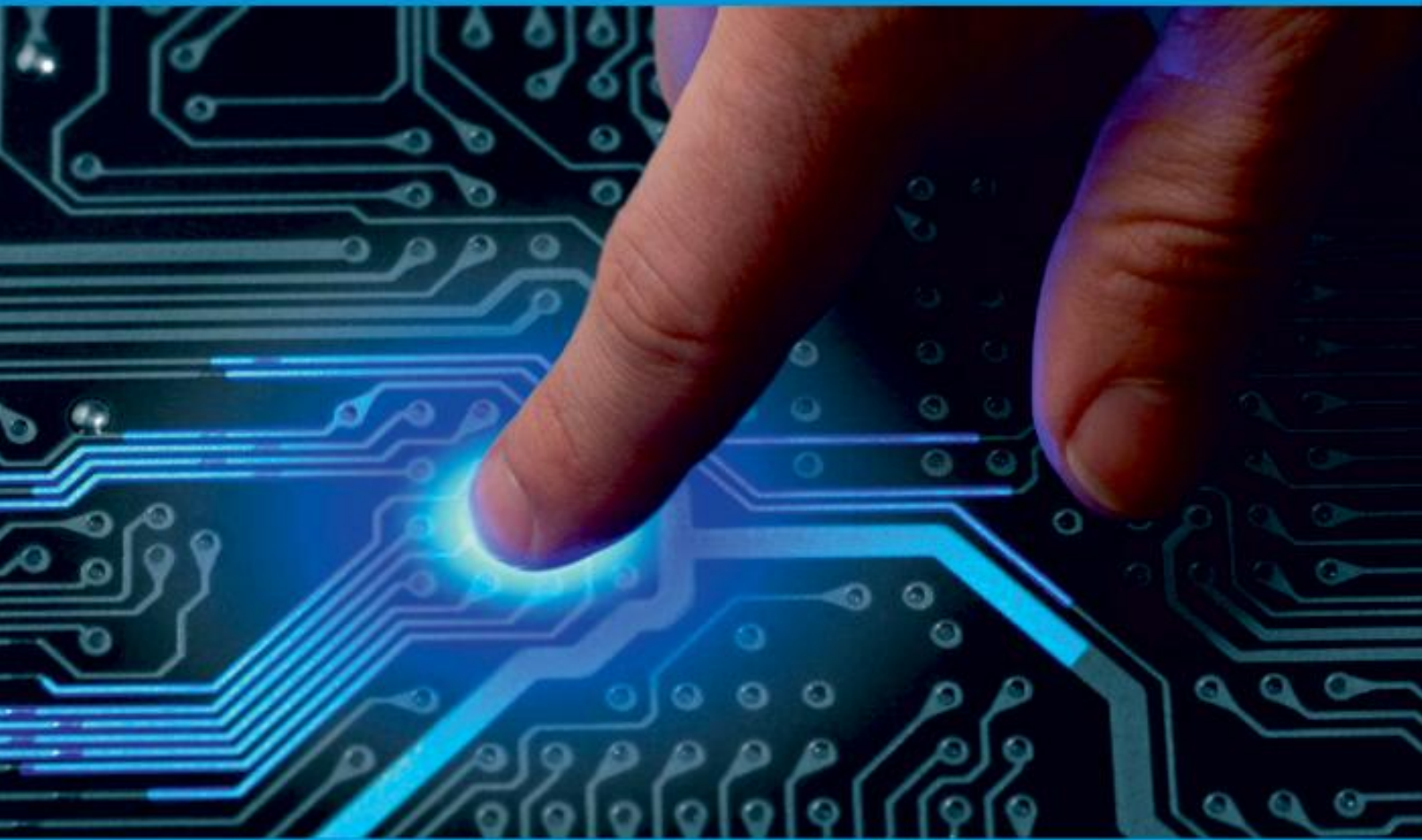




**IJIRCCCE**

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

**Volume 9, Issue 6, June 2021**

**ISSN** INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA

**Impact Factor: 7.542**



9940 572 462



6381 907 438



ijircce@gmail.com



www.ijircce.com

# Character Text Generation Using Machine Learning

Ch.Anuritha<sup>1</sup>, K.Sandhya Rani<sup>2</sup>, R.Radhika<sup>3</sup>, S.YAishwarya<sup>4</sup>, Abhay Kumar<sup>5</sup>

B.Tech. Student, Department of Computer Science and Engineering, JB Institute of Engineering and Technology,  
Moinabad, Telangana, India<sup>1,2,3,4</sup>

Associate Professor, Department of Computer Science and Engineering, JB Institute of Engineering and Technology,  
Moinabad, Telangana, India<sup>5</sup>

**ABSTRACT:** Machine learning methods possess many processing layers to know the stratified representation of data and have achieved state-of-art leads to several domains. Recently, machine learning model designs and architectures have unfolded within the context of natural language Processing (NLP). This survey presents a quick description of the advances that have occurred within the area of Deep Generative modeling. This work considers most of the papers from 2015 onwards. during this paper, we review many deep learning models that are used for the generation of text. We also summarize the assorted models and have recommend the detailed understanding of past, present, and way forward for text generation models in deep learning. Furthermore, machine learning approaches that are explored and evaluated in numerous application domains in NLP are included during this survey. There is a large amount of knowledge that may be categorized as sequential. it is present within the style of audio, video, text, statistic, sensor data, etc. A special thing about this kind of data is that if two events are occurring in an exceedingly particular time frame, the occurrence of event A before event B is a wholly different scenario as compared to the occurrence of event A event B. However, in conventional machine learning problems, it hardly matters whether a particular data point was recorded before the other. Text, a stream of characters lined up one after another, could be a difficult thing to crack. this can be because when handling text, a model could also be trained to create very accurate predictions using the sequences that have occurred previously but one wrong prediction has the potential to create the complete sentence meaningless. However, in case of a numerical sequence prediction problem, whether or not a prediction goes entirely south, it could still be considered a legitimate prediction. Text generation may be a popular problem in data science and machine learning, and it's an acceptable task for recurrent neural.

**KEYWORDS:** Text generation, Recurrent Neural Network, Long Short Term Memory, Neural Networks.

## I. INTRODUCTION

The report consists of two documents describing the state of the art of computer text generation of natural language text. Both were prepared by a panel of individuals who are active in research on text generation. The first document assesses the state of the art, identifying four kinds of technical developments which will shape the art in the coming decade: linguistically justified grammars, knowledge representation methods, models of the reader, and models of discourse. The second document is a comprehensive bibliography on text generation, the first of its kind. In addition to citations of documents, it includes descriptions of on-going research efforts.

## II. LITERATURE SURVEY

Machine learning has evolved many algorithms in the field of NLP like Recurrent Neural Networks (RNNs) (for sequence modeling), Recurrent Neural Networks with external memory (RNN-EM) (to improve memory capacity of RNN) (Peng and Yao, 2015), Gated Feedback Recurrent Neural Networks (GF-RNN) (stacking multiple recurrent layers with gating units) (Chunget al., 2015), Conditional Random Fields as Recurrent Neural Networks (CRF-RNN) (for probabilistic graphical modeling) (Zheng et al., 2015), Quasi- Recurrent Neural Networks (Q-RNN) (using parallel time-steps for sequence modeling) (Bradbury et al., 2016), Memory Networks (for Question Answering (QA)) (Weston et al., 2014), Augmented Neural Networks (Neural Turing Machines) (Olah and Carter, 2016). Recurrent neural networks were based on David

Rumelhart's work in 1986.[8] Hopfield networks - a special kind of RNN - were discovered by John Hopfield in 1982. In 1993, a neural history compressor system solved a "Very Deep Learning" task that required more than 1000 subsequent layers in an RNN unfolded in time. Recurrent Neural Networks (RNNs) (for sequence modeling), Recurrent Neural Networks with external memory (RNN-EM)(to improve memory capacity of RNN) (Peng and Yao,2015), Gated Feedback Recurrent Neural Networks (GF-RNN) (stacking multiple recurrent layers with gating units) (Chunget al., 2015), Conditional Random Fields as Recurrent Neural Networks (CRF-RNN)(for probabilistic graphical modeling) (Zheng et al., 2015), Quasi- Recurrent Neural Networks (Q-RNN)(using parallel time-steps for sequence modeling) (Bradbury et al., 2016),Memory Networks (for Question Answering (QA)) (Weston et al.,2014), Augmented Neural Networks (Neural Turing Machines)(Olah and Carter, 2016). LSTM Edit Long short-term memory (LSTM) networks were invented by Hochreiter and Schmidhuber in 1997 and set accuracy records in multiple applications domains.Around 2007, LSTM started to revolutionize speech recognition, outperforming traditional models in certain speech applications.[11] In 2009, a Connectionist Temporal Classification (CTC)-trained LSTM network was the first RNN to win pattern recognition contests when it won several competitions in connected handwriting recognition. In 2014, the Chinese company Baidu used CTC-trained RNNs to break the 2S09 Switchboard Hub5'00 speech recognition dataset benchmark without using any traditional speech processing methods.LSTM also improved large-vocabulary speech recognition and text to-speech synthesis and was used in Google Android.In 2015, Google's speech recognition reportedly experienced a dramatic performance jump of 49%[citation needed] through CTC-trained LSTM.LSTM broke records for improved machine translation,Language Modeling and Multilingual Language Processing. LSTM combined with convolutional neural networks (CNNs) improved automatic image captionin.

### III.RELATED WORK

This section will target recurrent nets with time-varying inputs (as opposition nets with stationary inputs and xpoint-based gradient calculations, e.g., Almeida 1987, Pineda 1987).Gradient-descent variants. The approaches of Elman (1988), Fahlman (1991), Williams(1989), Schmidhuber (1992a), Pearlmutter (1989), and plenty of of the related algorithms in Pearlmutter's comprehensive overview (1995) suer from the identical problems as BPTT and RTRL (seeSections 1 and 3).Time-delays. Other methods that appear practical for brief time lags only are Time-Delay Neural Networks (Lang et al. 1990) and Plate's method (Plate 1993), which updates unit activations supported a weighted sum of old activations (see also de Vries and Principe 1991). Lin et al.(1995) propose variants of time-delay networks called NARX networks.Time constants. To accommodate very long time lags, Mozer (1992) uses time constants inuencing changes of unit activations (deVries and Principe's above-mentioned approach (1991) may of course be viewed as a mix of TDNN and time constants). For very long time lags, however, the time constants need external ne tuning (Mozer 1992). Sun et al.'s alternative approach (1993) updates the activation of a recurrent unit by adding the old activation and also the (scaled) current net input. The net input, however, tends to perturb the stored information, which makes long-term storage impractical.Ring's approach. Ring (1993) also proposed a technique for bridging while lags. Whenever a unit in his network receives conicting error signals, he adds a better order unit inuencing appropriate connections. Although his approach can sometimes be extremely fast, to bridge a pause involving 100 steps may require the addition of 100 units. Also, Ring's net doesn't generalize to unseen lag durations.Bengio et al.'s approaches. Bengio et al. (1994) investigate methods like simulated annealing, multi-grid random search, time-weighted pseudo-Newton optimization, and discrete error propagation. Their \latch" and \2-sequence" problems are very almost like problem 3a with minimal break 100 (see Experiment 3). Bengio and Frasconi (1994) also propose an EM approach for propagating targets. With n so-called \state networks", at a given time, their system is in one amongst only different states. cf. beginning of Section 5. But to unravel continuous problems like the adding problem" (Section 5.4), their system would require an unacceptable number of states (i.e., state networks).Kalman lters. Puskorius and Feldkamp (1994) use Kalman lter techniques to boost recurrent net performance. Since they use \a derivative discount factor imposed to decay exponentially the consequences of past dynamic derivatives," there's no reason to believe that their Kalman Filter Trained Recurrent Networks are going to be useful for very long minimal time lags.Second order nets. we'll see that LSTM uses multiplicative units (MUs) to guard error.

### IV.PROPOSED SYSTEM

Long Short Term Memory Network is an advanced RNN,a sequential network,that allows information to persist.It is capable of handling the vanishing gradient problem faced by RNN.As, the LSTM contains memory states (previous

state, current memory, and the input) which are combined to solve this vanishing gradient problem. LSTM also uses constant error carousel (CEC) which constantly eradicating error flow. The LSTM contains memory states (previous state, current memory, and also the input) which are combined to resolve problems like vanishing/exploding gradient. Many LSTM based text generation models are proposed. (Pawade et al., 2018; Chen et al., 2019; Wang et al., 2019). GRU is another extension of standard RNNs (Cho et al., 2014) which modifies LSTM architecture with a gating network which generates signals that control this input and previous memory to update the present activation and current network state. It's simpler than LSTM within which the parameter updation is additionally used for gates in line with the algorithm. Many deep generative architectures have used GRUs for the generation of text (Mangal et al., 2019; Hong et al., 2018).

### 3.1 System Architecture

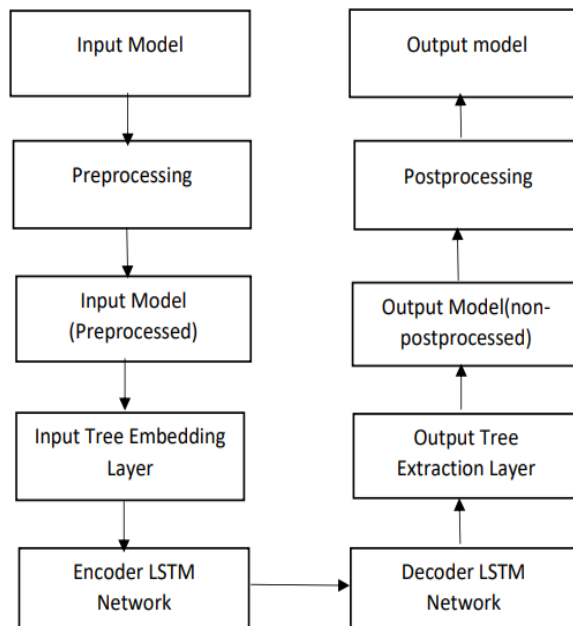


Fig 1. Architecture

### 3.2 Module Description

#### 3.2.1 Dataset:

Creates a text dataset contains the one-hot encoded text data. It produces batches of sequences of encoded labels.

We split the text data into batches are accustomed train the RNN, which we sample a random chunk of the text (with given length) to gauge the performance of our model.

#### 3.2.2 Model Selector:

Performs randomized search and rank the models by accuracy.

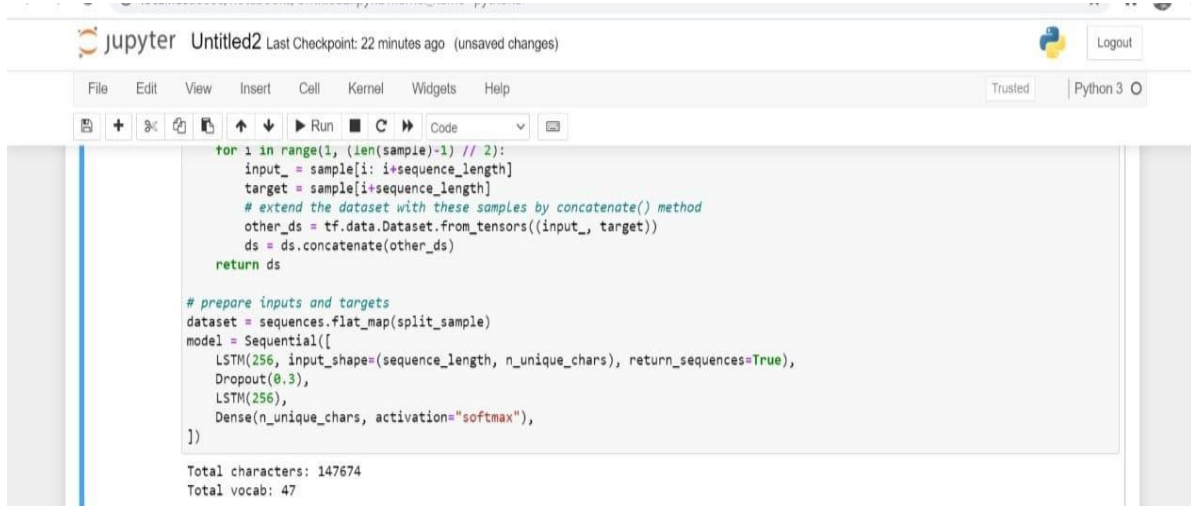
It selects the most effective ranking models and allows lengthy searching.

#### 3.2.3 RNN Text Generator:

Creates a recurrent neural network with a TensorFlow RNN cell.

It has an output projection layer which produces the last word probability for every character class.

### V.EXPERIMENTAL RESULTS

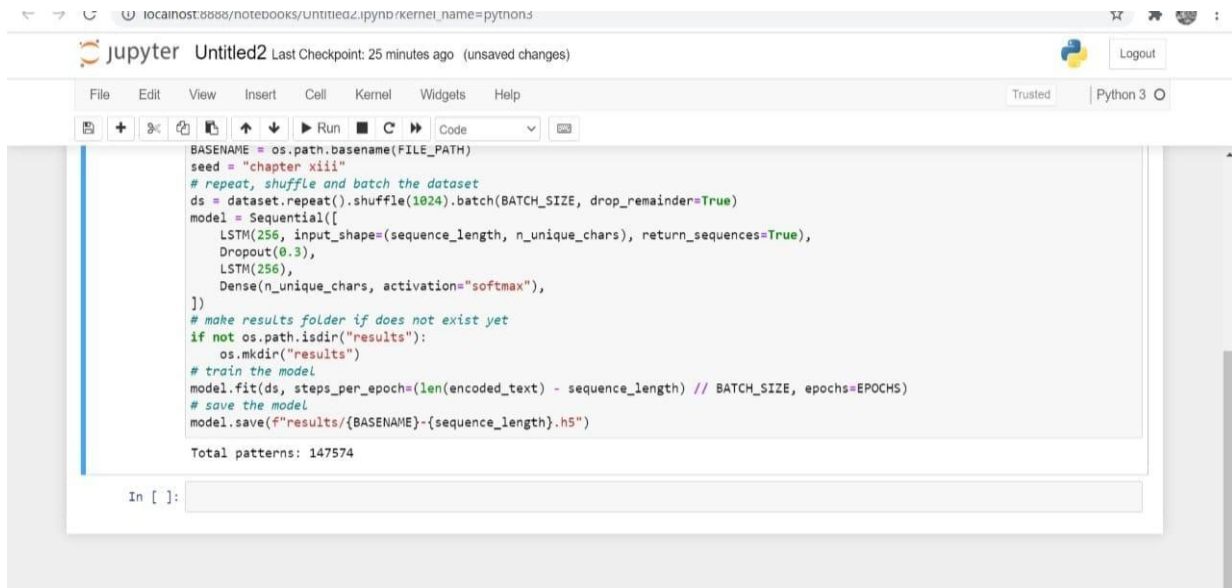


```
for i in range(1, (len(sample)-1) // 2):
    input_ = sample[i: i+sequence_length]
    target = sample[i+sequence_length]
    # extend the dataset with these samples by concatenate() method
    other_ds = tf.data.Dataset.from_tensors((input_, target))
    ds = ds.concatenate(other_ds)
return ds

# prepare inputs and targets
dataset = sequences.flat_map(split_sample)
model = Sequential([
    LSTM(256, input_shape=(sequence_length, n_unique_chars), return_sequences=True),
    Dropout(0.3),
    LSTM(256),
    Dense(n_unique_chars, activation="softmax"),
])

Total characters: 147674
Total vocab: 47
```

Fig.2 No of characters in data set

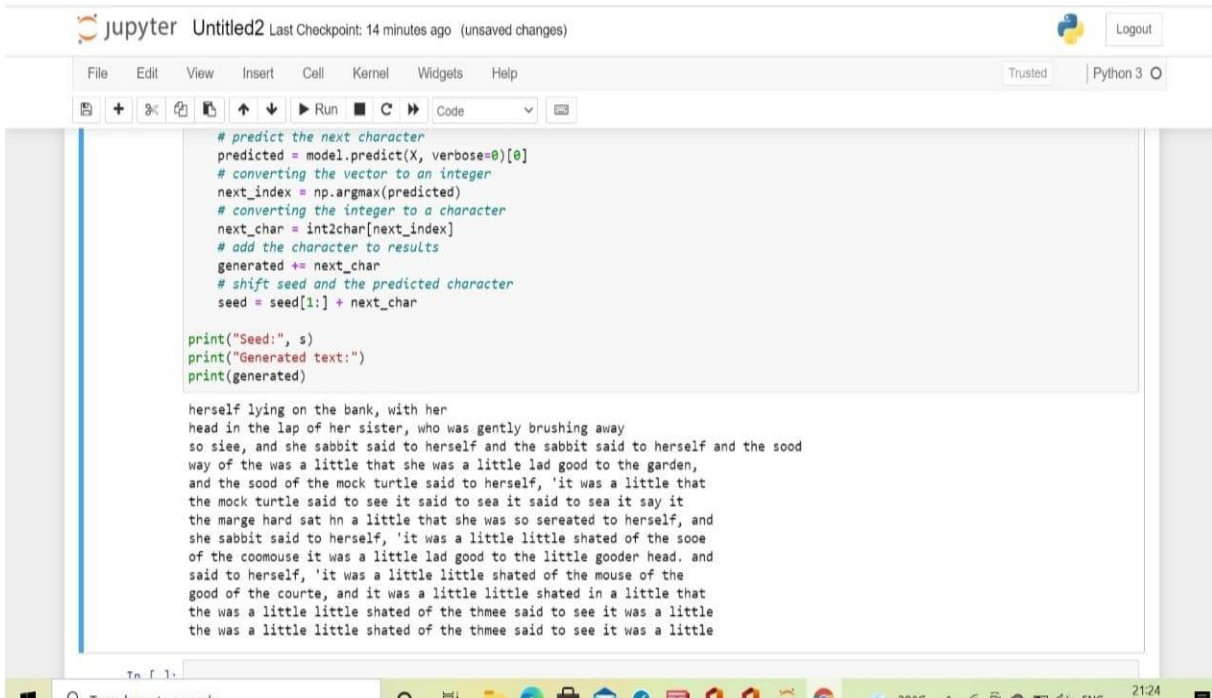


```
BASENAME = os.path.basename(FILE_PATH)
seed = "chapter xiii"
# repeat, shuffle and batch the dataset
ds = dataset.repeat().shuffle(1024).batch(BATCH_SIZE, drop_remainder=True)
model = Sequential([
    LSTM(256, input_shape=(sequence_length, n_unique_chars), return_sequences=True),
    Dropout(0.3),
    LSTM(256),
    Dense(n_unique_chars, activation="softmax"),
])
# make results folder if does not exist yet
if not os.path.isdir("results"):
    os.mkdir("results")
# train the model
model.fit(ds, steps_per_epoch=(len(encoded_text) - sequence_length) // BATCH_SIZE, epochs=EPOCHS)
# save the model
model.save(f"results/{BASENAME}-{sequence_length}.h5")

Total patterns: 147574

In [ ]:
```

Fig.3 Total no of patterns present in data set



```

# predict the next character
predicted = model.predict(X, verbose=0)[0]
# converting the vector to an integer
next_index = np.argmax(predicted)
# converting the integer to a character
next_char = int2char[next_index]
# add the character to results
generated += next_char
# shift seed and the predicted character
seed = seed[1:] + next_char

print("Seed:", s)
print("Generated text:")
print(generated)

herself lying on the bank, with her
head in the lap of her sister, who was gently brushing away
so siee, and she sabbit said to herself and the sabbit said to herself and the sood
way of the was a little that she was a little lad good to the garden,
and the sood of the mock turtle said to herself, 'it was a little that
the mock turtle said to see it said to sea it said to sea it say it
the marge hard sat hn a little that she was so sereated to herself, and
she sabbit said to herself, 'it was a little little shated of the sooe
of the coomouse it was a little lad good to the little gooder head. and
said to herself, 'it was a little little shated of the mouse of the
good of the courte, and it was a little little shated in a little that
the was a little little shated of the thmee said to see it was a little
the was a little little shated of the thmee said to see it was a little
    
```

Fig.4 Generated text with random seed test

### VI.CONCLUSION

Even though there are various solutions for character text generation, LSTM somehow solved various issues raised by this problem. Still we cannot this can be the ultimate solution as LSTM has its own disadvantages. But we will say it are often one in every of most appropriate solutions among many Scientists. Scientists are still researching on these areas and it's still most interesting problem in nowadays also to unravel .Moreover, LSTM by introducing Constant Error Carousel (CEC) units, LSTM deals with the exploding and vanishing gradient problems in most appropriate way.

### VII.FUTURE ENHANCEMENT

The scope of Machine Learning isn't limited to the investment sector. Rather, it's expanding across all fields like banking and finance, information technology, media & entertainment, gaming, and also the automotive industry. as the Machine Learning scope is extremely high, there are a number of the areas where researchers are working toward revolutionizing the world for the longer term. LSTMs became popular because they might solve the problem of vanishing gradients. But it seems, they fail to get rid of it completely. the matter lies within the undeniable fact that the data still needs to move from cell to cell for its evaluation. Moreover, the cell has become quite complex now with the extra features (such as forget gates) being brought into the image. They require plenty of resources and time to induce trained and become ready for real-world applications. I technical terms, they have high memory-bandwidth because of linear layers present in each cell which the system usually fails to provide for. Thus, hardware-wise, LSTMs become quite inefficient With the increase of knowledge mining, developers are searching for a model that may remember past information for a extended time than LSTMs. The source of inspiration for such kind of model is that the human habit of dividing a given piece of data into small parts for simple remembrance. LSTMs get tormented by different random weight initializations and hence behave quite like that of a feed-forward neural net. they like small weight initializations instead .LSTMs are liable to overfitting and it's difficult to use the dropout algorithm to curb this issue. Dropout may be a regularization method where input and recurrent connections to LSTM units are probabilistically excluded from activation and weight updates while training a network. while, LSTMs are a awfully promising solution to sequence and statistic related problems.



## REFERENCES

1. Almeida, L. B. (1987). A learning rule for asynchronous perceptron with feedback in a combinatorial environment. In IEEE 1st International Conference on Neural Networks, San Diego, volume 2, pages 609-618.
2. Baldi, P. and Pineda, F. (1991). Contrastive learning and neural oscillator. *Neural Computation*, 3:526-545.
3. Bengio, Y. and Frasconi, P. (1994). Credit assignment through time: Alternatives to backpropagation. In Cowan, J. D., Tesauero, G., and Alspector, J., editors, *Advances in Neural Information Processing Systems 6*, pages 75-82. San Mateo, CA: Morgan Kaufman.
4. Cleeremans, A., Servan-Schreiber, D., and McClelland, J. L. (1989). Finite-state automata and simple recurrent networks. *Neural Computation*, 1:372-381.
5. de Vries, B. and Principe, J. C. (1991). A theory for neural networks with time delays. In Lippmann, R. P., Moody, J. E., and Touretzky, D. S., editors, *Advances in Neural Information Processing Systems 3*, pages 162-168. San Mateo, CA: Morgan Kaufman.
6. Doya, K. (1992). Bifurcations in the learning of recurrent neural networks. In *Proceedings of 1992 IEEE International Symposium on Circuits and Systems*, pages 2777-2780.
7. Doya, K. and Yoshizawa, S. (1989). Adaptive neural oscillator using continuous-time backpropagation learning. *Neural Networks*, 2:375-385.
8. Elman, J. L. (1988). Finding structure in time. Technical Report CRL Technical Report 8801, Center for Research in Language, University of California, San Diego.
9. Fahlman, S. E. (1991). The recurrent cascade-correlation learning algorithm. In Lippmann, R. P., Moody, J. E., and Touretzky, D. S., editors, *Advances in Neural Information Processing Systems 3*, pages 190-196. San Mateo, CA: Morgan Kaufman.
10. Hochreiter, J. (1991). Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München. See [www7.informatik.tu-muenchen.de/~hochrei](http://www7.informatik.tu-muenchen.de/~hochrei). Lang, K., Waibel, A., and Hinton, G. E. (1990).



**INNO**  **SPACE**  
SJIF Scientific Journal Impact Factor  
**Impact Factor: 7.542**



**ISSN** INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
**INDIA**



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

 **9940 572 462**  **6381 907 438**  **ijircce@gmail.com**



[www.ijircce.com](http://www.ijircce.com)

Scan to save the contact details