Predicting and detecting anomalous edges in large computer networks

Student: Francesco Sanna Passino
francesco.sanna-passino16@imperial.ac.uk
Supervisor: Dr Nick Heard

1. Problem

Monitoring and detecting anomalies in computer networks is an extremely challenging task. The quantity of data available is massive and large networks are constantly target of attacks from potential intruders. Traditionally, the approaches to cyber-security have been based on detecting signatures in packets, but it is possible to employ more advanced techniques, based on statistical models, in order to better identify suspicious patterns within the network. In particular, in this project we focus on modelling the network graph with two main purposes in mind:

- predicting future links
- identifying anomalous edges

2. Data

The data used are the NetFlow data collected by Imperial College London. An example of a data line, representing a connection between two randomly generated IPs, on June 1, 2017, from 12.30 to 12.31, is:

2|1496316548|785|1496316602|865|6|0|0|0|3292176969|49745|0|0|0|2902214576|443|0|0|945|1338|0|0|18|4347

The most relevant entries (separated by the vertical bar "|") are: 7 to 10 – source IP address (first 3 entries are $0 \rightarrow$ IPv4), 11 – source port, 12 to 15 – destination IP address, 16 – destination port, 23 – number of packets sent, 24 – number of bytes sent. On average approximately 14 Terabytes per month of data are available.

3. Setup

Given a set of Netflow records within a given time interval, we can construct a directed graph $\mathbb{G} = (V_c, V_s, E)$ where:

- V_c is the set of clients, $|V_c| = n_c$,
- V_s is the set of servers, $|V_s| = n_s$,
- E is the edge set, containing dyads (i, j), $i \in V_c$, $j \in V_s$.

We draw an edge if a client $i \in V_c$ connects to server $j \in V_s$ within the time interval, and we write $(i, j) \in E$.

From $\mathbb{G}$, we can obtain a rectangular adjacency matrix $\mathbf{A} = \{A_{ij}\}$, of dimension $n_c \times n_s$. We have:

$$A_{ij} = \begin{cases} 1 & \text{if } (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Clients} \begin{cases} \overbrace{\begin{pmatrix} 1 & 1 & 0 & \cdots & 0 & 1 \\ 0 & 0 & 1 & \cdots & 1 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & 1 & \cdots & 1 & 1 \end{pmatrix}}^{\text{Servers}} \end{cases}$$

Note that this object is **hugely sparse**.

It is useful to consider a weighted version $\mathbf{W} = \{W_{ij}\}$ of the rectangular adjacency matrix. Weights associated with each dyad $(i, j) \in E$ can be obtained in many ways. Here we consider the number of connections N_{ij} between two nodes within the time interval.

4. Exploratory Data Analysis

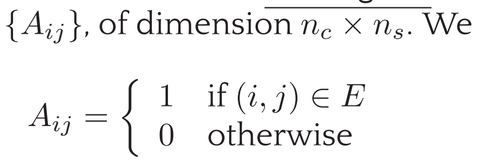

Figure 1: *Network activity on June 1, 2017*

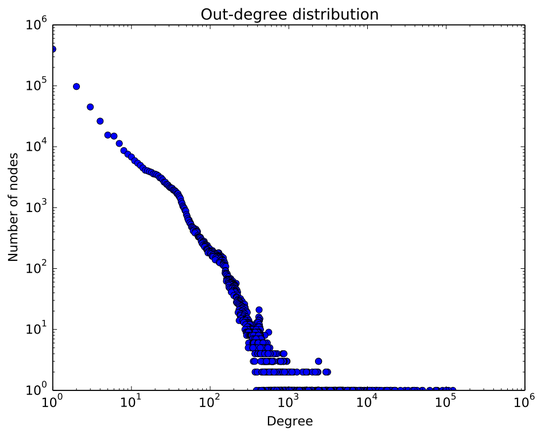

Figure 2: *Out-degree distribution on June 1, 2017*

5. Some features of the Imperial network

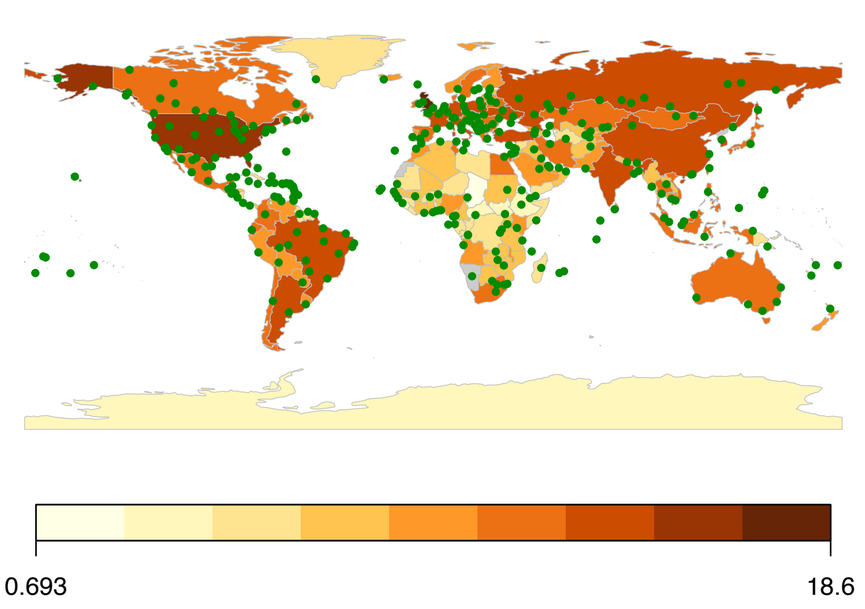

Figure 3: *Client locations (June 1, 2017, 10-11am), log-scale of intensity*

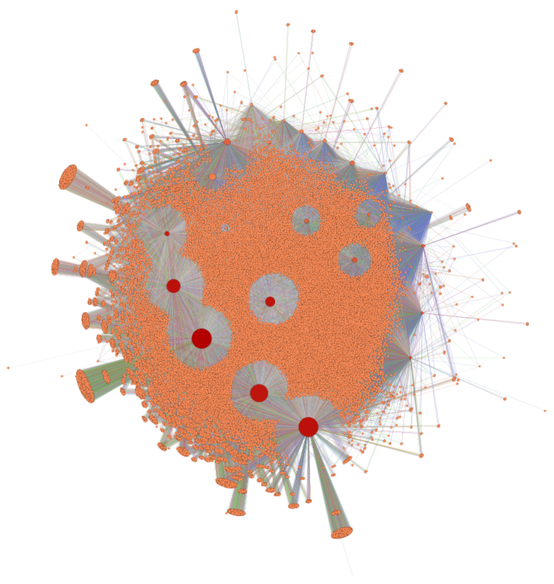

Figure 4: *Graph of CIDR/16 subnets communications (June 1, 2017 - 10–11am)*

6. A simple link prediction procedure based on the SVD

Singular Value Decomposition (SVD) of a rectangular matrix $\mathbf{A}$

$$\underset{n_c \times n_s}{\mathbf{A}} = \underset{n_c \times n_c}{\mathbf{U}} \times \underset{n_c \times n_s}{\mathbf{D}} \times \underset{n_s \times n_s}{\mathbf{V}^\top}$$

assuming $n_c < n_s$ (not always the case)

Eckart–Young theorem for best rank r approximation of $\mathbf{A} \rightarrow$ truncated SVD (tSVD)

$$\underset{n_c \times n_s}{\mathbf{A}} \approx \underset{n_c \times r}{\mathbf{U}_r} \times \underset{r \times r}{\mathbf{D}_r} \times \underset{r \times n_s}{\mathbf{V}_r^\top} \Rightarrow \mathbf{A} \approx \sum_{i=1}^{r} d_i \mathbf{u}_i \mathbf{v}_i^\top$$

- the Singular Value Decomposition (SVD) is a common matrix factorisation technique for link prediction, see [1]
- the **truncated SVD (tSVD)** is a feature extraction procedure: we set r and we obtain client and server-specific features from $\mathbf{U}_r$ and $\mathbf{V}_r^\top$
- it is simple to compute it in a sparse regime using the Implicitly Restarted Arnoldi Method (IRAM) – ARPACK – that requires only $\mathcal{O}[\text{nnz}(\mathbf{A})]$ operations

- apply the tSVD with $r = 100$ to the rectangular adjacency matrix obtained on June 1, 2017, from 10 to 11am
- ≈ 1.1 million nodes (only IPv4) and ≈ 10.7 million edges
- extract the new links from the graph obtained on the same day between 11 and 12am – $\approx$ 3 million connections in common, ≈ 7 million new links
- build a simple binary classifier ($1 =$ edge present, $0 =$ edge absent), sample absent edges to have balanced classes
- probability of an edge between i and j is modelled as a monotonic function of $\mathbf{u}_i^\top \mathbf{D}_r \mathbf{v}_j$, following [2].

7. Results

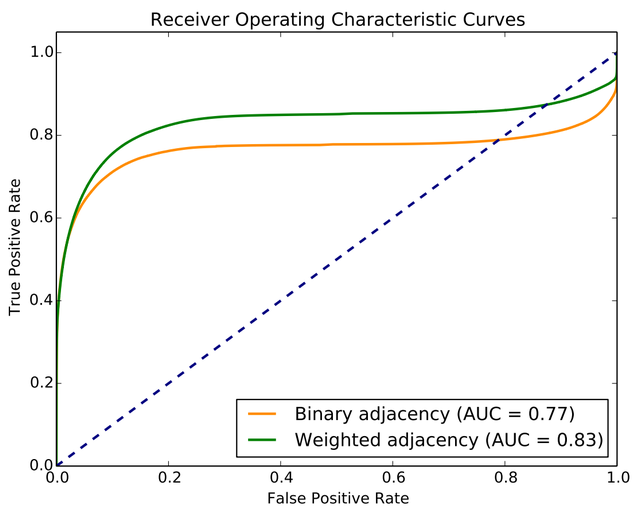

Figure 5: *ROC curve for the binary link classification procedure based on the tSVD with $r = 100$*

More information clearly gives better results, but even an AUC of 0.77 for binary data for 15 millions predictions is not terrible, since the two networks have completely different structures.

8. Future work

◎ So far we have used the raw and weighted adjacency matrices $\rightarrow$ consider other covariates to have a better understanding of the evolution of the network:

- IP outside/inside college
- geographic locations
- bytes sent
- duration of the connection
- ports used

◎ Focus on anomaly detection $\rightarrow$ given a model for the adjacency matrix, which nodes and edges are outliers?

References

[1] A.K. Menon, C. Elkan (2011). Link Prediction via Matrix Factorization. In: Gunopulos D. et al., Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2011. Springer, Berlin, Heidelberg

[2] P. Rubin-Delanchy, N.M. Adams, N.A. Heard (2016). Disassortativity of Computer Networks. In 14th IEEE International Conference on Intelligence and Security Informatics - Cybersecurity and Big Data (IEEE ISI)