Source Code Usage Instruction

Source Code Usage Instruction

A List of things you need to keep in mind

- The DIG is implemented in CompetitionGraph Package.
- The machine learning algorithms and policy model are implemented in ML_Models package.
- For challenge and developer feature encoding and some data preprocessing modules of the system, refer to the DataPre package.
- The Utility package contains some personalized tag definition, user function and testing scripts.
- Before running the system, make sure to configure following settings:
  - install python 3.x;
  - pip install keras, tensorflow, scikit-learn, imbalance-learn, numpy. Pymysql, networkx;
  - install mysql database;
  - refer to the topcoder project at: https://github.com/lifeloner/topcoder for newest data crawler implemented in JAVA.
  - customize local mysql database ip and port according to local machine configuration.
- Make sure that the hierarchy of data folder is same in local disk.
- Run DataPre package script to start Data Extractor and generate input data for the system.
- Run ML_Models package script to start meta-model training and optimal meta feature searching.
- Test Policy Model.

Start Cyber Crawler to collect Data

We do have a database in our laboratory, but due to the size and continuously updating of our database, it is not a good way to put the database here. Instead, we put the tools for data collection here, thus everyone can get enough data as they want. If you are eager for our data, contact me via the anonymous email mail@{1196641807@qq.com}.

- Install mysql database into your computer with a linux system, and configure mysql ip and port according to the instruction of https://www.mysql.com/.
- refer to the topcoder project at: https://github.com/lifeloner/topcoder for newest data crawler implemented in JAVA.
- After downloading the java crawler maven project, please use intelliJ idea at: https://www.jetbrains.com/idea/ to deploy the crawler jar package in your machine
- Configure the ip and port of your crawler according to the the configure of mysql database
- Start run the crawler by the following command which will run in background:

  nohup java –jar crawler.jar &

Prepare Your Programming Environment

- We develop the whole system using python, so we recommend you to install an anaconda virtual python3.6 environment at: https://www.anaconda.com/
- We need to connect to mysql database, apply machine learning and deep learning algorithms, use page rank to generate developer rank score, etc. So readers have to install following python packages:
  - Pymysql, scikit-learn, lightgbm, xgboost, numpy, tensorflow, keras, network, imbalance-learn,pandas
- Besides, due to the computation and ram requirement is quite large, we recommend you prepare a machine with following requirement
  - 256G RAM or more
  - At least 12 cores of CPU
  - 1TB+ Disk memory
  - TitanXP NVIDIA GPU is recommended for boosting computation
  - Make sure the bandwidth is at least 1000Mb/s if the mysql database is not in your programming machine

Construct Input Data

- Configure the datra/dbSetup.xml and set ip and port as same as the machine running mysql database, copy data/viewdef.sql and run it in your mysql client to create view for initial data cleaning.
- You need to encode Developer and Challenge features at first
  - Run TaskContent.py of DataPre package to generate challenge feature encoding vectors and build clustering model
  - Run UserHistory.py of DataPre package to generate developer history data
  - Run DIG.py of CompetitionGraph package to generate developer rank score data
- Run TaskUserInstances.py of DataPre package to generate input data
  - Adjust the maxProcessNum of DataInstances class to adapt your computer CPU and RAM
  - For training,set global variant testInst=False. The value of variant mode in global means 0-registration training data input, 1-submission training data input, 2-winning training data input. You have to run the script under the 3 values.
  - Generate test input data via set mode=2 and testinst=True

- After finished running all the above scripts, check whether the generate traing input and test input data is completed via running the TopcoderDataset.py

Train Meta Models

- Run XGBoostModel.py of ML_Models package
  - Feed “keepd” as key of tasktypes and run the script for 3 times with mode =0,1,and 2
  - Feed “clustered” as key of tasktrypes and run the script for 3 times with mode=0,1,and 2
  - After finished this, the meta model implemented using XGBoost algorithms can extract registration meta-feature, submission meta-feature and winning met-feature of all datasets
- Run DNNModel.py of ML_Models package in the same way as XGBoostModel.py
- Run EnsembleModel.py of ML_Models package in the same way as XGBoostModel.py

- Generate the performance of all the winning meta models via running MetaModelTest.py of ML_Models package
  - Readers can build winning predictor based on the performance results

Run Policy Model and Baselines

- Run BaselineModel.py of ML_Models package to build the baseline models we mentioned in the paper
  - After building baseline models, run the MetaModelTest.py of ML_Models package again but pass the model name as the names of classes of the baseline model in BaselineModel.py to generate performance results

- Run PolicyModelTuning.py of ML_Models package to apply meta-learning meta-feature selection process and generate the meta data (meta-feature, performance).
  - Readers can refer to MetaLearning.py of ML_Models package which implemented some new learning process but may not be global optima