Technical Disclosure Commons

Defensive Publications Series

April 11, 2018

LONG SHORT TERM MEMORY BASED TOTAL TRAFFIC PREDICTION FOR CONTAINER LOAD BALANCING

Manoj Ragupathy
Cisco Systems, Inc.

Xueqiang Sherman Ma
Cisco Systems, Inc.

LONG SHORT TERM MEMORY BASED TOTAL TRAFFIC PREDICTION FOR CONTAINER LOAD BALANCING

AUTHORS:
Manoj Ragupathy
Xueqiang Sherman Ma

CISCO SYSTEMS, INC.

ABSTRACT

Predictive load balancing is becoming increasingly relevant with the rapid adoption of machine learning models. Current load balancing approaches only considers the current state of the system, even though the current state of the system often varies. Accordingly, dispatch requests are provided herein which not only consider the current load on the system, but also the future load.

DETAILED DESCRIPTION

Container deployment frequently suffers from latency issues due to short sightedness. In particular, when making decisions relating to dispatching a new container, it is assumed that the load at the current moment in time is going to continue.

In one example, there are limited available resources (e.g., only a few containers). Suppose machine A has 50% Central Processing Unit (CPU) free and has containers of type A1 and A2, and machine B has 40% free CPU and has containers of types A2 and A3. If another container of type A2 needs to be run, a load balancer for application deployment would assign machine A to handle it since it has more resources available at that point in time. However, if it could predict that the A1 application will have more demand in a few minutes, then the container orchestration tool could anticipate and allocate the load to machine B.

The methodology involves calculating load anticipation of different application types (e.g., a set of endpoint groups in Cisco Systems, Inc.'s Application Centric Infrastructure (ACI) architecture). Long Short Term Memory (LSTM) in Recurrent Neural Networks (RNNs) calculates the future load based on the current and past load. It does not have to create a different model for time of the day, day of the week, or day of the year. LSTM automatically remembers which values of the past data contribute to the future data.

The load balancer described herein includes two LSTM based prediction systems. The first prediction system predicts the future load for each application type. Using this data it can estimate the average load for each machine at any given time. For example, if the load for application type A is predicted to be 300% for the next hour and there are six machines hosting the type A container, the average load for the three hosts must be at least 50% for the next hour. The second prediction system predicts how long the current request (flow duration) will last. The future load at the target system should be less than 100% for this duration at least.

Figure 1 below illustrates an example overview of a system configured to implement the techniques described herein.

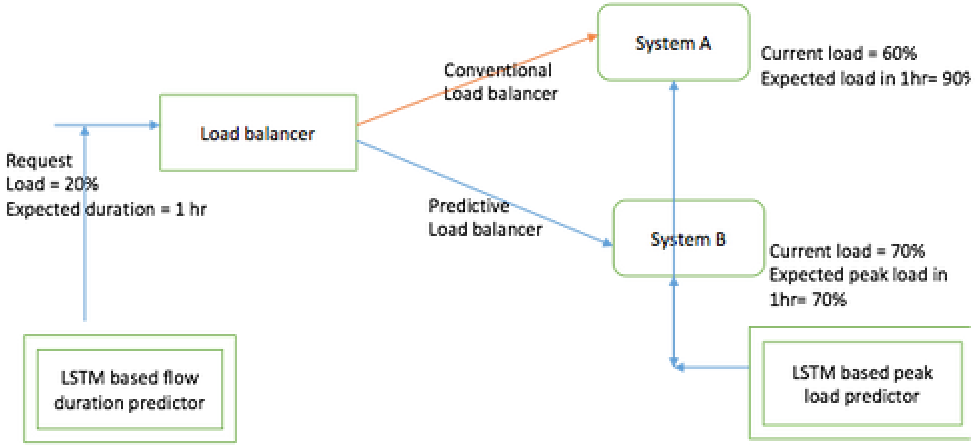

Figure 1

Both prediction systems are recurrent neural networks based on LSTM cells. LSTM cells are relatively new phenomena for time series prediction. An LSTM cell consists of three gates: the forget, input, and output gates. The forget gate decides which values to keep and for how long. Thus LSTM gates forget irrelevant recent information in favor of relevant older information. The input gate decides which values of input features are important for the future state. The output gate decides which value of input is relevant for the output.

Both the flow duration and traffic expectation is dependent on a variety of interwoven variables such as source IP address, destination IP address, past request duration, day of the week, day of the year, hour of the day, byte length of the Hypertext Transfer Protocol (HTTP) request, source port, destination port, network protocols, application protocols, compression algorithms, encryption algorithms, etc. Oftentimes the current flow may be dependent on a past occurrence. LSTM, which has the capability to remember past events for a longer time and forget short term events, can remember seasonal changes in network traffic. Thus, LSTM is more suitable for data networking than other time series prediction tools such as autoregressive integrated moving average (ARIMA). For current flow duration prediction the inputs are source IP address, source port, destination IP address, destination port, and protocol, and the output is the number of bytes and duration.

For total traffic prediction, the inputs are the total number of bytes sent or received for each ten-second period. The output is the same but for time steps (t + 10). The load balancer may also be applied to balance non-network metrics such as CPU load, memory, disk input/output operations, etc.

Example load balancer pseudocode is provided as follows:

1. For each request predict the total flow duration due to current request. Let that be “t”.
2. For each container that can receive the request calculate the future peak load during “t”, choose the best node that has the least probability of being overloaded at any time during “t”.

Figure 2 below illustrates example flow duration prediction.

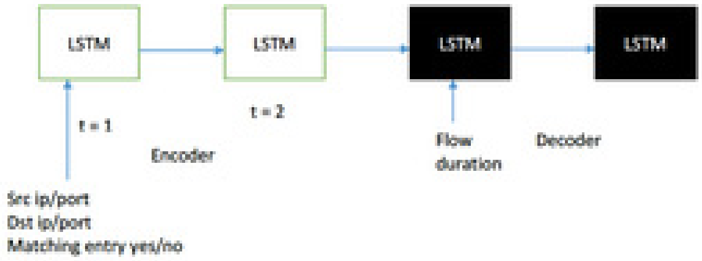

Figure 2

Figure 3 below illustrates example performance/load prediction.

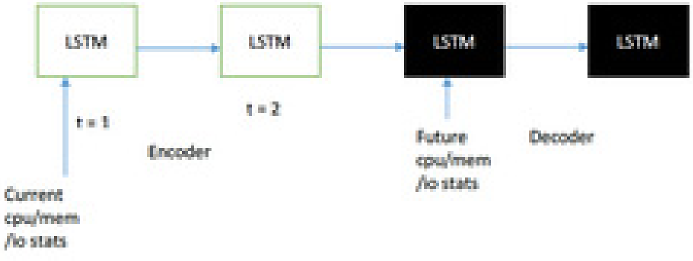

Figure 3

Conventional approaches do not use LSTM cells to predict the duration of network flow. This sets bounds on how far ahead in the future the load on the servers needs to be predicted. This also determines the configuration (number of LSTM cells) of the recurrent neural network used to predict the future load. The unique combination of these two different LSTM models provides the requisite accuracy to perform this kind of prediction.

In summary, predictive load balancing is becoming increasingly relevant with the rapid adoption of machine learning models and the popularity of containers. Containers running the same type of application exhibit similar packet flow characteristics. Current load balancing approaches only considers the current state of the system, even though the current state of the system often varies, particularly in virtual machine / container computing environments. Accordingly, dispatch requests are provided herein which not only consider the current load on the system, but also the future load.