Attention Networks for Time Series Regression and Application to Congestion Control

Paper Presentation

Victor PERRIER, *TéSA*
Emmanuel Lochin, *ENAC*
Jean-Yves Tourneret, *IRIT/ENSEEIHT*
Nicolas Kuhn Patrick Gelard, *CNES*

October 17, 2022

TéSA Télécommunications Spatiales et Aéronautiques · cnes CENTRE NATIONAL D'ÉTUDES SPATIALES · isae Institut Supérieur de l'Aéronautique et de l'Espace SUPAERO

Outline

1. Congestion Control
2. COPA
3. Contribution
4. Attention Mechanism presentation
5. Experimental Setup and Results

Congestion Control

- Congestion control (CC) algorithm has 3 main objectives

Congestion Control

- Congestion control (CC) algorithm has 3 main objectives
  1. use the maximum of the available capacity;

Congestion Control

- Congestion control (CC) algorithm has 3 main objectives

    1. use the maximum of the available capacity;

    2. reduce the latency;

Congestion Control

- Congestion control (CC) algorithm has 3 main objectives
  1. use the maximum of the available capacity;
  2. reduce the latency;
  3. Stay fair.

Congestion Control

- Congestion control (CC) algorithm has 3 main objectives

  1. use the maximum of the available capacity;
  2. reduce the latency;
  3. Stay fair.

TCP Cubic is the main CC algorithm on the internet

Congestion Control

- Congestion control (CC) algorithm has 3 main objectives

  1. use the maximum of the available capacity;
  2. reduce the latency;
  3. Stay fair.

TCP Cubic is the main CC algorithm on the internet

Issue

TCP Cubic is not able to achieve these 3 goals.

In quest of the optimal control point

To improve CC performances, algorithms should be operating around the Optimal Control Point.

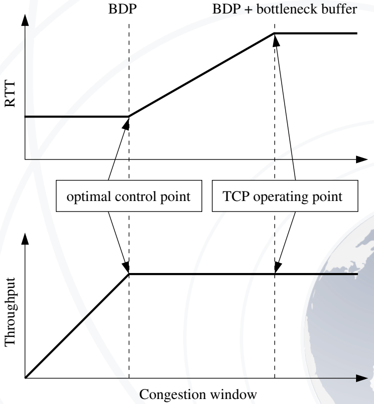


Figure: Operating points of different CC algorithms

State of the art

Some new CC mechanisms try to reach that optimal working point.

State of the art

Some new CC mechanisms try to reach that optimal working point.

- BBR tries to guess the optimal congestion window by estimating the throughput and the delay of the link.

State of the art

Some new CC mechanisms try to reach that optimal working point.

- BBR tries to guess the optimal congestion window by estimating the throughput and the delay of the link.
- Copa tries to estimate the queuing delay, to keep it close to the minimum.

State of the art

Some new CC mechanisms try to reach that optimal working point.

- BBR tries to guess the optimal congestion window by estimating the throughput and the delay of the link.
- Copa tries to estimate the queuing delay, to keep it close to the minimum.
- PCC tries to optimize a custom reward function with online learning.

Copa mechanism

We interest ourselves more specifically in the COPA mechanism :

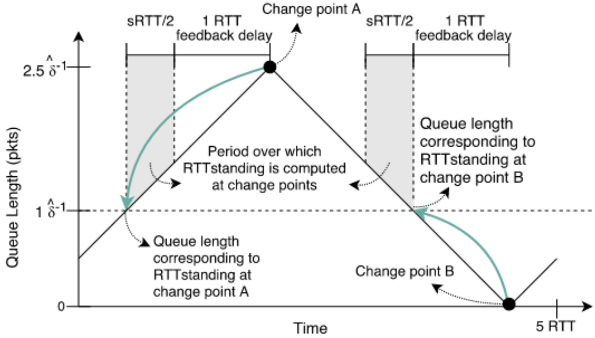


Figure: COPA mechanism. The aim is to stabilize the total queue length around δ^{-1}.

Copa mechanism

This problem has been solved using an estimator defined by:

$$\min_{i \in [t-L_1, t]} \boldsymbol{x}_i - \min_{i \in [t-L_2, t]} \boldsymbol{x}_i, \tag{1}$$

with $L_1 << L_2$ and where $\boldsymbol{x}_i$ is the time series of RTT, i.e., the round trip times of packets in the network.

Copa mechanism

This problem has been solved using an estimator defined by:

$$\min_{i \in [t-L_1, t]} \boldsymbol{x}_i - \min_{i \in [t-L_2, t]} \boldsymbol{x}_i, \tag{1}$$

with $L_1 << L_2$ and where $\boldsymbol{x}_i$ is the time series of RTT, i.e., the round trip times of packets in the network.

(1) tries to estimate the total amount of time a packet spend in the all the queues in the network by a packet. The idea behind the estimator is that the queue should periodically be empty, thus $\min_{i \in [t-L_2, t]} \boldsymbol{x}_i$ should approximate the RTT without any congestion. One limitation of this estimator is that it is hard to handle competition between CC mechanisms.

Our contribution

In a previous contribution, we showed that it is possible to learn some internal metric of the network, such as the total time spend in the queues in a network by a packet (used by COPA) or the bottleneck queuing size and evolution. However, a limitation was the complexity of the estimator.

Our contribution

In a previous contribution, we showed that it is possible to learn some internal metric of the network, such as the total time spend in the queues in a network by a packet (used by COPA) or the bottleneck queuing size and evolution. However, a limitation was the complexity of the estimator.

The objective is to propose a supervised Deep Learning method that can improve the accuracy of (1), handle cases where there is competition between CC algorithm, and have a lower complexity than our previous method.

Our contribution

In a previous contribution, we showed that it is possible to learn some internal metric of the network, such as the total time spend in the queues in a network by a packet (used by COPA) or the bottleneck queuing size and evolution. However, a limitation was the complexity of the estimator.

The objective is to propose a supervised Deep Learning method that can improve the accuracy of (1), handle cases where there is competition between CC algorithm, and have a lower complexity than our previous method.

Attention Networks

Attention is a time series processing mechanism originally built to perform language translation tasks. However, it can easily be extended to other domains, such as time series regression.

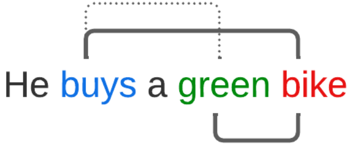


Figure: Example of the attention mechanism for a sentence. A dashed line corresponds to a weak connection between words, whereas a plain line is used for a strong connection.

Attention Networks

Notations	
Symbol	Signification
d	dimension of the Time Series
L	Length of the Time Series
$\mathbf{X} \in \mathbb{R}^{L \times d}$	Multivariate Time Series
$\mathbf{W}_k \in \mathbb{R}^{d \times d}$ $\mathbf{W}_q \in \mathbb{R}^{d \times d}$ $\mathbf{W}_v \in \mathbb{R}^{d \times d}$	Matrices of Attention parameters
$\mathbf{P} \in \mathbb{R}^{L \times d}$	Position encoding matrix concatenated with $\mathbf{X}$

$$\text{ATTENTION}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}}\right)\boldsymbol{V}, \tag{2}$$

with

$$\text{softmax}(\boldsymbol{X})_{i,j} = \frac{e^{\boldsymbol{X}_{i,j}}}{\sum_{k=0}^{d} e^{\boldsymbol{X}_{i,k}}},$$

and

$$\begin{cases} \boldsymbol{K} = \mathbf{W}_k \boldsymbol{X}, \\ \boldsymbol{Q} = \mathbf{W}_q \boldsymbol{X}, \\ \boldsymbol{V} = \boldsymbol{W}_v \boldsymbol{X}, \end{cases}$$

The rationale behind these formulas is that the resulting vector after the *softmax* operation is a linear combination of the vector $\boldsymbol{X}$. It can extract interesting component of $\boldsymbol{X}$, depending on the choice of $\boldsymbol{K}$, $\boldsymbol{Q}$ and $\boldsymbol{V}$, just like an SQL algorithm.

Attention : Pros and Cons

Attention sees the entire dataset, and is able to establish logical links between elements with ease in comparison to more classic Deep Learning methods such as LSTMs. Also, estimating a maximum or a minimum with attention is natural because of the *softmax* operation.

Attention : Pros and Cons

Attention sees the entire dataset, and is able to establish logical links between elements with ease in comparison to more classic Deep Learning methods such as LSTMs. Also, estimating a maximum or a minimum with attention is natural because of the *softmax* operation.

However, it is very expensive to compute the attention over a time series at each time step. We need $\mathcal{O}(L^3)$ computation steps to compute attention over a time series of size L.

Proposed Algorithm

To overcome both shortcomings of LSTM and Attention networks, we propose a new hybrid architecture defined as follows:

1. an LSTM layer is constructed:

$$\mathbf{H_i} = \mathbf{LSTM}(\mathbf{x}_1, ...\mathbf{x}_i) \in \mathbb{R}^{J \times d},$$

2. an Attention network is constructed as follows:

$$\mathbf{Y}_i = \mathbf{ATTENTION}(\mathbf{W}_q\mathbf{H}_i, \mathbf{W}_k\mathbf{X}_{1;i}, \mathbf{W}_v\mathbf{X}_{1;i}),$$

   $\boldsymbol{H}_i$ is used to determine which are the most important past elements. The idea of this architecture is not to use self-Attention directly (since if is too computationally intensive), but to generate the requests with an LSTM network.

3. A non-linear layer (RELU activation function) is finally introduced as in many DL architectures:

$$\mathbf{Z} = \mathbf{FeedForward}(\mathbf{Y})$$

The previous steps 1), 2) and 3) can be repeated for each of the M layers of the network.

Experimental setup

- Emulation
- Parking lot topology
- We have perfect information
- How did we generate traffic :
  - TCP flows are used for the traffic
  - The number of TCP flows changes
  - The bottleneck changes
  - At each node we may have implemented some type of scheduling

Experimental setup

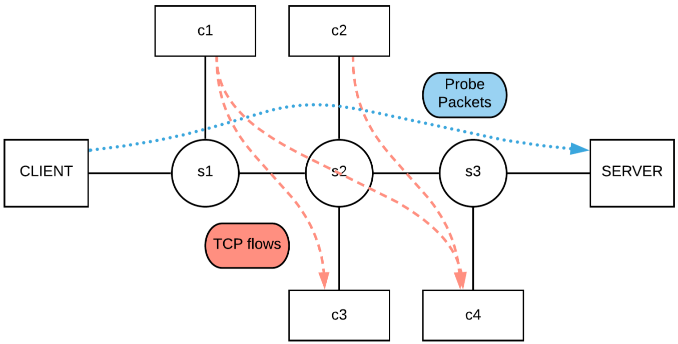
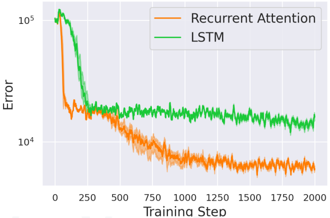


Figure: Parking lot topology

Results for an emulated dataset



(a) Training loss. The colored envelopes represent the maximal and minimal errors for 10 trained models.

(b) Estimation of the bottleneck load (with emulation) using LSTM and Attention architectures.

Figure: Results for a concrete task of estimating the current load at the bottleneck in a network path.

Conclusion and future work

This work show that it is possible to obtain better estimations of the internal state of the network with deep learning mechanisms of limited complexity: $\mathcal{O}(L^2)$.

Conclusion and future work

This work show that it is possible to obtain better estimations of the internal state of the network with deep learning mechanisms of limited complexity: $\mathcal{O}(L^2)$.

We are currently working on still improving the complexity of the attention mechanism.