* **New**[Community](https://community.analyticsvidhya.com/?automatic_login=true&utm_source=blog_navbar&utm_medium=community_button)
* [Articles](javascript:void(0);)
  + [Guides](https://www.analyticsvidhya.com/blog/category/guide/?utm_source=blog_navbar&utm_medium=machine_learning_button)
  + [Machine Learning](https://www.analyticsvidhya.com/blog/category/machine-learning/?utm_source=blog_navbar&utm_medium=machine_learning_button)
  + [Deep Learning](https://www.analyticsvidhya.com/blog/category/deep-learning/?utm_source=blog_navbar&utm_medium=deep_learning_button)
  + [NLP](https://www.analyticsvidhya.com/blog/category/nlp/?utm_source=blog_navbar&utm_medium=_button)
  + [Computer Vision](https://www.analyticsvidhya.com/blog/category/computer-vision/?utm_source=blog_navbar&utm_medium=article_button)
  + [Data Visualization](https://www.analyticsvidhya.com/blog/category/data-visualization/?utm_source=blog_navbar&utm_medium=_button)
  + [Interview Questions](https://www.analyticsvidhya.com/blog/category/interview-questions/?utm_source=blog_navbar&utm_medium=career_button)
  + [More](javascript:void(0);)

[Write an Article](https://lekhak.analyticsvidhya.com/write/)

* [Courses](javascript:void(0);)
  + [Certified AI & ML BlackBelt Plus](https://blackbelt.analyticsvidhya.com/plus?utm_source=blog_navbar&utm_medium=blackbelt_button)
  + [Data Science Immersive Bootcamp](https://bootcamp.analyticsvidhya.com/?utm_source=blog_navbar&utm_medium=bootcamp_button)
  + [Machine Learning](https://courses.analyticsvidhya.com/courses/Machine-Learning-Certification-Course-for-Beginners?utm_source=blog_navbar&utm_medium=start_here_button)
  + [Deep Learning](https://courses.analyticsvidhya.com/courses/getting-started-with-neural-networks?utm_source=blog_navbar&utm_medium=start_here_button)
  + [NLP](https://courses.analyticsvidhya.com/courses/Intro-to-NLP?utm_source=blog_navbar&utm_medium=start_here_button)

[All Courses](https://courses.analyticsvidhya.com/collections?utm_source=blog_navbar&utm_medium=all_courses_button)

* [Blogathon](https://datahack.analyticsvidhya.com/blogathon/?utm_source=blog&utm_medium=nav_bar)
* [MasterSeries](https://masterseries.analyticsvidhya.com/?utm_source=AVwebsite&utm_medium=blog-navbar)
* Login/Signup

[Journey To Success] This Roadmap is followed by 10000+ Data Science Professionals...

[Learn Now](https://blackbelt.analyticsvidhya.com/plus?utm_source=blog_india&utm_medium=desktop_flashstrip&utm_campaign=25-Aug-2023||&utm_content=roadmap)

×

1. [Home](https://www.analyticsvidhya.com/blog/)

* **New**[Community](https://community.analyticsvidhya.com/?automatic_login=true&utm_source=blog_navbar&utm_medium=community_button)
* [Articles](javascript:void(0);)
  + [Guides](https://www.analyticsvidhya.com/blog/category/guide/?utm_source=blog_navbar&utm_medium=machine_learning_button)
  + [Machine Learning](https://www.analyticsvidhya.com/blog/category/machine-learning/?utm_source=blog_navbar&utm_medium=machine_learning_button)
  + [Deep Learning](https://www.analyticsvidhya.com/blog/category/deep-learning/?utm_source=blog_navbar&utm_medium=deep_learning_button)
  + [NLP](https://www.analyticsvidhya.com/blog/category/nlp/?utm_source=blog_navbar&utm_medium=_button)
  + [Computer Vision](https://www.analyticsvidhya.com/blog/category/computer-vision/?utm_source=blog_navbar&utm_medium=article_button)
  + [Data Visualization](https://www.analyticsvidhya.com/blog/category/data-visualization/?utm_source=blog_navbar&utm_medium=_button)
  + [Interview Questions](https://www.analyticsvidhya.com/blog/category/interview-questions/?utm_source=blog_navbar&utm_medium=career_button)
  + [More](javascript:void(0);)

[Write an Article](https://lekhak.analyticsvidhya.com/write/)

* [Courses](javascript:void(0);)
  + [Certified AI & ML BlackBelt Plus](https://blackbelt.analyticsvidhya.com/plus?utm_source=blog_navbar&utm_medium=blackbelt_button)
  + [Data Science Immersive Bootcamp](https://bootcamp.analyticsvidhya.com/?utm_source=blog_navbar&utm_medium=bootcamp_button)
  + [Machine Learning](https://courses.analyticsvidhya.com/courses/Machine-Learning-Certification-Course-for-Beginners?utm_source=blog_navbar&utm_medium=start_here_button)
  + [Deep Learning](https://courses.analyticsvidhya.com/courses/getting-started-with-neural-networks?utm_source=blog_navbar&utm_medium=start_here_button)
  + [NLP](https://courses.analyticsvidhya.com/courses/Intro-to-NLP?utm_source=blog_navbar&utm_medium=start_here_button)

[All Courses](https://courses.analyticsvidhya.com/collections?utm_source=blog_navbar&utm_medium=all_courses_button)

* [Blogathon](https://datahack.analyticsvidhya.com/blogathon/?utm_source=blog&utm_medium=nav_bar)
* [MasterSeries](https://masterseries.analyticsvidhya.com/?utm_source=AVwebsite&utm_medium=blog-navbar)
* Login/Signup

**The Complete LSTM Tutorial With Implementation**

[**Sukanya Bag**](https://www.analyticsvidhya.com/blog/author/sukanya3/)**— Published On January 31, 2022 and Last Modified On November 22nd, 2022**

[Deep Learning](https://www.analyticsvidhya.com/blog/category/deep-learning/) [Intermediate](https://www.analyticsvidhya.com/blog/category/intermediate/) [Python](https://www.analyticsvidhya.com/blog/category/python-2/)

This article was published as a part of the [Data Science Blogathon](https://datahack.analyticsvidhya.com/contest/blogathon-16/).

Source-Datafloc

Introduction

[Artificial Neural Networks](https://www.analyticsvidhya.com/blog/2021/06/artificial-neural-networks-better-understanding/) (ANN) have paved a new path to the emerging AI industry since decades it has been introduced. With no doubt in its massive performance and architectures proposed over the decades, traditional machine-learning algorithms are on the verge of extinction with deep neural networks, in many real-world AI cases.

But, every new invention in technology must come with a drawback, otherwise, scientists cannot strive and discover something better to compensate for the previous drawbacks. Similarly, Neural Networks also came up with some loopholes that called for the invention of recurrent neural networks.

**Why Recurrent?**

In Neural Networks, we stack up various layers, composed of nodes that contain hidden layers, which are for learning and a dense layer for generating output. But, the central loophole in neural networks is that it does not have memory. Since no memory is associated, it becomes very difficult to work on sequential data like text corpora where we have sentences associated with each other, and even time-series where data is entirely sequential and dynamic.

Here, [Recurrent Neural Networks](https://www.analyticsvidhya.com/blog/2022/03/a-brief-overview-of-recurrent-neural-networks-rnn/) comes to play. RNN addresses the memory issue by giving a feedback mechanism that looks back to the previous output and serves as a kind of memory. Since the previous outputs gained during training leaves a footprint, it is very easy for the model to predict the future tokens (outputs) with help of previous ones.

**Why LSTM when we have RNN?**

A sentence or phrase only holds meaning when every word in it is associated with its previous word and the next one. LSTM, short for Long Short Term Memory, as opposed to RNN, extends it by creating both short-term and long-term memory components to efficiently study and learn sequential data. Hence, it’s great for Machine Translation, Speech Recognition, time-series analysis, etc.

グラフィカル ユーザー インターフェイス

低い精度で自動的に生成された説明

**Ready to start your data science journey?**

Master 23+ tools & learn 50+ real-world projects to transform your career in Data Science.

[Enroll Now](https://blackbelt.analyticsvidhya.com/plus?utm_source=blog_india&utm_medium=desktop_banner_between_articles&utm_campaign=15-Aug-2023||&utm_content=tools)

Tutorial Overview

In this tutorial, we will have an in-depth intuition about LSTM as well as see how it works with implementation!

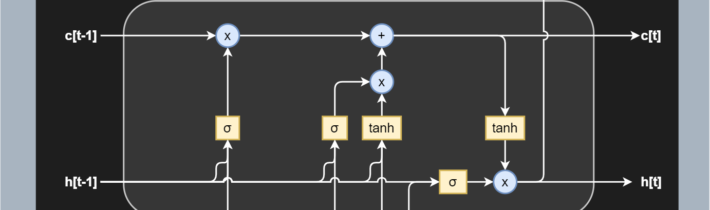
Let’s have a look at what we will cover-

1. A Quick Look into LSTM Architecture
2. Why does LSTM outperform RNN?
3. *Deep Learning* about LSTM gates
4. An Implementation is Necessary!
5. Wrap Up with Bonus Resources

So, let’s dive into the LSTM tutorial!

A Quick Look into LSTM Architecture

Let’s see how a simple LSTM black box model looks-



Source -MachineCurve

 To give a gentle introduction, LSTMs are nothing but a stack of neural networks composed of linear layers composed of weights and biases, just like any other standard neural network. The weights are constantly updated by backpropagation.

Now, before going in-depth, let me introduce a few crucial LSTM specific terms to you-

1. Cell — Every unit of the LSTM network is known as a “cell”. Each cell is composed of 3 inputs —

* *x(t) —*token at timestamp t​
* *h(t*−1) — previous hidden state
* *c(t-1) —*previous cell state,

and 2 outputs —

* *h(t) —*updated hidden state, used for predicting the output
* *c(t)* — current cell state

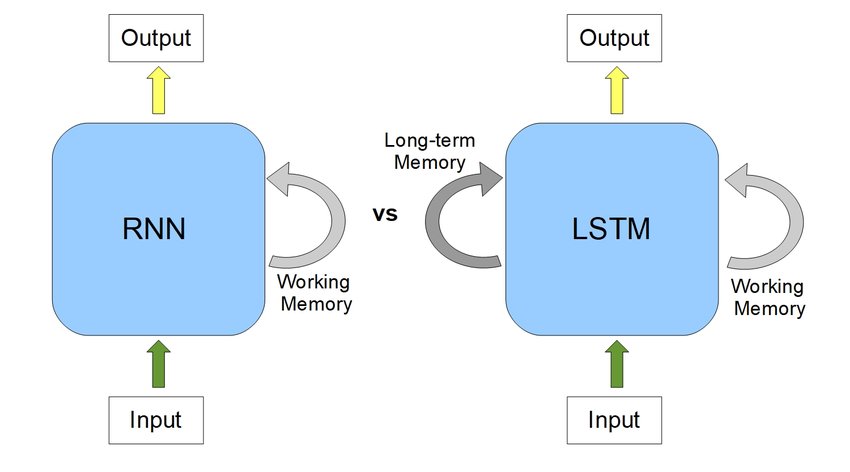
2. Gates — LSTM uses a special theory of controlling the memorizing process. Popularly referred to as gating mechanism in LSTM, what the gates in LSTM do is, store the memory components in analog format, and make it a probabilistic score by doing point-wise multiplication using sigmoid activation function, which stores it in the range of 0–1. Gates in LSTM regulate the flow of information in and out of the LSTM cells.

Gates are of 3 types —

* Input Gate — This gate lets in optional information necessary from the current cell state. It decides which information is relevant for the current input and allows it in.
* Output Gate — This gate updates and finalizes the next hidden state. Since the hidden state contains critical information about previous cell inputs, it decides for the last time which information it should carry for providing the output.
* Forget Gate — Pretty smart in eliminating unnecessary information, the forget gate multiplies 0 to the tokens which are not important or relevant and lets it be forgotten forever.

Why does LSTM outperform RNN?

First, let’s take a comparative look into an RNN and an LSTM-

Source – ResearchGate

RNNs have quite massively proved their incredible performance in sequence learning. But, it has been remarkably noticed that RNNs are not sporty while handling long-term dependencies.

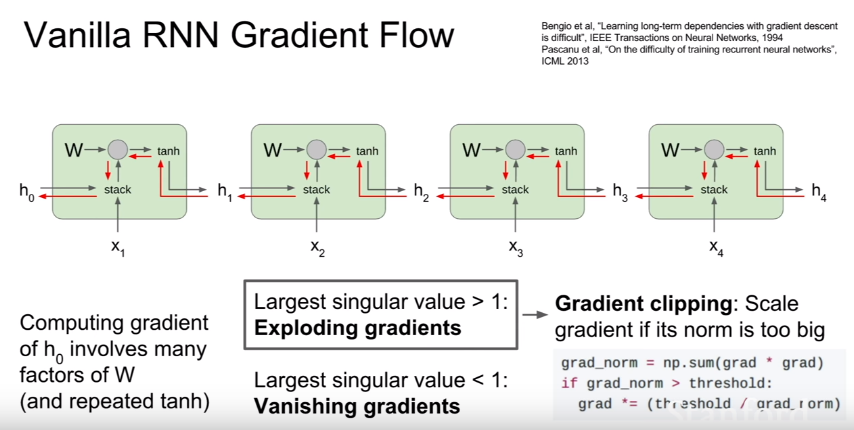
Why so?

**Vanishing Gradient Problem**

Recurrent Neural Networks uses a hyperbolic tangent function, what we call the tanh function. The range of this activation function lies between [-1,1], with its derivative ranging from [0,1]. Now we know that RNNs are a deep sequential neural network. Hence, due to its depth, the matrix multiplications continually increase in the network as the input sequence keeps on increasing. Hence, while we use the chain rule of differentiation during calculating backpropagation, the network keeps on multiplying the numbers with small numbers. And guess what happens when you keep on multiplying a number with negative values with itself? It becomes exponentially smaller, squeezing the final gradient to almost 0, hence weights are no more updated, and model training halts. It leads to poor learning, which we say as “cannot handle long term dependencies” when we speak about RNNs.

**Exploding Gradient Problem**

Similar concept to the vanishing gradient problem, but just the opposite of the process, let’s suppose in this case our gradient value is greater than 1 and multiplying a large number to itself makes it exponentially larger leading to the explosion of the gradient.



Source – Stanford NLP

This also leads to the major issue of Long Term Dependency.

**Long Term Dependency Issue in RNNs**

Let us consider a sentence-

“I am a data science student and I love machine \_\_\_\_\_\_.”

We know the blank has to be filled with ‘learning’. But had there been many terms after “I am a data science student” like, “I am a data science student pursuing MS from University of…… and I love machine \_\_\_\_\_\_”.

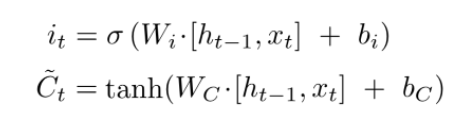
This time, however, RNNS fails to work. Likely in this case we do not need unnecessary information like “pursuing MS from University of……”. What LSTMs do is, leverage their forget gate to eliminate the unnecessary information, which helps them handle long-term dependencies.

*Deep Learning* about LSTM gates

Let’s learn about LSTM gates in-depth.

**The Input Gate**

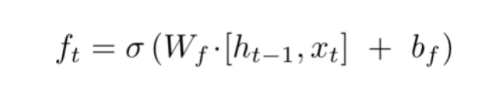
As discussed earlier, the input gate optionally permits information that is relevant from the current cell state. It is the gate that determines which information is necessary for the current input and which isn’t by using the sigmoid activation function. It then stores the information in the current cell state. Next, comes to play the tanh activation mechanism, which computes the vector representations of the input-gate values, which are added to the cell state.



Source – Stanford NLP

**The Forget Gate**

We already discussed, while introducing gates, that the hidden state is responsible for predicting outputs. The output generated from the hidden state at (t-1) timestamp is h(t-1). After the forget gate receives the input x(t) and output from h(t-1), it performs a pointwise multiplication with its weight matrix with an add-on of sigmoid activation which generates probability scores. These probability scores help it determine what is useful information and what is irrelevant.



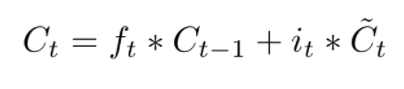
Source – Stanford NLP

**Cell State Update Mechanism**

Replacing the new cell state with whatever we had previously is not an LSTM thing! An LSTM, as opposed to an RNN, is clever enough to know that replacing the old cell state with new would lead to loss of crucial information required to predict the output sequence.

That’s where LSTM beats RNN, remember?!

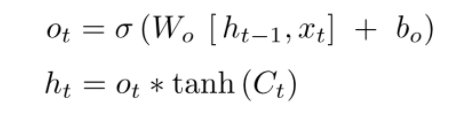
By now, the input gate remembers which tokens are relevant and adds them to the current cell state with tanh activation enabled. Also, the forget gate output, when multiplied with the previous cell state C(t-1), discards the irrelevant information. Hence, combining these two gates’ jobs, our cell state is updated without any loss of relevant information or the addition of irrelevant ones.

Source – Stanford NLP

**Output Gate**

This gate, which pretty much clarifies from its name that it is about to give us the output, does a quite straightforward job. The output gate decides what to output from our current cell state. The output gate, also has a matrix where weights are stored and updated by backpropagation. This weight matrix, takes in the input token x(t) and the output from previously hidden state h(t-1) and does the same old pointwise multiplication task. However, as said earlier, this takes place on top of a sigmoid activation as we need probability scores to determine what will be the output sequence.

After we get the sigmoid scores, we simply multiply it with the updated cell-state, which contains some relevant information required for the final output prediction. A final tanh multiplication is applied at the very last, to ensure the values range from [-1,1], and our output sequence is ready!



An Implementation is Necessary!

Step 1: Import the dependencies and code the activation functions-

import random

import numpy as np

import math

def sigmoid(x):

return 1. / (1 + np.exp(-x))

def sigmoid\_derivative(values):

return values\*(1-values)

def tanh\_derivative(values):

return 1. - values \*\* 2

# createst uniform random array w/ values in [a,b) and shape args

def rand\_arr(a, b, \*args):

np.random.seed(0)

return np.random.rand(\*args) \* (b - a) + a

Step 2: Initializing the biases and weight matrices

class LstmParam:

def \_\_init\_\_(self, mem\_cell\_ct, x\_dim):

self.mem\_cell\_ct = mem\_cell\_ct

self.x\_dim = x\_dim

concat\_len = x\_dim + mem\_cell\_ct

# weight matrices

self.wg = rand\_arr(-0.1, 0.1, mem\_cell\_ct, concat\_len)

self.wi = rand\_arr(-0.1, 0.1, mem\_cell\_ct, concat\_len)

self.wf = rand\_arr(-0.1, 0.1, mem\_cell\_ct, concat\_len)

self.wo = rand\_arr(-0.1, 0.1, mem\_cell\_ct, concat\_len)

# bias terms

self.bg = rand\_arr(-0.1, 0.1, mem\_cell\_ct)

self.bi = rand\_arr(-0.1, 0.1, mem\_cell\_ct)

self.bf = rand\_arr(-0.1, 0.1, mem\_cell\_ct)

self.bo = rand\_arr(-0.1, 0.1, mem\_cell\_ct)

# diffs (derivative of loss function w.r.t. all parameters)

self.wg\_diff = np.zeros((mem\_cell\_ct, concat\_len))

self.wi\_diff = np.zeros((mem\_cell\_ct, concat\_len))

self.wf\_diff = np.zeros((mem\_cell\_ct, concat\_len))

self.wo\_diff = np.zeros((mem\_cell\_ct, concat\_len))

self.bg\_diff = np.zeros(mem\_cell\_ct)

self.bi\_diff = np.zeros(mem\_cell\_ct)

self.bf\_diff = np.zeros(mem\_cell\_ct)

self.bo\_diff = np.zeros(mem\_cell\_ct)

Step 3: Multiplying forget gate with last cell state to forget irrelevant tokens

#stacking x(present input xt) and h(t-1)

xc = np.hstack((x, h\_prev))

#dot product of Wf(forget weight matrix and xc +bias)

self.state.f = sigmoid(np.dot(self.param.wf, xc) + self.param.bf)

#finally multiplying forget\_gate(self.state.f) with previous cell state(s\_prev)

#to get present state.

self.state.s = self.state.g \* self.state.i + s\_prev \* self.state.f

Step 4: Sigmoid Activation decides which values to take in and tanh transforms new tokens to vectors

#xc already calculated above

self.state.i = sigmoid(np.dot(self.param.wi, xc) + self.param.bi)

#C(t)

self.state.g = np.tanh(np.dot(self.param.wg, xc) + self.param.bg)

Step 5: Calculate the present cell state

#to calculate the present state

self.state.s = self.state.g \* self.state.i + s\_prev \* self.state.f

Step 6: Calculate the output state

#to calculate the output state

self.state.o = sigmoid(np.dot(self.param.wo, xc) + self.param.bo)

#output state h

self.state.h = self.state.s \* self.state.o

So, this is how a single node of LSTM works! You can check the entire implementation [here](https://gist.github.com/sukanyabag/fec853cee5c77ec362820e2131777911).

Wrap Up with Bonus Resources

Hope you have clearly understood how LSTM works and why is it better than RNN! If you are still curious and want to explore more, you can check on these awesome resources –

1. [Colah’s Blog](https://colah.github.io/posts/2015-08-Understanding-LSTMs/)

2. [Stanford Lectures](https://web.stanford.edu/class/cs224n/readings/cs224n-2019-notes05-LM_RNN.pdf)

Conclusion

Long Short Term Memories are very efficient for solving use cases that involve lengthy textual data. It can range from speech synthesis, speech recognition to machine translation and text summarization. I suggest you solve these use-cases with LSTMs before jumping into more complex architectures like Attention Models.

Read more articles on [LSTM](https://www.analyticsvidhya.com/?s=LSTM).

If you liked this article, feel free to share it with your network😄. For more articles about Data Science and AI, follow me on [Medium](https://sukanyabag.medium.com/) and [LinkedIn](https://www.linkedin.com/in/sukannya/).

Happy Machine Learning.

**The media shown in this article is not owned by Analytics Vidhya and are used at the Author’s discretion.**

**Related**

[](https://www.analyticsvidhya.com/blog/2023/02/neuro-symbolic-ai-enhancing-common-sense-in-ai/?relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0)

[Neuro Symbolic AI: Enhancing Common Sense in AI](https://www.analyticsvidhya.com/blog/2023/02/neuro-symbolic-ai-enhancing-common-sense-in-ai/?relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=0)

[](https://www.analyticsvidhya.com/blog/2023/02/introduction-to-neural-network-build-your-own-network/?relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1)

[Introduction to Neural Network: Build your own Network](https://www.analyticsvidhya.com/blog/2023/02/introduction-to-neural-network-build-your-own-network/?relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=1)

[](https://www.analyticsvidhya.com/blog/2020/09/quick-history-neural-networks/?relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2)

[A Quick History of Neural Networks](https://www.analyticsvidhya.com/blog/2020/09/quick-history-neural-networks/?relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2&relatedposts_hit=1&relatedposts_origin=87822&relatedposts_position=2)

[blogathon](https://www.analyticsvidhya.com/blog/tag/blogathon/)[Neural network](https://www.analyticsvidhya.com/blog/tag/neural-network/)

[グラフィカル ユーザー インターフェイス, アプリケーション, Teams

自動的に生成された説明](https://community.analyticsvidhya.com/c/live-events/datahour-exploring-the-potential-of-generative-ai/?utm_source=sidebanner&utm_medium=blog-outside-india&utm_campaign=https%3A%2F%2Fwww.analyticsvidhya.com%2Fblog%2F2022%2F01%2Fthe-complete-lstm-tutorial-with-implementation%2F)

Natural Language Processing

[**Become a full stack data scientist**](https://blackbelt.analyticsvidhya.com/plus?utm_source=ReadingList&utm_medium=blog)

Introduction to NLP

Text Pre-processing

NLP Libraries

Regular Expressions

String Similarity

Spelling Correction

Topic Modeling

Text Representation

Information Retrieval System

Word Vectors

Word Senses

Dependency Parsing

Language Modeling

Getting Started with RNN

Different Variants of RNN

[**Shortcomings of RNN**](https://www.analyticsvidhya.com/blog/2021/06/a-visual-guide-to-recurrent-neural-networks/?utm_source=reading_list&utm_medium=https://www.analyticsvidhya.com/blog/2022/01/the-complete-lstm-tutorial-with-implementation/)[**What is Long Short Term Memory (LSTM)**](https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/?utm_source=reading_list&utm_medium=https://www.analyticsvidhya.com/blog/2022/01/the-complete-lstm-tutorial-with-implementation/)[**Implementing LSTM**](https://www.analyticsvidhya.com/blog/2021/06/lstm-for-text-classification/?utm_source=reading_list&utm_medium=https://www.analyticsvidhya.com/blog/2022/01/the-complete-lstm-tutorial-with-implementation/)[**Build Your Own Fake News Classification Model**](https://www.analyticsvidhya.com/blog/2021/07/detecting-fake-news-with-natural-language-processing/?utm_source=reading_list&utm_medium=https://www.analyticsvidhya.com/blog/2022/01/the-complete-lstm-tutorial-with-implementation/)[**What is Gated Recurrent Unit (GRU)?**](https://www.analyticsvidhya.com/blog/2021/03/introduction-to-gated-recurrent-unit-gru/?utm_source=reading_list&utm_medium=https://www.analyticsvidhya.com/blog/2022/01/the-complete-lstm-tutorial-with-implementation/)[**Implementing GRU**](https://www.analyticsvidhya.com/blog/2018/04/a-comprehensive-guide-to-understand-and-implement-text-classification-in-python/?utm_source=reading_list&utm_medium=https://www.analyticsvidhya.com/blog/2022/01/the-complete-lstm-tutorial-with-implementation/)

Machine Translation and Attention

Self Attention and Transformers

Transfomers and Pretraining

Question Answering

Text Summarization

Named Entity Recognition

Coreference Resolution

Audio Data

ASR

Audio Separation

Chatbot

Auto NLP

**About the Author**

[Sukanya Bag](https://www.analyticsvidhya.com/blog/author/sukanya3/)

**Our Top Authors**

* 

[**view more**](https://www.analyticsvidhya.com/blog/creators/?utm-medium=blog-footer&utm_source=top-authors)

**DownloadAnalytics Vidhya App for the Latest blog/Article**

**Previous Post**[Multi-label Text Classification Using Transfer Learning powered by “Optuna”](https://www.analyticsvidhya.com/blog/2022/01/multi-label-text-classification-using-transfer-learning-powered-by-optuna/)

**Next Post**[Text Analysis app using Spacy, Streamlit, and Hugging face Spaces](https://www.analyticsvidhya.com/blog/2022/01/text-analysis-app-using-spacy-streamlit-and-hugging-face-spaces/)

Leave a Reply**Your email address will not be published. Required fields are marked \***

フォームの始まり







 Notify me of follow-up comments by email.

 Notify me of new posts by email.

Submit

フォームの終わり

Top Resources

[[グラフィカル ユーザー インターフェイス が含まれている画像

自動的に生成された説明](https://www.analyticsvidhya.com/blog/2023/08/ai-skills/)](https://www.analyticsvidhya.com/blog/2023/08/ai-skills/)

[Top 10 AI Skills to Have for Landing a Job..](https://www.analyticsvidhya.com/blog/2023/08/ai-skills/)

[avcontentteam -](https://www.analyticsvidhya.com/blog/author/avcontentteam/)AUG 28, 2023

[[テキスト が含まれている画像

自動的に生成された説明](https://www.analyticsvidhya.com/blog/2023/08/must-listen-data-science-podcasts-for-every-enthusiast/)](https://www.analyticsvidhya.com/blog/2023/08/must-listen-data-science-podcasts-for-every-enthusiast/)

[10 Must-Listen Data Science Podcasts for Every Enthusiast](https://www.analyticsvidhya.com/blog/2023/08/must-listen-data-science-podcasts-for-every-enthusiast/)

[avcontentteam -](https://www.analyticsvidhya.com/blog/author/avcontentteam/)AUG 28, 2023

[[ダイアグラム

低い精度で自動的に生成された説明](https://www.analyticsvidhya.com/blog/2023/08/ai-image-generators/)](https://www.analyticsvidhya.com/blog/2023/08/ai-image-generators/)

[10 Best AI Image Generator Tools to Use in 2023](https://www.analyticsvidhya.com/blog/2023/08/ai-image-generators/)

[avcontentteam -](https://www.analyticsvidhya.com/blog/author/avcontentteam/)AUG 17, 2023

[[グラフ

自動的に生成された説明](https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/)](https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/)

[Everything you need to Know about Linear Regression!](https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/)

[KAVITA MALI -](https://www.analyticsvidhya.com/blog/author/kavita65/)OCT 04, 2021

[](https://www.analyticsvidhya.com/)

Download App

* Analytics Vidhya

[About Us](https://www.analyticsvidhya.com/about/)[Our Team](https://www.analyticsvidhya.com/team/)[Careers](https://www.analyticsvidhya.com/careers/)[Contact us](https://www.analyticsvidhya.com/contact/)

* Data Scientists

[Blog](https://www.analyticsvidhya.com/blog/)[Hackathon](https://datahack.analyticsvidhya.com/)[Join the Community](https://community.analyticsvidhya.com/?utm_source=Blogs&utm_medium=Footer)[Apply Jobs](https://jobsnew.analyticsvidhya.com/)

* Companies

[Post Jobs](https://www.analyticsvidhya.com/corporate/)[Trainings](https://courses.analyticsvidhya.com/)[Hiring Hackathons](https://datahack.analyticsvidhya.com/)[Advertising](https://www.analyticsvidhya.com/contact/)

* Visit us

© Copyright 2013-2023 Analytics Vidhya.

[Privacy Policy](https://www.analyticsvidhya.com/privacy-policy/) [Terms of Use](https://www.analyticsvidhya.com/terms/) [Refund Policy](https://www.analyticsvidhya.com/refund-policy/)

We use cookies on Analytics Vidhya websites to deliver our services, analyze web traffic, and improve your experience on the site. By using Analytics Vidhya, you agree to our [Privacy Policy](https://www.analyticsvidhya.com/privacy-policy/) and [Terms of Use](https://www.analyticsvidhya.com/terms/).Accept