# Recurrent Neural Network for Short-Term Traffic flow Prediction

Atif Mukhtiar
Nebyu Derebe
Micheal Solomon

Date:

**NOTES:**

1. **THIS is a SAMPLE PROJECT REPORT. In case you work on this Topic, please DO NOT copy anything from this REPORT.**
2. **The Project Report should be between 3-6 pages.**
3. **You can add sub-topics focusing on your "Novelty" of the work in your project as appropriate.**

Abstract

We develop the Recurrent Neural Network model (LSTM) to predict the traffic flows. LSTM systems are appropriate to characterizing, handling and making forecasts dependent on time series information. We predict traffic flow on the basis of previous record street by street. In our dataset/record, we are presenting SITE_CODE as the street. So we classify our data using the SITE_CODE and feed the data to the model to predict the traffic of that street (SITE_CODE).

## 1. Introduction

### 1.1. Traffic flow Prediction

As we all know that we have real-time traffic predictor Google Maps which gives you live traffic at the moment but machine learning can give you real-time (15-40 minute) forecasting gives travelers the ability to choose better routes and authorities the ability to manage the transportation system.

### 1.2. Recurrent Neural Network

Recurrent neural networks (RNNs) are a type of artificial neural network that adds additional weights to the network to create cycles in the network graph in an effort to maintain an internal state. RNN is a powerful and robust type of neural networks and belongs to the most promising algorithms out there at the moment because they are the only ones with an internal memory.

They are networks with loops in them, allowing information to persist. RNN's are relatively old, like many other deep learning algorithms. They were initially created in the 1980's, but can only show their real potential since a few years, because of the

increase in available computational power, the massive amounts of data that we have nowadays and the invention of LSTM in the 1990's.

Because of their internal memory, RNN's are able to remember important things about the input they received, which enables them to be very precise in predicting what's coming next.

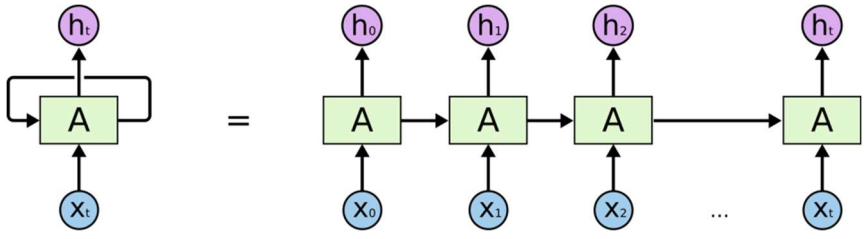
Recurrent Neural Networks produce predictive results in sequential data that other algorithms can't.

But when do you need to use a Recurrent Neural Network ?

"Whenever there is a sequence of data and that temporal dynamics that connects the data is more important than the spatial content of each individual frame."

– Lex Fridman (MIT)

Traffic flow forecast has a long history is as yet a troublesome issue due to naturally profoundly nonlinear and stochastic attributes of complex transportation frameworks.

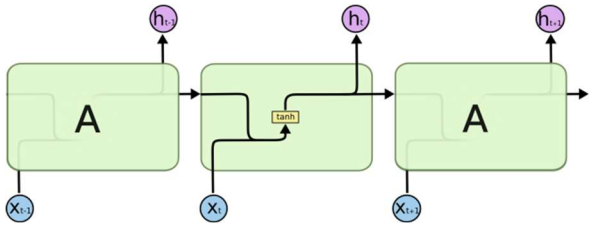



## 1.3. Long short-term memory

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), 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.

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.



## 2. Dataset

### 2.1. Problem with dataset

In order to predict any problem, we need to have enough dataset to train the model. Let's say we want to predict a street traffic. We need to have enough of record to train the model for that street. Now the problem with data we had before is that there were not any multiple records of the street to train the model.

### 2.2. Dataset

Now we have our dataset with multiple records of every street. So we can train our model that dataset and predict the traffic of the street.

## 3. Data Preparation

### 3.1. Problem in data preparation

In our case, we are grouping the traffic data street by street. To use this grouped dataset we need to have categorical LSTM algorithm. Now the problem is LSTM doesn't work with categorical data like predicting traffic flow based on a streets or locations or cities. So in order to do categorical LSTM we have to train the algorithm separately.

### 3.2. Data preparation

We can do some workaround to group data so classify the data by creating files of each street. Each file will represent the street separately. Now we can train our model for each street using file as street data.

## 4. Training and Result

First we were facing over-fitting as shown below Fig1, and then we start drawing the early-stopping graph to get the max epoch and we got the result as shown in Fig 2

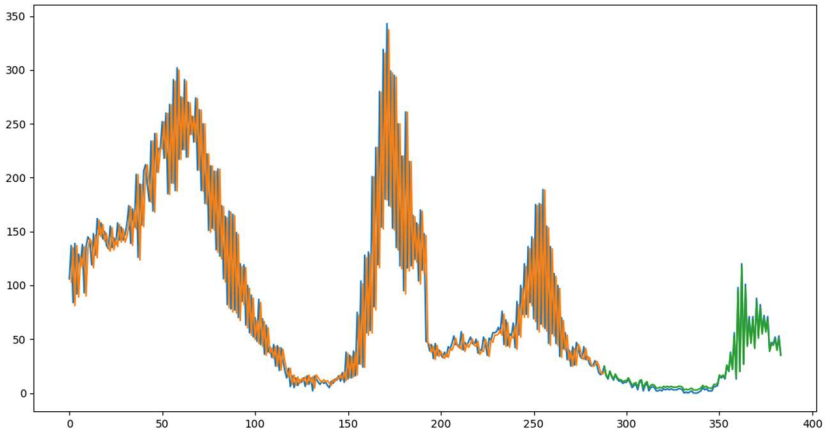



Fig1 over-fitting

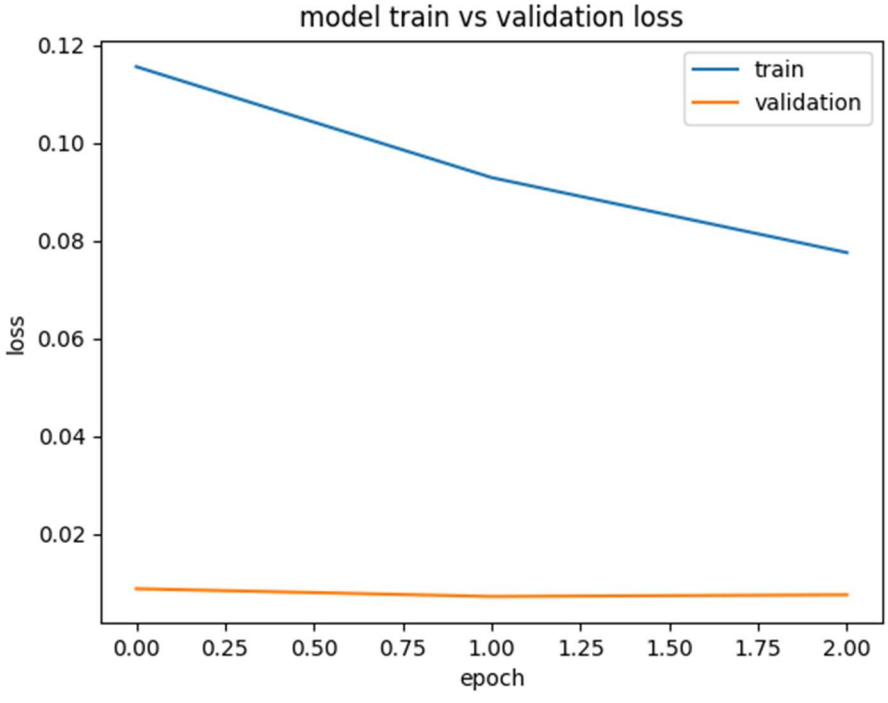



Fig2a Early stopping

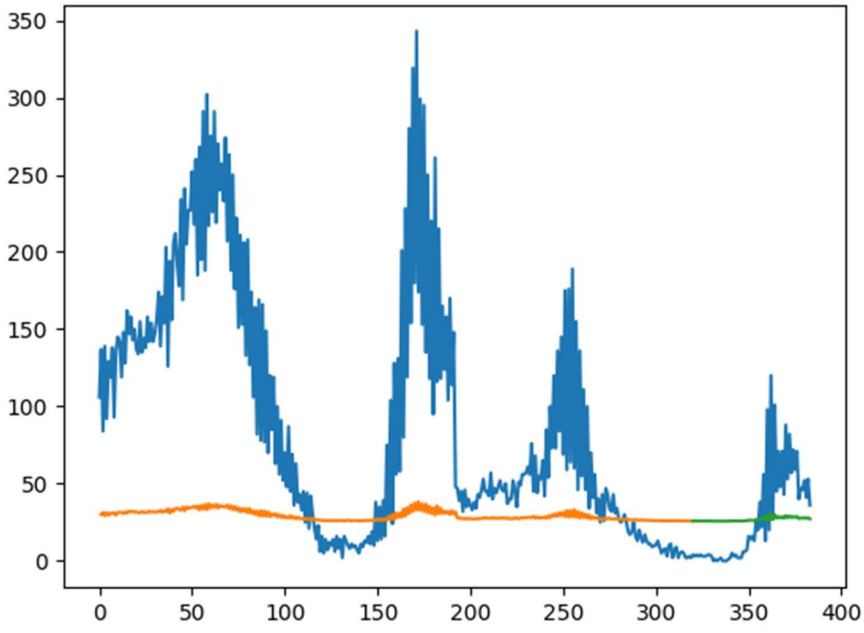



Fig2b  training and evaluation result

We limit the epochs to 2 and we got the result we as shown above in Fig2a and we got the result by plotting the training and evaluation dataset along with the actual dataset as shown in Fig2b.

At first we thought it's a bad seeing the result of training and evaluation but it turns out we got the good prediction result like the one in below Fig3.

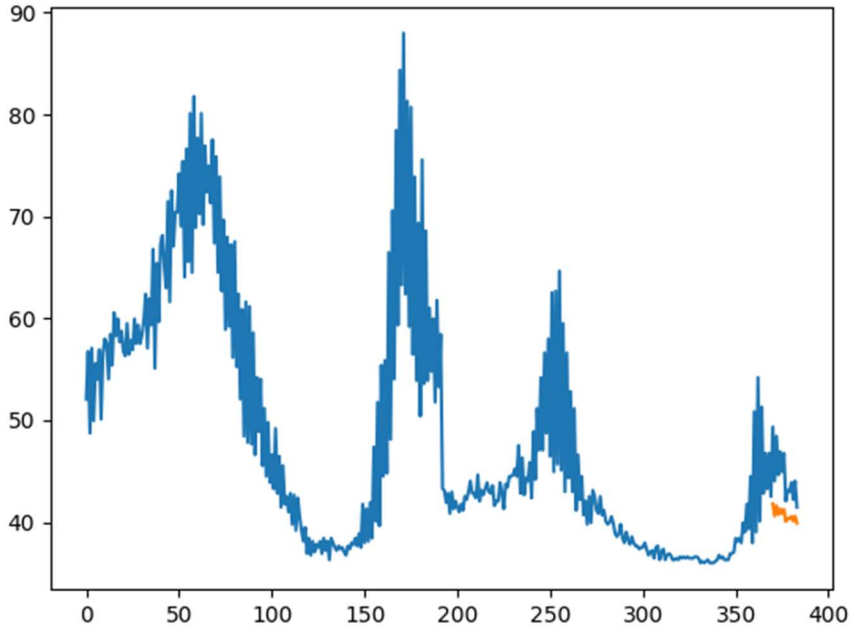



Fig3

## 5. Setup

It got two application one which

- The prediction application
- Client application

### 5.1. The prediction application

The application got 3 files

- Let'sGo
- Training
- Forecasting

- To train the algorithm run Training.py

- To test prediction run Forecasting.py

### 5.2. The client application

The Client application got war java(Spring) file that can be run in tomcat server.

- Run the java file
- To see the congestion select source and destination and the system shows the in Red line