AI-based TCP Performance Modelling

B.Piger and B.V.Ghita

Centre for Information Security and Network Research,
University of Plymouth, Plymouth, United Kingdom
e-mail: info@cscan.org

Abstract

This project aims to analyse the efficiency of artificial neural networks when modelling the performance of Transmission Control Protocol (TCP). First of all we need to understand what TCP performance depends on. We have therefore researched existing mathematical models. We have chosen to implement the Cardwell (2000) model. By using this model we can retrieve the data transfer time thanks to the timeout, loss rate, round trip time and initial congestion window. We have next built a network features dataset thanks to tcpdump and tcptrace. This set has enabled us to test our mathematical model and to build our artificial neural network. We have chosen to include the result of the mathematical model, the data packet number sent per connection and the average data packet size in the input of the artificial neural network in order to try to improve its efficiency. In order to analyse their performance we have chosen to use the correlation and the average relative error. By analysing our model we have shown the importance of the data. Indeed, we have to choose carefully their type and scope. We have shown also that our implementation of the mathematical model was inefficient. At the same time, we have reached a better accuracy with our artificial neural network model: 86.45% of correlation and 37.4% of average relative error.

Keywords

TCP performance prediction, Artificial neural network model, Mathematical model, Efficiency analysis.

1 Introduction

TCP is one of the core protocols used within networks. Most Internet services are based on this protocol. We can manage more efficiently a network and its services if we know its performance. It is therefore interesting to identify and predict the performance of TCP traffic. We know the performance of this protocol is based on network characteristics such as lost rate and timeout. Existing mathematical model uses some of these network features in order to retrieve the performance of TCP. AI-based modelling is a different approach we can use to determine this performance. Artificial neural network is one of the methods used to create AI-based model. Their implementation will enable us to model the performance of any TCP traffic. By understanding the efficiency of such models we can improve them. This project aims then to analyse the efficiency of artificial neural network when modelling the performance of TPC traffic.

This type of work as already been achieved. We know from a previous research paper (Bogdan and Furnell, 2008) that the creation of such model is possible. This

current paper will suggest a model based on different network features. In order to achieve this aim prior works are necessary. First of all we need to choose which mathematical model we want to implement. Indeed, in addition to many network features, we want to include in the dataset the mathematical model output. Once we have retrieved the dataset we can build an artificial neural network. In practise the first results were bad. We have then decided to filter the dataset. We will then analyse this filtering.

A background part will describe how we have retrieved the dataset and both mathematical and artificial neural network models we have implemented. The second part will present the traffic performance before the filtering. The third part will present the analysis of the filtering process. The last part will show the influence of the filtering on each network feature.

2 Background

This background part will introduce the mathematical model we have chosen and the dataset we will use. Then we will describe how we construct our artificial neural network.

2.1 Mathematical model

We will describe here three mathematical models used to estimate TCP performance.

The first formula comes from Bellcore Labs (Ott et al., 1996). The paper studies the stationary behaviour of the congestion window within an ideal TCP congestion avoidance. Ideal congestion avoidance is defined by independent loss, equal equity to occur and no timeout reached. In order to model this specific behaviour they use Selective Acknowledgments (SACKs).

The second model (Padhye et al., 1998) was developed to determine the steady state throughput thanks to the packet loss rate and the round trip time. This model extends the previous model and takes in consideration the influence of the timeout. Nevertheless it cannot model connection with no loss.

The last model (Cardwell et al., 2000) extends the second model. This time the data transfer time is estimate thanks to the round trip time, packet loss, timeout and initial congestion window. This model can handle connection with no loss. It is design to model short-lived connection.

From these three models we will chose the last one. Indeed, this model takes in consideration more network features and seems to be the most efficient. It uses and extend the two others. In addition, this model suit short-lived connections. This type of connection is likely to happen nowadays. For all these reason we have implemented a C program to retrieve the data transfer time.

2.2 Network Features

Now we know that the mathematical model we have chosen use the round trip time (RTT), timeout, packet loss and initial congestion window, we use them all to build our artificial neural network. In order to help it we will also include the average packet size and the average packet number per connection. In addition to these six network features the input part of our data will be also composed by the result of the mathematical model. The output use in our dataset will be the data transfer time.

In order to construct this dataset we have used tcpdump and tcptrace. Indeed we have captured a trace with tcpdump from the University of Plymouth network. This trace gathers the information of all the connection to the Internet. We specified to capture only TCP packets to ease to subsequent use. Once we have this trace we use tcptrace to retrieve the seven network features we wanted.

2.3 Artificial neural network

From the dataset we can start to train and test our artificial neural network. Our dataset will be split in two parts: 90% to make the training set and 10% to create the testing set. In order to build this model we use Stuttgart Neural Network Simulator (SNNS). We have then decided to use a very basic neural network configuration: one input layer, one hidden layer and one output layer. On the input layer we have to fit seven data types, we have then seven neurones. On the hidden layer we have four neurones. On the output layer we have just one neurone representing the data transfer time. Figure 1 shows us this initial neural network configuration.

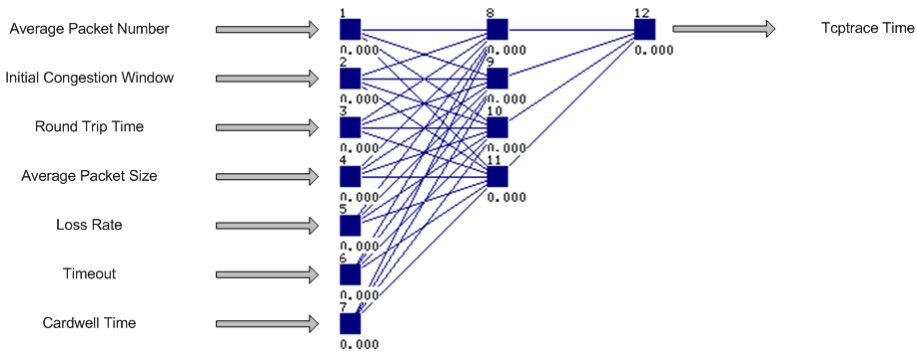

Figure 1: Initial Network Configuration (7-4-1)

Concerning the learning process, we have used the standard Backpropagation function with a learning rate at 0.3 and a desired error at 0.001. We have also chosen to run the creation process with 500 cycles.

In order to evaluate the efficiency of our models we will use the correlation and the average relative error. The correlation give us the relation between the estimated data transfer time and the real one. The relative error gives us the precision of the model between those same two values.

3 Traffic Performance

This part will present the performance of the traffic we have captured. This traffic is characterised by the seven network features plus the output of the Cardwell (2000) model. The performance analysed is the distribution of each data composing the traffic.

The scope the number of data packet sent per connection is 2 to 9,100. A single packet can transport 1380 bytes of information. The corresponding scope of the amount of data transported start from 2.69 kilobytes to finish at 11.97 megabytes. These values can happen in practice. Nevertheless the value of the 99th percentile is 383 packets. There is then a presence of extremum values over this point.

The initial congestion size is ranged from 2 to 13 segments. This scope can exist in real life. Nevertheless 89% of the connections possess an initial window of 2 segments. This configuration is the basic configuration of TCP. Furthermore 99% of our data are below or equal to 5 segments. Therefore a connection with 13 segments is very rare.

The round trip time is ranged from 0 to 99,812 milliseconds. A round trip time of zero millisecond cannot exist in practice. That means the transmission of the data within the whole network takes no time. Furthermore, 96% of the data are between 10 and 1,000 millisecond. These values are more likely to be good. For example, a round trip time between London and Sydney is roughly 330 milliseconds. The values over 1,000 milliseconds need then to be removed.

The average packet size is ranged from 136 to 1,379 bytes. The value of the 5th percentile is 585 bytes. The main part of the connection has then an average data packet size between 585 and 1,379 bytes. The maximum value is reached for 2% of the connections. The maximum amount of data a packet can transport is 1,380 bytes. As the last packet finishes the transmission, its size is very unlikely to be exactly equal to 1,380 bytes. That is why an average size of 1379 bytes happens many times.

The loss rate is ranged from 0 to 0.5. The value of the 99th percentile is 0.058824. Furthermore only 5% of the connections possess a loss. The main part of the losses is included between 0.01 and 0.1. This performance is good and possible. For the loss rate a filtering by the top can be done.

A timeout can by set up by the network administrator. It is generally ranged between 1 and 5 seconds. In the trace, the timeout is ranged from 0 to 191,867 milliseconds. The value of the 99th percentile is 3,894 milliseconds. Furthermore only 9% of the connections reach a timeout. This performance is then normal despite of maximum values which we need to filter.

The data transfer time calculated with the Cardwell (2000) model is ranged from 0 to 429,135 milliseconds. First of all, a transmission time of zero millisecond is not possible in reality. A connection cannot transfer data in nil time. Such values do not have to be taken under consideration. A value of 429 seconds can potentially happen in practise, but this phenomenon is still very rare. 98% of the connections are ranged

between 10 and 10,000 milliseconds. The values not in this scope may need to be filtered.

The data transfer time retrieved from tcptrace is ranged from 0 to 1,719,515 milliseconds. Here again the value of zero millisecond need to be removed. 3% of the connections possess a zero millisecond data transfer time. That is very strange and let us thinks of a weakness in tcptrace. The maximum value is also possible but very rare. 94% of the connections are ranged between 10 and 100,000 milliseconds. The values not in this scope may need to be filtered. By comparing this data transfer time with the Cardwell data transfer time, we can see that they have almost the same distribution. Nevertheless the scale is not the same at all.

4 Dataset Filtering

This part will present the filtering process. In order to see its impact on the artificial neural network model efficiency, we will also present its initial and new performance.

4.1 Initial Performance

First of all we will describe the initial efficiency of both mathematical and artificial neural network models. Table 1 presents these results.

Model	Correlation	Relative Error
Mathematical model	0.3221	2.5661
Artificial neural network model	0.1978	30.4014

Table 1: Initial Performance

We can see in this table that both models possess very low correlation. The mathematical model has power of the neural network one. Figure 2 illustrates the performance of the artificial neural network. The red dots represent the response of our model. The x-axis is the real data transfer time and the y-axis is the estimated one. The green line represent a correlation of 1, its function is y=x.

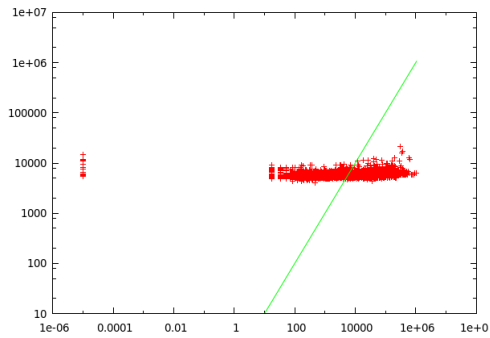

Figure 2: First efficiency

In this figure we can see that our response is ranged between 4,000 and 20,000 milliseconds while the real scope is 0.00001 to 1,000,000 milliseconds. Furthermore our response is far to follow the green line. We can then not estimate the whole range. That the result of a bad correlation. As our response is quite excessive we need to pay attention to our maximum values. Moreover a data transfer time that low is suspicious. We then need to take care of the minimum values as well.

4.2 Filtering Process

To filter our data we start to cut from maximum values. Once we obtain an interesting response we cut from the minimum values.

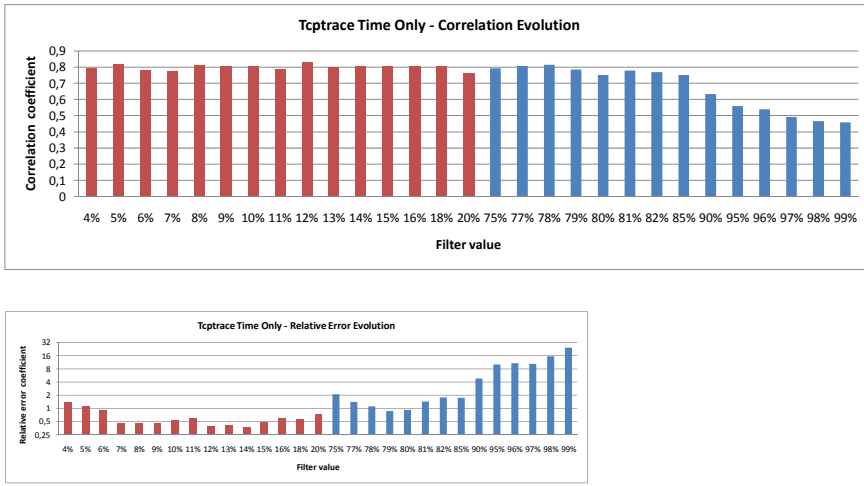

Figure 3: Filtering process

We can see that the efficiency is increasing when filtering from the top. By cutting just one percent we obtain already a correlation more than twice better at 47.79%. This correlation keeps increasing until it reaches a limit over 80%. The best correlation is 81.6% and it is reached at 78% of filtering. We can see also the effect of the filtering on the relative error. It is decreasing until it becomes lower than 1 for 79% and 80% of filtering. After this point the relative error is increasing we are then ignoring relevant information. At this point we will then keep the filtering at 79% as it possesses the lowest relative error with 0.86. The corresponding correlation is 78.55% which is nearly four times better than the initial one.

By filtering from the minimum values we can see that the correlation is not improved in a significant way. Nevertheless the precision is becoming lower. It even becomes lower than 0.5 for many values. The best value is 0.38 and it is found when we filter at 12%. The corresponding correlation is also the best: 82.88%.

4.3 New performance

After the filtering process we have retrieve the efficiency of our both models. Table 2 resumes these new results.

Model	Correlation	Relative Error
Mathematical model	0.1308	1.1890
Artificial neural network model	0.8288	0.3847

Table 2: New Performance

At the end of the filtering process we have then increased the performance of the artificial neural network. Nevertheless the mathematical has not been improved in the same way. The correlation has become very low. However, the relative error becomes twice better. Figure 4 illustrates the efficiency of our new artificial neural network model.

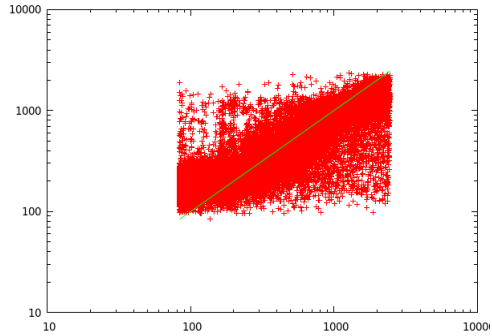

Figure 4: New efficiency

We can see here the scope taken by our response corresponds to the real one. Moreover, the general behaviour is closer to the optimum efficiency. That the effect of a better correlation. The enhancement of the relative error determines how close to the green line our response is. We can see that we need still need to improve it.

5 New Traffic Performance

After the filtering process, the distribution of all the data has changed. This part will describe the new distribution of each data.

The data packet number sent per connection is now ranged from 2 to 1,666 packets. The value of the 99th was 383 packets. After filtering the dataset, this range keeps more than 99% of its initial distribution. Moreover the scope from 2 to 100 packets represents now 99% of the data against 97% before the filtering process.

The initial congestion window size scope remains the same as before the filtering process. Furthermore, its distribution is almost the same.

The round trip time is now ranged from 0 to 42,148 milliseconds. Here again the new scope keeps 99% of the original distribution. 97% of the data represent now the scope 10 to 500 milliseconds, before it was representing the scope 10 to 1,000 milliseconds.

The average segment size is now ranged from 206 to 1,379 bytes. The previous minimum was 136. The value of the first percentile was 140 bytes, it is now 512 bytes. This data has then been filtered by the minimum.

The minimum loss rate has not been changed. Nevertheless, the connections having a loss represent now 2% against 5% before. The maximum was 0.5, it is now 0.33. The main scope is still from 0.01 to 0.1.

The minimum timeout has not been changed. Nevertheless, the connections reaching a timeout represent now 4% against 9% originally. The maximum was 191,867, it is now 4,899 milliseconds.

The minimum Cardwell data transfer time is still zero milliseconds. Its maximum falls from 429 seconds to 156,349 milliseconds. 99% of the original distribution has been kept. The tcptrace time is now ranged from 84 to 2,456 milliseconds. This scope represents 95% of the scope of the Cardwell time. This filtering process has then the right effect on our data.

6 Conclusion

In this current paper we have shown the performance of an artificial neural network when modelling TCP traffic. The characteristic we were targeting was the data transfer time. After optimising the artificial neural network, we have obtained a performance of 86.45% of correlation and 0.374 of relative error. This paper has also shown the importance of the scope of data we use. When mastering the data and the artificial neural network we can improve this performance. We can then use this technique to model other network protocols.

7 References

Cardwell, N., Savage, S. and Anderson, T. (2000). Modeling TCP latency [Online]. Department of Computer Science and Engineering, University of Washington. [Accessed the 24th of February 2009] http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.41.1513

Ghita, B.V. and Furnell, S. (2008). Neural network estimation of TCP performance, Proceedings of the 2008 International Conference on Communication Theory, Reliability, and Quality of Service (CTRQ 2008), Bucharest, Romania, 29 June – 5 July, pp53-58

Ott, T.J., Kemperman, J.H.B. and Mathis, M. (1996). The stationary behaviour of ideal TCP congestion avoidance [Online]. [Accessed the 24th February 2009] http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.50.9373

Padhye, J., Firoiu, V., Towsley , D. and Kurose, J. (1998). Modeling TCP throughput a simple model and its empirical validation [Online]. University of Massachusetts. [Accessed the 24th February 2009] http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.34.7826