International Journal of New Technology and Research (IJNTR)
ISSN:2454-4116, Volume-4, Issue-3, March 2018 Pages 116-119

Handwriting Recognition using LSTM Networks

Sarita Yadav, Ankur Pandey, Pulkit Aggarwal, Rachit Garg, Vishal Aggarwal

***Abstract*— Recognizing digits in an optimal way is a challenging problem. Recent deep learning based approaches have achieved great success on handwriting recognition. English characters are among the most widely adopted writing systems in the world. This paper presents a comparative evaluation of the standard LSTM RNN model with other deep models on MNIST dataset.**

***Index Terms*— LSTM, MNIST dataset.**

I. INTRODUCTION

Recurrent neural networks (RNN) have recently shown great promise in tackling various sequence modeling tasks in machine learning, such as automatic speech recognition [1-2], language translation [3-4], and generation of language descriptions for images [5-6]. Simple RNNs, however, are difficult to train using the stochastic gradient descent and have been reported to exhibit the so-called "vanishing" gradient and/or "exploding" gradient phenomena [7-8]. This has limited the ability of simple RNN to learn sequences with relatively long dependencies. To address this limitation, researchers have developed a number of techniques in network architectures and optimization algorithms [9-11], among which the most successful in applications is the Long Short-term Memory (LSTM) units in RNN [9, 12]. They were introduced by Hochreiter & Schmidhuber (1997), and were refined and popularized by many people in following work.1 They work tremendously well on a large variety of problems, and are now widely used. A LSTM unit utilizes a "memory" cell that may maintain its state value over a long time, and a gating mechanism that contains three non-linear gates, namely, an input, an output and a forget gate. The gates' intended role is to regulate the flow of signals into and out of the cell, in order to be effective in regulating long-range dependencies and achieve successful RNN training. Since the inception of the LSTM unit, many modifications have been introduced to improve performance. Gers et al. [13] have introduced "peephole" connections to the LSTM unit that connects the memory cell to the gates so as to infer precise timing of the outputs. Sak et al. [14-15] introduced two units layer and the output layer, which resulted in significantly improved performance in a large vocabulary speech recognition task. LSTMs help preserve the error that can be backpropagated through time and layers. By maintaining a more constant error, they allow recurrent nets to continue to learn over many time steps (over 1000), thereby opening a channel to link causes and effects remotely

Sarita Yadav, Bharati Vidyapeeth's College of Engineering New Delhi, India 110063
Ankur Pandey, Bharati Vidyapeeth's College of Engineering New Delhi, India 110063
Pulkit Aggarwal, Bharati Vidyapeeth's College of Engineering New Delhi, India 110063
Rachit Garg, Bharati Vidyapeeth's College of Engineering New Delhi, India 110063
Vishal Aggarwal, Bharati Vidyapeeth's College of Engineering New Delhi, India 110063

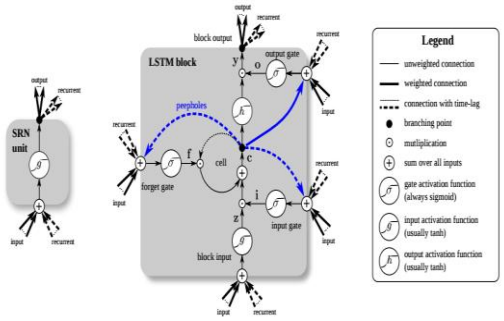


Figure 1. Detailed schematic of the Simple Recurrent Network (SRN) unit (left) and a Long Short-Term Memory block (right) as used in the hidden layers of a recurrent neural network.

The paper presents a comparative evaluation of the standard LSTM RNN model with other deep models on MNIST dataset.

II. THE RNN LSTM ARCHITECTURE

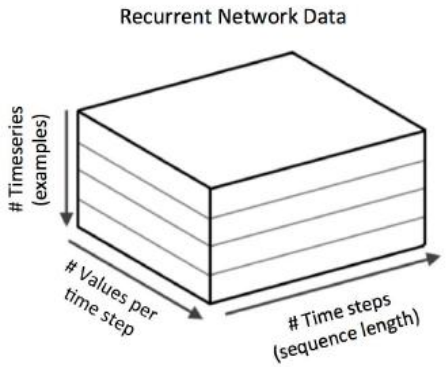
The RNN and LSTM are the class of Artificial Neural Network, which allow the network to preserve the dependency among sequence of input data. Unlink feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition[1] or speech recognition.

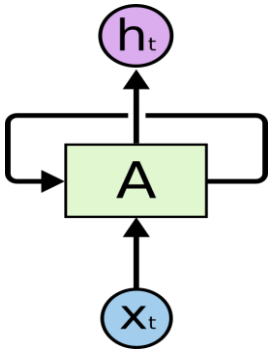
A. Recurrent Neural Network (RNN)

The RNN is a natural generalization of the feedforward neural networks to sequences [30]. Given a general input sequence [x1, x2, . . . , xk] where xi $\in$ R^d (different samples may have different sequence length k), at each time-step of RNN modeling, a hidden state is produced, resulting in a hidden sequence of [h1, h2, . . . , hk].

Input of RNN and its variant has 3D Volumetric input which has following dimensions:

1. Mini-batch Size
2. Number of columns in our vector per time-step
3. Number of time-steps





In the above diagram, a chunk of neural network, A A, looks at some input xtxt and outputs a value htht. A loop allows information to be passed from one step of the network to the next.

The activation of the hidden state at time-step t is computed as a function f of the current input xt and previous hidden state ht-l as:

$$ht = f(xt, ht - 1). \quad (4)$$

At each time-step, an optional output can be produced by yt = g(ht), resulting in an output sequence [y1, y2, . . . , yk], which can be used for sequence-to-sequence tasks, for example, based on the CTC framework [31]. In this section, the input sequence is encoded into a fixed-length vector for final classification, due to the recursively applied transition function f. The RNN computes activations for each timestep which makes them extremely deep and can lead to vanishing or exploding gradients [28]. The choice of the recurrent computation f can have a big impact on the success of RNN because the spectrum of its Jacobian controls whether gradients tend to propagate well (or vanish or explode).

Backpropagation in RNN is performed using BPTT algorithm. But BPTT has some limitations which are as follow:

1. Vanishing Gradients
2. Exploding Gradients

Unfortunately, gradients often get smaller and smaller as the algorithm progresses down to the lower layers. As a result, the Gradient Descent update leaves the lower layer connection weights virtually unchanged, and training never converges to a good solution. This is called the vanishing gradients problem. In some cases, the opposite can happen: the gradients can grow bigger and bigger, so many layers get insanely large weight updates and the algorithm diverges. This is the exploding gradients problem, which is mostly encountered in recurrent neural networks. More generally, deep neural networks suffer from unstable gradients; different layers may learn at widely different speeds.

LSTM and GRU[17] are used to overcome the above limitations. In this paper, we use both long short term memory (LSTM) [15] [16] for RNN modeling.

B. *Long Short Term Memory (LSTM)*

LSTM [15] [16] is widely applied because it reduces the vanishing and exploding gradient problems and can learn longer term dependencies. With LSTMs, for time-step t, there is an input gate it, forget gate ft, and output gate ot:

$$it = sign(Wixt + Ui\ ht - 1 + bi), \quad (5)$$
$$ft = sign(Wfxt + Uf\ ht - 1 + bf), \quad (6)$$
$$ot = sign(Woxt + Uo\ ht - 1 + bo), \quad (7)$$
$$cet = tan\ h(Wcxt + Uc\ ht - 1 + bc), \quad (8)$$
$$ct = it\ cet + ft\ ct - 1, \quad (9)$$
$$ht = ot\ tan\ h(ct), \quad (10)$$

where W* is the input-to-hidden weight matrix, U* is the state-to-state recurrent weight matrix, and b* is the bias vector. The operation denotes the element-wise vector product. The hidden state of LSTM is the concatenation of (ct, ht). The long-term memory is saved in ct, and the forget gate and LSTM/GRU mean pooling and dropout logistic regression.

Function of three gate units:

1. The input gate protects the unit from irrelevant input events.
2. The forget gate helps the unit forget previous memory contents.
3. The output gate exposes the contents of the memory cell (or not) at the output of the LSTM unit.

Each LSTM unit has two types of connections:

1. Connections from the previous time-step (outputs of those units).
2. Connections from the previous layer.

The memory cell in an LSTM network is the central concept that allows the network to maintain state over time. The input, forget, and output gates in an LSTM unit have sigmoid activation functions for [0, 1] restriction. The LSTM block input and output activation function (usually) is a tanh activation function. An activation output of 1.0 means "remember everything" and activation output of 0.0 means "forget everything."

Back propagation in LSTM can be done in two ways:

1. BPTT (Backpropagation Through Time)
2. Truncated BPTT (Efficient when time steps are more than 100)

III. METHODOLOGY

A. ***Preprocessing:*** The preprocessing phase can be considered as the first stage of the recognition system. The main goal of this step is to modify the images in a way that will make it easier and faster for the recognizer to learn from them.

Objectives of Pre- processing:

1. Binarization: Process of converting a gray-scale image into a binary image. It is done by thresholding.
2. Noise reduction: Process of improves the quality of image by removing noise from image. It is done by Normalization.
3. Stroke width normalization
4. Skew correction: Process of alignment correction of object in image. It is done by correlation, projection profiles and Hough transform.

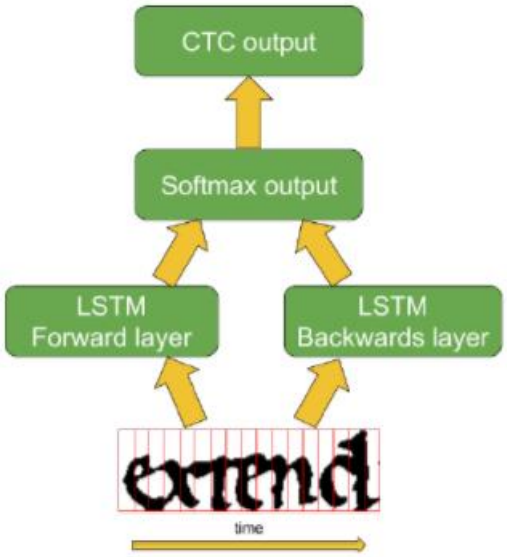
B. ***Feature extraction:*** It is a second phase of converting each image enclosing the digits into feature vector. Feature vector is a tensor representation of image.

It removes the redundant features, so that only those features are present in data which affects the image classification algorithms

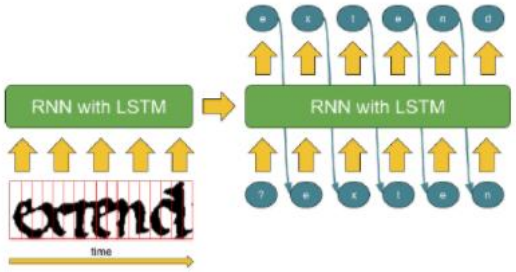
C. ***Classification:*** It is the most important phase of image recognition where LSTM network take feature vector as input and classify them in one of the 10 classes.

D. ***Connectionist Temporal Classification approach:*** The feed-forward approach is similar to the original recurrent neural networks by the fact both architectures require a direct alignment between the input features and target variables. However in the real-world handwriting recognition problems it is much easier to segment text into words rather than individual characters. Achieving direct alignment between the image of input character and character target label would require a prior segmentation step. Being a very hard problem by itself its complexity keeps increasing in time since there is a high tendency of encountering possible errors already in the segmentation step. As a result this would also limit the context of the data learned by the RNN.

To target this problem the Connectionist Temporal Classification (CTC) approach was introduced,originally for speech recognition and afterwards also for handwriting recognition [7]. CTC makes it possible to avoid the previously mentioned direct alignment between the input variables and the target labels by interpreting the output of the network as a probability distribution over all possible label sequences on the given input sequence.



E. ***Sequence-to-Sequence Learning approach:*** Sequence-to-Sequence Learning (Seq2Seq) is based on the approach developed by Google researchers for the automatic sentence translation from English to French [15]. The main idea is to use two connected RNNs, the first RNN for reading an input sentence and mapping it to a fixed-dimensional vector representation and the second RNN for decoding an output sequence from that representation. The approach is very suitable for machine translation, as input and output sequences can be of various lengths and ordering of words in each language can be different. While in handwriting recognition it can generally be assumed that handwritten characters can be mapped to labeled characters in the same order, Seq2Seq approach still provides the advantage that nospecific mapping between positions of characters in input image and target character labels is required.The model consists of two connected RNNs, namely: the Image-RNN for the image encoding and the Label-RNN for the generation of the text label as shown in Figure 6. Each RNN is constituted of one or more LSTM or BLSTM layers. Weighted cross-entropy loss between a target sequence and a predicted sequence of characters is used as a cost function.



IV. EXPERIMENTS

We conducted the experiments using LSTM, MLP, RNN on MNIST data set.

Following is the configuration of the LSTM Network:

1. Number of hidden units of LSTM: 128
2. Time steps to unroll: 28
3. Input size to LSTM cell: 28
4. Number of epochs: 10
5. Total layers: 2

Following is the configuration of the MLP Network:

1. Number of nodes in input layer: 784
2. Number of output nodes: 10
3. Number of epochs: 10
4. Total layers: 2

Following is the configuration of the RNN:

1. Number of hidden units of RNN: 128
2. Time steps to unroll: 28
3. Input size to LSTM cell: 28
4. Number of epochs: 10
5. Total layers: 2

Model of SVM: Linear svc

V. OBSERVATIONS

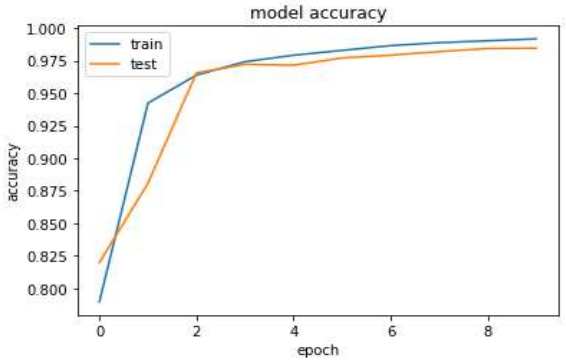
MODEL	ACCURACY
LINEAR SVC	91.8%
RNN	97.57%
MLP	98.17%
LSTM	98.46%

LSTM are better in recognising Handwritten digits of MNIST data set than SVM, RNN, MLP.

VI. RESULT

Final accuracy of LSTM network on MNIST dataset: 98.46%

Below is the graph of epoch versus accuracy.



VII. CONCLUSION AND FUTURE WORK

LSTM has shown to perform well on MNIST data set because of its ability to capture long term dependency in a sequence data and storing information from trained data and using it to classify test data.

In the future, LSTM can be combined with the CNN to create a Convolutional LSTM deep neural network model which can capture both spatial and temporal dependency to classify data more accurately.

REFERENCES

[1] A. Graves, Supervised Sequence Labelling with Recurrent Neural Networks, Berlin, Heidelbergz: Springer-Verlag, 2012.
[2] A. Graves, "Speech recognition with deep recurrent networks," in Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, 2013, pp. 6645-6649.
[3] K. Cho, B. van Merienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2014, pp. 1724-1734.
[4] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in Proceedings of Advances in Neural Information Processing Systems 27, NIPS, 2014, pp. 3104–3112, 2014.
[5] A. Karpathy, F. F. Li, "Deep visual-semantic alignments for generating image descriptions," in Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, IEEE, 2015, pp. 3128-3137.
[6] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. S. Zemel, and Y. Bengio. "Show, attend and tell: Neural image caption generation with visual attention," in Proceedings of the 32nd International Conference on Machine Learning, 2015, vol. 37, pp. 2048-2057.
[7] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," IEEE Transactions on Neural Networks, vol. 5, no. 2, pp. 157–166, 1994.
[8] R. Pascanu, T. Mikolov, and Y, Bengio, "On the difficulty of training recurrent neural networks," in Proceedings of the 30 th International Conference on Ma-chine Learning, 2013. JMLR: W&CP volume 28.
[9] S. Hochreiter, J. Schmidhuber. "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997. [10] Q.V. Le, N. Jaitly, G.E. Hinton, "A simple way to initialize recurrent networks of rectified linear units." arXiv preprint arXiv: 1504.00941, 2015.
[10] J. Martens and I. Sutskever, "Training deep and recurrent neural networks with Hessian-Free optimization." Neural Networks: Tricks of the Trade, Springer, 2012, pp. pp 479-535.
[11] F. A. Gers, J. Schmidhuber, and F. Cummins. "Learning to forget: Continual prediction with LSTM." in Proceedings of the 9th International Conference on Artificial Neural Networks, IEEE, 1999, vol. 2, pp. 850- 855.
[12] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber. "Learning precise timing with LSTM recurrent networks," Journal of Machine Learning Research, vol. 3, pp. 115-143, 2003.
[13] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling." ISCA, pp. 338-342, 2014.
[14] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," ArXiv e-prints, 2014. [16] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. "Empirical evaluation of gated recurrent neural networks on sequence modeling," arXiv: 1412.3555, 2014.
[15] K. Greff, R. K. Srivastava, J. Koutnk, B. R. Steunebrink, and J. Schmidhuber. "LSTM: a search space odyssey," arXiv: 1503.04069