Implementing Similarity Network Fusion in Neo4J

Evan Stene, Bahavana, Sumanth, Thrupthi

Problem Statement and Background

The Similarity Network Fusion (SNF) is a powerful technique for clustering medical patients utilizing a variety of data types in order to assist in diagnosing diseases such as cancer. Each patient is associated with a wide range of data from a variety of medical tests. The results of each test can, as a whole, be considered a separate data type, and patients can be clustered according to a specific data type to form a similarity network. Networks spanning multiple data types can then be combined through iterative fusion steps to form a single network that captures the overall similarity between patients in the network. Given the nature of medical data, patients should be clustered according to certain attributes from their test data assuming that those in a cluster have received a similar diagnosis. The goal then is twofold: identify the defining attributes for each cluster, and place new patients into appropriate clusters in order to assign them a possible diagnosis.

SNF is useful for data with a very low ratio of samples to features, which is true of medical data that spans multiple data types. Data of this nature tends to produce a low signal to noise ratio in analysis, a problem that affects the predictive power of the model.

Currently, the method for creating an SNF requires using a mathematical or statistical modeling language such as MATLAB or R. We believe that this approach lacks the ability to scale to larger datasets both in terms of creation time of the network and in providing a framework for querying the network. Our goal is to implement the SNF creation in the native graph database application Neo4J. This

Related Works

SNF provides a framework for discovering mechanisms responsible for diseases, clustering or sub-typing samples (patients), and predicting outcomes (survival). There are other strategies that attempt to achieve one or more of these goals, but each has drawbacks that prevent it from working with the same success as SNF.

Independent Analysis

Although possible to analyze each attribute independently, this is time intensive and inconsistent.

Feature Concatenation

Another strategy is to simply concatenate each data type into a single feature list before performing traditional clustering techniques. However, when data is producing a low signal to noise ratio, lengthening the feature list only serves to exacerbate the problem.

Gene Preselection

Genes known to be important may also be preselected before analysis to identify which are effected in each subtype. This strategy is biased toward existing knowledge and does not facilitate new gene exploration or possible interactions between multiple effected areas.

Solution

Our method is based on the SNF implementation method described in [1]. The original implementation written by the authors of the paper is available at http://compbio.cs.toronto.edu/SNF/SNF/Software.html and is provided in both MATLAB and R formats, each containing the necessary code to build the network and provide some statistical analysis on the result. Data for five different diseases, each spanning three data types for around 200 patients, was also supplied on the website and is the data used by the authors of [1].

Our work is an attempt to expand the SNF for use with higher volumes of data by implementing the network as a graph database. As more and more data is becoming available, the SNF must be able to adapt to much larger sample sizes or conversely, to generate the transpose SNF where features are clustered rather than samples.

Results

Conclusion and Future Work

During the course of implementing our graph-based SNF, we encountered a few interesting problems related to the data management of the network.

Data Loading

Neo4j allows the uploading of files in csv format located on the local machine. Data used in the SNF fits well in csv format with both patient ID's and each data point labeled appropriately in the raw data. However, when reading a csv file, Neo4j creates an attribute for every column in the file and, when dealing with large and possibly variable numbers of attributes, this can make for difficulties querying the data later on. The underlying patient data may be more useful stored in a relational database accessed from each node which in turn can be accessed through the patient ID and its relation of similarity to neighboring nodes.

Cluster Definition and Statistics

Once the similarities of the nodes have been computed, defining the number and members of each resulting cluster is critical to the usefulness of the SNF.

Our solution attempts to go one step further by adding feature recognition in order to define a most probable identifying data pattern for a cluster. Having the knowledge of

possible indicators for subtypes would not only be useful information for researchers, but also for implementing further network queries.

Insert Queries

Inserting new patients into the network is difficult to implement efficiently because new patients could potentially have similarity to any of the existing members of the network. Assigning a patient to a cluster could require comparing every attribute among every existing patient to define new similarities. However, with the addition of recognized features for each cluster, it may be possible to only compare a select few features at first to identify a cluster then focus on similarity only to other patients in that cluster.

Team Contributions

References

[1] Bo Wang, Aziz M Mezlini, Feyyaz Demir, Marc Fiume, Zhuowen Tu, Michael Brudno, Benjamin Haibe-Kains, and Anna Goldenberg. Similarity network fusion for aggregating data types on a genomic scale. *Nat Meth*, 11(3):333–337, 03 2014.