An overview

Looking at possible relationships between different features of network data

Using Canonical Correlation Analysis

Status:Ongoing

We are looking at what relationships exist between different features of representations of network traffic data.

We have collected data consisting of ip addresses, time of start, source and destination ports, number of bytes and number of packets and so on.

The first representation of the traffic is where each node of the graph is a distinct IP address. An edge between two nodes is put if there is some communication between those two IP addresses. The edge weight can be the bytes transferred.

The second representation of the data is such that a flow between two IP addresses (a flow is characterized by IP1,IP2,time,bytes) is one single node. We will put an edge between two nodes if both the flows corresponding to the nodes are similar in nature. We were initially thinking of using Euclidian distance as the similarity metric, but we decided against it, as each feature of the flow has different scale and different variance. Also different features have different weights. We have recently decided upon using the Mahalanobis distance. This is basically a normalized distance. The weight of each feature is the inverse of the variance of that feature.

One of the challenges for CCA is that both these graphs should have the same number of node. We have to look at what is the optimal way to achieve this. We are currently looking at adding dummy nodes to get this.

Canonical Correlation analysis: Canonical Correlation Analysis (CCA) is a method for measuring the linear relationship between two multidimensional variable. Given two random vectors of the same dimension, x and y, CCA is concerned with finding a pair of linear transformations such that one component within each set of transformed variables is correlated with a single component in the other set. This is done by maximizing correlations between linear combinations of features in each set. This problem of maximizing correlations can be reduced to an eigen vector problem, and taking the top k eigen vectors.

The analysis of these graphs focuses on analysis of the adjacency matrix. This is treated as a vector representation of the data. We then perform CCA on these d-dimensional vector representations of the networks. We have implemented CCA from scratch using math libraries in Java. We have tested this on some data. We have looked at various coefficients like adequacy coefficient, factor loadings, communality coefficient and redundancy coefficients. These coefficients tell us what percent of variance of one set of variables is explained by the other, how well the transformed variables explain the initial variables and so on. Using these we hope to get some understanding of the relationships between different features.

The data can give very different results based on the time of sampling. The ports of transmission of data can give us different results. These are some directions we are looking at.

Note: This is a project which I recently started. Before we began work on this, we looked at the basics of Machine learning and learnt about some statistical techniques like Dimensionality reduction using PCA.