Long short-term memory recurrent neural network architectures for Urdu acoustic modeling

Tehseen Zia[1] · Usman Zahid[1]

Received: 21 July 2018 / Accepted: 29 October 2018 / Published online: 8 November 2018

Abstract

Recurrent neural networks (RNNs) have achieved remarkable improvements in acoustic modeling recently. However, the potential of RNNs have not been utilized for modeling Urdu acoustics. The connectionist temporal classification and attention based RNNs are suffered due to the unavailability of lexicon and computational cost of training, respectively. Therefore, we explored contemporary long short-term memory and gated recurrent neural networks Urdu acoustic modeling. The efficacies of plain, deep, bidirectional and deep-directional network architectures are evaluated empirically. Results indicate that deep-directional has an advantage over the other architectures. A word error rate of 20% was achieved on a hundred words dataset of twenty speakers. It shows 15% improvement over the baseline single-layer LSTMs. It has been observed that two-layer architectures can improve performance over single-layer, however the performance is degraded with further layers. LSTM architectures were compared with gated recurrent unit (GRU) based architectures and it was found that LSTM has an advantage over GRU.

Keywords Recurrent neural networks · Long short-term memory · Acoustic modeling · Speech recognition · Urdu

1 Introduction

Speech is a complex time-variant signal with complex temporal dependencies at the span of different timescales. Recurrent neural networks (RNNs) are a type of neural networks with cyclic connections. This configuration makes them more potent instrument for sequence modeling than feed-forward neural networks. That is why RNN have achieved great milestones in different sequence processing applications such as handwritten recognition (Graves and Schmidhuber 2009), language modeling (Mikolov et al. 2010), polyphonic music prediction (Pascanu et al. 2013a) and acoustic modeling (Sak et al. 2014) etc. For acoustic modeling, however the potential of RNNs have not been fully exploited. Although feed-forward networks known as deep neural networks (DNNs) are considered as established state-of-art in acoustic modeling (Hinton et al. 2012; Graves et al. 2013a), they have different disadvantages over RNNs. They operate over a fixed time window of acoustic frames and therefore can only achieve a limited temporal modeling. Further, as they can only model data within the time window, they are unable to tackle variable speaking rates and long term dependencies. In contract, RNNs model the temporal dependencies by maintaining an internal state (historical contextual information) and updating it at each time step on the basis of previous state and current input. This retention of internal state enables RNNs to hold long term dependencies. The structure also allows RNN to adaptively change the size of contextual window instead of being fixed as in feed-forward networks, which enables them to process different speaking rates.

In state-of-art acoustic model, hidden Markov models (HMMs) are employed for sequence modeling while the states of HMMs are modeled using DNNs. A well-known limitation of HMMs is the Markov assumption which strictly restricts the models to capture temporal correlations within the sequences over a longer span of time (Juang and Rabiner 1991). Since RNNs do not make such an assumption, they have potential to capture longer temporal dependencies. Another advantage of RNNs over HMMs is the greater representational power of neural networks and their ability

✉ Tehseen Zia
tehseen.zia@comsats.edu.pk

Usman Zahid
usman.zahid@comsats.edu.pk

[1] COMSATS University Islamabad, Islamabad, Pakistan

to perform intelligent smoothing by taking into account syntactic and semantic features (Lipton et al. 2015).

In RNN, there are two ways to incorporate contextual information into the sequence processing tasks. Firstly, by modeling temporal variations with recurrent activation function (Schuster and Paliwal 1997). In this approach, RNN has a known limitation called gradient vanishing or exploding problem that severely restricts its ability to capture medium and long term dependencies (Pascanu et al. 2013b). A widely adopted solution of the problem is to employ sophisticated activation functions known as long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997). The development has shown to significantly improve the performance of RNNs in many tasks, e.g., language modeling, handwritten recognition and acoustic modeling of phonemes (Amodei et al. 2016; Hannun et al. 2014; Sak et al. 2014). Another shortcoming of recurrent activation function is that as the input is processed in temporal order, so the output tends to rely mostly on the previous context. Although methods are proposed to incorporate future contexts (e.g., by adding delay between output and targets), backward dependencies are usually not fully exploited. An elegant solution to the problem is to employ bidirectional RNNs (Amodei et al. 2016; Graves et al. 2013a). The bidirectional LSTM (BLSTM) have outperformed the state-of-art HMMs in handwritten recognition (Graves and Schmidhuber 2009). Similarly in continuously speech recognition and large-scale acoustic modeling, BLSTM has shown superior performance than DNNs (Amodei et al. 2016; Chan and Lane 2015; Sak et al. 2014).

The second approach to incorporate contextual information into RNNs is to model structured hierarchy of slow changing and fast changing components by stacking multiple levels of RNNs (a.k.a. deep RNN). This approach is primarily inspired from the remarkable successes of DNNs over swallow networks particularly for object recognition, speech recognition and natural language modeling (Graves et al. 2013a; Krizhevsky et al. 2012; Mikolov et al. 2010). The utility of deep RNNs for language modeling, polyphonic music prediction and handwritten recognition are shown in different studies (Pascanu et al. 2013a). In speech recognition, benchmark results are reported when deep RNNs are employed for phoneme recognition task (Chan and Lane 2015; Graves et al. 2013a; Sak et al. 2014).

This paper is inspired from the recent developments in acoustic modeling using LSTMs. We have explored well-acknowledged RNNs architectures for Urdu acoustic modeling. The investigated architectures include deep, bidirectional, and deep bidirectional LSTMs. The modeling and investigations of these architectures for Urdu speech recognition is the main contribution of this work. A novel deep bidirectional LSTM architecture is proposed to exploit the advantages of both deep and bidirectional networks. This work is a crucial step towards designing end-to-end continuous Urdu speech recognition system which is extremely convenient and efficient as compared to conventional hand-engineered pipeline of speech recognition. The paper is organized as: Related work on recent developments on acoustic modeling and Urdu speech recognition is presented in Sect. 2. The architectures of recurrent neural networks are introduced in Sect. 3. Experimental setup is described in Sect. 4, results are presented in Sect. 5 and conclusion in Sect. 6.

2 Related work

Designing an end-to-end ASR system with RNNs is an active research area (Amodei et al. 2016; Graves and Jaitly 2014; Hannun et al. 2014) which has shown compelling results in the context of scoring DNN-HMM outputs. There are two commonly used approaches for mapping audio sequences of variable lengths to transcripts of varying lengths. The first approach is based attention based paradigm (Sutskever et al. 2014) where an encoder is employed to transform input signal into input feature vectors and a decoder is used to generate a sequence of transcriptions by dynamically attending input feature vectors (Bahdanau et al. 2016). In (Chan et al. 2016), a vanilla attention based model is used with a window based technique in order to reduce the number of input feature vectors that can be attended. In order to further reduce the number of input vectors, a pyramid structure is employed to transform input feature vectors into abstract feature vectors that can be attended while transliterating. The most recent work on using attention based models for acoustic modeling is proposed in (Chiu et al. 2017). Various advancements are considered to achieve superior performance over other architectures. For example, scheduled sampling is employed that consider previous predicted labels rather than ground truth during training period for consistent training and testing. Also, label smoothing is used to prevent model from being over-confident in predictions. Moreover, an external language model is incorporated to deal with misclassified predictions. Although, attention based models have shown their potential for acoustic modeling, they are harder to train due to their higher computational complexity (Yu and Li 2017).

In the second approach, CTC loss function is associated with RNNs for modeling temporal sequences (Amodei et al. 2016; Graves et al. 2013a, b). While a typical acoustic model deals with processing complete sequence of word or phoneme acoustics to output a label, CTC enables the model to generate frame-by-frame labeling with traditional cross entropy objective function. In (Sak et al. 2015), the research has shown that context-dependent phone-based frame labeling achieve better performance than monophone-based frame labeling. It is also shown that CTC based method can achieve comparable performance with LSTM based model. The CTC method is used in deep speech model to directly predict characters instead of phonemes. The character-based CTC method is shown to have advantage of robustness for the accented speech because an accent can impose a smaller effect on graphoneme sequence than the phoneme pronunciation (Rao and Sak 2017). In (Zweig et al. 2017), the researchers have explored other CTC units for labeling rather than phoneme, character and word based labels. Though, CTC based RNNs are found to be effective, choosing the basic CTC unit remains a design challenge and depends on a lexicon. In the above mentioned studies, a fixed pre-determined sequence of decomposed acoustic with corresponding unit label is employed. Nevertheless, such a pre-defined fixed decomposition does not serve the purpose of end-to-end learning and may not necessarily optimal (Yu and Li 2017). Also, these methods cannot readily be adopted for processing acoustic of languages where such a fixed pre-determined lexicon is not available. Furthermore, the CTC based methods are difficult to train in comparison to LSTM with cross-entropy training (Yu and Li 2017). In (Sak et al. 2015), LSTM with cross-entropy are also employed to initialize LSTM network with CTC based training as the randomly initialized CTC network are shown to be very harder to train.

Most of the research on acoustic modeling for speech recognition has been done for English language. Very few efforts have been seen in the context of Urdu acoustic modeling where two techniques are mainly used: neural networks (NNs) and HMMs. The NNs based approaches are employed for recognition of Urdu digits, small vocabulary isolated words and continuous speech recognition (Ahad et al. 2002; Azam et al. 2007; Hasnain and Awan 2008). The main disadvantages of these approaches are limited temporal modeling due to sliding window based processing and inability to handle varying speaking rates. Moreover, the employed networks are shallow and do not possess the representational power as promised by DNNs. Despite NNs and HMMs, the effectiveness of other machine learning approaches such as support vector machine, random forest and linear discriminant analysis have also been analyzed (Ali et al. 2016). However, these methods also have same drawbacks as of NNs. In HMMs based approaches, GMMs are used for modeling states of the models while temporal variations are modeled with state transition probabilities. The key shortcomings of these approaches are: inefficient state modeling due to GMMs (as described above), limited temporal modeling due to Markov assumption and limited representational capability of HMMs (Ashraf et al. 2010; Sarfraz et al. 2010).

3 Recurrent neural networks (RNNs)

The objective of the acoustic model is to take speech audio as input and generate Urdu text transcriptions as output. Consider a single utterance **x** and corresponding label y as an example of a training set$X = \{(\mathrm{x}^1, y^1), (\mathrm{x}^2, y^2), \dots \}$ where the utterance x^i is a vector of time series of length T^i and a time slice x_t^i is an audio representation. The y is a one-hot representation vector of a label. Given an input vectorx_t at time t, the task of the model will be to produce a vector of word probabilities over a set of Urdu words, $O_t = p(W_t|\mathrm{x}_t)$. The mel-frequency cepstral coefficient of speech is usually employed for representing speech signal since it is found to be most effective speech representation technique (Graves et al. 2013a; Hinton et al. 2012; Juang and Rabiner 1991; Rabiner 1989). Therefore, a representation at each time slice $x_{t,p}^i$ will be a cepstral coefficient of the pth frequency bin in the audio frame at time t. The hidden activation of the RNN layer at time t will be computed as a function of input sequence x_t at time t and hidden state at time $t-1$ as[1]:

$$h_t = \sigma\left(W_x x_t + W_h h_{t-1}\right), \tag{1}$$

where σ is a sigmoid function and W_x and W_h are weight matrices for input sequence and hidden state.[2] The output O_t is then computed as a function of activations at time t as:

$$O_t = \sigma\left(W_y h_t\right), \tag{2}$$

where $\mathrm{W_y}$ is a weight matrix for hidden activations. A known drawback of RNN is inability to learn long term dependencies due to gradient vanishing and exploding problem (Pascanu et al. 2013b). A well-adopted solution of the problem is to employ sophisticated activation function known as long short-term memory (LSTM), introduced below.

3.1 LSTM

The key idea is to use a memory cell for maintaining the state of the network over a long period of time. The input, output and forgets operations on the cell (c_t) are regulated with learnable gates known as input gate (i_t), output gate (o_t) and forget gate (f_t). Using these components, the cell content at any time t is computed as (Greff et al. 2016; Hochreiter and Schmidhuber 1997; Sutskever et al. 2014):

$$c_t = f_t \odot c_{t-1} + i_t \odot \hat{c}_t\ , \tag{3}$$

[1] The sequence processing using neural networks is usually performed by operating over a context window at the first layer. We have not considered context window in this section for notational convenience.

[2] Biases are omitted throughout the paper for simplicity.

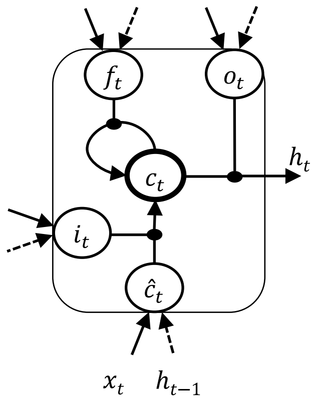


Fig. 1 Block diagram of LSTM

where ⊙ and + respectively denotes element wise multiplication and addition. i_t and f_t are defined as sigmoid functions and $\hat{c}_t$ as tangent function of x_t and h_{t-1}:

$$\hat{c}_t = \tanh\left(W_{xc}\mathrm{x}_t + W_{hc}h_{t-1}\right), \tag{4}$$

$$i_t = \sigma\left(W_{xi}\mathrm{x}_t + W_{hi}h_{t-1}\right), \tag{5}$$

$$f_t = \sigma\left(W_{xf}\mathrm{x}_t + W_{hf}h_{t-1}\right), \tag{6}$$

where W_{x*} and W_{h*} (* denotes the gating function) are weight matrices respectively for input sequence and hidden state (note: these notations will be used throughout the paper unless specified otherwise). It can be noticed that cell content c_t is a linear function of c_{t-1} and therefore not affected from gradient vanishing and exploding problem. Based on c_t, the output is computed as:

$$h_t = o_t \odot \tanh\left(c_t\right), \tag{7}$$

where o_t is defined as:

$$o_t = \sigma\left(W_{xo}\mathrm{x}_t + W_{ho}h_{t-1}\right). \tag{8}$$

The block diagram of the model is shown in Fig. 1.

4 Deep LSTMs

The acoustic waveforms often contain slow changing and fast changing components which are structured in hierarchical manner. This hierarchy is encoded by employing multiple levels of LSTMs on top of each other as shown in Fig. 2.

Another key objective of constructing deep architectures is to achieve better generalization. Since the inputs are processed with many nonlinear layers, these models are more robust against overfitting. This approach is followed in recently developed architectures and has shown promising results (Graves et al. 2013a; Hannun et al. 2014; Sak et al. 2014). At level l the activations of the LSTM are computed in the same way as mentioned in Sect. 3.1. However, in this case the input will be comprised of activations of layer $l-1$. In case of $l=1$, the input sequence h_t^{l-1} is equal to x_t. After computing hidden state of top most layer L, the output can be obtained as:

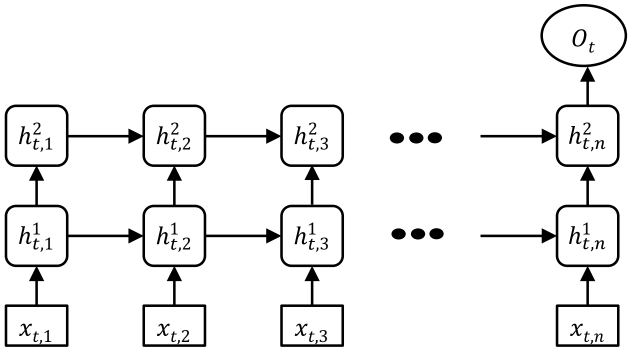


Fig. 2 Stacked LSTMs

$$O_t = \sigma\left(W^L_{hy}\, h^L_t\right). \tag{9}$$

4.1 Bidirectional LSTMs

The bidirectional-RNN (BRNN) is turning-out to be the most effective RNN architecture for acoustic modeling, since it preserves both past and future context to predict the present unlike conventional RNN which only preserves the history (Chan and Lane 2015; Graves et al. 2013a; Schuster and Paliwal 1997). Based on the results of previous work showing the effectiveness of BLSTMs, we adopted this architecture for modeling Urdu speech waveforms. In this architecture, the activations of hidden state can be calculated through forward (h_t^f) and backward (h_t^b) recurrence as:

$$h_t^f = o_t^f \odot \tanh\left(c_t^f\right), \tag{10}$$

$$h_t^b = o_t^b \odot \tanh\left(c_t^b\right). \tag{11}$$

It is worth mentioning that h_t^f will be computed sequentially from $t = 1$ to $t = T^i$ for ith utterence and h_t^b in reverse from $t = T^i$ to$t = 1$. After computing forward and backward activations, the next layer of the model takes both h_t^f and h_t^b as input and produce final activations as:

$$h_t = \sigma\left(W^f_{hy}h^f_t + W^b_{hy}h^b_t\right). \tag{12}$$

Illustration of the model in shown in Fig. 3.

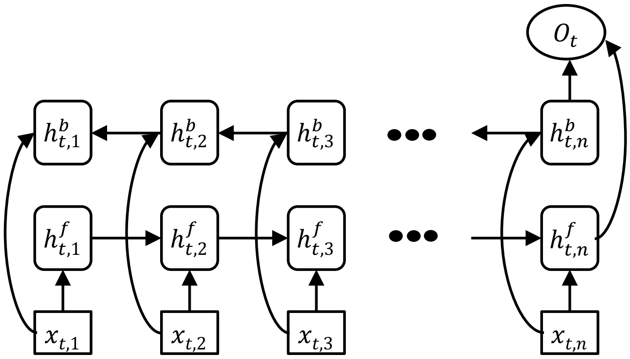


Fig. 3 Bidirectional LSTMs

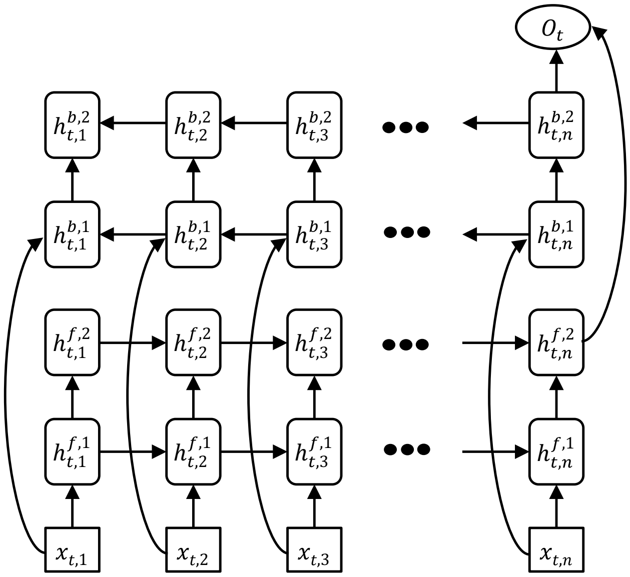


Fig. 4 Deep bidirectional LSTMs

5 Deep bidirectional LSTMs

We propose a novel architecture of deep bidirectional LSTMs (DBLSTMs) where multiple LSTM layers are stacked to compute forward and backward activations (given in Eqs. 10 and 11) before computing final representations (given in Eq. 12). The objective is to extract long term high-level (i.e., abstract) representations of historical and future context before aggregating them to capture full range of temporal dependencies. The architecture is shown in Fig. 4.

The key difference between our and recently proposed architectures (Chan and Lane 2015) is that they have used feed-forward networks to model temporal dependencies between given range of elements before using BLSTMs while we have replaced feed-forward networks with LSTM layers. The main advantage of the proposed architecture is that we are allowing the model to capture long range temporal contexts rather than fixed size limited context. By employing more hidden layers, we are aiming to model temporal dependencies at higher timescale. Another architecture of DBLSTMs is proposed in (Chan and Lane 2015) where DBLSTMs are used for modeling states of the HMMs. In contrast, our proposed model is end-to-end trainable RNN architecture.

6 Experimental setup

6.1 Data collection and representation

We employed an isolated words dataset of 2500 Urdu audio samples readily available on Kaggle (https://www.kaggle.com/hazrat/urdu-speech-dataset). The development process of the dataset is explained in details in (Ali et al. 2016). The lexicon for dataset contains most frequently spoken 250 words (including digits from 0 to 9). The list of these words is provided by center of language engineering.[3] To utter the words, ten speakers from different geographic locations and ages are employed. The recording is conducted by using Sony Linear PCM Recorder. The recorded files are stored with sampling rate of 16,000 Hz in .wav format. Average duration for each recording is half a second. The dataset is organized as a master directory containing ten subdirectories corresponding to each speaker. Each sub-directory comprises of 250 acoustic records in .wav format. The information about the speaker is encoded with six letters (e.g., AKMNG2) and used as a name the sub-directory as: first two letters represent name of the person, third letter symbolize gender of the speaker, fourth letter denotes whether the speaker is native or non-native and finally the last two letters represent the age of the person.

We have closely followed the experimentation procedure of (Ali et al. 2016) to be comparable with baseline results. We randomly partitioned the dataset into train and test sets respectively with the ratio of 7:3. The utterances are preprocessed with Mel-Frequency Cepstrum Coefficient (MFCC) technique by employing python-speech-features toolkit.[4] The parameters that are optimized through empirical validation include frame duration (Tw), number of filterbank channels (M), number of cepstral coefficients (C) and cepstral sine lifter parameters (L). Results of this evaluation are shown in Fig. 5. The frame shift is kept half of the frame

[3] "Center for Language Engineering" [Online]. Available: http://www.cle.org.pk.

[4] "Python_speech_features toolkit" [Online]. Available: https://python-speech-features.readthedocs.io/en/latest/.

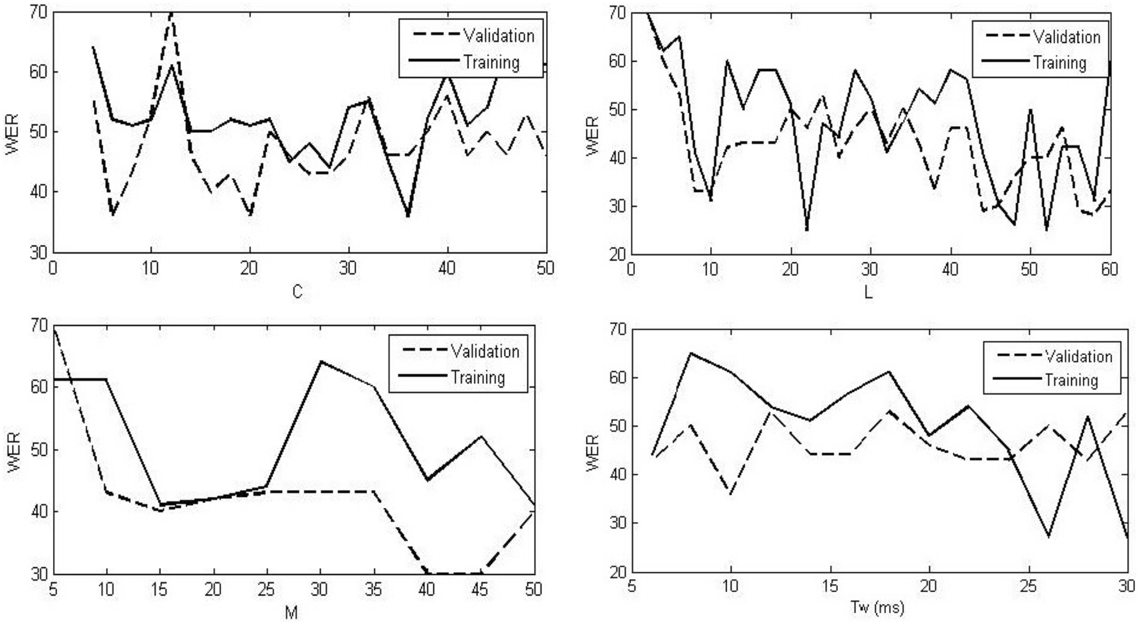


Fig. 5 Speech recognition results of single layer (with 50 hidden units) LSTM network with different values of number of cepstral coefficients (C), cepstral sine lifter parameter (L), number of filter bank channels (M) and frame duration. The efficacy of these parameters is measured in terms of % of word error rate

duration while experimenting with frame duration. A single parameter is optimized at a time while using either default or optimized values for other parameters. The experimentations are performed with baseline single layer LSTMs. As the result of this analysis, we have used a frame size of 10 ms with 5 ms of frame shift, 40 filterbank channels, 20 cepstral coefficients and 58 cepstral sine lifter parameters.

6.2 Network architectures

We have explored three LSTM architectures including deep, bidirectional and deep bidirectional LSTMs for Urdu acoustic modeling. For implementation of these networks, we employed Keras toolkit (Chollet 2015) which is a high-level API for implementing neural network, built on the top of Python and Tensorflow. Further for training on GPU, we employed Google s’ cloud services known as Colaboratory[5] which provides GPU based Jupyter notebook environment.

6.2.1 Input and output layers

The sizes of input and output layer in the entire networks is taken 13 (one for each coefficient in MFCC time window) and 100 (one for each word) respectively. As the size of input layer depends on number of coefficients in a time window which depends on the size of the window, this parameter is empirically selected through experimentations. The input layer is fully connected to recurrent layer and the recurrent layer is fully connected to itself and output layer. For some of the complex network topologies (e.g., multiple LSTM cells per block, pipehole connection, direct feedback connection between input and output layer), we relied on the results of (Greff et al. 2016) where they have shown that such complex topologies reduces the performance of the networks.

6.2.2 Hidden layers

The choice of number of units at hidden layer for single layer LSTM is performed through an empirical validation. The units are tested in order of multiple of tens from interval 20–400 and results are shown in Fig. 6. The number

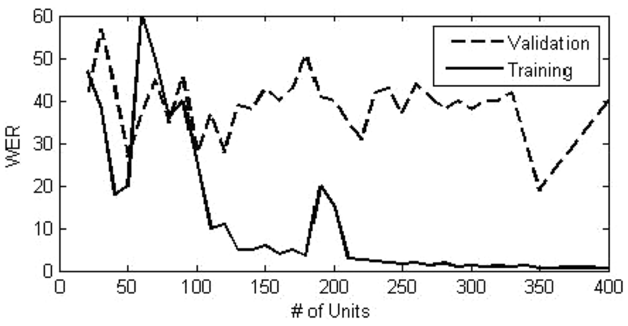
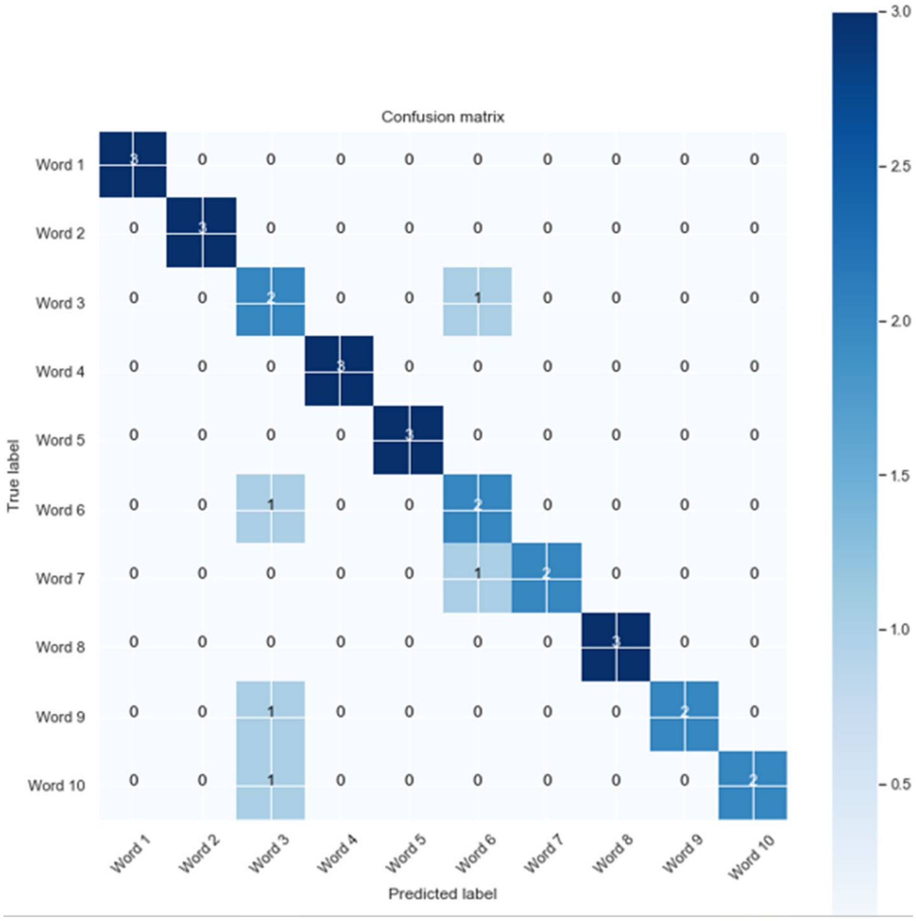


Fig. 6 Number of hidden units versus % of word error rate

[5] “Colaboratory”, Available: https://colab.research.google.com/notebooks/welcome.ipynb#recent=true.

Fig. 7 Confusion matrix of 2L-BLSTM on test data



of hidden layers in deep LSTMs is also validated empirically. In this regard, the number of hidden units at each layer is found by dividing the number of units of single layer LSTM with depth of the network. This procedure is usually adopted while performing comparison between different architectures. We further fine-tuned the choice of hidden units for deep LSTMs by testing different variations around the obtained number. The result of this analysis is shown in Fig. 7.

6.2.3 Architectures

In this study, we analyzed the efficacy of four commonly used LSTM architectures: single layer unidirectional LSTM, two layer unidirectional LSTM, single layer bidirectional LSTM and two layer bidirectional LSTM. The objective of testing the single layer and two layer architectures was to analyze whether the performance of the networks improve while adding the hierarchical depth while keeping number of parameters fixed. The goal of testing bidirectional architectures was to observe that whether the future contexts also matters or simply historical context is enough for capturing the pattern of acoustic waveform. Lastly, we observed the joint impact of historical and future contexts along with depth by using deep bidirectional LSTMs. The specification of these model architectures is as follows:

- Single layer LSTMs (1L-LSTMs) with 350 hidden units
- Two layer LSTMs (2L-LSTMs) with 150 hidden units at each layer
- Bidirectional LSTM (1L-BLSTMs) with 145 hidden units at each layer
- Deep bidirectional LSTMs (2L-BLSTMs)

6.2.4 GRU architectures

In this type, the main objective of the activation function is to simplify the complexity of LSTM yet maintaining the core characteristic of LSTM (i.e. linear dependencies between recurrent layers) (Chung et al. 2014). The separate memory cell is no longer maintained. The activation of the hidden state is computed as a weighted sum of partial activations at time $t\hat{h}_t$ and hidden state at time $t - 1$, h_{t-1}.

$$h_t = \left(1 - z_t\right)h_{t-1} + z_t\hat{h}_t, \tag{13}$$

where

$$\hat{h}_t = \tanh\left(W_h \mathrm{x}_t + r_t \odot \left(U h_{t-1}\right)\right), \tag{14}$$

$$z_t = \sigma\left(W_z \mathrm{x}_t + U_z h_{t-1}\right), \tag{15}$$

$$r_t = \sigma\left(W_r \mathrm{x}_t + U_r h_{t-1}\right), \tag{16}$$

where z_t is known as update gate and determines the participation of partial activations and previous hidden state in the computation of actual activations. r_t is called reset gate and determined the participation of previous hidden state in the computation of partial activations. W and U represent weight matrices for input sequence (x_t) and previous hidden state (h_{t-1}). In deep GRU (DGRUs) architectures, each layer l (where $l \geq 2$) takes activations of below layer $l-1$ as input; therefore $\mathbf{x}_t$ is replaced with h_t^{l-1}. In case of bidirectional GRU (BGRU), the activations (given in Eq. 13) is computed twice; in forward direction (i.e., h_t^f) as well as in backward direction (i.e., h_t^b), before computing the final representations by using Eq. 12.

6.2.5 Network training

In order to predict probabilities for transcriptions, we have employed standard softmax function where the probability of kth character is computed as:

$$O_{t,k} = \hat{y}_{t,k} = p\left(W_t|\mathrm{x}_t\right) = \frac{\exp\left(W_k h_t + b^L\right)}{\sum_j \exp\left(W_j\, h_t + b^L\right)}. \tag{17}$$

On computing $p(W_t|\mathrm{x}_t)$, we will use cross entropy objective (or loss) function $\mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y})$ to measure the error in prediction. The gradient of the objective function $\Delta_{\hat{y}}\mathcal{L}(\hat{\boldsymbol{y}}, \boldsymbol{y})$ is calculated with respect to training labels $\boldsymbol{y}$. After computing gradient of loss function, the gradients for model parameters are calculated for the network using back-propagation through time algorithm. In order to train the model parameters, the stochastic gradient decent algorithm is applied.

The networks are trained with truncated back-propagation through time (TBTT) method. This method restrains the backpropagation to the specified time delay and has already been used for acoustic modeling (Williams and Peng 1990). We restricted the backpropagation till the start of the utterance; the internal state of the networks (i.e., activation of hidden state $\mathrm{h_t}$ in SRN and GRU and memory cell content $\mathrm{c_t}$ in LSTM) is reset at the end of each utterance. The main reason to use TBTT is to process utterances independently. Since the experimentations are performed on isolated words, the contextual information of other utterances may not be required. The initialization of weights for all the networks is performed with a uniform distribution in the range (− 0.02–0.02). The learning rates are decayed exponentially during the training process. As the main objective of the study is to analyze the effects of varying architectures, single learning algorithm is employed for training the networks. However, different learning algorithms may affect the performance of different architectures and can show the performance improvements.

Table 1 Recognition rate of LSTM architectures

Architectures	Training	Testing
1L-LSTMs	1	0.68
2L-LSTMs	0.92	0.76
1L-BLSTMs	0.88	0.76
2L-BLSTMs	0.88	0.83

7 Results

7.1 Comparison between LSTM architectures

The performance of contemporary LSTM architectures for Urdu acoustic modeling is presented in Table 1. The deep LSTM architecture with two hidden layers has performed better than single layer LSTMs. However, as described earlier, adding more than 2 hidden layers can cause sharp decrease in the performance of the networks with number of layers (see Fig. 7). The bidirectional LSTMs has shown better performance over simple LSTMs. The proposed model has shown superior performance over other architectures.

7.2 Comparison with GRUs

We have chosen the number of hidden units in order to equalize the number of parameters with corresponding LSTM architecture. This procedure is adopted in (Chung et al. 2014) while comparing LSTMs with GRUs. Following architectures are used for evaluating the performance of GRUs in comparison with LSTMs:

- Single layer GRU (1L-GRUs) with 385 hidden units and approximately 0.5 million parameters. This architecture is approximately equivalent in terms of parameters with 1L-LSTMs with 350 units.
- Two layer GRU (2L-GRUs) with 195 hidden units and approximately 0.35 million parameters. This architecture is equivalent in terms of parameters with 2L-LSTMs.
- Bidirectional GRU (1L-BGRU) with 190 hidden units and approximately 0.35 million parameters. This architecture is equivalent to 1L-BLSTM.

In Table 2, the results of GRU architectures are summarized. It can be observe that GRU based architectures

Table 2 Recognition rate of GRU architectures

Architectures	Training WER (%)	Testing WER (%)
1L-GRUs	0.92	0.68
1L-BGRU	0.88	0.76
2L-GRUs	0.88	0.76

Table 3 Recognition rate of 2L-BLSTM, SVM, RF and LDA

Word No.	Recognition rate (2L-BLSTM)	Recognition rate (SVM) []	Recognition rate (RF) []	Recognition rate (LDA) []
1	1	1	0.66	1
2	1	0.66	0.33	0.33
3	0.66	0.66	1	1
4	1	1	0.66	0.66
5	1	0.66	0.66	0.66
6	0.66	0.33	0.66	0.66
7	0.66	0.66	0.66	0.66
8	1	1	0	0
9	0.66	0.66	0.33	0.33
10	0.66	0.66	1	1

perform lower than LSTM based architectures in this comparison. We hypothesized that LSTM is gaining this advantage due to maintaining a memory cell in order to retain contextual information in more explicit manner. However, in terms of convergence time, GRU architectures are observed to have advantage over LSTM architectures.

7.3 Comparison with benchmark results

In Table 3, best performing LSTM architecture (i.e., 2L-BLSTM) is compared with the benchmark results of three classifiers including support vector machine (SVM), random forest (RF) and linear discriminant analysis (LDA) as reported in (Ali et al. 2016). We have employed accuracy as metric for this comparison because benchmark results are reported on the basis of this metric. Three waveforms of each word from the test data is used for obtaining these results. The performance value 1 shows that all three waveforms are correctly classified and 0 shows that none of them is correctly classified. We have achieved an overall accuracy of 83% which is 10% higher than maximum achieved accuracy of SVM (i.e., 73%) in the previous study. The confusion matrix of the proposed 2L-BLSTM is shown in Fig. 7. It can be noticed that the model correctly classifies all three words in most of the cases.

8 Conclusion

In this paper, we presented a study of RNN architectures for Urdu acoustic modeling and proposed a novel acoustic modeling architecture. The studied architectures include deep, bidirectional, and deep bidirectional. We modeled the architectures with long short term memory and gated recurrent units. Results have shown that bidirectional LSTMs are best suited paradigm for this task. The GRU is found to be less effective than LSTMs in all architectures.

In future, we would also like to apply this model for large-vocabulary and continuous Urdu speech recognition. Another future work will be to employ the model with HMMs to analyze the effectiveness of hybrid BLSTM + HMM based approach. We are interested in analyzing the proposed model with different objective functions (such as connectionist temporal classification) and training algorithms to obtain performance improvements. An empirical study can also be made to analyze the effects of various regularization techniques to improve the performance of the system. Though bidirectional LSTMs has shown superior performance, they may not be useful in online settings since they require complete future context along with history. We are also interested to modify the proposed architecture to enable it to work in online settings. Finally, to deal with gradient vanishing and explosion problem between timescales, advanced network architectures such as highway networks and residual networks can also be employed to take full advantage of deep architectures.

References

Ahad, A., Fayyaz, A., & Mehmood, T. (2002). Speech recognition using multilayer perceptron. In *Proceedings of IEEE students conference* (Vol. 1, pp 103–109).

Ali, H., Ahmad, N., & Hafeez, A. (2016). Urdu speech corpus and preliminary results on speech recognition. In *International conference on engineering applications of neural networks* (pp 317–325). New York: Springer.

Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., & Chen, J. (2016). Deep speech 2: End-to-end speech recognition in English and Mandarin. In *International conference on machine Learning* (pp 173–182).

Ashraf, J., Iqbal, N., Khattak, N. S., & Zaidi, A. M. (2010). Speaker independent Urdu speech recognition using HMM. In *7th IEEE international conference on informatics and systems (INFOS)* (pp 1–5).

Azam, S. M., Mansoor, Z. A., Mughal, M. S., & Mohsin, S. (2007). Urdu spoken digits recognition using classified MFCC and backpropgation neural network. In *IEEE conference on computer graphics, imaging and visualisation* (pp 414–418).

Bahdanau, D., Chorowski, J., Serdyuk, D., Brakel, P., & Bengio, Y. (2016). End-to-end attention-based large vocabulary speech recognition. In *IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 4945–4949). IEEE.

Chan, W., Jaitly, N., Le, Q., & Vinyals, O. (2016). Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 4960–4964). IEEE.

Chan, W., & Lane, I. (2015), Deep recurrent neural networks for acoustic modelling. arXiv Preprint arXiv:1504.01482.

Chiu, C. C., Sainath, T. N., Wu, Y., Prabhavalkar, R., Nguyen, P., Chen, Z., & Jaitly, N. (2017). State-of-the-art speech recognition with sequence-to-sequence models. arXiv Preprint arXiv:1712.01769.

Chollet, F. (2015). Keras.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv Preprint arXiv:1412.3555.

Graves, A., & Jaitly, N. (2014). Towards end-to-end speech recognition with recurrent neural networks. In *Proceedings of the 31st international conference on machine learning (ICML-14)* (pp 1764–1772).

Graves, A., Mohamed, A. R., & Hinton, G. (2013a). Speech recognition with deep recurrent neural networks. In *IEEE international conference on acoustics, speech and signal processing* (pp 6645–6649).

Graves, A., Jaitly, N., & Mohamed, A. R. (2013b). Hybrid speech recognition with deep bidirectional LSTM. In *IEEE workshop on automatic speech recognition and understanding (ASRU)*, pp 273–278.

Graves, A., & Schmidhuber, J. (2009). Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in neural information processing systems* (pp 545–552).

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2016). LSTM: A search space odyssey. In IEEE transactions on neural networks and learning systems.

Hannun, A., Case, C., Casper, J., Catanzaro, B., Diamos, G., Elsen, E., Prenger, R., Satheesh, S., Sengupta, S., Coates, A., & Ng, A. Y. (2014). Deep speech: Scaling up end-to-end speech recognition. arXiv Preprint arXiv:1412.5567.

Hasnain, S. K., & Awan, M. S. (2008). Recognizing spoken Urdu numbers using Fourier descriptor and neural networks with Matlab. In *Second international IEEE conference on electrical engineering* (pp 1–6).

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine, 29*(6), 82–97.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.

Juang, B. H., & Rabiner, L. R. (1991). Hidden Markov models for speech recognition. *Technometrics, 33*(3), 251–272.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp 1097–1105).

Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). A critical review of recurrent neural networks for sequence learning. arXiv Preprint arXiv:1506.00019.

Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., & Khudanpur, S. (2010). Recurrent neural network based language model. *Interspeech, 2*, 3.

Pascanu, R., Gulcehre, C., Cho, K., & Bengio, Y. (2013). How to construct deep recurrent neural networks. arXiv Preprint arXiv:1312.6026.

Pascanu, R., Mikolov, T., & Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *International conference on machine learning* (pp 1310–1318).

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE, 77*(2), 257–286.

Rao, K., & Sak, H. (2017). Multi-accent speech recognition with hierarchical grapheme based models. In *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, (pp. 4815–4819). IEEE.

Sak, H., Senior, A., & Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth Annual Conference of the International Speech Communication Association*.

Sak, H., Senior, A., Rao, K., & Beaufays, F. (2015). Fast and accurate recurrent neural network acoustic models for speech recognition. arXiv Preprint arXiv:1507.06947.

Sarfraz, H., Hussain, S., Bokhari, R., Raza, A. A., Ullah, I., Sarfraz, Z., Pervez, S., Mustafa, A., Javed, I., & Parveen, R. (2010). Large vocabulary continuous speech recognition for Urdu. In *Proceedings of the 8th ACM international conference on frontiers of information technology* (p 1).

Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing, 45*(11), 2673–2681.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014), Sequence to sequence learning with neural networks. In *Advances in neural information processing systems* (pp 3104–3112).

Williams, R. J., & Peng, J. (1990). An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural computation, 2*(4), 490–501.

Yu, D., & Li, J. (2017). Recent progresses in deep learning based acoustic models. *IEEE/CAA Journal of Automatica Sinica, 4*(3), 396–409.

Zweig, G., Yu, C., Stolcke, D. J., A. (2017). Advances in all-neural speech recognition. In: *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 4805–4809). IEEE.