Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam

Graph Mining: Project presentation

Davide Mottin, Konstantina Lazaridou

Hasso Plattner Institute

Graph Mining course Winter Semester 2016

Lecture road

Slides from previous lectures

Project description

Useful links

… let us go back …

Slides are available in:

https://hpi.de/fileadmin/user_upload/fachgebiete/mueller/courses/graphmining/GraphMining-06-NodeClassification.pdf

Lecture road

Slides from previous lectures

Project description

Useful links

What is the project about?

- The project is an analysis of a **SIGNIFICANTLY** large network (>10k nodes/edges)
- You should choose an algorithm and a network for your analysis
- We will provide some links to graph datasets and available algorithms, but you are welcome to choose others as well
- You should evaluate the performance of the approach and its results, report your findings, show why the analysis is useful and propose ideas/directions to improve it

Algorithms: four options

1. Request the code from the authors of a paper you like or implement the technique by yourself (be careful, it requires a **lot** of time!)
2. Implement a **SIMPLE** algorithm for a specific task by yourself
3. Use one of the tools we will present (or any other tool for graph analysis)
4. Compare two or more existing techniques in at least one dimension (efficiency, effectiveness, scalability, accuracy, specific limitations/characteristics, maximum data size they can handle,...)

Project's general goal

- The main idea is to have the insights of the network with a specific technique, retrieve interesting facts, and critically analyze the algorithm's performance and results
- The project should be fun and give you an idea of the properties of the network and the algorithm of choice!
- The analysis should not be trivial but also not too sophisticated

Project rules

- The project can be done in groups of 1 or 2 people
- All projects must be different
- The handout is a report of **max 5** pages with a set of statistics and insights
- Presentation of the results must be **understandable** and conclusive (visual examples, different use cases)
  - You could visualize some important parts of the graph: for node classification you can show different classified nodes, for graph querying how the algorithm behaves with particular patterns (stars, triangles, cores ...), ..

Report rules

- Use the following LaTex template: **Link**
- The report must be handed over on **January 24, 2017**, no extension is allowed
- The report should be sent via email in a single PDF file in the format: **[Last-name-1]-[Last-name-2]-report.pdf**

Report content (more details in the tex file)

1. Main characteristics of the dataset(s)
2. Data preprocessing steps and storing choices
3. Description of the methodology and the kind of analysis
4. Presentation of the results - Show usefulness and novelty of the results
5. Compare the runtime and the scalability of the used method
6. Comment the limitations of the used method

Lecture road

Slides from previous lectures

Project description

Useful links

Graph data repositories

- https://snap.stanford.edu/data/
- http://konect.uni-koblenz.de/
- http://irl.cs.ucla.edu/topology/#data
- http://networkrepository.com/
- https://networkdata.ics.uci.edu/index.php
- http://nexus.igraph.org/api/dataset_info
- http://www.geonames.org/export/
- https://github.com/caesar0301/awesome-public-datasets
- http://www.icwsm.org/2016/datasets/datasets/
- https://datahub.io/dataset?tags=ontology&_tags_limit=0

Tools and libraries for graph analysis

Graph indexing and graph querying

- Graph creation, graph generation, computing structural properties, visualization
  - iGraph (R, Python, C)
  - Snappy (Python)
  - Jung (Java)
  - NetworkX (Python)
- GRAIL: reachability queries
- RQ-tree: Reliability search in uncertain graphs

More features of the above-mentioned libraries

- **iGraph R/Python/C**, **NetworkX**: attribute-based querying
- **iGraph C**: computing spanning trees, clustering coefficient, examining graph isomorphism
- **Snappy**: finding communities, computing node centrality
- **Jung**: computing maximum flow
- **NetworkX**: long list of algorithms!

Frequent subgraph mining

- Gradoop (Java, Hadoop)
  - distributed graph analytics including a frequent subgraph mining approach
- Xifeng Yan software packages
    - gSpan, frequent subgraph mining algorithm (Java/C++)
    - Gupta 2014, approach for interesting subgraph discovery in social networks (Java)
- MUSK: Markov Chain MonteCarlo method to guarantee maximally frequent subgraphs to be sampled

Social network analysis

- Analysis
  - Snappy (Python) : community detection, connected component computation, k-core computation, etc.
- Analysis and visualization
  - Windows software packages
    - Ucinet (text-book)
    - Pajek
  - Other tools
    - Gephi visualization tool: computation of simple metrics, visualizing timestamps, finding clusters
    - NodeXL : Microsoft excel tool
- More tools for SNA

Multi-purpose libraries

- Pegasus (Java, Hadoop)
  - parallel computation of node degree, PageRank score, connected components, random walks with restart
- Gradoop (Java, Hadoop)
  - distributed graph analytics
- Giraph (Java, Hadoop)
  - large-scale graph processing

Multi-purpose libraries (2)

- NetworkX (Python)
  - Algorithms: cliques, homophily, communities, clusters, link prediction, etc.
- GraphX (Spark, Scala)
  - parallel computation of triangles, connected components, PageRank score
- Graph Databases
  - neo4j (Cypher, Java) : tutorial and features
  - sparksee (.Net, C++, Python, Objective-C, Java)

Multi-purpose libraries (3)

- Graph-tool (Python)
  - graph statistics, centrality measures, topological algorithms, network inference
- jGraphT (Java)
  - graph theory algorithms
  - visualization

Diffusion models

- Netinf (SNAP)
  - information cascade tracking
- NDVT (R)
  - R-based tool that renders dynamic network data from 'networkDynamic' objects as movies, interactive animations, or other representations of changing relational structures and attributes (documentation)
- EpiModel (R)
  - R-based tool for simulating and analyzing mathematical models of infectious disease (documentation)

Node classification and similarity

- FaBP
  - Danai Koutra, PKDD 2011: Fast Belief Propagation for node classification in graphs

Link prediction

- NetworkX (Python)
  - algorithm list including a link prediction approach
- Github projects
  - LPmade (Java)
    - manual
  - LinkPred (Python)

Graph summarization

- Vog (Python)
  - Danai Koutra, Statistical Analysis and Data mining 2015: Summarizing and Understanding Large Graphs

Data and algorithm selection

- You are welcome to choose the dataset and algorithm/tool you prefer, even outside the list!
- Let us know about your decision before you begin working on your analysis, so that we can give you feedback and help if necessary :)

Time for discussion