HIGH ORDER RECURRENT NEURAL NETWORKS FOR ACOUSTIC MODELLING

C. Zhang & P. C. Woodland

Cambridge University Engineering Dept., Trumpington St., Cambridge, CB2 1PZ U.K.

{cz277,pcw}@eng.cam.ac.uk

ABSTRACT

Vanishing long-term gradients are a major issue in training standard recurrent neural networks (RNNs), which can be alleviated by long short-term memory (LSTM) models with memory cells. However, the extra parameters associated with the memory cells mean an LSTM layer has four times as many parameters as an RNN with the same hidden vector size. This paper addresses the vanishing gradient problem using a high order RNN (HORNN) which has additional connections from multiple previous time steps. Speech recognition experiments using British English multi-genre broadcast (MGB3) data showed that the proposed HORNN architectures for rectified linear unit and sigmoid activation functions reduced word error rates (WER) by 4.2% and 6.3% over the corresponding RNNs, and gave similar WERs to a (projected) LSTM while using only 20%–50% of the recurrent layer parameters and computation.

1. INTRODUCTION

A recurrent neural network (RNN) is an artificial neural network layer where hidden layer outputs from the previous time step form part of the input used to process the current time step [1, 2]. This allows information to be preserved through time and is well suited to sequence processing problems, such as acoustic and language modelling for automatic speech recognition [3, 4]. However, training RNNs with sigmoid activation functions by gradient descent can be difficult. The key issues are *exploding and vanishing gradients* [5], *i.e.*, the long-term gradients, which are back-propagated through time, can either continually increase (explode) or decrease to zero. This causes RNN training to either fail to capture long-term temporal relations or for standard update steps to put parameters out of range.

Many methods have been proposed to solve the gradient exploding and vanishing problems. While simple gradient clipping has been found to work well in practice to prevent gradient exploding [4], circumventing vanishing gradients normally requires more sophisticated strategies [6]. For instance [7] uses Hessian-Free training which makes use of second-order derivative information. Modifying the recurrent layer structure is another approach. The use of both rectified linear unit (ReLU) and sigmoid activation functions with trainable amplitudes were proposed to maintain the magnitude of RNN long-term gradients [8–10]. A gating technique is used in the long short-term memory (LSTM) model where additional parameters implement a memory circuit which can remember long-term information from the recurrent layer [11]. A model similar to the LSTM is the gated recurrent unit [12]. More recently, additional residual [13] and highway connections [14] were proposed to train very deep feed-forward models, which allows gradients to pass more easily through many layers. Various similar ideas have been applied to recurrent models [15–20]. Among these approaches, the LSTM has recently become the dominant type of recurrent architecture. However LSTMs, due to the extra parameters associated with gating, use four times more parameters as standard RNNs with the same hidden layer size, which significantly increases storage and computation in both training and testing.

In this paper, we propose another RNN modification, the high order RNN (HORNN), as an alternative to the LSTM. It handles vanishing gradients by adding connections from hidden state values at multiple previous time steps to the RNN input. By interpreting the RNN layer hidden vector as a continuous valued hidden state, the connections are termed high order since they introduce dependencies on multiple previous hidden states. Acoustic modelling using HORNNs is investigated for both sigmoid and ReLU activation functions. In the sigmoid case, it is found that additional high order connections are beneficial. Furthermore, analogous to the projected LSTM (LSTMP) [22], a linear recurrent projection layer can be used by HORNNs to reduce the number of parameters, which results in the projected HORNN (HORNNP). Experimental results show that the HORNN/HORNNP (both sigmoid and ReLU) have similar word error rates (WERs) to LSTM/LSTMP models with the same hidden vector size, while using fewer than half the parameters and computation. Furthermore, HORNNs were also found to outperform RNNs with residual connections in terms of both speed and WER.

This paper is organised as follows. Section 2 reviews RNN and LSTM models. The (conditional) Markov property of RNNs is described in Sec. 3, which leads to HORNNs and architectures for both sigmoid and ReLU activation functions. The experimental setup and results are given in Sec. 4 and Sec. 5, followed by conclusions.

2. RNN AND LSTM MODELS

In this paper, an RNN refers to an *Elman network* [2] that produces its output hidden vector at step t, $\mathbf{h}_t$, based on the previous output $\mathbf{h}_{t-1}$ and the current input $\mathbf{x}_t$ by

$$\mathbf{h}_t = f(\mathbf{a}_t) = f(\mathbf{W}\mathbf{x}_t + \mathbf{U}\mathbf{h}_{t-1} + \mathbf{b}), \tag{1}$$

where $\mathbf{W}$ and $\mathbf{U}$ are the weights, $\mathbf{b}$ is the bias, and $f(\cdot)$ and $\mathbf{a}_t$ are the activation function and its input activation value. In general, $\mathbf{h}_t$ is processed by a number of further layers to obtain the final network output. It is well known that when $f(\cdot)$ is the sigmoid denoted $\sigma(\cdot)$, RNNs suffer from the vanishing gradient issue since

$$\frac{\partial \sigma(\mathbf{a}_t)}{\partial \mathbf{a}_t} = \sigma\left(\mathbf{a}_t\right)(1 - \sigma(\mathbf{a}_t)) \leqslant \frac{1}{4},$$

which enforces gradient magnitute reductions in backpropagation [3]. Note that ReLU RNNs suffer less from this issue.

In contrast to a standard RNN, the LSTM model resolves gradient vanishing by using an additional linear state $\mathbf{c}_t$ at each step of the sequence, which can be viewed as a memory cell. At each

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

step, a new cell candidate $\tilde{\mathbf{c}}_t$ is created to encode the information from the current step. $\mathbf{c}_t$ is first updated by interpolating $\mathbf{c}_{t-1}$ with $\tilde{\mathbf{c}}_t$ based on the forget gate $\mathbf{f}_t$ and input gate $\mathbf{i}_t$, and then converted to the LSTM hidden state by transforming with hyperbolic tangent (tanh) and scaling by the output gate $\mathbf{o}_t$. This procedure simulates a memory circuit where $\mathbf{f}_t$, $\mathbf{i}_t$, and $\mathbf{o}_t$ are analogous to its logic gates [11]. More specifically, an LSTM layer step t is evaluated as

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{V}_i \odot \mathbf{c}_{t-1} + \mathbf{b}_i),\\
\mathbf{f}_t &= \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{V}_f \odot \mathbf{c}_{t-1} + \mathbf{b}_f),\\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{W}_c\mathbf{x}_t + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{b}_c),\ \mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t,\\
\mathbf{o}_t &= \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{V}_o \odot \mathbf{c}_t + \mathbf{b}_o),\\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t),
\end{aligned}$$

where $\odot$ represents element-wise product, and the $\mathbf{V}$ matrices are diagonal which serve as a "peephole". Although LSTMs work very well on a large variety of tasks, it is computationally very expensive. The representations for each temporal step, $\tilde{\mathbf{c}}_t$, are extracted in the same way as the RNN $\mathbf{h}_t$. However, the additional cost of finding $\mathbf{c}_t$ over $\tilde{\mathbf{c}}_t$, requires three times the computation and parameter storage since $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t$ all need to be calculated.

3. HIGH ORDER RNN ACOUSTIC MODELS

In this section, HORNNs are proposed by relaxing the first-order Markov conditional independence constraint.

3.1. Markov Conditional Independence

The posterior probability of the T frame label sequence $y_{1:T}$ given the T frame input sequence $\mathbf{x}_{1:T}$ can be found by integrating over all possible continuous hidden state sequences $\tilde{\mathbf{h}}_{1:T}$

$$\begin{aligned}
P(y_{1:T}|\mathbf{x}_{1:T}) &= \int P(y_{1:T}|\tilde{\mathbf{h}}_{1:T}, \mathbf{x}_{1:T}) p(\tilde{\mathbf{h}}_{1:T}|\mathbf{x}_{1:T})\, d\tilde{\mathbf{h}}_{1:T}\\
&= \int \prod_{t=1}^{T} P(y_t|y_{1:t-1}, \tilde{\mathbf{h}}_{1:T}, \mathbf{x}_{1:T}) p(\tilde{\mathbf{h}}_t|\tilde{\mathbf{h}}_{1:t-1}, \mathbf{x}_{1:T})\, d\tilde{\mathbf{h}}_{1:T}.
\end{aligned}$$

When implemented using an RNN,

$$P(y_t|y_{1:t-1}, \tilde{\mathbf{h}}_{1:T}, \mathbf{x}_{1:T}) = P(y_t|\tilde{\mathbf{h}}_t),$$

which is produced by the layers after the RNN layer. From Eqn. (1), $\tilde{\mathbf{h}}_t$ depends only on $\tilde{\mathbf{h}}_{t-1}$ and $\mathbf{x}_t$, *i.e.*,

$$p(\tilde{\mathbf{h}}_t|\tilde{\mathbf{h}}_{1:t-1}, \mathbf{x}_{1:T}) = p(\tilde{\mathbf{h}}_t|\tilde{\mathbf{h}}_{t-1}, \mathbf{x}_t). \tag{2}$$

Since the initial hidden state is given (often set to $\mathbf{h}_0 = \mathbf{0}$), all subsequent states $\mathbf{h}_{1:T}$ are determined by Eqn. (1), which means $p(\tilde{\mathbf{h}}_t|\mathbf{h}_{t-1}, \mathbf{x}_t)$ is a Kronecker delta function

$$p(\tilde{\mathbf{h}}_t|\mathbf{h}_{t-1}, \mathbf{x}_t) = \begin{cases} 1 & \text{if } \tilde{\mathbf{h}}_t = \mathbf{h}_t \\ 0 & \text{otherwise} \end{cases}.$$

Hence $P(y_{1:T}|\mathbf{x}_{1:T}) = \prod_{t=1}^{T} P(y_t|\mathbf{h}_t = f(\mathbf{W}\mathbf{x}_t + \mathbf{U}\mathbf{h}_{t-1} + \mathbf{b}))$.

Eqn. (2) is the 1st-order *Markov conditional independence property* [23]. It means that the current state $\mathbf{h}_t$ depends only on its immediately preceding state $\mathbf{h}_{t-1}$ and the current input $\mathbf{x}_t$. This property differs from the 1st-order *Markov property* by also conditioning on $\mathbf{x}_t$.[1] Note that this property also applies to bidirectional RNNs [24], which is easy to show by defining a new hidden state $\mathbf{h}_t^{\text{bid}} = \{\mathbf{h}_t^{\text{fwd}}, \mathbf{h}_t^{\text{bwd}}\}$, where $\mathbf{h}_t^{\text{fwd}}$ and $\mathbf{h}_t^{\text{bwd}}$ are the forward and backward RNN hidden states.

[1]For language modelling, $\mathbf{h}_t$ has the standard Markov property as the RNN models $P(y_{1:T})$ without conditioning on $\mathbf{x}_{1:T}$.

3.2. HORNNs for Sigmoid and ReLU Activation Functions

In this paper, the gradient vanishing issue is tackled by relaxing the first-order Markov conditional independence constraint. Hence, not only the direct preceding state $\mathbf{h}_{t-1}$ but also previous states $\mathbf{h}_{t-n}(n > 1)$ are used when calculating $\mathbf{h}_t$. This adds additional high order connections to the RNN architecture and results in a HORNN. From a training perspective, including high order states creates shortcuts for backpropagation to allow additional long-term information to flow more easily. Specifically, the gradients w.r.t. $\mathbf{h}_{t-1}$ of a general n-order RNN can be obtained by

$$\frac{\partial \mathcal{F}}{\partial \mathbf{h}_{t-1}} = \sum_{i=1}^{n} \frac{\partial \mathcal{F}}{\partial \mathbf{h}_{t-i-1}} \frac{\partial \mathbf{h}_{t-i-1}}{\partial \mathbf{h}_{t-1}}, \tag{3}$$

where $\mathcal{F}$ is the training criterion. For $n > 1$, Eqn. (3) sums multiple terms to prevent the gradient vanishing. From an inference (testing) perspective, an RNN assumes sufficient past temporal information has been embedded in the representation $\mathbf{h}_{t-1}$, but using a fixed sized $\mathbf{h}_{t-1}$, means that information from distant long-term steps may not be properly integrated with new short-term information. The HORNN architecture allows more direct access to the past long-term information.

There are many alternative ways of using $\mathbf{h}_{t-n}$ in the calculation of $\mathbf{h}_t$ in the HORNN framework. This paper assumes that the high order connections are linked to the input at step t. It was found to be sufficient to use only one high order connection at the input, *i.e.*

$$\mathbf{h}_t = f(\mathbf{W}\mathbf{x}_t + \mathbf{U}_1\mathbf{h}_{t-1} + \mathbf{U}_n\mathbf{h}_{t-n} + \mathbf{b}). \tag{4}$$

Here $\mathbf{h}_{t-n}$ can be viewed as a kind of "memory" whose temporal resolution is modified by $\mathbf{U}_n$. From our experiments the structure in Eqn. (4) allowed ReLU HORNNs to give similar WERs to LSTMs. However, when using sigmoid HORNNs, a slightly different structure is needed to reach a similar WER. This has an extra high order connection from $\mathbf{h}_{t-m}$ to the sigmoid function input, *i.e.*

$$\mathbf{h}_t = f(\mathbf{W}\mathbf{x}_t + \mathbf{U}_1\mathbf{h}_{t-1} + \mathbf{U}_n\mathbf{h}_{t-n} + \mathbf{h}_{t-m} + \mathbf{b}). \tag{5}$$

Here, $\mathbf{h}_{t-m}$ is directly added to the sigmoid input without impacting the temporal resolution at t since $\mathbf{h}_{t-m}$ is from a previous sigmoid output. Eqns. (4) and (5) are used for ReLU and sigmoid HORNNs throughout the paper.

3.3. Parameter Control using Matrix Factorisation

Comparing Eqns. (4) and (5) to Eqn. (1), HORNN increases the number of RNN layer parameters from $(D_x + D_h)D_h + D_h$ to $(D_x + 2D_h)D_h + D_h$, where D_x and D_h are the sizes of $\mathbf{x}_t$ and $\mathbf{h}_t$. One method to reduce the increase in parameters is to project the hidden state vectors to some a dimension D_p with a recurrent linear projection $\mathbf{P}$ [22]. This factorises $\mathbf{U}_1$ and $\mathbf{U}_n$ in Eqns. (4) and (5) to $\mathbf{U}_{p1}\mathbf{P}$ and $\mathbf{U}_{pn}\mathbf{P}$ with a *low-rank approximation*. The projected HORNNs (denoted by HORNNP) for ReLU and sigmoid activations are hence defined as

$$\mathbf{h}_t = f(\mathbf{W}\mathbf{x}_t + \mathbf{U}_{p1}\mathbf{P}\mathbf{h}_{t-1} + \mathbf{U}_{pn}\mathbf{P}\mathbf{h}_{t-n} + \mathbf{b}) \tag{6}$$

and

$$\mathbf{h}_t = f(\mathbf{W}\mathbf{x}_t + \mathbf{U}_{p1}\mathbf{P}\mathbf{h}_{t-1} + \mathbf{U}_{pn}\mathbf{P}\mathbf{h}_{t-n} + \mathbf{h}_{t-m} + \mathbf{b}), \tag{7}$$

and the number of parameters used is $D_hD_p + (D_x + 2D_p)D_h + D_h$. The resulting parameter reduction ratio is approximately $2D_h/3D_p$ (given $D_h > D_p \gg D_x$). Note that the same idea was used by

the projected LSTM (LSTMP) to factorise $\mathbf{U}_i$, $\mathbf{U}_f$, $\mathbf{U}_c$, and $\mathbf{U}_o$ [22], which reduces the number of LSTM parameters from $4(D_x + D_h)D_h + 7D_h$ to $D_h D_p + 4(D_x + D_p)D_h + 7D_h$.

Next we compare the computational complexity of LSTMs and HORNNs. Given that multiplying a $l \times m$ matrix by a $m \times n$ matrix ($l \neq m \neq n$) requires lmn multiply-adds, and ignoring all element-wise operations, the testing complexity for a HORNNP layer is $\mathcal{O}(T(D_x + 3D_p)D_h)$, whereas for an LSTMP it is $\mathcal{O}(TD_hD_p + 4T(D_x + D_p)D_h)$. This shows that HORNNPs use less than 3/5 of the calculations of LSTMPs. It has been found that HORNNPs often result in a 50% speed up over LSTMPs in our current HTK implementation [25–27].

3.4. Related Work

After independently developing the HORNN for acoustic modelling, we found that similar ideas had previously been applied to rather different tasks [28–32]. However, both the research focus and model architectures were different to this paper. In particular, the model proposed in [28, 31] is equivalent to Eqn. (4) without subsampling the high order hidden vectors, and [32] applied that model to TIMIT phone recognition. Furthermore, previous studies didn't discuss the high order connections in the Markov property framework.

Adding $\mathbf{h}_{t-m}$ to the input of the sigmoid function in Eqn. (5) is similar to the residual connection in residual networks [13]. A residual RNN (ResRNN) with a *recurrent kernel depth* of two ($d = 2$) can be written as

$$\mathbf{h}_t = f(\mathbf{U}_{d2} f(\mathbf{W}\mathbf{x}_t + \mathbf{U}_{d1}\mathbf{h}_{t-1} + \mathbf{b}) + \mathbf{h}_{t-m}), \qquad (8)$$

where $m = 1$ [17]. Another related model is the recent residual memory network [21], which can be viewed as an unfolded HORNN defined in Eqn. (4) with $\mathbf{U}_1$ and $\mathbf{b}$ being zero, $\mathbf{W}$ being distinct untied parameters in each unfolded layer, and $n \geqslant 1$ being any positive integer. In addition, since highway networks [14] can be viewed as a generalised form of the residual networks, highway RNNs and LSTMs are also related to this work [15, 19]. Note that it is also possible to combine the residual and highway ideas with HORNNs by increasing the recurrent depth.

4. EXPERIMENTAL SETUP

The proposed HORNN models were evaluated by training systems on multi-genre broadcast (MGB) data from the MGB3 speech recognition challenge task [33, 34]. The audio is from BBC TV programmes covering a range of genres. A 275 hour (275h) full training set was selected from 750 episodes where the sub-titles have a phone matched error rate $< 40\%$ compared to the lightly supervised output [35] which was used as training supervision. A 55 hour (55h) subset was sampled at the utterance level from the 275h set. A 63k word vocabulary [36] was used with a trigram word level language model (LM) estimated from both the acoustic transcripts and a separate 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. This is a subset of the official full development set (**dev17a**) with data that overlaps training and test sets excluded. System outputs were evaluated with confusion network decoding (CN) [37] as well as 1-best Viterbi decoding.

All experiments were conducted with an extended version of HTK 3.5 [25, 26]. The LSTM was implemented following [22]. A 40d log-Mel filter bank analysis was used and expanded to an 80d vector with its Δ coefficients. The data was normalised at the utterance level for mean and at the show-segment level for variance [38]. The inputs at each recurrent model time step were single frames delayed for 5 steps [22, 39]. All models were trained using the cross-entropy criterion and frame-level shuffling used. All recurrent models were unfolded for 20 time steps, and the gradients of the shared parameters were normalised by dividing by the sharing counts [26]. The maximum parameter changes were constrained by update value clipping with a threshold of 0.32 for a minibatch with 800 samples.

About 6k/9k decision tree clustered triphone tied-states along with GMM-HMM/DNN-HMM system training alignments were used for the 55h/275h training sets. One hidden layer with the same dimension as $\mathbf{h}_t$ was added between the recurrent and output layers to all models. The NewBob$^+$ learning rate scheduler [26, 27] was used to train all models with the setup from our previous MGB systems [38]. An initial learning rate of 5×10^{-4} was used for all ReLU models, while an initial rate of 2×10^{-3} was used to train all the other models. Since regularisation plays an important role in RNN/LSTM training, weight decay factors were carefully tuned to maximise the performance of each system.

5. EXPERIMENTAL RESULTS

5.1. 55 Hour Single Layer HORNN Experiments

Initial experiments studied various HORNN architectures in order to investigate suitable values of n for the ReLU model in Eqn. (4), and for both m and n for the sigmoid model in Eqn. (5). To save computation, the 55h subset was used for training. All models had one recurrent layer with the $\mathbf{h}_t$ size fixed to 500. An LSTM and a standard RNN were created as baselines, which had 1.16M and 0.29M parameters in the recurrent layers respectively. A ResRNN, defined by Eqn. (8) was also tested as an additional baseline using both ReLU and sigmoid functions.[2] ResRNNs had the same number of parameters (0.54M) as the HORNNs. Note that rather than the standard case with $m = 1$ [17], $m \in [1, 4]$ were examined which falls into the high order framework when $m > 1$. For HORNNs, $n \in [2, 6]$ were tested; m was fixed to 2 for all sigmoid HORNNs. From the results shown in Figure 1, the LSTM gives lower WERs than a standard RNN, but the ReLU ResRNN with m set to 1 or 2 had a similar WER to the LSTM.

ReLU HORNNs gave WERs at least as low as the LSTM and the best ReLU ResRNN systems. Sigmoid HORNNs gave better WERs than sigmoid ResRNNs and similar WERs to those from the LSTM. The performance can be further improved by using p-sigmoid [40] as the HORNN activation function which associates a linear scaling factor to each recurrent layer output unit and makes it more similar to a ReLU. In addition, HORNNs were faster than both LSTM and ResRNNs. ResRNNs were slightly slower than HORNNs since the second matrix multiplication depends on the first one at each recurrent step. For the rest of the experiments, all ReLU HORNNs used $n = 4$, and all sigmoid HORNNs used $m = 1$ and $n = 2$.

5.2. Projected and Multi-Layered HORNN Results

Next, projected LSTMs and projected HORNNs were compared. First, D_h (the size of $\mathbf{h}_t$) and D_p (the projected vector size) were fixed to 500 and 250 respectively for the single recurrent layer (1L) LSTMP and HORNNP models. The LSTMP baseline $\mathrm{L}_1^{55\mathrm{h}}$ had 0.79M parameters and HORNNP system $\mathrm{S}_1^{55\mathrm{h}}$ and $\mathrm{R}_1^{55\mathrm{h}}$ had 0.42M parameters. From Table 1, the HORNNPs have similar WERs to the LSTMP. By further reducing D_p to 250, the HORNN systems, $\mathrm{S}_2^{55\mathrm{h}}$ and $\mathrm{R}_2^{55\mathrm{h}}$, reduced the number of parameters to 0.23M and gave

[2] This is also the first time to apply such ResRNNs to acoustic modelling.

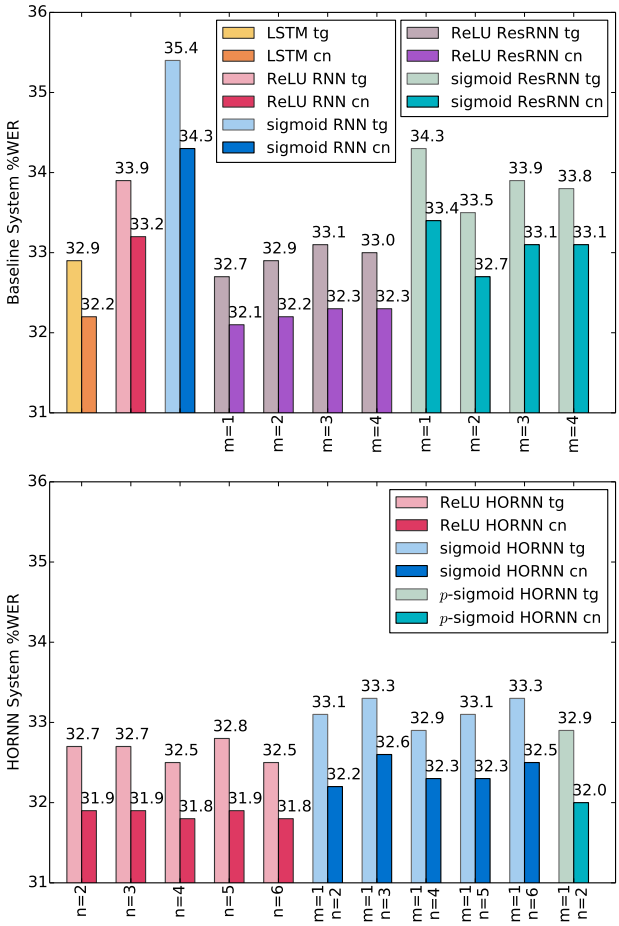

Fig. 1. *%WERs of 55h systems on dev17b. Systems use a trigram LM with Viterbi decoding (tg) or CN decoding (cn).*

similar WERs to LSTM and LSTMP (L_1^{55h}) with only 20% and 29% of the recurrent layer parameters.

The values of D_h and D_p for HORNNs were increased to 800 and 400 respectively to make the overall number of recurrent layer parameters (1.02M) closer to that of the 500d LSTM (1.16M). This produced system S_3^{55h} and R_3^{55h}. The LSTMP was also modified to $D_h = 600$ and $D_p = 300$ to have 1.10M parameters. From the results in Table 1, S_3^{55h} and R_3^{55h} both outperformed L_2^{55h} by a margin since the 800d representations embed more accurate temporal information than with 600d. The p-sigmoid function was not used for HORNNPs since the linear projection layer also scales $\mathbf{h}_t$.

Finally, the LSTMP and HORNNP were compared by stacking another recurrent layer. With two recurrent layers (2L) of $D_h = 500$ and $D_p = 250$, the 2L HORNNP systems S_4^{55h} and R_4^{55h} had 0.92M parameters and still produced similar WERs to the 2L LSTMP system L_3^{55h} (with 1.91M parameters). These results indicate that rather than spending most of the calculations on maintaining the LSTM memory cell, it is more effective to use HORNNs and use the computational budget for extracting better temporal representations using wider and deeper recurrent layers.

5.3. Experiments on 275 Hour Data Set

To ensure that the previous results scale to a significantly larger training set, some selected LSTMP and HORNNP systems were built on

ID	System	D_h	D_p	tg	cn
L_1^{55h}	1L LSTMP	500	250	32.9	32.1
L_2^{55h}	1L LSTMP	600	300	32.7	32.0
L_3^{55h}	2L LSTMP	500	250	31.3	30.6
S_1^{55h}	1L sigmoid HORNNP	500	250	32.8	31.9
S_2^{55h}	1L sigmoid HORNNP	500	125	33.0	32.1
S_3^{55h}	1L sigmoid HORNNP	800	400	31.6	30.9
S_4^{55h}	2L sigmoid HORNNP	500	250	31.4	30.7
R_1^{55h}	1L ReLU HORNNP	500	250	32.0	31.4
R_2^{55h}	1L ReLU HORNNP	500	125	32.5	31.8
R_3^{55h}	1L ReLU HORNNP	800	400	31.4	30.7
R_4^{55h}	2L ReLU HORNNP	500	250	31.4	30.7

Table 1. *%WERs for various 55h system on dev17b. Systems use a trigram LM with Viterbi decoding (tg) or CN decoding (cn).*

the full 275h set. Here D_h and D_p were set to 1000 and 500, which increased the number of recurrent layer parameters to better model the full training set. From Table 2, for both single recurrent layer and two recurrent layer architectures, HORNNs still produced similar WERs to the corresponding LSTMPs. This validates our previous finding on a larger data set that the proposed HORNN structures can work as well as the widely used LSTMs on acoustic modelling by using far fewer parameters. In addition, along with the multi-layered structure, HORNNs can also be applied to other kinds of recurrent models by replacing RNNs and LSTMs, such as the bidirectional [24] and grid [39, 41, 42] structures *etc*. Finally, a 7 layer (7L) sigmoid DNN system, D_1^{275h}, was built following [38] as a reference.

ID	System	D_h	D_p	tg	cn
L_1^{275h}	1L LSTMP	1000	500	26.5	26.0
S_1^{275h}	1L sigmoid HORNNP	1000	500	26.4	25.8
R_1^{275h}	1L ReLU HORNNP	1000	500	26.4	25.9
L_3^{275h}	2L LSTMP	1000	500	25.7	25.2
S_4^{275h}	2L sigmoid HORNNP	1000	500	25.6	25.2
R_4^{275h}	2L ReLU HORNNP	1000	500	25.3	25.0
D_1^{275h}	7L sigmoid DNN	1000		28.4	27.5

Table 2. *%WERs for a selection of 275h system on dev17b. Systems use a trigram LM with Viterbi decoding (tg) or CN decoding (cn).*

6. CONCLUSIONS

This paper proposed the use of HORNNs for acoustic modelling to address the vanishing gradient problem in training recurrent neural networks. Two different architectures were proposed to cover both ReLU and sigmoid activation functions. These yielded 4%-6% WER reductions over the standard RNNs with the same activation function. Furthermore, additional structures were investigated: reducing the number of HORNN parameters with a linear recurrent projected layer; and adding another recurrent layer. In all cases, compared to the projected LSTMs and the residual RNNs, it was shown that HORNNs gave similar WER performance while being significantly more efficient in computation and storage. When the savings in parameter number and computation are used to implement wider or deeper recurrent layers, (projected) HORNNs gave a 4% relative reduction in WER over the comparable (projected) LSTMs .

7. REFERENCES

[1] D.E. Rumelhart, J.L. McClelland, & the PDP Research Group *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, MIT Press, 1986.

[2] J.L. Elman, "Finding structure in time", *Cognitive Science*, vol. 14, pp. 179–211, 1990.

[3] T. Robinson, M. Hochberg and S. Renals. "The use of recurrent neural networks in continuous speech recognition", In *Automatic Speech and Speaker Recognition*, pp. 233–258, Springer, 1996.

[4] T. Mikolov, *Statistical Language Models based on Neural Networks*, Ph.D. thesis, Brno University of Technology, Brno, Czech Republic, 2012.

[5] 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.

[6] R. Pascanu, T. Mikolov, & Y. Bengio, "On the difficulty of training recurrent neural networks", *Proc. ICML*, Atlanta, 2013.

[7] I. Sutskever, J. Martens, & G. Hinton, "Generating text with recurrent neural networks", *Proc. ICML*, New York, 2011.

[8] E. Salinas & L.F. Abbott, "A model of multiplicative neural responses in parietal cortex", *Proc. National Academy of Science U.S.A.*, vol. 93, pp. 11956–11961, 1996.

[9] R.L.T. Hahnloser, "On the piecewise analysis of networks of linear threshold neurons", *Neural Networks*, vol. 11, pp. 691–697, 1998.

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

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

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

[13] K. He, X. Zhang, S. Ren, & J. Sun, "Deep residual learning for image recognition", *Proc. CVPR*, Las Vegas, 2016.

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

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

[16] 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.

[17] Y. Wang & F. Tian, "Recurrent residual learning for sequence classification", *Proc. EMNLP*, Austin, 2016.

[18] A. van den Oord, N. Kalchbrenner, & K. Kavukcuoglu, "Pixel recurrent neural networks", *Proc. ICML*, New York, 2016.

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

[20] J. Kim, M. El-Khamy, & J. Lee, "Residual LSTM: Design of a deep recurrent architecture for distant speech recognition", *Proc. Interspeech*, Stockholm, 2017.

[21] M.K. Baskar, M. Karafiát, L. Burget, K. Veselý, F. Grézl, & J.H. Černocký, "Residual memory networks: Feed-forward approach to learn long-term temporal dependencies", *Proc. ICASSP*, New Orleans, 2017.

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

[23] Y. Bengio & P. Frasconi, *Creadit assignment through time: Alternatives to backpropagation*, *Advances in NIPS 6*, Hong Kong, 1993.

[24] M. Schuster & K.K. Paliwal, "Bidirectional recurrent neural networks", *IEEE Transactions on Signal Processing*, vol. 45, pp. 2673–2681, 1997.

[25] 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.

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

[27] 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.

[28] T. Lin, B.G. Horne, P. Tiňo, & C. Lee Giles, "Learning long-term dependencies in NARX recurrent neural networks", *IEEE Transactions on Neural Networks*, vol. 7, pp. 1329–1338, 1996.

[29] P. Tiňo, M. Čerňanský, & L. Beňušková, "Markovian architectural bias of recurrent neural networks", *IEEE Transactions on Neural Networks*, vol. 15, pp. 6–15, 2004.

[30] I. Sutskever & G. Hinton, "Temporal-kernel recurrent neural networks", *Neural Networks*, vol. 23, pp. 239–243, 2010.

[31] R. Soltani & H. Jiang, "Higher order recurrent neural networks", *arXiv.org*, 1605.00064, 2016.

[32] H. Huang & B. Mak, "To improve the robustness of LSTM-RNN acoustic models using higher-order feedback from multiple histories", *Proc. Interspeech*, Stockholm, 2017.

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

[34] 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.

[35] 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.

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

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

[38] 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.

[39] B. Li & T.N. Sainath, "Reducing the computational complexity of twodimensional LSTMs", *Proc. Interspeech*, Stockholm, 2017.

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

[41] N. Kalchbrenner, I. Danihelka, & A. Graves, "Grid long short-term memory", *Proc. ICLR*, San Juan, 2016.

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