# Semantic Proximity

## Overview

This tool aims to perform semantic proximity search between nodes on the graph based on anchored metagraphs or node embeddings. The anchored metagraph is introduced in our TKDE19 paper, an extension of metagraph in our ICDE16 paper.

## Citation

**Metagraph-based Learning on Heterogeneous Graphs**.
Y. Fang, W. Lin, V. W. Zheng, M. Wu, J. Shi, K. C.-C. Chang and X. Li
*TKDE* 33(1), 2019, pp. 154--168.

**Semantic Proximity Search on Graphs with Metagraph-based Learning**.
Y. Fang, W. Lin, V. W. Zheng, M. Wu, K. C.-C. Chang and X. Li.
In *ICDE* 2016, pp. 277--288.

## Code and Data

Compiled Java binary (cross platform): Download
Requirement: JRE 1.8 and Trove 3.0.3.
Sample datasets are included. The source of these datasets can be found in the above citations.

## Usage (anchored metagraphs as features)

### Command line

```
java -cp <String> -Dconfig=<String> -Dsplits=<Integer> -Dfdb=<String>
-Dgt=<String> -Dtest=<String> -Dtrain=<String> -Dpred=<String> exec.Learn
```

### Command line arguments

1. `-cp <String>`
   Java class path for the JRE files.

2. `-Dconfig=<String>`
   Model configuration file, which stores the hyper-parameters.

3. `-Dsplits=<Integer>`
   Number of splits for train/test sets (see more details in `-Dtest` below)

4. `-Dfdb=<String>`
   A file in Metagraph Feature Format.

5. `-Dgt=<String>`
   The ground truth file.

6. `-Dtest=<String>`
   The *prefix* of test set filename. The program will automatically append numbers 1, 2, 3, ... (up

to the number of splits) to the prefix as the actual filenames. For example, if `-Dtest=test` and `-Dsplit=10`, then the program will attempt to load test set files named `test1`, `test2`, ..., `test10`.

7. `-Dtrain=<String>`
   The *prefix* of train set filename. Similar to the `-Dtest` argument.

8. `-Dpred=<String>`
   The *prefix* of prediction filename. Similar to the `-Dtest` argument.

The format of configuration, ground truth, train, test and prediction files can be found in the *File Format* section below.

### Sample command line

```
java -cp lib/*: -Dconfig=config -Dsplits=10 -Dfdb=data/dblp/feature.dblp
-Dgt=data/dblp/advisor/ground_truth.txt -Dtest=data/dblp/advisor/test
-Dtrain=data/dblp/advisor/train -Dpred=data/dblp/advisor/pred exec.Learn
```

## Usage (node embeddings as features)

### Command line

```
java -cp lib/*: -Dconfig=<String> -Dsplits=<Integer> -Demb=<String> -
Dgt=<String> -Dtest=<String> -Dtrain=<String> -Dpred=<String> exec.LearnEmb
```

### Command line arguments

1. `-Demb=<String>`
   A file containing node embeddings in text format. The first line contains two integers, the number of nodes and number of dimensions. Each subsequent line contain the embedding of a node. All lines are space delimited.

2. All other arguments are the same as using anchored metagraphs as features above.

## File Format

### Configuration file

Stores the following hyperparmeters of the learner. Default values are included in the sample file.

1. `GD_STEP=1`
   Learning rate for gradient descent.

2. `GD_EPSILON=1E-5`
   Maximum relative different before gradient descent stops.

3. `GD_TRY=5`
   Number of trials to perform gradient descent using different seeds.

4. `GA_FW=1.01`
   Factor to adjust the learning rate when the loss is decreasing after each iteration.

5. `GA_BW=0.5`
   Factor to adjust the learning rate when the loss in not decreasing after each iteration.

6. `MU=5`
   Scaling factor in Eq. (4) of our TKDE19 paper.

7. `LOG_FEATURE=true`
   Whether to apply `log1p` transformation on the metagraph features.

8. `SILENT=false`
   Whether to suppress messages.

### Ground truth file

Each line represents one query, delimited by tabs. The first column in each line represents the query node ID, and subsequent columns represent the ground-truth (positive) nodes ranked by proximity.

### Train file

Each line contains a triple `(q, a, b)`, where `q` denotes the query node, and `a, b` are two nodes such that node `a` should be ranked higher than node `b` w.r.t. `q`.

### Test file

Each line represents a query, delimited by tabs. The first column in each line represents the query node ID, and subsequent columns represent the candidate nodes (both positive and negative) in randomized order.

### Prediction file

Each line represents a query, delimited by tabs. The first column in each line represents the query node ID, and subsequent columns represent the candidate nodes (both positive and negative) in ranked order. Note that the query nodes may appear in an arbitrary order, different from the corresponding test file.

## Disclaimer