Z-networks and transformers

Introduction

Consider that the end points of a z-network are state machines and we call them z-machines.
A z-machine is a vector where each component is a state machine. Every component has at least a state: "on" or "off". If the state is "on" the incoming packet is applied to that machine, otherwise that machine does not do any action and the output is the same packet.

Let's now consider a z-system.
A z-system is a collection of z-machines that share data packets using common channels and protocols.
What characterizes a z-system is that all the z-machines are exactly the same, except in the configuration. A configuration is the resulting binary vector that tells when the ith machine is "on" (1) or "off"(0).

The purpose of using transformers is to predict the next best configuration based on the current traffic. It means that the resulting transformer model accepts a sequence of packets and using the current configuration it predicts the next one to be used. This happens on every z-machine.

Definitions

z-Machine

State Vector: A z-machine, Z, is represented as a binary vector, $z = (z_n)$, where n is the number of state machines within Z, and $z_i \in \{0, 1\}$. The *i*-th component indicates whether the *i*-th state machine is "off" (0) or "on" (1).

Packet Processing Function: Each component z_i has an associated state machine function f_i that processes incoming packets if $z_n = 1$. Otherwise, the output is the unaltered packet.

State Transition: A transition function T determines the next state of Z based on the input packet and the current state vector.

z-System

Homogeneity: A z-system, S, is a collection of z-machines (Z_n) that are structurally identical but may differ in their state vector configurations.

Shared Communication: All z-machines in S communicate via shared channels and adhere to common communication protocols.

Configuration Space: The configuration space of S is the set of all possible state vectors for z-machines within S.

Transformer Model for z-System Configuration Prediction

Input: A sequence of packets $P = \{p_k\}$and the current configuration vector $z_{current}$.

Output: The next configuration vector z_{next} .

Model Structure: The transformer employs layers of self-attention mechanisms to process the sequence of packets and the current state vector, utilizing positional encoding to maintain the order of packets.

Training Objective: The model is trained to minimize the prediction error of the next configuration vector, given the current configuration and the sequence of packets.

Functional Formalization

1- Packet Sequence Representation: Let $P = \{p_k\}$ represent the embedding of packets into a suitable vector space where p_i corresponds to the i-th packet in the sequence.
2- Current State Encoding: Encode the current configuration $z_{current}$ as an embedding $c_{current}$, potentially with additional features to provide the transformer model with more context about the state of the z-system.

3- Transformer Architecture: Design the transformer model, T, to take P and $c_{current}$ as input and output a prediction for z_{next} , represented as $c_{predicted}$. The model operates as follows:

a- Self-Attention Mechanism: To weigh the importance of each packet in the sequence and the current state vector.
b- Feed-Forward Neural Network: To process the attended information and predict the next state.
c- Training: The model is trained on historical data of packet sequences and corresponding optimal configurations.

4- State Transition Function: The predicted configuration embedding $c_{predicted}$ is then transformed into a binary vector z_{next} which represents the predicted configuration of each z-machine in the z-system.

Mathematical Formalization

Let $z_{current}$ be the current state vector and P the sequence of packets. The transformer model T operates on these inputs as follows:

$$z_{next} = T(P, z_{current})$$

The objective function O for training the model T is given by:

$$O = \sum_{D} L(z_{next}, z_{true})$$

Conclusion

This formalization serves as a blueprint for developing a transformer-based predictive model within the z-networks and z-systems. The success of such a model hinges on its ability to learn from historical data and accurately predict the optimal configuration for a given sequence of packets, facilitating dynamic and responsive network management.