INTERNSHIP REPORT

Submitted by :- Pratyush Prateem Bhatta

Roll no :- 170710007044

College :- Jorhat Engineering College

Acknowledgement

I would like to thank ASTU and IIIT Guwahati for teaching the new concepts on ML and AI, different methods and techniques, also like to thank ECKOVATION for providing such a nice platform where we can test our skills and learn more. Thank you for giving me the opportunity to work on this project. During the project I have got to learn a lot of things and came to know different method

**Responsibilities:**

**Topic: INFOSYS Stock Price Prediction**

**Problem Statement:** Importing necessary libraries,

Import Dataset from the given link, Perform exploratory data analysis and Creating Training and Test Data.

**Data Set Link:** https://www.kaggle.com/rohanrao/nifty50-stock-market-data?select=INFY.csv

**Solution:**

* **Module used :**

**Numpy, Matplotlib, Pandas, Tensorflow,**

**Sklearn**

* **The given set of data is divided into two half , such that 75% training and 25% testing.**

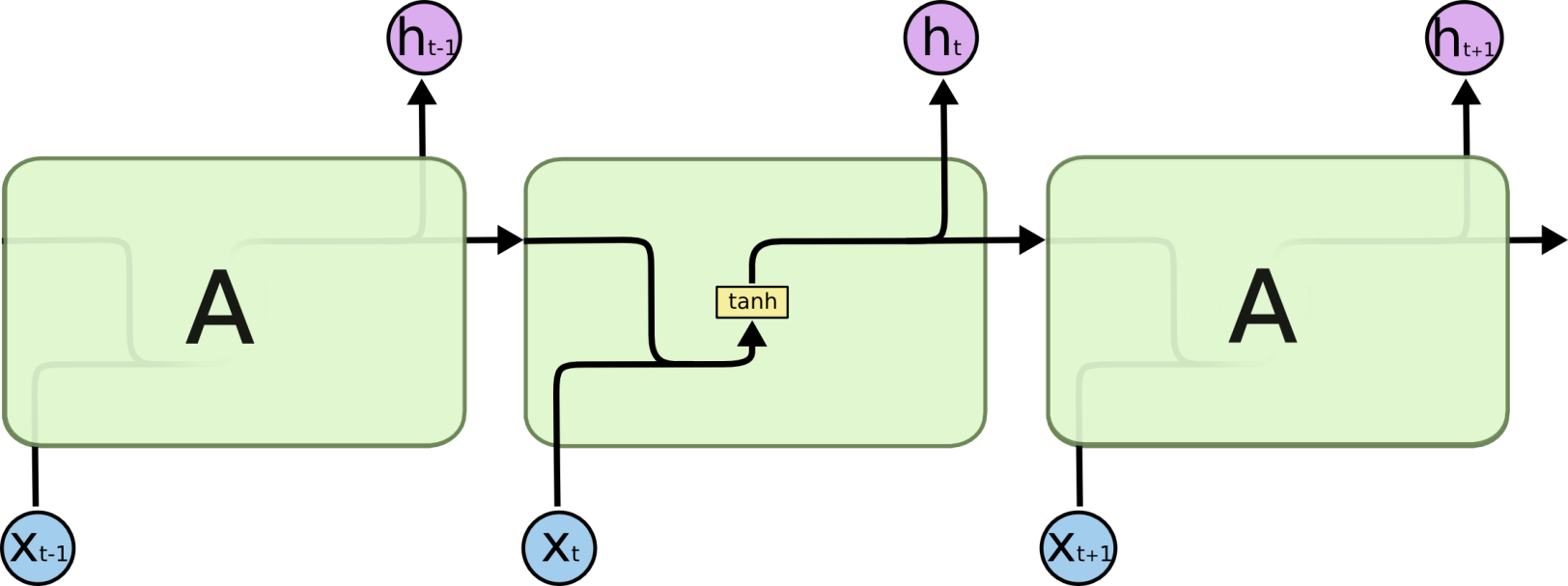
**LSTM Networks**

## LSTM Networks

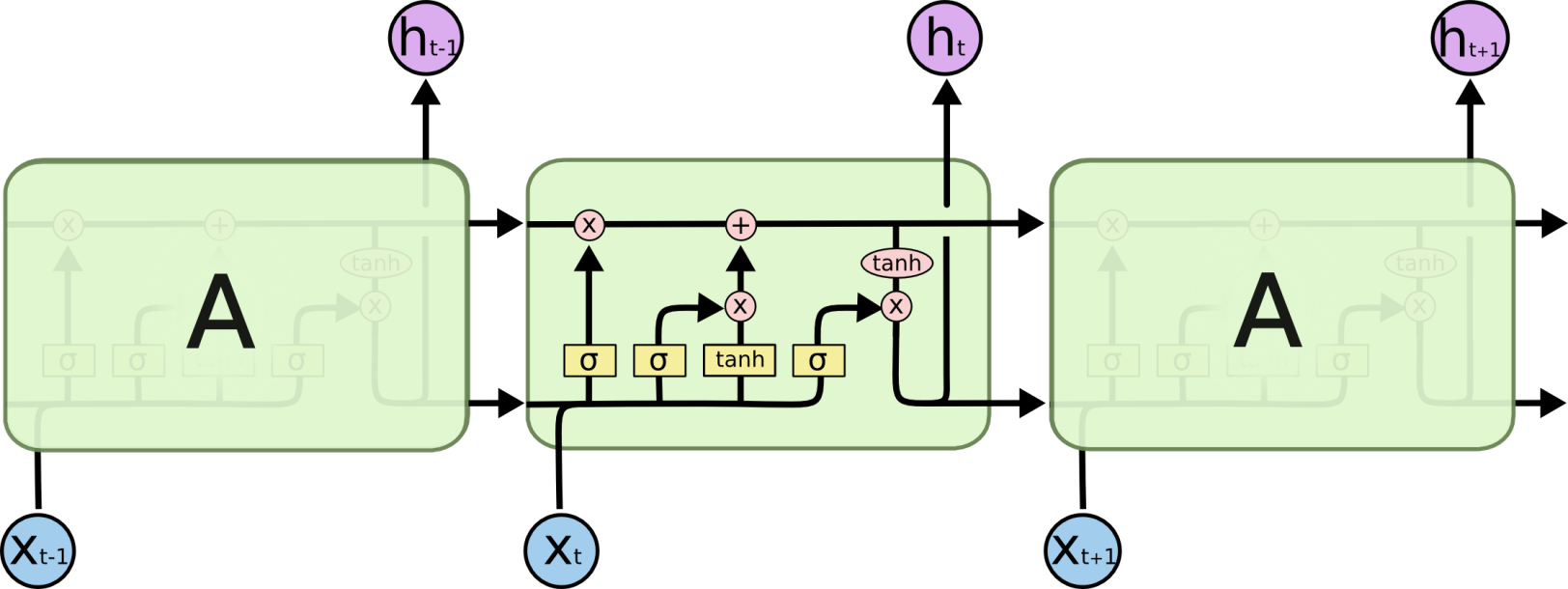
Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. They were introduced by [Hochreiter & Schmidhuber (1997)](http://www.bioinf.jku.at/publications/older/2604.pdf), and were refined and popularized by many people in following work.[1](http://colah.github.io/posts/2015-08-Understanding-LSTMs/#fn1) They work tremendously well on a large variety of problems, and are now widely used.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.



**The repeating module in a standard RNN contains a single layer.**



**The repeating module in an LSTM contains four interacting layers.**

Don’t worry about the details of what’s going on. We’ll walk through the LSTM diagram step by step later. For now, let’s just try to get comfortable with the notation we’ll be using.



In the above diagram, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes are learned neural network layers. Lines merging denote concatenation, while a line forking denote its content being copied and the copies going to different locations.

* **the MinMaxScaler() function to scale the training dataset to range 0-1 and reduce their difference.**
* **Building the LSTM model, in five step,**

**regressior.add(LSTM(units=60,activation='relu',return\_sequences = True, input\_shape=(x\_train.shape[1],5)))**

**regressior.add(Dropout(0.05))**

**regressior.add(LSTM(units=60,activation='relu',return\_sequences = True))**

**regressior.add(Dropout(0.05))**

**regressior.add(LSTM(units=80,activation='relu',return\_sequences = True))**

**regressior.add(Dropout(0.05))**

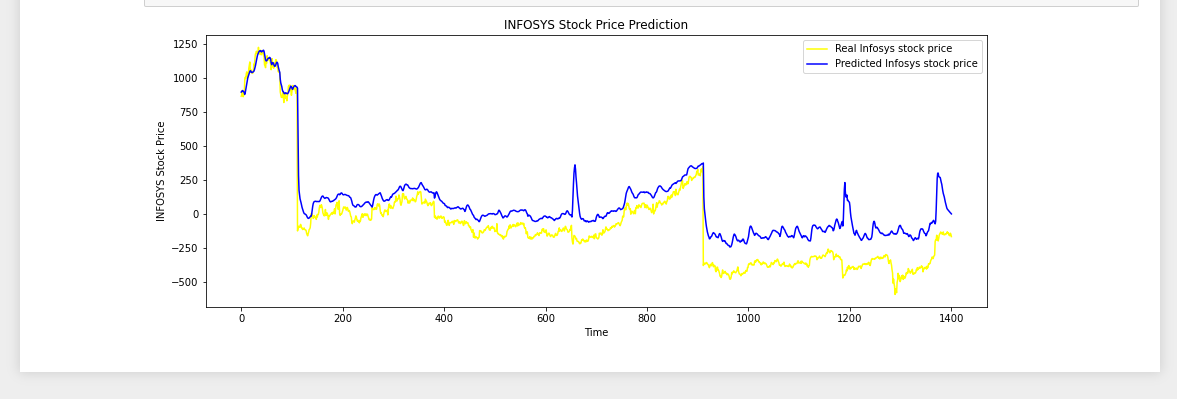
**regressior.add(LSTM(units=120,activation='relu'))**

**regressior.add(Dropout(0.05))**

**regressior.add(Dense(units=1))**

* **Training the model on the build-up model with the training data set**
* **Testing the build model on the testing dataset. Result was not good at the beginning . Need to increase epochs (=50), reduce loss , and also reduce dropouts(=0.05). Finally obtained a good result on the data set.**

**Final prediction, predicted by the model**

****