CS 550: Massive Data Mining and Learning – Project Description

Option 1: Recommender System

1. Overview
   Recommendation systems widely exist in many online applications, in this project, we work on typical and classical recommendation system scenarios, such as product recommendation in E-commerce (i.e., Amazon), or movie recommendation in online review systems (i.e., MovieLens). For the e-commerce dataset, we have 24 product domains from Amazon (e.g., cell phones, clothing, beauty, etc.), and each product domain corresponds to a dataset, the structure of a dataset will be exposited in the next section; for the movie recommendation dataset, we have a "small" dataset (1M) and a "full" dataset (265M). **You only need to work on one dataset, whichever you like.**

   After you select a particular dataset to work on, this project will mainly consist three steps: 1) Data Processing, create the training and testing dataset from the original dataset; 2) Conduct rating prediction and make evaluation based on MAE and RMSE; 3) Conduct Top-N Recommendation and make evaluation based on Precision, Recall, F-measure, and NDCG.

2. Dataset
   a. E-commerce Dataset

   A complete Amazon dataset publicly available online (https://nijianmo.github.io/amazon/index.html). This dataset contains user reviews (numerical rating and textual comment) towards amazon products on 29 product categories, and there is an independent dataset for each product category. **We use the "Small subsets for experiment" (the 5-core dataset) on the website, which can be downloaded directly from the website. You can select one product domain to work on.**

   The structure of the dataset has been explained on the website with detailed examples. Basically, each entry in a dataset is a user-item interaction record, including the following fields:
   - **user-id**: which is denoted as "reviewerID" in the dataset
   - **product-id**: which is denoted as "asin" in the dataset
   - **rating**: a 1-5 integer star rating, which is the rating that the user rated on the product, it is denoted as "overall" in the dataset
   - **review**: a piece of review text, which is the review content that the user commented about the product, it is denoted as "reviewText" in the dataset
   - **title**: the title of the review, which is denoted as "summary" in the dataset
   - **timestamp**: time that the user made the rating and review
   - **helpfulness**: contains two numbers, i.e., [#users that think this review is *not* helpful, #users that think this review is helpful]

- **Image:** for each product, the dataset product the image of the product in a form of a 4096-dimensional vector that is learned by a CNN deep neural network (these vectors are provided in an independent dataset "Visual Features", also in the website, which is very large)
- **Metadata:** some metadata information for each product, including product title, price, image URL, brand, category, etc. It is also provided as an independent dataset ("Metadata"), which is also very large.

b. Movie Review Dataset
The dataset can be downloaded from here:
https://grouplens.org/datasets/movielens/latest/
It includes the user_ID, item_ID, user-item ratings, time, user-movie tags, as well as the metadata of the movies, such as title and genre. Details can be seen in this readme file: http://files.grouplens.org/datasets/movielens/ml-latest-README.html

Depending on the algorithm that you want to design, you may not use all the information available in the dataset. In the simplest case, you may only use the user-id, item-id, and ratings to finish the project, but these information is sufficient enough to design very complicated algorithms (most collaborative filtering and matrix factorization algorithms are only based on ratings).

If you want to design more advanced recommendation algorithms that may achieve better prediction and recommendation performance, you may use other information sources such as review text (based on NLP techniques), timestamps (for time-aware recommendation), images (for visual recommendation), or metadata (for content-based recommendation).

3. Required Tasks
   a. **Data selection and preprocessing:**
   First you need to select a dataset. If you are not very familiar with very large scale data processing or your computing facility (e.g. your laptop) is not powerful enough to process very big dataset, you may select relatively smaller dataset to work on.
   After you select a dataset, you need to create a training dataset and a testing dataset from therein for the experiment. A recommended standard pre-processing strategy is that: for each user, randomly select 80% of his/her ratings as the training ratings, and use the remaining 20% ratings as testing ratings. At last, the training ratings from all users consist the final training dataset, and the testing ratings from all users consist the final testing dataset.
   Because each rating is accompanied with a single piece of textual review, so the reviews are also automatically split into training and testing sets.
   b. **Rating Prediction**
   Based on the training dataset, i.e., the information that we treat as

already known, you should develop a model/algorithm to conduct rating prediction, i.e., to predict the ratings in the testing set as if we didn't know them. You may use any existing popular algorithm (e.g., user-based CF, item-based CF, Slope One, Matrix Factorization) or develop new algorithms by yourself.

After predicting the ratings in the testing set, evaluate your predictions by calculating the MAE[1] and RMSE[2].

c. **Item Recommendation**

The final step is to create a recommendation list (a ranking list of recommended items) for each user, and the length of recommendation list should be 10. Note that, the recommended items should be items that the user didn't purchase before, i.e., you should avoid recommending an item that the user has already rated in the training dataset, instead, your algorithm should try the best to recommend the items in the testing set.

A simple strategy to create such a recommendation list for a user is to predict the ratings on all the items that user didn't buy before (as in step 2), then rank the items in descending order of the predicted rating, and finally take the top 10 items as the recommendation list. Of course, you may develop other recommendation algorithms to create a recommendation list.

After the recommendation list is created (we called it a top-10 recommendation list), you should evaluate the quality of the recommendation list. Remember that you have already holdout 20% purchased items for each user as testing items, then you can calculate the following measures for evaluation:

1. Precision: percentage of testing items in the 10 recommended items, calculate the precision for each user first, then average the numbers from all users
2. Recall: percentage of recommended testing items in all the testing items for a user, also average the recall of all users to get the final recall
3. F-measure: F=2*Precision*Recall / (Precision + Recall)
4. NDCG: Normalized Discounted Cumulative Gain[3].

4. Optional Tasks

Except for the above required tasks that focuses on recommendation accuracy, you may consider one (or more) optional tasks focusing on trustworthiness of recommender systems.

a. Transparency and Explainability of Recommender Systems
b. Fairness and Unbiases of Recommender Systems
c. Controllability of Recommender Systems
d. Privacy Protection for Recommender Systems
e. Robustness and Anti-attacks for Recommender Systems

---

[1] https://en.wikipedia.org/wiki/Mean_absolute_error

[2] https://en.wikipedia.org/wiki/Root-mean-square_deviation

[3] https://en.wikipedia.org/wiki/Discounted_cumulative_gain

This is a totally open-ended task, you can feel free to come up with your project problem, solution, and evaluation methods for any one or more of the above trustworthiness perspectives. Depending on the quality of implementation, you will earn up to 5 bonus points for the optional tasks.

Option 2: Graph Analysis

1. Overview
   Graph is a widely used structure in various applications such as social networks, citation networks, molecular graphs, etc. This project option provides several graph data for graph analysis, including link prediction and node classification tasks.

2. Dataset
   a. Citation network dataset
      The Cora citation network dataset consists of 2708 scientific publications classified into one of seven classes. The citation network consists of 5429 links. Each publication in the dataset is described by a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 1433 unique words. The dataset description and the data download link can be found in the following link: https://relational.fit.cvut.cz/dataset/CORA

   b. The OGB Large Graph
      The OGB large graph dataset is provided by the KDD Cup competition 2021 (https://ogb.stanford.edu/kddcup2021/). It proposes a large-scale graph ML competition, OGB Large-Scale Challenge (OGB-LSC), to encourage the development of state-of-the-art graph ML models for massive modern datasets. Specifically, it presents three datasets: MAG240M, WikiKG90M, and PCQM4M, that are unprecedentedly large in scale and cover prediction at the level of nodes, links, and graphs, respectively. **You only need to select on of the three datasets to work on.**
      The dataset description and the data download link can be found in the following link: https://ogb.stanford.edu/kddcup2021/

3. Required Tasks
   The tasks are very similar to Project Option 1, it also contains three steps:
   a. Step 1: Data prepressing and split, create a training dataset and a testing dataset for experiment
   b. Step 2: Link Prediction, develop an algorithm to predict the links in the testing set based on the information (nodes, features, edges) in the training dataset, and evaluate the predictions based on precision, recall, F-measure.
   c. Step 3: Node classification, predict the node labels in the testing dataset and evaluate the performance based on precision, recall, F-measure.

4. Optional Tasks
   Similar as the recommendation option, except for the above required tasks that focuses on graph prediction accuracy, you may consider one (or more) optional tasks focusing on trustworthiness of graph analysis.
   a. Transparency and Explainability of Graph-based Prediction
   b. Fairness and Unbiases of Graph-based Prediction
   c. Controllability of Graph-based Prediction
   d. Privacy Protection for Graph-based Prediction
   e. Robustness and Anti-attacks for Graph-based Prediction

   Also, this is a totally open-ended task, you can feel free to come up with your project problem, solution, and evaluation methods for any one or more of the above trustworthiness perspectives. Depending on the quality of implementation, you will earn up to 5 bonus points for the optional tasks.

Project Submission

1. The code of your project
2. A project report written using the provided latex template
3. The presentation slides of your project

All of the above documents should be put into one single folder and compressed into one .zip file, and name the zip file using the NetID of all team members, e.g., NetID1_NetID2_NetID3.zip.

References

1. You can design your model and conduct experiments based on some open-source toolkits, such as PyTorch or TensorFlow, however, you should not simply and directly use the packaged algorithms in the toolkits as your own model, instead, you can use the functions in these toolkits as building blocks to design and implement *your own* model.
2. There are a lot of research papers using this dataset, some of the examples are listed in the following, you may refer to these papers if you want to try something cool and develop the recommendation algorithm for yourself.
   a. Joint Representation Learning for Recommendation with Heterogeneous Information Sources (https://dl.acm.org/citation.cfm?id=3132892)
   b. Towards conversational search and recommendation: System ask, user respond (https://dl.acm.org/citation.cfm?id=3271776)
   c. Recommendation as Language Processing (RLP): A Unified Pretrain, Personalized Prompt & Predict Paradigm (P5) (https://arxiv.org/abs/2203.13366)
   d. Neural Collaborative Reasoning (https://arxiv.org/abs/2005.08129)
   e. Causal Collaborative Filtering (https://arxiv.org/abs/2102.01868)
   f. User-oriented Fairness in Recommendation (https://arxiv.org/abs/2104.10671)

g. Personalized Transformer for Explainable Recommendation (https://aclanthology.org/2021.acl-long.383/)
h. Counterfactual Explainable Recommendation (https://arxiv.org/abs/2108.10539)