Graph Mining: Project presentation

Davide Mottin, Anton Tsitsulin

Hasso Plattner Institute

Graph Mining course Winter Semester 2017

Lecture road

Project description

Datasets

Tools and libraries

Lecture road

Project description

Datasets

Tools and libraries

What is the project about?

- The project is an analysis of a SIGNIFICANTLY large network (>10k nodes/edges)
- Choose one dataset, convert into a graph and perform an analysis (predictions, statistics, communities, …) to highlight certain aspects of the data
- Choose the analyses you want to perform and highlight and the methods you want to use (e.g., correlations with degrees, label propagation, …)
- We will provide some links to 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: five 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 present (or similar 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,...)
5. Look up GitHub search ;)

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
- Start by **some hypothesis** on the data, for instance:
  - people in Berlin tend to like electronic music
  - fantasy movies are rated higher among young people
  - Chess players with the same score will play together more often
- The project **should be fun** and give you an idea of the **properties** of the network and the algorithm(s) of choice!
- You can **change the algorithm** to fit specific purposes
- The analysis should not be trivial but also not too sophisticated

Project rules

- The project can be done in groups of **2 or 3 people**
- All projects must be different
- The handout is a report of **max 3-5** pages with a set of statistics and insights
- Presentation of the results must be **understandable** and conclusive (visual examples, different use cases)
  - You MUST 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: tiny.cc/GMREP
- The report **must** be handed over on Feb 16th, 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
    1. Show **usefulness** and **novelty** of the results
    2. Compare the **runtime** and the **scalability** of the used method
5. Comment on the limitations of the used method

Important dates

- **22/11**: Groups and preliminary ideas on data
- **20/12:** Informal feedback on the projects (no evaluation)
- **31/01 - 07/02**: Project in-class presentation
- **16/02:** Report hand over (via email)

Community Detection: Analysis of Political communities in Reddit

- **Data**
  - Reddit is a social news website
  - Discussions are organized in groups (subreddits)
  - 234 Million users, >100k subreddits
  - The project analyses 2014 politics subreddit
- **Preprocessing**
  - Nodes are users, edges are interactions among them
  - Keep only connected components
  - Two different subsets
- **Algorithms**
  - Modularity optimization with Girvan-Newmann
  - Spectral clustering
- **Network analysis**
  - People with similar political thoughts cluster together

Preprocess the data in a smart way

Node Classification: League of Legends

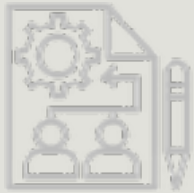
- **Data**
  - League of Legends is a Multiplayer Online Battle Arena (MOBA)
  - 100 Million players/month
  - 5 players compete each other in every battle
  - The project predicts user skills and classification of champion roles
- **Preprocessing**
  - Nodes are players, edges are battles or hero selections
  - 180.000 matches, 6 patches
  - Every patch changes the dynamics of the game
- **Algorithms**
  - Spectral clustering on normalized adjacency
  - Clustering coefficient algorithm + several variants
- **Network analysis**
  - Hypothesis: Cluster represent roles in the game, since every hero is usually assigned to the same role.

Try small variations of algorithms (e.g. weighted)!

Other good example projects

- Temporal Analysis of Resilience in the US Airline Network
- Structure and Evolution of Bitcoin Transaction Network
- Player Rankings and Outcome Prediction in Tournaments
- League of Legends: The Meta as a Function of Time and Rank
- Predicting Reverts in WikiGraph Properties
- The Interplay of Money and Politics in the United States House of Representatives
- Video Game Community Analysis - Destiny Clan Wars
- Identifying Efficient Learning Paths
- Predicting New Ratings and Creating Business Recommendations on Yelp
- The Time they are a-Changin': Evolving Communities in a Musician Network
- Detecting and Analyzing Political Communities in Poltics Sub-Reddit Comments
- Link Prediction for Pinterest Board Recommendation
- Community Detection and Network Analysis on Random Acts of Pizza
- Understanding Distance Effects in Global Food Trade Through Network Analysis
- Predicting Stock Movements Using Market Correlation Networks
- Arrest Charges' (Non)-Independence in Black and White Men
- Hot or Not? Finding Star Projects in Coding Social Network

Lecture road



Project description

Datasets

Tools and libraries

Datasets

- Common sources:
  - Wikipedia, Wikidata,…
  - Academic torrents
  - /r/datasets
  - govdata.de, daten.berlin.de – Germany Open Data Law
  - Archive Team
  - SNAP
- We can provide some datasets/samples
- Preprocessing should not eat up all your project time
  - Seek for partially preprocessed data
- Don’t be afraid to subsample!

Wikipedia

- Mother of many graph mining projects
  - Many types of links, rich data
  - Complete history is available
- Wikidata offers free preprocessing
  - Automatically extracts links from wiki in different languages
- Don’t forget other projects:
  - Wikiquote

Wikipedia: example projects

- Hidden influencers in human history
  - Person-to-person graph from wikidata
  - Compare influencers from graph perspective to article lengths
- Twin Towns (Sister cities)
  - Berlin is a sister city of Moscow, London, and Windhoek (Namibia)
  - Is there any structure in how they are created? Communities?
- War network between countries
  - France and England are arch-nemesis
  - Italy and literally any country around
  - Can we detect more?

Chess networks

- lichess.org
- FIDE rated games

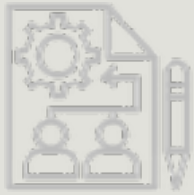
- Example questions:
  - Are players of similar ratings grouped together?
  - Do players learn openings from each other?
  - How connected are the low-rated players?
  - Do chess openings form “player types”?

DoTA networks

- Yet another MOBA game
- 1.1 **billion** matches available

- Example analyses:
  - How do new heroes affect the game
  - How the hero centrality changes over time
  - … you are probably better experts

Lecture road



Project description

Datasets for the project

Tools and libraries

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

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
- LinBP and SSLH
  - Linearized and improved single pass belief propagation and semi-supervised with homophily
- Any classifier (such as Linear Regression in Numpy)

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

Graph Embeddings

- Deepwalk: https://github.com/xgfs/deepwalk-c
- Node2vec: https://github.com/aditya-grover/node2vec
- NEU: https://github.com/thunlp/NEU
- LINE: https://github.com/tangjianpku/LINE
- HOPE: https://github.com/AnryYang/HOPE

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