Capstone Project Proposal

Louis Tian

August 28, 2017

1 Domain Background

Since the dawn of humanity, we have been seeking the future telling crystal ball. There are countless applications ranging from traditional science, finance to robotic. In the last five year, the world has been amazed by the rapid development machine learning, especially the success achieved by deep learning algorithms. Despite its practical importance, during the last five years, relatively little attention has been paid to applied those new advancements in deep learning technique to time series prediction problems.

This proposed project is an attempt to applied some of those learning and techniques from deep learning to time series problem, predicting Wikipedia web page traffic based on its historical traffic volume.

1.1 Challenges

1.1.1 scarce of data

For one, compared to other fields of machine learning, data in time series problem are relatively scarce. One of the most successful areas in machine learning is computer vision, the prevalence of smart phone and digital camera created a huge wealth of data for computer vision, easily available to researchers. On the other hand, time series data are much harder to come by. For example, financial market data are not only limited and prohibitively expensive to acquire. Even for the more accessible type of time series data, like server log, the relative size of available data is still small due to fundamental of time series data generating process. It takes strictly a day to produce a single data point daily web traffic.

1.1.2 Intrinsically difficulty

Time series problems are intrinsically harder than computer vision or speech recognition, in the sense that those are cognitive tasks that can be easily solved by a child. On the other hand, time series prediction problem has always been on the frontier of human's ability. While a three-year-old could tell apart a dog from a cat, only the very best of us are able to make an accurate prediction of the future.

1.1.3 Non-stationary & concept drifting

In addition to the above problem faced by time series problems in general, there are additional challenges that is specific to the type of data we are facing here.

Web traffic data often contains extreme values (i.e. spikes) that are highly stochastic. For example, whenever a TV show releases new season, one would see a spike of traffic. This kind of spikes are impossible to predict using the historical data along and because those spikes are generally quite extreme, it could cause problem for certain machine learning that are sensitive to outliers.

Web traffic typically exhibit short term seasonality but are not non-stationary in the longer terms due to concept drift over time. For example, one would typically see a higher traffic for pages describing causal topics (i.e. TV shows) on a weekend. But the weekly mean traffic will change over time. For example, you would see higher mean traffic during the time a TV show is airing. This kind of problem is typically dealt with change point detecting algorithm.

2 Problem Statement

2.1 Descriptions

The proposed project attempts to predict the page visit traffic for english Wikipedia web pages based on their historical daily visits data.

2.2 Definitions

Let x_i^t be web traffic on day t for web page i, and use $\mathbf{x}^{p,q}$ to represent the web traffic from time p to time q for all pages.

$$\mathbf{x}_{p,q} = \begin{bmatrix} x_1^p & x_1^{p+1} & \dots & x_1^t \\ x_2^p & x_2^{p+1} & \dots & x_2^t \\ \vdots & \vdots & \ddots & \vdots \\ x_m^t & x_m^{p+1} & \dots & x_m^t \end{bmatrix}$$

where m is the total number of web pages in the dataset.

Then the learning problem can be expressed formally expressed as finding a function f such that:

$$f = \underset{f}{\operatorname{argmin}} \quad (\mathbf{x}_{t+1,t+n}, f(\mathbf{x}_{0,t}))$$

where n is the days of prediction into the future. For example, when $n = 1$, only the next day is predicted. And L is the loss funcation that is discussed later in the "Evaluation Metrics" section.

3 Datasets and Inputs

The data set used for this project is available at Kaggle.com. The original data set includes daily web traffic data for a total of wikipedia 145,063 pages for the period from 01/07/2015 to 31/12/2016.

This data set does not distinguish between traffic values of zero and missing values. A missing value may mean the traffic was zero or that the data is not available for that day.

The following table is a snapshot the dataset (first five days of the first five pages).

index	2015-07-01	2015-07-02	2015-07-03
_zh.wikipedia.org_mobile-web_all-agents	89.0	80.0	70.0
_zh.wikipedia.org_all-access_all-agents	342.0	283.0	239.0
Jackie_Coogan_en.wikipedia.org_all-access_spider	36.0	27.0	26.0

4 Solution Statement

The general goal of this project proposed to transform this traditional time series problem into a supervised learning problem and use neural nets to forecast the web traffic for a fixed length out of sample period.

4.1 Architecture of Neural Net Work

There are at least two standard neural network architecture can be used to formulate this problem. One is a standard fully connected neural networks and another is a recurrent neural network.

4.2 Rolling forcast or Batch forcast

Regardless what type of neural network architecture we will be using, there is a choice of how the forcast is predicted.

One way is to make prediction one period at a time and rolls the prediction model forward for multi period prediction. Standard LSTM recurrent network fits well within this class. Alternatively, a fully connected neural network with one-dimensional output can used.

Another solution is to build a neural network that makes predictions for n days in advance directly. Examples of this class of model include Sequence-to-Sequence recurrent and fully connected neural network with two-dimensional outputs.

4.3 One-to-one, Many-to-many and Many-to-one

A neural network can be built for each series individually, or we can train a single model for all of the pages simultaneously.

One obvious upside of the many-to-x approaches over the one-to-one approach is that the models might be able to discover some relationships between pages and make better predictions based on that. On the other hand, that formulation significantly reduces the number of samples can be used. The dataset used in the project contains only 550 days of web traffic. This means if the many-to-x approach is used, the maximum number of sample size is 550 (when the rolling window size is 1). This is too small for training neural networks.

Also, it is practically impossible to use a many-to-x as the memory size required for a many-to-x model required increases exponentially as the number of web pages increases.

Traditionally, univariate time series solutions like ARMIA, a rolling window is typically used for model fitting. Similarly, this technique can also be used when a fully connected neural network is used. However, it can be problematic is many-to-x approach is used. That is because some page existed for less than 550 days. In this case, web traffic is represented as NaN, which can drop when a one-to-one approach is used but must be filled if a many-to-many approach is taken, other wise significant sacrifices in the size of training data must be made. This is not acceptable as the training data will be already limited by the choice of many-to-x approach as discussed above.

One the other hand, RNN is designed to handle variable input size and handles this type of problems natural.

4.4 The proposed approach

Given the discussion above, I will use a sequence-to-sequence recurrent network and take a one-to-ones type formulation. Each row of the dataset (i.e. a single page) will be feed to network one at a time.

5 Benchmark Model

The traditional approach to time series problem is to fit the time series with ARIMA model. ARIMA model is considered as the most general form time series model in traditional statistically methods. However, the general form of ARIMA requires a significant amount of hyper-parameter tuning makes is unsuitable for our purpose as the benchmark model. For benchmarking, we will restrict ourselves to ARIMA(1,1,0), which assumes the time series is generated by a random walk process. Under this setting, the best prediction is simply the latest observed value in the time series.

$$y_{\hat{t+1}} = f(,) = (,)$$

6 Evaluation Metrics

The official evaluation criteria used in the Kaggle competition is Symmetric Mean Absolute Percentage Error (SMAPE) defined as:

$$SMAPE = \frac{1}{n} \sum_i \frac{|\hat{y}_i - y_i|}{|\hat{y}_i| + |y_i|}$$

One particular advantage of using this metrics is that it is robust again the spikes, as the value of SMAPE is that it is capped between 0 to 1. On the downside, this metric penalized under prediction heavier than over prediction.

7 Project Design & Workflow

The overall work flow of the proposed project is summarised in this section.

1. Data Exploration and Visualisation Calculate basic statistics about the data set and produce visualisations to understand the fundamental characteristics of the dataset.
2. Data Preparation Reserve the last 14 days of data as the validation data set
3. Calculate the benchmark model performance metric ARIMA(1,1,0) benchmark metrics should be calculated.
4. Build the models in Tensorflow The model will be built in Tensorflow for performance and flexibility. Tensorflow has high level API for sequence-to-sequence model which can be used directly.
5. Hyperparameter tuning with Grid Search Given that only 100 pages will be used, it is computationally feasible to tune our parameter by using simple grid search.
6. Model Evaluation Train the model on the entire training set and evaluate the performance by making the 14 days out of sample prediction. Using the predictions and the validation set to calculate the SMAPE score and compare against the benchmark value.