Project 1 – Community Detection in a Facebook network

Aim:

The aim of this project is to build a tool to identify communities and influencers in a social network. The tool should be able to:

- Load a social graph
- Run community detection and some analytical algorithms
- Visualize the network

Tasks:

1- Load the dataset

- Load the following Facebook dataset (The nodes represent the users and links represent the friendships between users) provided from:

http://snap.stanford.edu/data/egonets-Facebook.html

2- Implementation

- Implement Louvain method of community detection
- Implement Degree centrality measure
- Implement Random Walk algorithm

Perform the following tasks using the previous implementations:

- Identify users' communities in the Facebook network using Louvain method.
- Identify top 2 users with the highest degree centrality in each community.
- Distribute three messages across the Facebook network using the random walk algorithm, start each walk from a different community using its highest degree centrality user. Terminate the walk when at least one node from each community has been reached.

3- Visualization

- Visualize the output of Louvain method by coloring nodes according to their assigned communities.
- Visualize the output of applying Random Walk algorithm, by highlighting the sequence of nodes selected in a path.

Contact person: Akansha Bhardwaj (akansha.bhardwaj@unifr.ch)

Project 2 - Community detection in an academic network

Aim:
The aim of this project is to build a tool to identify communities and influencers in a academic network. The tool should be able to:

- Load social graph
- Run community detection and centrality methods
- Visualize the network

Tasks:

1- Load the dataset

- Load the Author Network dataset provided in https://aminer.org/lab-datasets/soinf/

The graph consists of authors and coauthor relationships.

2- Implementation

- Implement Girvan-Newman clustering algorithm till 10th iteration level.
- Implement Pagerank algorithm.
- Implement Betweenness centrality measure.

Using the previous implementation, perform the following tasks:

- Use Girvan-Newman algorithm to find clusters of authors.
- Find the top-10 authors with highest betweenness centrality.

3- Visualization

- Visualize the output of Girvan-Newman algorithm by coloring nodes according to their assigned groups.
- Visualize the network and highlight the top 10 authors with the highest betweenness centrality, and top 10 edges with the highest betweenness centrality.

Contact person: Akansha Bhardwaj (akansha.bhardwaj@unifr.ch)

Project 3 - Community detection in a blog network

Aim:

The aim of this project is to build a tool to identify communities and influencers in a blog network. The tool should be able to:

- Load a social graph
- Run community detection and centrality measures
- Visualize the network

Tasks:

1. **Load the dataset**
- Load the blog catalog dataset: http://socialcomputing.asu.edu/datasets/BlogCatalog

2. **Implementation**
- Implement Clique Percolation Method (CPM).
- Implement a procedure to transform an undirected graph to a directed one, by randomly assigning a direction to each edge.
- Implement in-degree, out-degree centrality measure.
- Implement Betweenness centrality measure.

Perform the following tasks using the previous implementations:

- Find a reasonable value for CPM parameter. Apply CPM (with this parameter) to find the resulting k-clique communities.
- Transform the undirected blog catalog dataset into a new directed network.
- Use in-degree, out-degree centralities to find the 10 most influential authors for each centrality on the above transformed directed blog catalog dataset.
- Identify authors who connect several groups together.

3. **Visualization**

- Visualize the output of CPM algorithm highlighting the resulting communities.
- Visualize a network of 10 authors with highest in-degree centrality and their 1 degree neighbors.

Contact person: Rana Hussein (rana.hussein@unifr.ch)

Project 4 - Real World Network Simulation

Aim:

The aim of this project is to implement a tool for simulating real world networks. The tool should be able to:

- Load a social graph
- Run a number of models for approximating real networks
- Compute and plot network characteristics
- Visualize the simulated networks

Tasks:

1- **Load the dataset :** Load the following Emails network (The nodes represent the institution members and links represent the email communication between) provided at: https://snap.stanford.edu/data/email-Eu-core.html

2- **Implementation**

Implement the below models for approximating real networks:
- Random graph model
- Small world graph model

Implement procedures to compute the following properties for a network:
- Degree distribution
- Clustering coefficient
- Average path length

Perform the following tasks using the previous implementation:
- Simulate the Emails network using the two simulation models.
- Compute the three network properties of the real network and of each simulated model.

3- **Visualization**

- Plot the degree distribution graph of the real network and of each simulated model.
- Visualize the generated networks from the two simulation models.

Contact person: Rana Hussein (rana.hussein@unifr.ch)

Project 5 – Community detection in a College Network

Aim:

The aim of this project is to build a tool to identify communities and influencers in a college network. The tool should be able to:

- Load a social graph
- Run community detection and analytical algorithms
- Visualize the network

Tasks:

1- Load the dataset

- Load the following social network dataset (The nodes represent users and links represent messages between them) provided from:

http://snap.stanford.edu/data/CollegeMsg.html

2- Implementation

- Implement Louvain method of community detection
- Implement Pagerank algorithm
- Implement Vertex similarity
- Implement Jaccard similarity

Perform the following tasks using the previous implementation:

- Identify users' communities in the CollegeMsg network using Louvain method.
- Identify top 15 users with highest pagerank centrality in the network.
- Find the pairs with highest vertex, jaccard similarity from among the 15 users identified above.

3- Visualization

- Visualize the output of Louvain method by coloring nodes according to their assigned communities.
- Visualise the top 15 users identified above, highlight the pairs with highest vertex, jaccard similarity.

Contact person: Akansha Bhardwaj (akansha.bhardwaj@unifr.ch)

Project 6 - Blog network behaviour analysis

Aim:

The aim of this project is to analyze the social behavior in a blog network and identify communities and influencers.

The tool should be able to:

- Load social graph
- Run community detection and centrality methods
- Visualize the network

Tasks:

1- Load the dataset

- Load the blog catalog dataset : http://socialcomputing.asu.edu/datasets/BlogCatalog

2- Implementation

- Implement Girvan-Newman clustering algorithm till 15th iteration level.
- Implement degree centrality
- Implement Random walk algorithm.
- Implement a procedure to normalize the degree centrality

Using the previous implementation, perform the following tasks:

- Use Girvan-Newman algorithm to find clusters of authors
- Identify the top 10 users with highest degree centrality
- Distribute two messages across the network using the random walk algorithm. Start each walk from the highest degree centrality user. Terminate the walk when at least 200 nodes have been traversed.

3- Visualization

- Visualize the output of Girvan-Newman algorithm by coloring nodes according to their assigned groups.
- Visualize the network and highlight the top 10 authors with highest degree centrality.

Contact person: Akansha Bhardwaj (akansha.bhardwaj@unifr.ch)

Project 7 - Community detection in a movie network

Aim:

The aim of this project is to build a tool to identify communities and influencers in the film industry. The tool should be able to:

- Load a social graph
- Run centrality measures
- Visualize the network

Tasks:

1. **Load the dataset**

- Load the movie dataset from https://aminer.org/lab-datasets/soinf/ . The dataset consists of an actor-director-film-writer network, and for each actor, the categories of movies they starred in.

2. **Implementation**

- Implement Clique percolation method (CPM).
- Implement a procedure to generate an "actor graph" where an actor is connected to another actor if they starred in at least one common category of movies.
- Implement Betweenness centrality measure.
- Implement Cosine similarity measure.
- Implement Adamic-Adar similarity measure.

Using the previous implementations, perform the following tasks on the generated "actor graph":

- Find a reasonable value for CPM parameter starting from k=3. Apply CPM (with this parameter) to find the resulting k-clique communities.
- Find the top-10 actors with highest betweenness centrality on the above dataset.
- For each of the top 10 actors, find the nodes which are most similar according to cosine, Adamic-Adar measures.

3. **Visualization**

- Visualize the output of CPM algorithm highlighting the resulting communities.
- Visualize a network of top-10 actors with highest betweenness centrality and their 1 degree neighbors.

Contact person: Rana Hussein (rana.hussein@unifr.ch)

Project 8 - Social Network measures analysis

Aim:

The aim of this project is to implement a tool for analyzing the social behavior in graphs using centrality and similarity measures. The tool should be able to:

- Load two social networks
- Analyze both networks using centrality and similarity measures
- Visualize the networks

Tasks:

1. **Load the datasets**

- Load the following Facebook dataset (The nodes represent the users and links represent the friendships between users) provided from: http://snap.stanford.edu/data/egonets-Facebook.html
- Load the following Emails network (The nodes represent the institution members and links represent the email communication between) provided at: https://snap.stanford.edu/data/email-Eu-core.html

2. **Implementation**

- Implement degree centrality measure.
- Implement Betweenness centrality measure.
- Implement pagerank centrality
- Implement procedures to compute the following properties for a network:
    - Degree distribution
    - Clustering coefficient

Perform the following tasks using the previous implementations in both datasets:

- Apply all of the above centrality measures.
- Find the degree distribution.
- Find the clustering coefficient

3. **Visualization**

- Plot a frequency (number of nodes), centrality histogram for all of the above centrality measures for both datasets and compare the plots.
- Plot the degree distribution for both datasets.

Contact person: Rana Hussein (rana.hussein@unifr.ch)

Project 9 – Information Diffusion and Influence in Social Networks: Twitter

Aim:

Implement algorithms to simulate information flow via diffusion and influence in a network based on a real-world Twitter dataset.

Tasks:

1) **Loading**

- Load the dataset from here: https://snap.stanford.edu/data/higgs-twitter.html. You will need the following files:
    a. retweet_network.edgelist.gz
    b. reply_network.edgelist.gz
    c. mention_network.edgelist.gz

These files comprise sets of entries describing connections between users. Entry *u2 u1 n* means that user *u2* retweeted/replied/mentioned user *u1 n* times.

- Preprocess the dataset as follows:
    a) Reverse all edges in graphs, i.e. an entry *7 5 1* (edge from 7 to 5 with weight 1) should be converted into *5 7 1*. We need to do this because we want to model the flow of the information and the entry *7 5 1* means that the information has actually originated from user5.
    b) For tasks 2a, 2b, and 2d sum the edge weights for reply, mention and retweet set (we will not consider the different types of interaction for these tasks). Normalize the edge weight sums between 0 and 1.
    c) For task 2c, we will consider the interaction types, so you will have to maintain edge weights of different types.

2) **Implementation**

    a) The Independent Cascade Model (ICM) algorithm. As greater edge weights indicate a stronger channel, the activation should be based on a randomly generated number that is smaller than the edge weight.
    b) The greedy algorithm to find the best set of initial nodes to be activated in order to maximize the spread with the ICM algorithm.
    c) The computation of Pearson correlation between the level of influence a node has in the network. For this, each node should have 3 ordinal values: 1) the sum of all outgoing ‘mention’ edges, 2) the sum of all outgoing ‘retweet’ edges, and 3) the sum of all outgoing ‘reply’ edges. Pearson correlation should compare the similarity between adjacent nodes (undirected) on the same aspect (type of interaction).

d) The Linear Threshold Model (LTM) algorithm, where the edge weight represents the amount of influence exerted from one node to another. Initialize each node with a randomly generated threshold between 0 and 1 ("resistance to change"). A node is activated once the sum of influence it receives from all activated neighbors (incoming edges) exceeds the threshold.

Hint: Create and load a small graph to iteratively test the correctness of your implementations of the algorithms.

3) Analysis

a) ICM: Plot the cumulative number of activated nodes at each iteration
   i) With the normal ICM algorithm. As there is a randomness to the algorithm, obtain averages from multiple runs.
   ii) With an intervention after [2, 5, 7, 10] iterations by halving activation probabilities (=edge weights).

b) Plot the number of nodes reached (spread) as a function of different interesting values for the budget parameter *k* for the greedy algorithm.
   i) Reverse the directions of all edges and repeat the experiment. Analyze your results.

c) Compute Pearson correlation between the nodes in the network for mention, retweet and reply edges, and compare the values.

d) LTM: Plot the cumulative number of nodes activated at each iteration. Repeat multiple runs with different random thresholds and plot the mean of activated nodes and the standard deviation at each iteration.

You may use the networkx library to represent the graph, but may not use any pre-provided algorithms for the implementations stated above.

Contact person: Laura Rettig (laura.rettig@unifr.ch)

Project 10 – Information Diffusion and Influence in Social Networks: Flickr

Aim:

Implement algorithms to simulate information flow via diffusion and influence in a network based on a real-world Twitter dataset.

Tasks:

1. Loading

- Download the flickr-large dataset from https://aminer.org/data-sna at http://arnetminer.org/lab-datasets/flickr/flickr.rar containing different data files (see website for description).
- Build a graph reflecting different social interactions of the users as follows:
    a. Add an undirected edge of type 'relationship' and weight 1 between two users if there exists a link in user2user.txt
    b. Add an undirected edge between two users indicating their number of shared groups, i.e. the edge would have a type 'groups' and a weight corresponding to the mutual groups, as given based on user2group.txt
    c. Add a directed edge from user1 to user2 indicating the number of times the former left a comment on the image of the latter, i.e. an edge 'comment' with weight being the number of comments as per images_comts.txt
    d. No need to fuss with the clear user names - IDs are enough.
- For tasks 2a, 2b, and 2d sum the edge weights for relationship, groups and comments (we will not consider the different types of interaction for these tasks). Normalize the edge weight sums between 0 and 1.
- For task 2c, we will consider the interaction types, so you will have to maintain edge weights of different types.

2. Implementation

a. The Independent Cascade Model (ICM) algorithm. As greater edge weights indicate a stronger channel, the activation should be based on a randomly generated number that is smaller than the edge weight.
b. The greedy algorithm to find the best set of initial nodes to be activated in order to maximize the spread with the ICM algorithm.
c. The computation of Pearson correlation between the level of influence a node has in the network. For this, each node should have 3 ordinal values: 1) the sum of all outgoing 'comment' edges and 2) the sum of all adjacent 'groups' edges. Pearson correlation should compare the similarity between nodes where a relationship exists on the same aspect (type of interaction).

d. The Linear Threshold Model (LTM) algorithm, where the edge weight represents the amount of influence exerted from one node to another. Initialize each node with a randomly generated threshold between 0 and 1 ("resistance to change"). A node is activated once the sum of influence it receives from all activated neighbors (incoming edges) exceeds the threshold.

Hint: Create and load a small graph to iteratively test the correctness of your implementations of the algorithms.

3. Analysis

a. ICM: Plot the cumulative number of activated nodes at each iteration
   i. With the normal ICM algorithm. As there is a randomness to the algorithm, obtain averages from multiple runs.
   ii. With an intervention after [2, 5, 7, 10] iterations by halving activation probabilities (=edge weights).
b. Plot the number of nodes reached (spread) as a function of different interesting values for the budget parameter *k* for the greedy algorithm/
   i. Reverse the directions of all edges and repeat the experiment. Analyze your results.
c. Compute Pearson correlation between the nodes in the network for mention, retweet and reply edges, and compare the values.
d. LTM: Plot the cumulative number of nodes activated at each iteration. Repeat multiple runs with different random thresholds and plot the mean of activated nodes and the standard deviation at each iteration.

You may use the networkx library to represent the graph, but may not use any pre-provided algorithms for the implementations stated above.

Contact person: Laura Rettig (laura.rettig@unifr.ch)

Project 11 - Social Recommendation Systems (FilmTrust Dataset)

Aim:

The aim of this project is to build a tool that implements and compares different types of recommendation algorithms on a real-world dataset (using LibRec: https://www.librec.net/). The tool should be able to:

- Load a dataset, and show basic statistics of the dataset
- Run different recommendation algorithms on the dataset
- Compare and discuss the results, parameter sensitivity

Tasks:

1. **Loading**

Load and understand the FilmTrust dataset (https://www.librec.net/datasets.html)

2. **Implementation**

Implement a social recommendation system that consists of the following components:

a. Basic user based Collaborative Filtering Recommender with cosine similarity.
   hint: check userknn, CosineSimilarity in LibRec
b. Probabilistic matrix factorization algorithm
   hint: check pmf in LibRec
c. Trust-based matrix factorization algorithm
   hint: check trustmf and in LibRec
d. MAE and RMSE metrics for evaluation
   hint: check mae and rmse in LibRec

3. **Analysis**

Perform the following analytical tasks using the previous implementation:

a. Split the dataset into 80%-20% training-testing data
b. Apply the above algorithms on the training datasets and testing using cross-validation, compare the results of MAE and RMSE (fix the latent dimension as 10 for matrix factorization algorithm)
c. Investigate the sensitivity of the parameters for matrix factorization algorithm, in particular the latent dimension (factor.number in LibRec), e.g., 5, 10, 20, 30, 40, 50. Present your results in table or figure.
d. Investigate the sensitivity of the following parameters for trust-based recommendations: social regularization strength (social.regularization in LibRec).

Contact person: Natalia Ostapuk (natalia.ostapuk@unifr.ch)

Project 12 - Social Recommendation Systems (FilmTrust Dataset)

Aim:

The aim of this project is to build a tool that implements and compares different types of recommendation algorithms on a real-world dataset (using LibRec: https://www.librec.net/). The tool should be able to:

- Load a dataset, and show basic statistics of the dataset
- Run different recommendation algorithms on the dataset
- Compare and discuss the results, parameter sensitivity

Tasks:

1. Loading

Load and understand the FilmTrust dataset (https://www.librec.net/datasets.html)

2. Implementation

Implement a social recommendation system that consists of the following components:

a. Basic item based Collaborative Filtering Recommender with pearson correlation coefficient.
   hint: check itemknn , PCCSimilarity in LibRec
b. Non-negative matrix factorization algorithm
   hint: check nmf in LibRec
c. Social-based matrix factorization algorithm
   hint: check socialmf in LibRec
d. MAE and RMSE metrics for evaluation
   hint: check mea and rmse in LibRec

3. Analysis

Perform the following analytical tasks using the previous implementation:

a. Split the dataset into 80%-20% training-testing data
b. Apply the above algorithms on the training datasets and testing using cross-validation, compare the results of MAE and RMSE (fix the latent dimension as 10 for matrix factorization algorithm)
c. Investigate the sensitivity of the parameters for matrix factorization algorithm, in particular the latent dimension (factor.number in LibRec), e.g., 5, 10, 20, 30, 40, 50. Present your results in table or figure.

d. Investigate the sensitivity of the following parameters for social-based recommendations: social regularization strength (social.regularization in LibRec).

Contact person: Natalia Ostapuk (natalia.ostapuk@unifr.ch)

Project 13 - Social Recommendation Systems (CiaoDVD Dataset)

Aim:

The aim of this project is to build a tool that implements and compares different types of recommendation algorithms on a real-world dataset (using LibRec: https://www.librec.net/). The tool should be able to:

- Load a dataset, and show basic statistics of the dataset
- Run different recommendation algorithms on the dataset
- Compare and discuss the results, parameter sensitivity

Tasks:

1. **Loading**
   Load and preprocess the CiaoDVD dataset (https://www.librec.net/datasets.html)
   hint: you need to generate new file, containing *userId*, *movieId* and *reviewRating*

2. **Implementation**
   Implement a social recommendation system that consists of the following components:
   a. Basic user based Collaborative Filtering Recommender with pearson correlation coefficient.
      hint: check userknn, PCCSimilarity in LibRec
   b. Probabilistic matrix factorization algorithm
      hint: check pmf in LibRec
   c. Trust-based matrix factorization algorithm
      hint: check trustmf and in LibRec
   d. MAE and RMSE metrics for evaluation
      hint: check mae and rmse in LibRec
3. **Analysis**
   Perform the following analytical tasks using the previous implementation:
   a. Split the dataset into 80%-20% training-testing data
   b. Apply the above algorithms on the training datasets and testing using cross-validation, compare the results of MAE and RMSE (fix the latent dimension as 10 for matrix factorization algorithm)

c. Investigate the sensitivity of the parameters for matrix factorization algorithm, in particular the latent dimension (factor.number in LibRec), e.g., 5, 10, 20, 30, 40, 50. Present your results in table or figure.
d. Investigate the sensitivity of the following parameters for trust-based recommendations: social regularization strength (social.regularization in LibRec).

Contact person: Natalia Ostapuk (natalia.ostapuk@unifr.ch)

Project 14 - Social Recommendation Systems (CiaoDVD Dataset)

Aim:

The aim of this project is to build a tool that implements and compares different types of recommendation algorithms on a real-world dataset (using LibRec: https://www.librec.net/). The tool should be able to:

- Load a dataset, and show basic statistics of the dataset
- Run different recommendation algorithms on the dataset
- Compare and discuss the results, parameter sensitivity

Tasks:

1. **Loading**
   Load and preprocess the CiaoDVD dataset (https://www.librec.net/datasets.html)
   hint: you need to generate new file, containing userId, movieId and reviewRating

2. **Implementation**
   Implement a social recommendation system that consists of the following components:
   a. Basic item based Collaborative Filtering Recommender with cosine similarity.
      hint: check itemknn , PCCSimilarity in LibRec
   b. Non-negative matrix factorization algorithm
      hint: check nmf in LibRec
   c. Social-based matrix factorization algorithm
      hint: check socialmf in LibRec
   d. MAE and RMSE metrics for evaluation
      hint: check mea and rmse in LibRec

3. **Analysis**
   Perform the following analytical tasks using the previous implementation:
   a. Split the dataset into 80%-20% training-testing data

b. Apply the above algorithms on the training datasets and testing using cross-validation, compare the results of MAE and RMSE (fix the latent dimension as 10 for matrix factorization algorithm)
c. Investigate the sensitivity of the parameters for matrix factorization algorithm, in particular the latent dimension (factor.number in LibRec), e.g., 5, 10, 20, 30, 40, 50. Present your results in table or figure.
d. Investigate the sensitivity of the following parameters for social-based recommendations: social regularization strength (social.regularization in LibRec).

Contact person: Natalia Ostapuk (natalia.ostapuk@unifr.ch)

Project 15 – Human Computation and Crowdsourcing

Aim:

The aim of this project is to build a tool that implements and compares different types of output aggregation algorithms on real-world datasets. The tool should be able to:

- Load a dataset, and show basic statistics
- Run different output aggregation algorithms on the dataset
- Compare and discuss the results

Tasks:

1. Loading

a. Load and understand the Fashion dataset: https://drive.google.com/open?id=1X9E34IkWhRCAnWHhB3bF-KLX4-Vnr9hD
b. Show the distribution of #candidate influencers labeled per worker and the distribution of worker accuracy for the dataset

2. Implementation

a. Implement the majority voting algorithm
b. Implement the Dawid and Skene (DS) algorithm
c. Implement the Learning-from-Crowd (LFC) algorithm
d. Implement the accuracy and F1 metrics for evaluation

3. Analysis

Perform the following analytical tasks using the previous implementation:

a. Split the labeled candidate influencers (together with their social features) into training (60%), validation (20%) and test set (20%)

b. Train the algorithms on the worker answers and the training set (with negative sampling), and evaluate the algorithms against the gold labels in the test set using the accuracy and F1 metrics
c. Use the validation set to find the optimal negative sampling rate. Plot the change of evaluation results with sampling rate = {0, 0.1, 1, 10, 100}

Contact person: Natalia Ostapuk (natalia.ostapuk@unifr.ch)

Project 16 – Human Computation and Crowdsourcing

Aim:

The aim of this project is to build a tool that implements and compares different types of output aggregation algorithms on real-world datasets. The tool should be able to:

- Load a dataset, and show basic statistics
- Run different output aggregation algorithms on the dataset
- Compare and discuss the results

Tasks:

1. **Loading**
   a. Load and understand the Fashion dataset: https://drive.google.com/open?id=13pIHpp7E24vOVMCyRz1Um-2iDP1jtFSm
   b. Show the distribution of #candidate influencers labeled per worker and the distribution of worker accuracy for the dataset

2. **Implementation**
   a. Implement the majority voting algorithm
   b. Implement the GLAD algorithm with Gaussian priors
   c. Implement an enhanced version of the GLAD algorithm by incorporating the network of candidate influencers
   d. Implement the accuracy and F1 metrics for evaluation

3. **Analysis**

Perform the following analytical tasks using the previous implementation:

a. Split the labeled candidate influencers into training (60%), validation (20%) and test set (20%).

b. Train the algorithms on the worker answers and the training set (with negative sampling), and evaluate the algorithms against the gold labels in the test set using the accuracy and F1 metrics.
c. Use the validation set to find the optimal negative sampling rate. Plot the change of evaluation results with sampling rate = {0, 0.1, 1, 10, 100}.

Contact person: Natalia Ostapuk (natalia.ostapuk@unifr.ch)

Useful libraries:

- You can optionally use the following libraries to load your data:
  - NetworkX: https://networkx.github.io/
  - SNAP: https://snap.stanford.edu/snappy
  - iGraph http://igraph.org/python/
- For visualization: You can use the following libraries:
  - matplotlib http://matplotlib.org/
  - vis.js http://visjs.org/
  - d3.js https://d3js.org/