The Hong Kong Polytechnic University
SingularityNet

Causal analysis in Networks via Deep Neural-Symbolic Learning

Andres Suarez-Madrigal, Tesfa Yohannes, Ben Goertzel

November 16, 2019

Contents

1 Introduction

The current project explores a method to predict anomalous events in the dynamics of a network, via finding their causes. The main idea in the method is to learn the grammar of the network –its implicit dynamical structure– in an unsupervised manner, starting from telemetry data generated by the different components of the network. If such a grammar can be obtained, then it would be possible to predict an anomaly (with a certain confidence) when observing dynamics that preceed such a state in the grammar. This would be similar to a situation in the processing of a known natural language where, after observing a sequence of words that form a sentence's subject (e.g. determiner-adjective-noun), we can expect a verb to follow.

This project will leverage SingularityNet's Unsupervised Language Learning (ULL) pipeline, which attempts to find the grammar implicit in a given corpus of sentences. In order to process the network data with such pipeline, the process can be divided into three stages:

1. Abstract the network dynamics by converting the state of the network at each timestep into a real-valued vector (or a set of vectors).
2. Using symbolic dynamics techniques, convert the time trajectory of the network embedding vector(s) into a sequence of symbols; these sequences would be functionally equivalent to a natural language corpus in the ULL project.
3. Apply a suitable version of the ULL pipeline to the sequences obtained in step 2 (the sentences), in order to learn the network grammar.

The first step could be done in a number of ways. One popular and successful way to create distributed representations is using Neural Networks. Deep Neural Networks like graph2seq have been used to abstract graphs, with some success [2].
TESFA, YOU CAN ADD SOME STUFF HERE.

Another approach for creating embeddings is to use the features as they come from the telemetry data. In Section 2 we follow a similar processing as done by Putina et al.[1] and visualize the resulting vectors.

For step 3, we need to adapt the ULL pipeline to our purposes: instead of having a natural language sentence where every word is presented to the algorithm at the same time, we would be dealing with a continous stream of tokens representing the state of the network at different times. In particular, the parser used in the ULL pipeline needs to be replaced by one that handles a continous stream of tokens. For this purpose, we implemented a continuous version of the MST-parser proposed by Yuret [3][1]

[1] Code available at https://github.com/glicerico/stream-parser

2 Embeddings

A crucial part of the proposed pipeline is to represent the obtained raw telemetry data in a way that abstracts the state of the network, distiniguishing clearly when it is in an anomalous state. Deep neural networks have been used for non-dynamical graphs [2], where the structure of the graph is converted to a continous vector for each timestep. We need a DNN that can also process the dynamical behavior of our data into the graph.

Another possibility is to use the telemetry data directly, in a fashion similar to how Putina et al.[1] feed their clustering algorithm. However, what is important in our case is that the representations used can distinguished between normal and anomalous network operation. In order to explore this, we decided to visualize the data and find out if some of the given features could be projected into a lower dimensional space, where a distinction between anomalous and normal behavior is clear.

We started with the data available in the **telemetry** repository[2], which contains metrics from experiments with different failure conditions. The databases contain readings for a subset of the available features for all nodes at different timesteps, pertaining to different Yang paths. We first constructed a network embedding for each unique timestep, independent of its origin node or Yang path. Using the ground truth information provided with the datasets, we also tag each timestep as 'anomalous' or 'normal', to use during visualization. Given the high dimensionality of the resulting vectors, we use the online *Embedding Projector* feature of TensorFlow[3] to project and visualize them. Figure 2.1 shows the t-SNE projection from dataset #3, with blue points representing 'normal' behavior, and red ones stand for timesteps where an 'anomaly' is present. We notice that this embeddings projection does not show a clear division between the two states of interest everywhere. Instead, it seems that the projection groups the representations by the Yang path that produced them, as can be seen in figure 2.2.

Using the *Embedding projector* functionality we are able to isolate the datapoints that come from a given node, as well as from a specific Yang-path. We attempt a projection using only the **leaf1** node, and the **data-rate** path (the one with most entries), which is shown in figure 2.3. Although we can see distinction in parts of the data, there is still a big region that mixes the two network states of interest.

Instead of using pre-selected features, we decide to now project the raw telemetry data available at the **OutlierDenStream-BigDama18** repository[4], which comes separated by node and experiment. Following Putina et al., we also distinguish between *DataPlane* and *ControlPlane* features to build the node representations for each timestep, discard features which have zero standard deviation, and normalize the resulting data matrix. Additionaly, the ground truth is used to tag the timesteps where an anomaly is present in the experiment. The t-SNE projection of node **leaf1** using the *DataPlane* features is shown in figure 2.4, where we are able to see a clear distinction between red (anomalous)

[2] https://github.com/cisco-ie/telemetry
[3] Available at https://projector.tensorflow.org/
[4] https://github.com/anrputina/OutlierDenStream-BigDama18

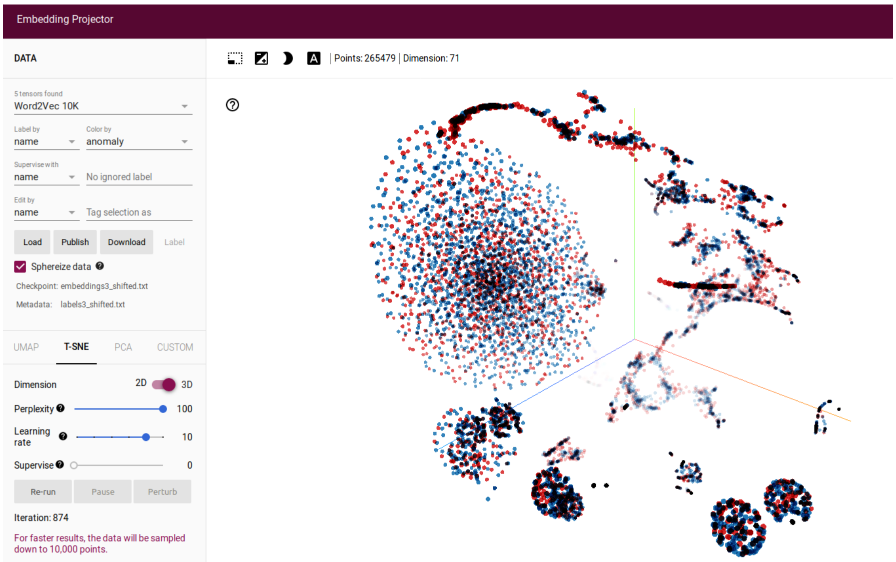

Figure 2.1: t-SNE projection of dataset #3. Red dots represent timesteps when the system is going through an 'anomaly'.

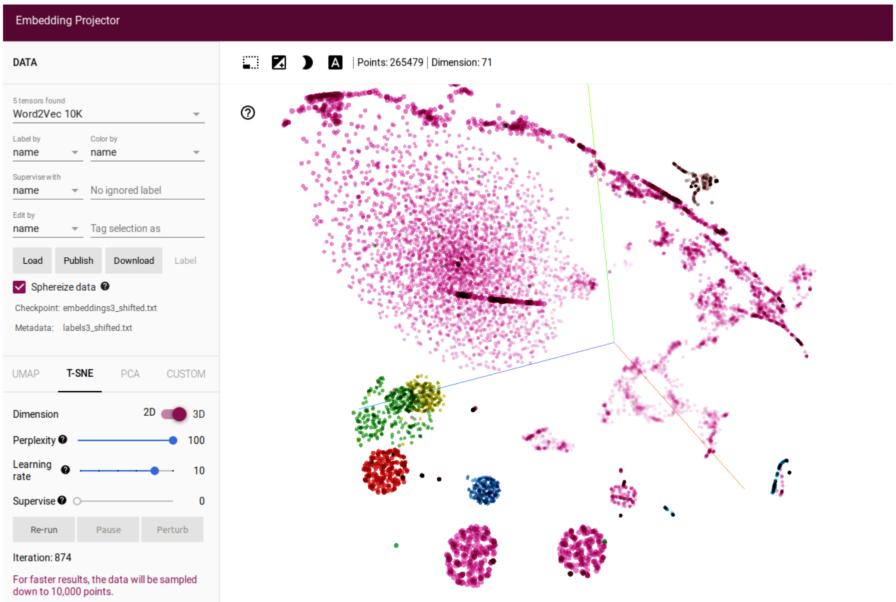

Figure 2.2: t-SNE projection of dataset #3. Color code represents the Yang path that produced each vector.

and blue (normal) network states for this node.

On the other hand, when using the *ControlPlane* for the same node and experiment, we obtain figure 2.5, which doesn't separate the states of interest as well.

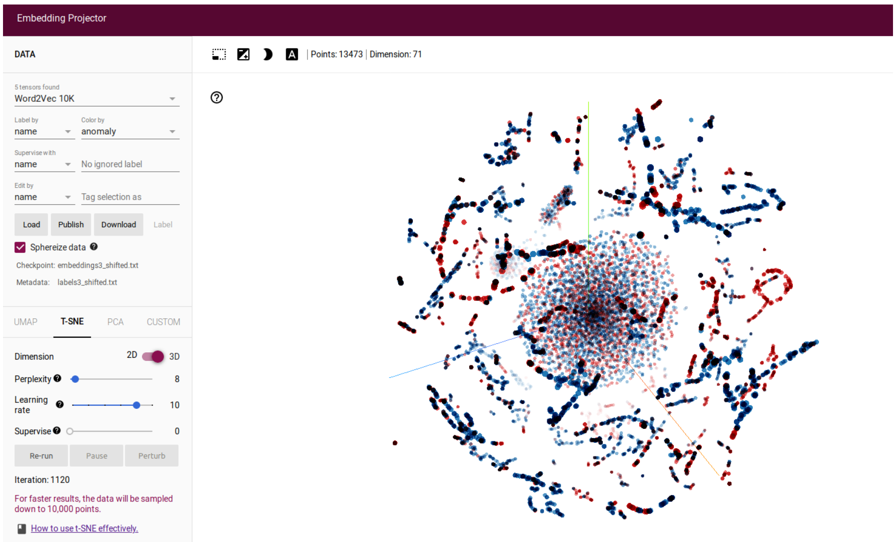

Figure 2.3: t-SNE projection of the **leaf1** node from dataset #3, using only the features from the **data-rate** Yang path. The node color distinguish anomalous (red) and normal (blue) states.

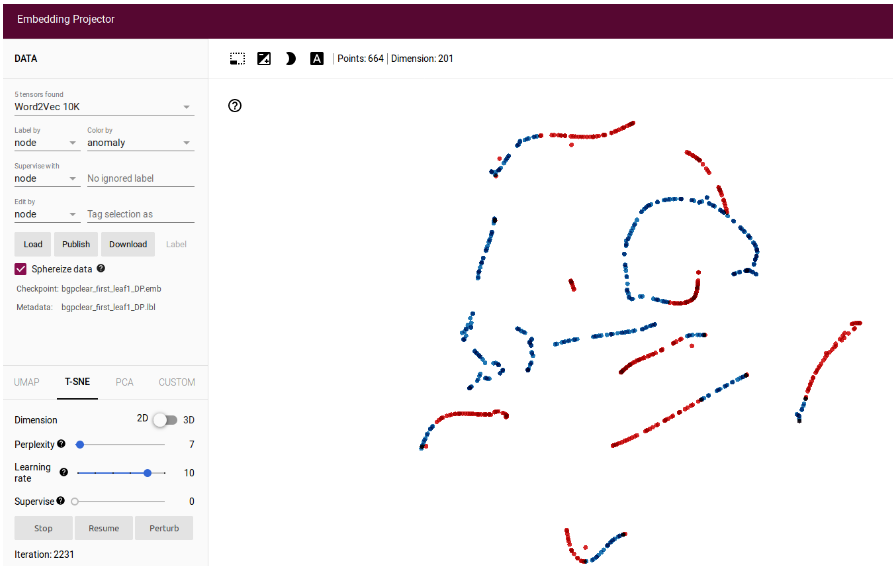

Figure 2.4: t-SNE projection of **bgpclear_first**'s **leaf1** node using *DataPlane* features. Red dots represent timesteps when the system is going through an 'anomaly'

Using data from a different experiment, figure 2.6 shows that a similar separation of states if found.

This is an interesting result, which shows that it's feasible to subdivide the high-dimensional space in which the given representations exist, and know where the normal

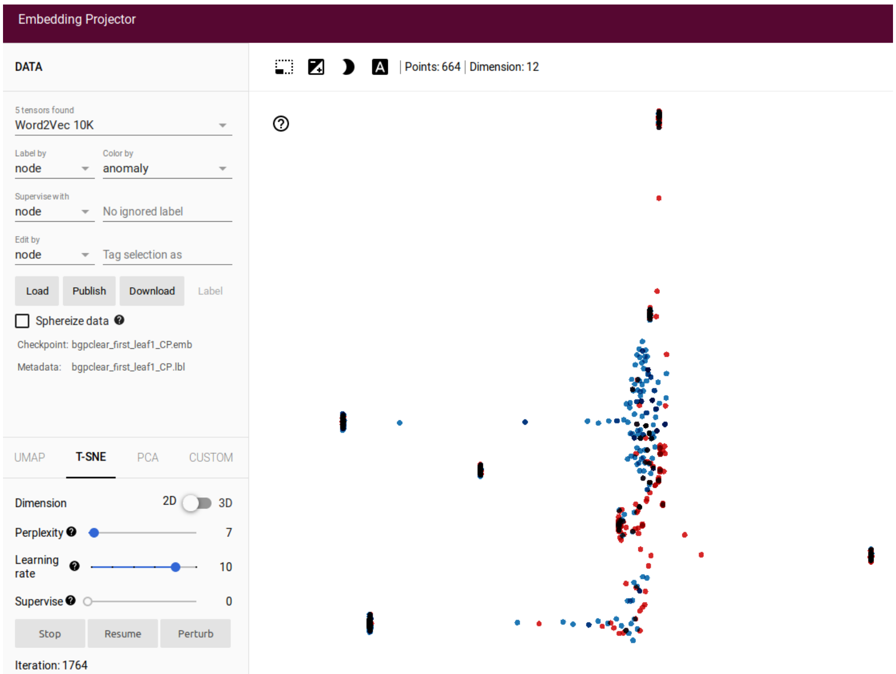

Figure 2.5: t-SNE projection of **bgpclear_first**'s **leaf1** node using *ControlPlane* features. Red dots represent timesteps when the system is going through an 'anomaly'

behavior of the network lies. Additionaly, this can be done without having an expert preselect the features to use, as done by Putina et al[1]. This subdivision of the space will allow to construct the sequence of tokens that would be processed by the Unsupervised Language Learning pipeline to obtain the network's grammar.

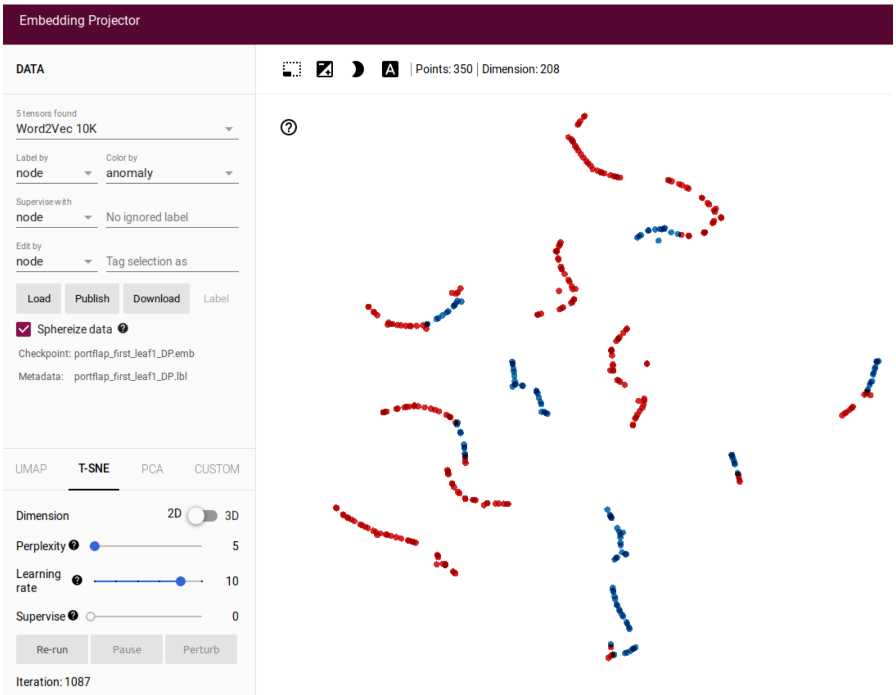

Figure 2.6: t-SNE projection of **portflap_first**'s **leaf1** node using *DataPlane* features. Red dots represent timesteps when the system is going through an 'anomaly'

References

[1] A. Putina, D. Rossi, A. Bifet, S. Barth, D. Pletcher, C. Precup, and P. Nivaggioli. Telemetry-based stream-learning of BGP anomalies. In *Proceedings of the 2018 Workshop on Big Data Analytics and Machine Learning for Data Communication Networks - Big-DAMA '18*, pages 15–20, Budapest, Hungary, 2018. ACM Press.

[2] S. B. Venkatakrishnan, M. Alizadeh, and P. Viswanath. Graph2seq: Scalable Learning Dynamics for Graphs. *arXiv:1802.04948 [cs]*, Feb. 2018. arXiv: 1802.04948.

[3] D. Yuret. Discovery of Linguistic Relations Using Lexical Attraction. *arXiv:cmp-lg/9805009*, May 1998. arXiv: cmp-lg/9805009.