Recent Trends in Application of Neural Networks to Speech Recognition

G Gnaneswari,
Research Scholar
Jain University,
Bangalore, India

S R VijayaRaghava,
Research Scholar
Jain University,
Bangalore, India.

A K Thushar,
Research Scholar
Jain University,
Bangalore, India

Dr.S.Balaji,
Centre for Emerging Technologies,
Jain Global Campus,
Jain University,
Bangalore, India

Abstract: In this paper, we review the research work that deal with neural network based speech recognition and the various approaches they take to bring in accuracy. Three approaches of speech recognition using neural network learning models are discussed: (1) Deep Neural Network(DNN) - Hidden Markov Model(HMM), (2) Recurrent Neural Networks(RNN) and (3) Long Short Term Memory(LSTM). It also discusses how for a given application one model is better suited than the other and when should one prefer one model over another.A pre-trained Deep Neural Network - Hidden Markov Model hybrid architecture trains the DNN to produce a distribution over tied triphone states as its output. The DNN pre-training algorithm is a robust and often a helpful way to initialize deep neural networks generatively that can aid in optimization and reduce generalization error. Combining recurrent neural nets and HMM results in a highly discriminative system with warping capabilities. To evaluate the impact of recurrent connections we compare the train and test characteristic error rates of DNN, Recurrent Dynamic Neural Networks (RDNN), and Bi-Directional Deep Neural Network (BRDNN) models while roughly controlling for the total number of free parameters in the model. Both variants of recurrent models show substantial test set characteristic error rate improvements over the non-recurrent DNN model. Inspired from the discussion about how to construct deep RNNs, several alternative architectures were constructed for deep LSTM networks from three points: (1) input-to-hidden function, (2) hidden-to-hidden transition and (3) hidden-to-output function. Furthermore, some deeper variants of LSTMs were also designed by combining different points.

Keywords — Neural *Networks, Speech Recognition, Deep Neural Networks(DNN), Hidden Markov Model(HMM), Recurrent Neural Networks(RNN), Long Short Term Memory(LSTM)*

*****

I. INTRODUCTION

In speech recognition,the deep form of artificial neural networks had been in research for many years now. Generally, both acoustic model and language model speech recognition algorithms are studied using HMMs. DNN uses multiple hidden layers between the input and the output layers. The main use of RNN is to identify latent temporal dependencies and use this information to perform the task of speech recognition. LSTM is an variant of RNN. The progress of research in speech recognition were speedup DNN training and decoding, Sequence discriminative training of DNNs, Feature processing by deep, Adaptation of DNNs and of related deep models, Multi-task and transfer learning by DNNs and related deep models, Convolution neural networks and design them to best exploit domain knowledge of speech, Recurrent neural network and its rich LSTM variants. [1]
Recentlystochastic modeling has been used to solve the speech recognition problem like Deep Neural Networks (DNN)-Hidden Markov Modeling (HMM) approach. Currently, researches on the speech recognition involve recognizing continuous speech from a large vocabulary using HMMs, Artificial Neural Networks (ANNs), or a hybrid form. These techniques are briefly explained and analyzed below. The general-purpose speech recognition systems are based on algorithms, the most extensively used ones are Hidden Markov Model (HMM), Recurrent Neural Networks (RNN), and Long Short Term Memory (LSTM).
The paper is organized as follows: Section 2 describes neural networks. In Section 3 the various neural network models such as DNN-HMM model, RNN model and LSTM model, their limitations and their advantages are extensively described. Section4 gives an overview of the modelsbased on character error rate(CER) and in Section 5, the lowest CER in every model is highlighted and presented as the conclusion.

II. NEURAL NETWORKS

Neural network models in artificial intelligence are usually referred to as ANNs; these are essentially simple mathematical models defining a function $f:X\rightarrow Y$or a distribution over Xor both $X$and Y, but sometimes models are also intimately associated with a particular learning algorithm or learning rule. A common use of the phrase ANN model really means the definition of a class of such functions (where members of the class are obtained by varying parameters, connection weights, or specifics of the architecture such as the number of neurons or their connectivity).

The word network in the term 'artificial neural network' refers to the interconnections between the neurons in the different layers of each system. An example system has three layers.

The first layer has input neurons which send data via synapses to the second layer of neurons, and then via more synapses to the third layer of output neurons. More complex systems will have more layers of neurons with some having increased layers of input neurons and output neurons. The synapses store parameters called "weights" that manipulate the data in the calculations.

An ANN is typically defined by three types of parameters: the interconnection pattern between the different layers of neurons, the learning process for updating the weights of the interconnections and the activation function that converts a neuron's weighted input to its output activation.

Mathematically, a neuron's network function $f(x)$is defined as a composition of other functions $g_i(x)$, which can further be defined as a composition of other functions. This can be conveniently represented as a network structure, with arrows depicting the dependencies between variables. A widely used type of composition is the nonlinear weighted sum,where $f(x)=K\left(\sum_i w_i g_i(x)\right)$, where K is a predefined function. It will be convenient to refer to a collection of functions $g_i$as simply a vector $g=(g_1,g_2,\ldots,g_n)$.

The first view is the functional view: the input $x$is transformed into a 3-dimensional vector h, which is then transformed into a 2-dimensional vector g, which is finally transformed into f. This view is most commonly encountered in the context of optimization. The second view is the probabilistic view: the random variable $F=f(G)$depends upon the random variable $G=g(H)$, which depends upon $H=h(X)$, which depends upon the random variable X. This view is most commonly encountered in the context of graphical models. The two views are largely equivalent. In either case, for this particular network architecture, the components of individual layers are independent of each other (e.g., the components of $g$are independent of each other given their input h). This naturally enables a degree of parallelism in the implementation.

Networks such as the previous one are commonly called feed forward, because their graph is a directed acyclic graph. Networks with cycles are commonly called recurrent.

III. NEURAL NETWORK MODELS

Both acoustic modeling and language modeling are important parts of modern statistically-based speech recognition algorithms. The following section discusses various approaches for Speech Recognition.

3.1. Deep Neural Networks-Hidden Markov Model

HMMs have been the dominant technique for Large Vocabulary Speech Recognition (LVSR) for at least two decades [2]. An HMM is a generative model in which observable acoustic features are assumed to be generated from a hidden Markov process that transitions between states S = $\{s_1, s_2 \ldots, s_k\}$. The key parameters in the HMM are the initial state probability distribution f = $\{p(q_0=s_i)\}$, , where qt is the state at time t; the transition probabilities $a_{ij}= p(q_t=s_j|qt-1=si)$; and a model to estimate the observation probabilities $p(x_t|s_i)$. In conventional HMMs used for Automatic Speech Recognition (ASR), the observation probabilities are modeled using Gaussian Mixture Models (GMM). These GMM-HMMs are typically trained to maximize the likelihood of generating the observed features.

Despite all their advantages, GMMs have a serious shortcoming that they are statistically inefficient for modeling data that lie on or near a non-linear manifold in the data space. For example, modeling the set of points that lie very close to the surface of a sphere only requires a few parameters using an appropriate model class, but it requires a very large number of diagonal Gaussians or a fairly large number of full-covariance Gaussians. Speech is produced by modulating a relatively small number of parameters of a dynamical system and this implies that its true underlying structure is much lower-dimensional than is immediately apparent in a window that contains hundreds of coefficients. Therefore, other types of models may work better than GMMs for acoustic modeling if they can more effectively exploit information embedded in a large window of frames. ANNs trained by back-propagating error derivatives have the potential to learn much better models of data that lie on or near a non-linear manifold. Over the last few years, advances in both machine learning algorithms and computer hardware have led to more efficient methods for training DNNs that contain many layers of non-linear hidden units and a very large output layer. The large output layer is required to accommodate the large number of HMM states that arise when each phone is modeled by a number of different “triphone”- HMMs that take into account the phones on either side. Even when many of the states of these triphone HMMs are tied together, there can be thousands of tied states. Using the new learning methods, several different research groups have shown that DNNs can outperform GMMs at acoustic modeling for speech recognition on a variety of datasets including large datasets with large vocabularies.

A DNN is a feed-forward while an ANN has more than one layer of hidden units between its inputs and its outputs [3]. Each hidden unit, j, typically uses the logistic function to map its total input from the layer below, x_j , to the scalar state, y_j that it sends to the layer above

$$y_j = \log(x_j) = \frac{1}{1+e^{-x_j j}}$$

$$x_j = b_j + \sum_i y_i w_{ij}$$

Where b_j is the bias of unit j, i is an index over units in the layer below, and w_{ij} is the weight on a connection to unit j from unit i in the layer below. For multiclass classification, output unit j converts its total input, x_j, into a class probability, p_j, by using the “softmax” non-linearity

$$\rho_j = \frac{\exp(x_j)}{\sum_k \exp(x_k)}$$

where k is an index over all classes.

DNN’s can be discriminatively trained by back-propagating derivatives of a cost function that measures the discrepancy between the target outputs and the actual outputs produced for each training case. When using the softmax output function, the natural cost function C is the cross-entropy between the target probabilities d and the outputs of the softmax, p

$$C = -\sum_j d_j \log \rho_j$$

where the target probabilities, typically taking values of one or zero, are the supervised information provided to train the DNN classifier.

DNN’s with many hidden layers are hard to optimize. Gradient descent from a random starting point near the origin is not the best way to find a good set of weights and unless the initial scales of the weights are carefully chosen, the back-propagated gradients will have very different magnitudes in different layers. In addition to the optimization issues, DNNs may generalize poorly to held out test data. DNNs with many hidden layers and many units per layer are very flexible models with a very large number of parameters. This makes them capable of modeling very complex and highly non-linear relationships between inputs and outputs. This ability is important for high-quality acoustic modeling, but it also allows them to model spurious regularities that are an accidental property of the particular examples in the training set, which can lead to severe overfitting. Weight penalties or early-stopping can reduce the overfitting but only by removing much of the modeling power. Very large training sets can reduce overfitting while preserving modeling power, but only by making training very computationally expensive. What we need is a better method of using the information in the training set to build multiple layers of non-linear feature detectors.

3.2. Recurrent Neural Networks

Recurrent Neural Network is an ANN that is represented in form of a directed cycle where each node is connected to the other. When communication takes place between any two units they become dynamic. They also use the internal memory like the feed forward networks to process sequence of inputs that arbitrary. This makes them a popular choice for tasks like speech recognition. The feature of a RNN is that the activations flow round in a loop since the network contains at least one feed-back connection enabling the networks to do temporal processing and learn sequences, like sequence recognition/reproduction.

Recurrent neural network architectures are of different types like a standard Multi-Layer Perceptron (MLP) plus added loops. Due to the powerful non-linear mapping capabilitiesMLP’s has some memory. Some have more uniform structures and have astochastic activation functions.

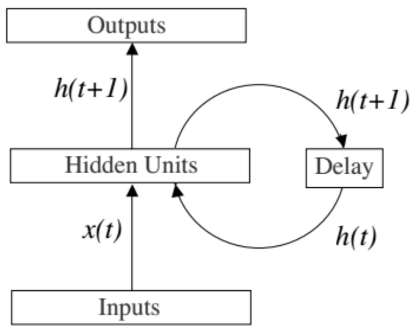

Fig. 1: Recurrent Neural Network [4]

The architecture of recurrent DNN used is shown in Fig. 1. The fundamental structure is a feed forward DNN. They have input and output layerand certain hidden layer with full recurrent connections.

Learning in RNN can be achieved even for simple architectures. The deterministic activation functions and similar gradient descent procedures can be used with those simple architectures. Back-propagation algorithm for feed forward networks will provide optimal results.

In Fig.1 we have used the simplest form of fully recurrent neural network that is an MLP with the previous set of hidden unitactivations (h(t)), feeding back into the network along with the input (h(t+1)). Here the time t has to be discretized, with the activations updated at each time step. The time scale might correspond to the operation of real neurons, or for artificial systems any time step size appropriate for the given problem can be used. A delay unit needs to be introduced to hold activations until they are processed at the next time step [4].

3.3. Long Short Term Memory

LSTM is an RNN architecture that contains LSTM blocks instead of, or in addition to, regular network units [5]. An LSTM block may be described as a "smart" network unit that can remember a value for an arbitrary length of time. An LSTM block contains gates that determine when the input is significant enough to remember, when it should continue to remember or forget the value, and when it should output the value. LSTMs have been successfully used for many sequence labeling and sequence prediction tasks. In the LSTM architecture, the recurrent hidden layer consists of a set of recurrently connected subnets known as “memory blocks”. Each memory block contains one or more self-connected memory cells and three multiplicative gates to control the flow of information. In each LSTM cell, the flow of information into and out of the cell is guarded by the learned input and output gates. Later, in order to provide a way for the cells to reset themselves, the forget gate is added [6]. A conventional LSTM can be defined as follows:

Given an input sequence x = (x_1, x_2 , . . . , x_T), a conventional RNN computes the hidden vector sequence h = (h_1, h_2, . . . , h_T) and output vector sequence y = (y_1, y_2, . . . , y_T) from t = 1 to T as follows:

$$h_t = H(W_{xh}x_t + W_{hh}h_t - 1 + b_h)$$
$$t = W_{hy}h_t + b_y$$

Where, the W denotes weight matrices, the b denotes bias vectors and H(·) is the recurrent hidden layer function.

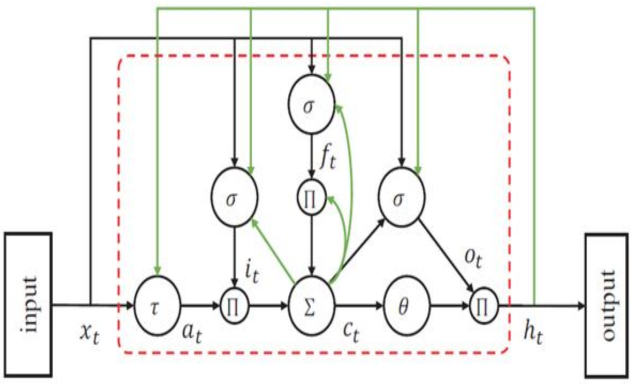

Fig. 2: The architecture of an LSTM network with one memory block, where green lines are time-delayed connections.

The advantages and disadvantages of LSTM are as listed below: [5].

Limitations of LSTM:

- The back propagation of LSTM stores only one of the inputs which will not reduce the error which will in turn not incrementally reduce the error even by solving the sub goal. This can be reduced by using full gradient.
- But full gradient is not recommended because 1) Its more complex 2) Error flow can be seen only when truncated LSTM is used 3)Full BPTT will not outperform truncated BPTT.
- Since 1 hidden unit is replaced by 3 units, the weight factor increases to 32. Thus each memory cell needs two additional cell blocks.
- LSTM has same problems as that of feed forward nets because it behaves like feedforward neural network trained by backpropagation neural network to see the complete input string.
- Counting the time steps is a practical problem in all the gradient based approaches; Even in LSTM the task is simpler when a signal occurred like 5 steps ago and was a problem if it occurred say a 100 steps ago.

Advantages of LSTM

- The constant error back propagation within memory cells results in LSTM's ability to bridge very long time lags in case of problems similar to those discussed above. For long time lag problems such as those discussed in this review, LSTM can handle noise, distributed representations, and continuous values. In contrast to finite state automata or hidden Markov models LSTM does not require an a priori choice of a finite number of states. In principle it can deal with unlimited state numbers.
- For problems discussed in this paper, LSTM generalizes well even if the positions of widely separated, relevant inputs in the input sequence do not matter. Unlike previous approaches, ours quickly learns to distinguish between two or more widely separated occurrences of a particular element in an input sequence, without depending on appropriate short time lag training exemplars.
- There appears to be no need for parameter ne tuning. LSTM works well over a broad range of parameters such as learning rate, input gate bias and output gate bias. For instance, to some readers the learning rates used in our experiments may seem large. However, a large learning rate pushes the output gates towards zero, thus automatically countermanding its own negative effects.
- The LSTM algorithm's update complexity per weight and time step is essentially that of BPTT, namely O (1). This is excellent in comparison to other approaches such as RTRL. Unlike full BPTT, however, LSTM is local in both space and time.

IV. OVERVIEW OF THE APPLICATIONS

4.1 Deep Neural Network – Hidden Markov Models

Deep Belief Networks with multiple hidden layers can learn a hierarchy of nonlinear feature detectors that can capture complex statistical patterns in data. The deep belief net training algorithm suggested in [7] first initializes the weights of each layer individually in a purely unsupervised1 way and then fine-tunes the entire network using labeled data.

Using pre-training to initialize the weights of a deep neural network has two main potential benefits that have been discussed in the literature. In [8], evidence was presented that is consistent with viewing pre-training as a peculiar sort of data-dependent regularizer whose effect on generalization error does not diminish with more data, even when the dataset is so vast that training cases are never repeated. The regularization effect from using information in the distribution of inputs can allow highly expressive models to be trained on comparably small quantities of labeled data. Additionally, [8], [9] and others have also reported experimental evidence consistent with pre-training aiding the subsequent optimization, typically performed by stochastic gradient descent. Thus, pre-trained neural networks often also achieve lower training error than neural networks that are not pre-trained (although this effect can often be confounded by the use of early stopping).

The Bing mobile voice search application allows users to do US-wide business and web search from their mobile phones via voice. The business search dataset used in our experiments was collected under real usage scenarios in 2008, at which time the application was restricted to do location and business lookup. All audio files collected were sampled at 8 kHz and encoded with the GSM codec. Some examples of typical queries in the dataset are “Mc-Donalds,” “Denny’s restaurant,” and “oak ridge church.” This dataset contains all kinds of variations: noise, music, side-speech, accents, sloppy pronunciation, hesitation, repetition, interruption, and different audio channels.

The sentence accuracy improves as more layers are added in the DNN-HMM [9]. When three hidden layers were used, the accuracy increased to 69.6%. The accuracy further improved to 70.2% with four hidden layers and 70.3% with five hidden layers. Overall, using the five hidden-layer models provides us with a 2.2% accuracy improvement over the single hidden-layer system when the same alignment is used. Although it is possible that using even more than five hidden layers would continue to improve the accuracy, percentage gain when compared to training effort is less.

Table 1: The Bing mobile voice search application results

Sl. No.	Model	CER (%)
1	DNN-HMM with 1 hidden layer and 1.5K hidden units	40.9
2	DNN-HMM with 1 hidden layer and 2K hidden units	31.9
3	DNN-HMM with 3 hidden layer and 2K hidden units	30.4
4	DNN-HMM with 4 hidden layer and 2K hidden units	29.8
5	DNN-HMM with 5 hidden layer and 2K hidden units	29.7

4.2 Recurrent Neural Networks

Combining recurrent neural nets and HMM results in a highly discriminative system with warping capabilities. By incorporating LSTM cells into recurrent neural nets even long time series can be modelled. Conventional methods were outperformed by the suggested tandem approach. [11]

To evaluate the impact of recurrent connections the train and test CERs of DNN, RDNN, and BRDNN models were compared while roughly controlling for the total number of free parameters in the model. The table below shows the results for each type of architecture. Both variants of recurrent models show substantial test set CER improvements over the non-recurrent DNN model.

In Table 2, all scores are word error rate(WER)/character error rate(CER) on the evaluation set; ‘LM’ is the Language model used for decoding and ‘14 Hr’ and ‘81 Hr’ refer to the amount of data used for training.

The performance for a DNN was reported of only 16.8M total parameters which are smaller than the total number of parameters used in both the RDNN and BRDNN models. They found that larger DNNs performed worse on the test set, suggesting that DNNs may be more prone to over-fitting for this task. Although the BRDNN has fewer parameters than the RDNN it performs better on both the training and test sets. Again this suggests that the architecture itself drives improved performance rather than the total number of free parameters. Conversely, because the gap between bi-directional recurrence and single recurrence is small relative to a non-recurrent DNN, on-line speech recognition using a singly recurrent network may be feasible without overly damaging performance. The RNN was first decoded with no dictionary or language model, using the space character to segment the character outputs into words, and thereby calculate the WER. The network was then decoded with a 146K word dictionary, followed by monogram, bigram and trigram language models. The dictionary was built by extending the default WSJ dictionary with 125K words using some augmentation rules. The language models were built on this extended dictionary, using data from the WSJ CD. The language model weight was optimized separately for all experiments. For the RNN experiments with no linguistic information, and those with only a dictionary, the beam search algorithm has been used for decoding. For the RNN experiments with a language model, an alternative method has been used, partly due to implementation difficulties and partly to ensure a fair comparison with the baseline system, an N-best list of at most 300 candidate transcriptions has been extracted from the baseline DNN-HMM and rescored by the RNN.

Table 2: Wall Street Journal Results (WSJ)

SYSTEM	LM	CER (%) - 14Hr	CER (%) - 81 HR
RNN-CTC	NONE	30.9	9.2
RNN-CTC	DICTIONARY	30	8
RNN-CTC	MONOGRAM	25.8	15.8
RNN-CTC	BIGRAM	15.5	10.4
RNN-CTC	TRIGRAM	13.5	8.7
RNN-WER	NONE	31.3	8.4
RNN-WER	DICTIONARY	31	7.3
RNN-WER	MONOGRAM	26	15.2
RNN-WER	BIGRAM	15.3	9.8
RNN-WER	TRIGRAM	13.5	8.2
BASELINE	NONE	—	—
BASELINE	DICTIONARY	56.1	51.1
BASELINE	MONOGRAM	23.4	19.9
BASELINE	BIGRAM	11.6	9.4
BASELINE	TRIGRAM	9.4	7.8
COMBINATION	TRIGRAM	—	6.7

4.3 Long Short Term Memory

Recently, LSTM networks have also been introduced on phoneme recognition task [12], robust speech recognition task [13], and large vocabulary speech recognition task [14], [15], [16], and shown state-of-the-art performances. Subsequently, the sequence discriminative training of LSTM networks is investigated in [17], and a significant gain was obtained. Long short-term memory (LSTM) based acoustic modeling methods have recently been shown to give state-of-the-art performance on some speech recognition tasks [18]. LSTMs and conventional RNNs have been successfully used for many sequence labeling and sequence prediction tasks. In language modeling, RNNs were used as generative models over word sequences, and remarkable improvements were achieved over the standard n-gram models. For handwriting recognition, LSTM networks have been applied for a long time, in which, the bidirectional LSTM (BLSTM) networks trained with connectionist temporal classification (CTC) has been demonstrated performing better than the HMM-based system. In speech synthesis, the BLSTM network has also been applied and a notable improvement was obtained. Inspired from the discussion about how to construct deep RNNs in, several alternative architectures were constructed for deep LSTM networks from three points: (1) input-to-hidden function, (2) hidden-to-hidden transition and (3) hidden-to-output function. Fig.3 details the various models applied for LSVR;

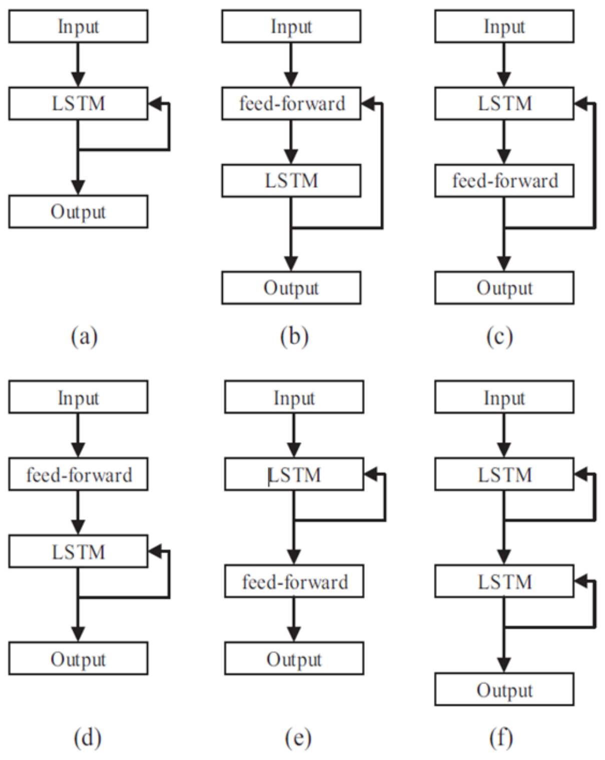

Fig. 3: Illustrations of different strategies for constructing LSTM based deep RNNs.

In fig.3, (a) a conventional LSTM; (b) a LSTM with input projection (LSTM-IP); (c) a LSTM with output projection. (LSTM-OP); (d) a LSTM with deep input-to-hidden function; (e) a LSTM with deep hidden-to-output function; (f) stacked LSTMs.

Furthermore, some deeper variants of LSTMs were also designed by combining different points. From the results listed in Table 3, the best performance can be obtained by combining the LSTM-OP and deep hidden-to-output function. These experimental results had revealed that deep LSTM networks benefit from the depth. Compared with the shallow LSTM network, a 13.98% relatively CER reduction can be obtained. Compared with the feed-forward DNNs, the deep LSTM networks can reduce the CER from 38.01% to 34.65%, which is an 8.87% relatively CER reduction.

Table 3: Speech recognition results of selected combinations for constructing deep LSTM networks

Sl. No.	Model Descriptions	CER (%)
1	LSTM	40.28
2	LSTM-IP	39.09
3	LSTM-OP	35.92
4	3-layer ReLU + LSTM	37.31
5	LSTM + 3-layer ReLU	37.16
6	Stack of LSTM (3-layer)	35.91
7	3-layer ReLU + LSTM-OP	36.73
8	LSTM-OP + 3-layer ReLU	34.65
9	Stack of LSTM-IP (3-layer)	35
10	Stack of LSTM-OP (3-layer)	34.84

V. CONCLUSIONS

The available models of Artificial Neural Networks have been applied to different problems in speech recognition with varying amount of success. The choice of algorithm depends on

1. The randomness or the repetitiveness of the patterns
2. The size of the testing and training data set under consideration
3. Specific areas of application

The application of recurrent neural networks for dictionary based language modelling shows a very low CER of 7.3. In the subfield of acoustic modelling, the DNN-HMM model gives a good CER of 29.7%, when 5 hidden layers are present in the system. This is however for a relatively medium sized data set like ordering for food in McDonalds.

For a very large data set like daily communication, LSTM based models outperform the others. The best CER of 34.56% is achieved by the use of LSTM-OP + 3-layer ReLU model.

Table 4: Comparison of Different Approaches for Speech Analysis

Model	Design	Application	Best CER (%)
DNN-HMM	DNN-HMM with 5 hidden layer and 2K hidden units	Context specific acoustic modeling	29.7
RNN	RNN-WER	Dictionary based language modeling	7.3
LSTM	LSTM-OP + 3-layer ReLU	General purpose acoustic modeling	34.65

Table 4 summarizes the application of different kinds of networks to various problems related to speech recognition. The best character error rate (CER) is also given.

REFERENCES

[1] L. Deng and D. Yu,"Deep Learning: Methods and Applications",http://research.microsoft.com/pubs/209355/DeepLearning-NowPublishing-Vol7-SIG-039.pdf, 2014

[2] G. E. Dahl, Dong Yu, Li Deng, and Alex Acero, "Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition," IEEE Transactions on Audio, Speech, And Language Processing, Vol. 20, No. 1, January 2012.

[3] Geoffrey Hinton, Li Deng, Dong Yu, George Dahl, Abdel-rahman Mohamed, NavdeepJaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara Sainath, Brian Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition," Signal Processing Magazine, November, 2012.

[4] Chao Weng1 , Dong Yu2, Shinji Watanabe3, Biing-Hwang (Fred) Juang , Recurrent Deep Neural Networks for Robust Speech Recognition, 1 Georgia Institute of Technology, Atlanta, GA, USA, 2 Microsoft Research, One Microsoft Way, Redmond, WA, USA, 3 Mitsubishi Electric Research Laboratories, Cambridge, MA, USA2014, IEEE International Conference on Acoustic Speech and Signal Processing (ICASSP).

[5] S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural Computation, 9(8):1735–1780, 1997.

[6] F. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm," *Neural Computation*, vol. 12, pp. 2451–2471, 2000.

[7] G. E. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, pp. 1527–1554, 2006.

[8] D. Erhan, A. Courville, Y. Bengio, and P. Vincent, "Why does unsupervised pre-training help deep learning?," in *Proc. AISTATS'10*, May 2010, vol. 9, pp. 201–208.

[9] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[10] G. E. Dahl, Dong Yu, Li Deng, and Alex Acero, "Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition," IEEE Transactions on Audio, Speech, And Language Processing, Vol. 20, No. 1, January 2012.

[11] Alex Graves, NavdeepJaitly, Towards End-to-End Speech Recognition with Recurrent Neural Networks, Department of Computer Science, University of Toronto, Canada, 31 st International Conference on Machine, Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

[12] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *ICASSP*, 2013, pp. 6645–6649.

[13] J. Geiger, X. Zhang, F. Weninger, and et al., "Robust speech recognition using long short-term memory recurrent neural networks for hybrid acoustic modelling," in *Interspeech*, 2014, pp. 631–635.

[14] A. Graves, N. Jaitly, and A. Mohamed, "Hybrid speech recognition with deep bidirectional lstm," in *ASRU*, 2013, pp. 273– 278.

[15] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Interspeech*, 2014, pp. 338–342.

[16] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," arXiv:1402.1128, 2014.

[17] H. Sak, O. Vinyals, G. Heigold, and et al., "Sequence discriminative distributed training of long short-term memory recurrent neural networks," in *Interspeech*, 2014, pp. 1209–1213.

[18] Li, Xiangang, and Xihong Wu. "Constructing Long Short-Term Memory based Deep Recurrent Neural Networks for Large Vocabulary Speech Recognition."*arXiv preprint arXiv:1410.4281* (2014).