Anomaly Detection in Time Evolving Networks

Anomaly detection refers to the problem of finding patterns in data which fail to conform to the expected standard. These anomalous patterns are useful to a number of business applications such as identifying trending topics on social media, identifying suspicious traffic on computer networks, credit card fraud detection, insurance fraud detection, e-auction fraud detection, etc.

There are a variety of anomaly detection algorithms available and we have to choose an appropriate for the the type of anomaly we are trying to detect. Different types of anomalies include point anomalies, contextual anomalies, collective anomalies, etc. This project's focus is on anomaly detection in time evolving graphs. Various methods like time series models, similarity based algorithms, and community based algorithms have been explored. Evaluation of these approaches is often not straightforward because of the lack of ground truth data. The algorithm implemented in this project is the DeltaCon algorithm. This link directs you to the original paper by Faloutsos et al.

Datasets

There are four datasets in the `datasets` folder: `autonomous`, `enron_by_day`, `p2p-Gnutella`, and `voices`. This data was primarily taken from the Stanford Large Network Dataset Collection. These are four time evolving graphs from different domains. For example, the enron graph represents an email network and the p2pGnutella graph represents a peer to peer network. Each graph is represented as a series of text files which define the graph at a given time point. Each text file is given as an edge list but with the first line stating the number of nodes and edges.

Getting Started

These instructions will get you a copy of the project up and running on your local machine.

Prerequisites

- python (version > 3.0)
- numpy, scipy and matplotlib python libraries

Instructions

Clone the repository, go to the root directory and run the following command in terminal (or any command shell):

> python3 anomaly.py your-dataset-name

For example, if you want to run the algorithm on the 'autonomous' dataset, you would run:

> python3 anomaly.py autonomous

Running the above command will output a text file named `autonomous_time_series.txt` which contains the similarity value time series for the autonomous dataset and a png file called `autonomous_time_series..png` which contains the plot of the same time series.

Note on running time

Depending on the size of the dataset and the size of each graph in the dataset, the program may take quite some time to run successfully. On my machine, which is a Macbook Pro with a 8 GB RAM and a 1.7GHz quad-core Intel Core i7 processor, the program took around 23 minutes to run on the p2p-Gnutella, the highest running time among the 4 datasets.

Author

- *Yashwanth Soodini* GitHub