTM layer at time t, the input, forget, and output gating vectors $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t$ are computed by

$$\mathbf{i}_t = \sigma(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{V}_i \circ \mathbf{c}_{t-1} + \mathbf{b}_i) \quad (1)$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{V}_f \circ \mathbf{c}_{t-1} + \mathbf{b}_f) \quad (2)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{V}_o \circ \mathbf{c}_t + \mathbf{b}_o), \quad (3)$$

where $\mathbf{x}_t$ and $\mathbf{h}_t$ are the input and hidden state values; $\mathbf{W}$ and $\mathbf{U}$ are weight matrices; $\mathbf{b}$ and $\mathbf{V}$ are the bias vector and diagonal "peephole" matrix; $\circ$ is component-wise multiplication; $[\sigma(\mathbf{a_t})]_j = 1/(1+e^{-a_{tj}})$ is the j^{th} component of the *vector sigmoid function* with input activation vector $\mathbf{a}_t$, and a_{tj} is the j^{th} component of $\mathbf{a}_t$. Given the j^{th} component of the *vector hyperbolic tangent function* as $[\tanh(\mathbf{a}_t)]_j = (e^{a_{tj}} - e^{-a_{tj}})/(e^{a_{tj}} + e^{-a_{tj}})$, the gating vectors are used to generate $\mathbf{h}_t$ based on the previous memory cell value $\mathbf{c}_{t-1}$ and the current candidate vector $\tilde{\mathbf{c}}_t$ by

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c\mathbf{x}_t + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{b}_c) \quad (4)$$

$$\mathbf{c}_t = \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tilde{\mathbf{c}}_t \quad (5)$$

$$\mathbf{h}_t = \mathbf{o}_t \circ \tanh(\mathbf{c}_t), \quad (6)$$

Thanks to Mark Gales & the MGB3 team for the MGB3 setup used.

10.21437/Interspeech.2018-2158

which manipulates the information flow to simulate the human long short-term memory mechanism, and helps solve the gradient vanishing problem [2].

The gating idea can also be used to attenuate the information loss in feedforward layers to allow the training of very deep models [5]. A *highway network* refers to a feedforward model with a stack of highway layers [4], with each of them defined as

$$\mathbf{m}_t = \sigma(\mathbf{W}_m\mathbf{x}_t + \mathbf{b}_m) \quad (7)$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r\mathbf{x}_t + \mathbf{b}_r) \quad (8)$$

$$\tilde{\mathbf{y}}_t = f(\mathbf{W}_y\mathbf{x}_t + \mathbf{b}_y) \quad (9)$$

$$\mathbf{y}_t = \mathbf{m}_t \circ \tilde{\mathbf{y}}_t + \mathbf{r}_t \circ \mathbf{x}_t, \quad (10)$$

where $\mathbf{y}_t$ is the output of the layer, $\tilde{\mathbf{y}}_t$ is the candidate vector, $\mathbf{m}_t$ and $\mathbf{r}_t$ are the gating vectors of the transform and carry gates, and $f(\cdot)$ is an activation function. For recurrent models, the feedforward highway layers can be located in-between the recurrent layers, which results in the recurrent highway network [6]. Furthermore, $\mathbf{r}_t$ can be replaced by $\mathbf{1} - \mathbf{m}_t$ to save one gating unit in each layer, and Eqn. (10) becomes

$$\mathbf{y}_t = \mathbf{m}_t \circ \tilde{\mathbf{y}}_t + (\mathbf{1} - \mathbf{m}_t) \circ \mathbf{x}_t. \quad (11)$$

This idea has also been applied to GRUs and quasi-RNNs by modifying Eqn. (6) in the same way [3, 9]. However, since Eqn. (6) is found to work better for highway networks [17], it is used throughout this paper.

3. Semi-tied Units

From Eqns. (1) – (4) and Eqns. (7) – (9), only a quarter or one third of the parameters (and calculations) are used to generate the candidate vectors in an LSTM and highway layers respectively, while the rest are associated with gating. The efficiency could be improved if there exists a shared "virtual unit" which distinguishes the gating and candidate units by cheaper operations than matrix multiplications. This is reasonable since the units have the same input and functional form. Based on this assumption, the STU is proposed that represents the "virtual unit" by parameters that are tied across all gating and candidate units, and modelling the difference between the "virtual unit" and every other unit by some extra untied parameters.

3.1. Parameterised Activation Function for STUs

In LSTMs and highway networks, since weight matrix multiplications take the most computation and storage cost, they are tied to form the "virtual unit", and the bias vectors are also tied. The type of the untied parameters is another important choice in an STU as they model the differences between the units. This paper uses additional linear factors to scale the output values from the "virtual unit" for this purpose, which is very efficient as it involves only component-wise operations. It is natural to associate such scaling factors with the activation functions that leads to the use of the parameterised activation functions proposed in [20]. The parameterised sigmoid function with additional parameter vectors $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$ is denoted as $\sigma_{\boldsymbol{\eta},\boldsymbol{\gamma}}(\cdot)$ and defined by

$$\sigma_{\boldsymbol{\eta},\boldsymbol{\gamma}}(\mathbf{a}_t) = \boldsymbol{\eta} \circ \sigma(\boldsymbol{\gamma} \circ \mathbf{a}_t),$$

where $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$ associates an independent parameter for every output node to scale its output and input values. Note that the scaling by $\boldsymbol{\eta}$ can mean that the range of the gating vector values is no longer constrained by 0-1, which can be seen as a generalisation of the original gating mechanism. In order to use STUs for LSTMs and rectified linear unit (ReLU) highway networks, the tanh and ReLU functions are also parameterised as

$$\tanh_{\boldsymbol{\eta},\boldsymbol{\gamma}}(\mathbf{a}_t) = \boldsymbol{\eta} \circ \tanh(\boldsymbol{\gamma} \circ \mathbf{a}_t)$$

$$\mathrm{ReLU}_{\boldsymbol{\eta}}(\mathbf{a}_t) = \boldsymbol{\eta} \circ \mathrm{ReLU}(\mathbf{a}_t),$$

where $[\mathrm{ReLU}(\mathbf{a}_t)]_j = \max(a_{tj}, 0)$. Here $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$ still refer to the output and input value scaling vectors for $\tanh_{\boldsymbol{\eta},\boldsymbol{\gamma}}(\cdot)$ and $\mathrm{ReLU}_{\boldsymbol{\eta}}(\cdot)$. Other types of parameterised activation functions have also been investigated for both conventional modelling [22–25] and speaker adaptation [26–29].

3.2. STUs for LSTMs and Highway Networks

3.2.1. STU based LSTMs (LSTM^{STU})

As discussed before, the weights and biases are tied across all gating and candidate units when using STUs. The shared part, or the "virtual unit", produces the common values $\mathbf{e}_t$ by

$$\mathbf{e}_t = \mathbf{W}\mathbf{x}_t + \mathbf{U}\mathbf{h}_{t-1} + \mathbf{b}.$$

Let i, f, o, and c be the subscripts of the activation function parameters for the input gate, forget gate, output gate, and the candidate unit, Eqns. (1) – (4) can be re-written as

$$\mathbf{i}_t = \sigma_{\boldsymbol{\eta}_i,\boldsymbol{\gamma}_i}(\mathbf{e}_t + \mathbf{V} \circ \mathbf{c}_{t-1})$$

$$\mathbf{f}_t = \sigma_{\boldsymbol{\eta}_f,\boldsymbol{\gamma}_f}(\mathbf{e}_t + \mathbf{V} \circ \mathbf{c}_{t-1})$$

$$\mathbf{o}_t = \sigma_{\boldsymbol{\eta}_o,\boldsymbol{\gamma}_o}(\mathbf{e}_t + \mathbf{V} \circ \mathbf{c}_t)$$

$$\tilde{\mathbf{c}}_t = \tanh_{\boldsymbol{\eta}_c,\boldsymbol{\gamma}_c}(\mathbf{e}_t).$$

Hence the weight matrices are tied to $\mathbf{W}$ and $\mathbf{U}$ respectively; the bias vectors and diagonal "peephole" matrices are tied to $\mathbf{b}$ and $\mathbf{V}$. Let X and H be the sizes of $\mathbf{x}_t$ and $\mathbf{h}_t$, STUs reduced the computation and storage complexities from $\mathcal{O}(4XH + 4H^2)$ to $\mathcal{O}(XH + H^2)$. Compared to the projected LSTM (LSTMP), whose recurrent matrices are factorised by a $H \times P$ projection matrix $\mathbf{P}$ to $\mathbf{U}_i\mathbf{P}$, $\mathbf{U}_f\mathbf{P}$, $\mathbf{U}_o\mathbf{P}$, and $\mathbf{U}_c\mathbf{P}$ [12], LSTM$^{\text{STU}}$ is even more efficient than LSTMP with $P = H/4$. The LSTMP also falls into the STU framework, by defining $\mathbf{e}_t = \mathbf{P}\mathbf{h}_{t-1}$.

3.2.2. STU based Highway Network (Highway^{STU})

Similar to the LSTM$^{\text{STU}}$ case, the "virtual unit" output value $\mathbf{e}_t$ in Highway$^{\text{STU}}$ is also shared among all gating units and the candidate unit. By tying both weights and biases, we have

$$\mathbf{e}_t = \mathbf{W}\mathbf{x}_t + \mathbf{b},$$

which is equal to the shared input activation values.

In a sigmoid highway network, i.e. $f(\cdot) = \sigma(\cdot)$, associating $\boldsymbol{\eta}_m$ and $\boldsymbol{\gamma}_m$ with the transform gate, $\boldsymbol{\eta}_r$ and $\boldsymbol{\gamma}_r$ with the carry gate, and $\boldsymbol{\eta}_y$ and $\boldsymbol{\gamma}_y$ with the candidate unit, Eqns. (7) – (9) are modified as

$$\mathbf{m}_t = \sigma_{\boldsymbol{\eta}_m,\boldsymbol{\gamma}_m}(\mathbf{e}_t)$$

$$\mathbf{r}_t = \sigma_{\boldsymbol{\eta}_r,\boldsymbol{\gamma}_r}(\mathbf{e}_t)$$

$$\tilde{\mathbf{y}}_t = \sigma_{\boldsymbol{\eta}_y,\boldsymbol{\gamma}_y}(\mathbf{e}_t).$$

This ties all weight matrices and bias vectors together, and reduces the calculation and storage complexities from $\mathcal{O}(3XH)$ to $\mathcal{O}(XH)$. If $f(\cdot)$ is ReLU, Eqn. (9) is then replaced by

$$\tilde{\mathbf{y}}_t = \mathrm{ReLU}_{\boldsymbol{\eta}_y}(\mathbf{e}_t).$$

where $\boldsymbol{\eta}_y$ is the ReLU output scaling factor vector. Note in both LSTM$^{\text{STU}}$ and Highway$^{\text{STU}}$, almost all parameters and calculations are used in candidate vector generation.

3.3. Training STUs

3.3.1. Training Activation Function Parameters

To train STUs by *error back propagation*, the derivatives of the parameterised activation functions w.r.t. to the function parameters and input activation values are required [20]. Let a_{tj}, η_j, and γ_j be the j^{th} components of $\mathbf{a}_t$, $\boldsymbol{\eta}$, and $\boldsymbol{\gamma}$, then

$$\begin{aligned}\partial\sigma_{\boldsymbol{\eta},\boldsymbol{\gamma}}(\mathbf{a}_t)/\partial a_{tj} &= \eta_j\gamma_j\, e^{-\gamma_j a_{tj}}/(1+e^{-\gamma_j a_{tj}})^2\\ \partial\sigma_{\boldsymbol{\eta},\boldsymbol{\gamma}}(\mathbf{a}_t)/\partial \eta_j &= 1/(1+e^{-\gamma_j a_{tj}})\\ \partial\sigma_{\boldsymbol{\eta},\boldsymbol{\gamma}}(\mathbf{a}_t)/\partial \gamma_j &= \eta_j a_{tj} e^{-\gamma_j a_{tj}}/(1+e^{-\gamma_j a_{tj}})^2,\end{aligned}$$

and

$$\begin{aligned}\partial\tanh_{\boldsymbol{\eta},\boldsymbol{\gamma}}(\mathbf{a}_t)/\partial a_{tj} &= 4\eta_j\gamma_j/(e^{\gamma_j a_{tj}}+e^{-\gamma_j a_{tj}})^2\\ \partial\tanh_{\boldsymbol{\eta},\boldsymbol{\gamma}}(\mathbf{a}_t)/\partial \eta_j &= (e^{\gamma_j a_{tj}}-e^{-\gamma_j a_{tj}})/(e^{\gamma_j a_{tj}}+e^{-\gamma_j a_{tj}})\\ \partial\tanh_{\boldsymbol{\eta},\boldsymbol{\gamma}}(\mathbf{a}_t)/\partial \gamma_j &= 4\eta_j a_{tj}/(e^{\gamma_j a_{tj}}+e^{-\gamma_j a_{tj}})^2.\end{aligned}$$

Similarly, for $\text{ReLU}_{\boldsymbol{\eta}}(\mathbf{a}_t)$, we have

$$\begin{aligned}\partial\,\text{ReLU}_{\boldsymbol{\eta}}(\mathbf{a}_t)/\partial a_{tj} &= \begin{cases}0 & \text{if } a_{tj}<0\\ \eta_j & \text{if } a_{tj}\geqslant 0\end{cases}\\ \partial\,\text{ReLU}_{\boldsymbol{\eta}}(\mathbf{a}_t)/\partial \eta_j &= \max(a_{tj},0).\end{aligned}$$

3.3.2. Normalising the Gradients of the Tied Parameters

When training the shared parameters in the "virtual unit" of an STU, e.g. $\mathbf{W}$, there are

$$\frac{\partial\mathbf{h}_t}{\partial\mathbf{W}} = \left(\frac{\partial\mathbf{h}_t}{\partial\mathbf{i}_t}\frac{\partial\mathbf{i}_t}{\partial\mathbf{e}_t} + \frac{\partial\mathbf{h}_t}{\partial\mathbf{f}_t}\frac{\partial\mathbf{f}_t}{\partial\mathbf{e}_t} + \frac{\partial\mathbf{h}_t}{\partial\mathbf{o}_t}\frac{\partial\mathbf{o}_t}{\partial\mathbf{e}_t} + \frac{\partial\mathbf{h}_t}{\partial\tilde{\mathbf{c}}_t}\frac{\partial\tilde{\mathbf{c}}_t}{\partial\mathbf{e}_t}\right)\frac{\partial\mathbf{e}_t}{\mathbf{W}}$$

for LSTM$^{\text{STU}}$ and

$$\frac{\partial\mathbf{y}_t}{\partial\mathbf{W}} = \left(\frac{\partial\mathbf{y}_t}{\partial\mathbf{m}_t}\frac{\partial\mathbf{m}_t}{\partial\mathbf{e}_t} + \frac{\partial\mathbf{y}_t}{\partial\mathbf{r}_t}\frac{\partial\mathbf{r}_t}{\partial\mathbf{e}_t} + \frac{\partial\mathbf{y}_t}{\partial\tilde{\mathbf{y}}_t}\frac{\partial\tilde{\mathbf{y}}_t}{\partial\mathbf{e}_t}\right)\frac{\partial\mathbf{e}_t}{\mathbf{W}}$$

for Highway$^{\text{STU}}$. Note that $\partial\mathbf{h}_t/\partial\mathbf{b}$, $\partial\mathbf{h}_t/\partial\mathbf{V}$, and $\partial\mathbf{y}_t/\partial\mathbf{b}$ are calculated in the same way. In addition, to use the same hyper parameters (e.g. learning rate) for both tied and untied parameters in training, the gradients of the unfolded LSTM layer parameters are further divided by the number of unfolded steps [38] (here 20 unfolded steps are used).

4. Experimental Setup

The proposed LSTM$^{\text{STU}}$ and Highway$^{\text{STU}}$ models were evaluated on multi-genre broadcast (MGB) data from the MGB3 speech recognition challenge task [30, 31]. The audio is from BBC TV programmes covering a wide range of genres. A 275 hour (275h) full training set was selected from 750 episodes where the training labels were from the sub-titles with a phone matched error rate $< 40\%$ compared to the lightly supervised output [32]. A 55 hour (55h) subset was uniformly sampled at the utterance level from the 275h set. A 63k word vocabulary [33] was used with a trigram word language model (LM) estimated from both the training labels and an extra 640 million word MGB subtitle archive. The test set, **dev17b**, contains 5.55 hours of audio data and 5,201 manually segmented utterances from 14 episodes of 13 shows. System outputs were evaluated with 1-best Viterbi decoding as well as confusion network decoding (CN) [34, 35].

All experiments were conducted with an extended version of HTK 3.5 [37, 38]. The ANN input features were 40d log-Mel filter bank along with their 40d Δ coefficients, which were normalised at the utterance level for mean and at the show-segment level for variance [36]. All models were trained as hybrid system acoustic models by *stochastic gradient descent* based on the cross-entropy criterion with the data shuffled at the frame-level in a 800 sample minibatch [39–41]. About 6k/9k decision tree clustered triphone tied-states along with appropriate training alignments were used for the 55h/275h training sets. The NewBob^{+} learning rate scheduler [21, 38] was used for all models with the setup from our previous MGB systems [36]. Weight decay factors were carefully tuned to maximise the performance of each system. More details about the LSTM implementation and training configuration can be found in [42, 43].

5. Experimental Results

5.1. Experiments on 55 Hour Data Set

5.1.1. LSTM^{STU} Experiments

The experiments started by investigating different STU settings with LSTMs. All 55h LSTMs had one feedforward hidden layer placed between the LSTM layers and output layer with $H = 500$. Two baseline systems with one LSTM layer (1L), $\text{L}_0^{55\text{h}}$ and $\text{L}_1^{55\text{h}}$, were trained, where $\text{L}_0^{55\text{h}}$ was a standard LSTM and $\text{L}_1^{55\text{h}}$ was an LSTMP with the projection size $P = 250$. LSTM$^{\text{STU}}$ systems with different settings were constructed: $\text{L}_2^{55\text{h}}$ followed those in Section 3.2.1; $\text{L}_3^{55\text{h}}$ used fixed $\boldsymbol{\eta} = \mathbf{1}$; $\text{L}_4^{55\text{h}}$ had untied "peephole" matrices; $\text{L}_5^{55\text{h}}$ had untied bias vectors. From Table 1, $\text{L}_3^{55\text{h}}$ had slightly higher word error rates (WER) than $\text{L}_2^{55\text{h}}$, which showed generalising gating to learn $\boldsymbol{\eta}$ was useful. $\text{L}_4^{55\text{h}}$ outperformed $\text{L}_2^{55\text{h}}$ due to the use of distinct "peepholes", but this also increased the training difficulty and was not used. $\text{L}_5^{55\text{h}}$ used untied bias vectors and was found to only improve the convergence speed by producing better criterion values in the early epochs in training and similar values in the end.

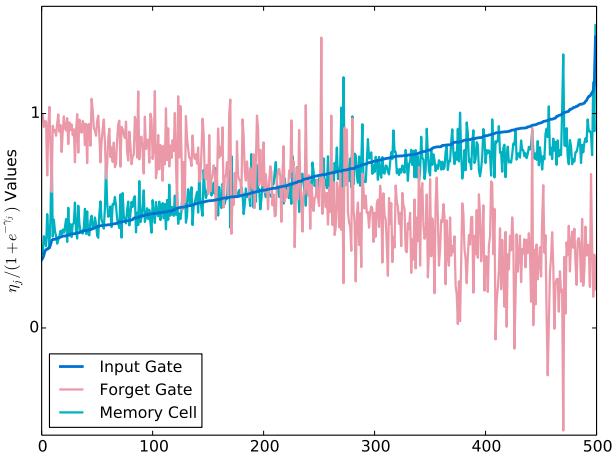

Figure 1: *55h LSTM^{STU} system L_2^{55h} $\eta_j/(1+e^{-\gamma_j})$ values. The node indexes j (x-axis) were re-ranked based on the input gate.*

To understand the STUs in $\text{L}_2^{55\text{h}}$, the units used in Eqn. (5): the input gate, forget gate and the candidate unit were shown in Fig. 1. By ignoring $\mathbf{V}\circ\mathbf{c}_t$ and taking $\mathbf{a}_t = \mathbf{1}$, $\mathbf{i}_t$, $\mathbf{f}_t$, and $\tilde{\mathbf{c}}_t$ can be approximately evaluated by $\eta_j/(1+e^{-\gamma_j})$. From Fig. 1, it can be seen that the input gate and forget gate follow roughly opposite trends, which coincides with replacing $\mathbf{i}_t$ by $\mathbf{1}-\mathbf{f}_t$ in Eqn. (11). The candidate vector values lie in between of those from the other two units, and are more similar to $\mathbf{i}_t$ since they are multiplied to function together in Eqn. (5). These showed

that STUs can still learn reasonable gating functions with the additional untied parameters.

Comparing L_2^{55h} with L_0^{55h} and L_1^{55h}, while producing similar WERs, L_2^{55h}, L_0^{55h}, and L_1^{55h} had 0.29 million (M), 1.16M, and 0.79M parameters in the LSTM layer. Hence, the use of STUs can reduce calculation and storage by a factor of four without increasing the WER. The LSTM systems with two stacked recurrent layers (2L) were also investigated, and the STU based system L_8^{55h} still generated similar WERs to LSTMP L_7^{55h}.

ID	System	tg	cn
L_0^{55h}	1L LSTM	32.9	32.2
L_1^{55h}	1L LSTMP $(P = 250)$	32.9	32.1
L_2^{55h}	1L LSTM^{STU}	32.6	31.9
L_3^{55h}	1L LSTM^{STU} ($\boldsymbol{\eta}$ fixed to $\mathbf{1}$)	33.3	32.5
L_4^{55h}	1L LSTM^{STU} (Untie $\mathbf{V}$)	32.8	32.0
L_5^{55h}	1L LSTM^{STU} (Untie $\mathbf{b}$)	33.1	32.2
L_7^{55h}	2L LSTMP $(P = 250)$	31.3	30.6
L_8^{55h}	2L LSTM^{STU}	31.4	30.8

Table 1: *55h LSTM system ($H = 500$) %WERs on dev17b. A trigram LM with Viterbi (tg) or CN (cn) decoding are used.*

5.1.2. Highway^{STU} Experiments

STUs were also used for both sigmoid and ReLU highway networks with $H = 500$, and the results were listed in Table 2. For sigmoid models, the 7 layer (7L) highway network S_1^{55h} had a 4.2% relative WER reduction (WERR) over the 7L deep neural network (DNN) S_0^{55h}. S_1^{55h} and S_0^{55h} had 4.83M and 1.61M hidden layer parameters. The Highway^{STU} model, S_2^{55h}, had almost the same WERs as the standard highway network and the same number of parameters as a DNN. The use of STUs retained the WER reduction obtained from highway connections while increasing the number of hidden layer parameters by only 1.1% rather than by 200% with the standard highway model. The 15 layer (15L) DNN S_3^{55h} gave a 3.9% WERR over the 7L DNN S_0^{55h}. Both standard and STU based highway systems, S_4^{55h} and S_5^{55h}, resulted in WERRs of 3.5% and 4.1% over S_3^{55h}, while using 6.51M and 0.04M extra parameters respectively.

The same experiments were also repeated for the ReLU systems. The 7L DNN baseline R_0^{55h} gave a 4.7% WERR over S_0^{55h}. 2.4% and 3.3% WERRs were obtained by using standard and STU based highway connections. The 15L ReLU DNN, R_3^{55h}, outperformed R_0^{55h} by a 2.4% WERR, and its relevant highway models R_4^{55h} and R_5^{55h} both outperformed R_3^{55h} by about 2% WERR. This showed the STU idea was also applicable to ReLU. Note that ReLU systems obtained smaller improvements from highway connections than the sigmoid systems, which is reasonable since ReLUs suffer less from information attenuation than sigmoids.

5.2. Experiments on 275 Hour Data Set

In order to ensure that the 55h results and findings can scale to a significantly larger training set, some selected LSTM and highway networks were built on the full 275h set. The hidden layer size H and LSTMP projection size P were increased to 1000 and 500, which quadrupled the parameters to better model the full training set. From Table 3, the 2L LSTMP L_3^{275h} gave a 3% WERR over 1L LSTMP L_1^{275h}. Comparing to S_2^{275h} and R_2^{275h}, the sigmoid and ReLU highway networks, S_4^{275h} and R_4^{275h}, had a 4.0% and a 3.7% WERRs respectively by increasing the model depths from 7L to 15L. All of the LSTM^{STU} and sigmoid/ReLU

ID	System	tg	cn
S_0^{55h}	7L sigmoid DNN	35.8	34.7
S_1^{55h}	7L sigmoid Highway	34.3	33.3
S_2^{55h}	7L sigmoid Highway^{STU}	34.3	33.2
S_3^{55h}	15L sigmoid DNN	34.4	33.4
S_4^{55h}	15L sigmoid Highway	33.2	32.2
S_5^{55h}	15L sigmoid Highway^{STU}	33.0	32.0
R_0^{55h}	7L ReLU DNN	34.1	33.1
R_1^{55h}	7L ReLU Highway	33.2	32.3
R_2^{55h}	7L ReLU Highway^{STU}	33.0	32.0
R_3^{55h}	15L ReLU DNN	33.2	32.2
R_4^{55h}	15L ReLU Highway	32.5	31.5
R_5^{55h}	15L ReLU Highway^{STU}	32.6	31.6

Table 2: *55h highway system ($H = 500$) %WERs on dev17b. A trigram LM with Viterbi (tg) or CN (cn) decoding are used.*

Highway^{STU} systems produced similar WERs to their corresponding conventional LSTMP and highway networks while using fewer than 40% of the parameters in the hidden layers. This validates our previous finding on a larger data set that the proposed STU can work as well as the widely used traditional gating units with far fewer parameters. The STU approach is a highly efficient way to perform general gating and information merging that can also be applied to other gated models, such as GRUs, recurrent highway networks, quasi-RNNs, and highway LSTMs *etc*.

ID	System	tg	cn
L_1^{275h}	1L LSTMP $(P = 500)$	26.5	26.0
L_2^{275h}	1L LSTM^{STU}	26.5	26.0
L_3^{275h}	2L LSTMP $(P = 500)$	25.7	25.2
L_4^{275h}	2L LSTM^{STU}	25.9	25.3
S_1^{275h}	7L sigmoid Highway	27.7	27.0
S_2^{275h}	7L sigmoid Highway^{STU}	27.6	27.0
S_4^{275h}	15L sigmoid Highway	26.3	25.8
S_5^{275h}	15L sigmoid Highway^{STU}	26.3	25.9
R_1^{275h}	7L ReLU Highway	27.2	26.4
R_2^{275h}	7L ReLU Highway^{STU}	27.2	26.3
R_4^{275h}	15L ReLU Highway	26.2	25.8
R_5^{275h}	15L ReLU Highway^{STU}	26.1	25.7

Table 3: *275h system ($H = 1000$) %WERs on dev17b. A trigram LM with Viterbi (tg) or CN (cn) decoding are used.*

6. Conclusions

This paper proposed the use of STUs for efficient gating in LSTMs and feedforward highway networks for acoustic modelling. The weight matrices and bias vectors from all units in the same target layer are tied together to save calculations and storage space, and additional linear input/output value scaling factors are associated with the activation functions of each hidden node individually, in order to learn distinct functions for all gating and candidate units. Experiments on both 55h and 275h MGB data sets found that STU-based LSTMs and highway networks produced similar WERs to the corresponding models with traditional gating units, while being several times more efficient. It was also shown that STUs learn reasonable gating functions, by using only a few thousand extra untied parameters in each sublayer for gating and candidate vector generation.

7. References

[1] Y. Bengio, P. Simard, & P. Frasconi, "Learning long-term dependencies with gradient descent is difficult", *IEEE Transactions on Neural Networks*, vol. 5, pp. 157–166, 1994.

[2] S. Hochreiter & J. Schmidhuber, "Long short-term memory", *Neural Computation*, vol. 9, pp. 1735–1780, 1997.

[3] J. Chung, C. Gulcehre, K.H. Cho, & Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling", *arXiv.org*, 1412.3555, 2014.

[4] R.K. Srivastava, K. Greff, & J. Schmidhuber, "Highway networks", *arXiv.org*, 1505.00387, 2015.

[5] R.K. Srivastava, K. Greff, & J. Schmidhuber, "Training very deep networks", *Advances in NIPS 28*, Montreal, 2015.

[6] J.G. Zilly, R.K. Srivastava, J. Koutník, & J. Schmidhuber, "Recurrent highway networks", *arXiv.org*, 1607.03474, 2016.

[7] K. Yao, T. Cohn, K. Vylomova, K. Duh, & C. Dyer, "Depth-gated LSTM", *arXiv.org*, 1508.03790, 2015.

[8] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, & W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting", *Advances in NIPS 28*, Montreal, 2015.

[9] J. Bradury, S. Merity, C. Xiong, & R. Socher, "Quasi-recurrent neural networks", *Proc. ICLR*, Toulon, 2017.

[10] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, & J. Schmidhuber, "A novel connectionist system for unconstrained handwriting recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 855–868, 2009.

[11] A. Graves, A.-R. Mohamed, G. Hinton, "Speech recognition with deep recurrent neural networks", *Proc. ICASSP*, Vancouver, 2013.

[12] H. Sak, A. Senior, & F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling", *Proc. Interspeech*, Singapore, 2014.

[13] H. Sak, A. Senior, K. Rao, F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition", *Proc. Interspeech*, Dresden, 2015.

[14] Y. Zhang, G. Chen, D. Yu, K. Yao, S. Khudanpur, & J. Glass, "Highway long short-term memory RNNs for distant speech recognition", *Proc. ICASSP*, Shanghai, 2016.

[15] L. Lu, X. Zhang, & S. Renals, "On training the recurrent neural network encoder-decoder for large vocabulary end-to-end speech recognition", *Proc. ICASSP*, Shanghai, 2016.

[16] G. Pundak & T.N. Sainath, "Highway-LSTM and recurrent highway networks for speech recognition", *Proc. Interspeech*, Stockholm, 2017.

[17] L. Lu & S. Renals, "Small-footprint deep neural networks with highway connections for speech recognition", *Proc. Interspeech*, San Francisco, 2016.

[18] Y. Zhang, W. Chan, & N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition", *Proc. ICASSP*, New Orleans, 2017.

[19] L. Tao, Y. Zhang, & Y.Artzi, "Training RNNs as fast as CNNs", *arXiv.org*, 1709.02755, 2017.

[20] C. Zhang & P.C. Woodland, "Parameterised sigmoid and ReLU hidden activation functions for DNN acoustic modelling", *Proc. Interspeech*, Dresden, 2015.

[21] C. Zhang, *Joint Training Methods for Tandem and Hybrid Speech Recognition Systems using Deep Neural Networks*, Ph.D. thesis, University of Cambridge, Cambridge, UK, 2017.

[22] S.L. Goh & D.P. Mandic "Recurrent neural networks with trainable amplitude of activation functions", *Neural Networks*, vol. 16, pp. 1095–1100, 2003.

[23] S.M. Siniscalchi, T. Svendsen, F. Sorbello, & C.-H. Lee, "Experimental studies on continuous speech recognition using neural architectures with "adaptive" hidden activation functions", *Proc. ICASSP*, Dallas, 2010.

[24] K. He, X. Zhang, S. Ren, & J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification", *Proc. ICCV*, Santiago, 2015.

[25] Z. Tüske, M. Sundermeyer, R. Schlüter, & H. Ney, "Integrating Gaussian mixtures into deep neural networks: Softmax layer with hidden variables", *Proc. ICASSP*, Brisbane, 2015.

[26] S.M. Siniscalchi, J. Li, & C.-H. Lee, "Hermitian polynomial for speaker adaptation of connectionist speech recognition systems", *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, pp. 2152–2161, 2013.

[27] Y. Zhao, J. Li, J. Xue, & Y. Gong, "Investigating online low-footprint speaker adaptation using generalized linear regression and click-through data", *Proc. ICASSP*, Brisbane, 2015.

[28] P. Swietojanski, J. Li, & S. Renals, "Learning hidden unit contributions for unsupervised acoustic model adaptation", *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, pp. 1450–1463, 2016.

[29] C. Zhang & P.C. Woodland, "DNN speaker adaptation using parameterised sigmoid and ReLU hidden activation functions", *Proc. ICASSP*, Shanghai, 2016.

[30] `http://www.mgb-challenge.org`

[31] P. Bell, M.J.F. Gales, T. Hain, J. Kilgour, P. Lanchantin, X. Liu, A. McParland, S. Renals, O. Saz, M. Wester, & P.C. Woodland, "The MGB challenge: Evaluating multi-genre broadcast media transcription", *Proc. ASRU*, Scottsdale, 2015.

[32] P. Lanchantin, M.J.F. Gales, P. Karanasou, X. Liu, Y. Qian, L. Wang, P.C. Woodland, & C. Zhang, "Selection of Multi-Genre Broadcast data for the training of automatic speech recognition systems", *Proc. Interspeech*, San Francisco, 2016.

[33] K. Richmond, R. Clark, & S. Fitt, "On generating Combilex pronunciations via morphological analysis", *Proc. Interspeech*, Makuhari, 2010.

[34] L. Mangu, E. Brill, & A. Stolcke, "Finding consensus in speech recognition: Word error minimization and other applications of confusion networks", *Computer Speech & Language*, vol. 14, pp. 373–400, 2000.

[35] G. Evermann & P. Woodland, "Large vocabulary decoding and confidence estimation using word posterior probabilities", *Proc. ICASSP*, Istanbul, 2000.

[36] P.C. Woodland, X. Liu, Y. Qian, C. Zhang, M.J.F. Gales, P. Karanasou, P. Lanchantin, & L. Wang, "Cambridge University transcription systems for the Multi-Genre Broadcast challenge", *Proc. ASRU*, Scottsdale, 2015.

[37] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, A. Ragni, V. Valtchev, P. Woodland, & C. Zhang, *The HTK Book (for HTK version 3.5)*, Cambridge University Engineering Department, 2015.

[38] C. Zhang & P.C. Woodland, "A general artificial neural network extension for HTK", *Proc. Interspeech*, Dresden, 2015.

[39] H.A. Bourlard & N. Morgan, "Connectionist Speech Recognition: A Hybrid Approach", Kluwer Academic Publishers, Norwell, MA, USA 1993.

[40] G.E. Dahl, D. Yu, L. Deng, & A. Acero, 'Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition", *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, pp. 30–42, 2012.

[41] G. Saon, H. Soltau, A. Emami, & M. Picheny, "Unfolded recurrent neural networks for speech recognition", *Proc. Interspeech*, Singapore, 2014.

[42] C. Zhang & P.C. Woodland, "High order recurrent neural networks for acoustic modelling", *Proc. ICASSP*, Calgary, 2018.

[43] F.L. Kreyssig, C. Zhang, & P.C. Woodland, "Improved TDNNs using deep kernels and frequency dependent Grid-RNNs", *Proc. ICASSP*, Calgary, 2018.