Node classification and link prediction

You are given six smaller networks with some class information associated with each node.

- Zachary karate club network (2 clubs)
- Davis southern women network (3 groups)
- US college football network (12 conferences)
- Computer scientists collaborations (8 fields)
- Java class dependency network (54 packages)
- Norwegian directors collaborations (2 genders)

All networks are in Pajek format, whereas edge list and LNA formats are also available.

Within the following exercises you will be evaluating node classification and link prediction by using either node or edge features or their embeddings. For this you could use a machine learning library like scikit-learn that includes implementations of standard classifiers such as naive Bayes, SVM, decision tree, random forest, neural network or other. However, possibly the most straightforward way would be to simply generate a tab-separated file for each exercise with nodes or edges in rows and features or embeddings in columns, and upload the file into the Orange data mining software.

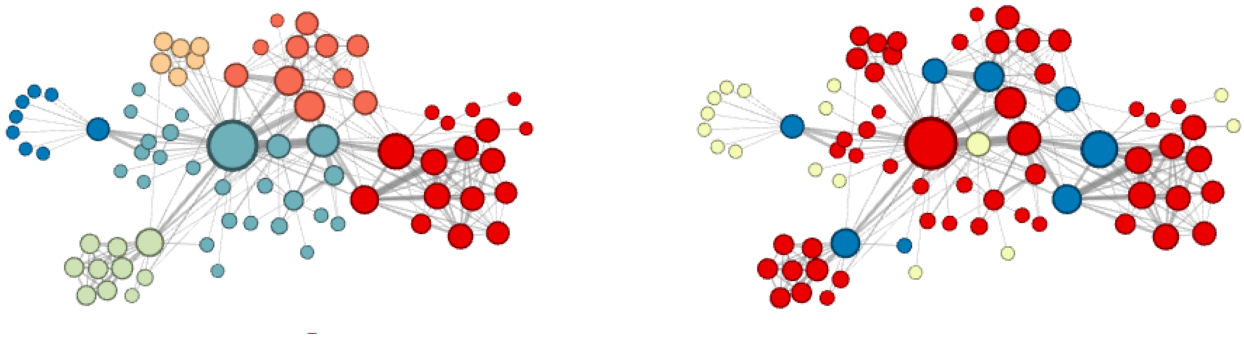
I. Node classification with features

(code) Consider different features of nodes you can compute from network structure. These include node centrality measures, link analysis algorithms, graphlet orbit counts, node community affiliation, stochastic block models, ego-network analysis etc. Train a machine learning classifier on the selected node features and evaluate its performance using some standard measure.

(answer) How well can you classify the nodes by using their network-based features? Is the performance sufficient for practical applications? Which features are most informative for each network?

II. Node classification with embeddings

(code) Consider the node2vec algorithm for computing low-dimensional node embeddings based on biased random walks on a network. Since the networks are relatively small, you might consider reducing the number of embedding dimensions to ≤ 32 and also adjust some other default parameters (e.g., number and length of random walks, random walk biases). Train a machine learning classifier on the generated node embeddings and evaluate its performance using some standard measure. (*Note that the implementation works correctly only if node identifiers are strings.*)



(answer) How well can you classify the nodes by using their network-based embeddings? Is the performance sufficient for practical applications? How does the performance compare to using only network-based features?

III. Link prediction with features

(code) Consider different features of pairs of nodes (either connected or not) you can compute from network structure. These include link prediction indices, node distances, structural equivalence or topological overlap, node community co-affiliation, stochastic block models etc. Train a machine learning classifier on the selected features and evaluate its performance using some standard measure. (*Make sure to compute node features from a network with test links removed.*)

(answer) How well can you predict the links by using network-based features of pairs of nodes? Is the performance sufficient for practical applications? Which features are most informative for each network?

IV. Link prediction with embeddings

(code) Consider the node2vec package for computing low-dimensional embeddings of pairs of nodes (either connected or not) based on biased random walks on a network. These can be defined as simply concatenation or some aggregation of embeddings of individual nodes (e.g., element-wise average or multiplication). Train a machine learning classifier on the generated embeddings and evaluate its performance using some standard measure. (*Make sure to compute node embeddings from a network with test links removed.*)

(answer) How well can you predict the links by using network-based embeddings of pairs of nodes? Is the performance sufficient for practical applications? How does the performance compare to using only network-based features?