**Exercise2 tasks:**

* information given sufficient for reproducibility:  
  workflow, data (subsets, test-train-split), random seeds, software (versions), hardware
* are reproduced results equal to paper?  
  if not: reasons?
* examine paper's experimental design:
  + have results been confirmed
  + CV
  + different random seeds compared
  + variance reported?
  + significance tests
  + flaws in the experimental design?
* documentation of workflow & tools  
  differences compared to paper

**Paper analysis**

**Material given for reproducing:**

* all material available as git-repository on github
* code as a single python file
* all settings for each dataset as config-files, that are loaded into the main code-file
* datasets included in repository:
  + already preprocessed
  + already split into train & test sets
* versions of used packages
* used CPU and size of RAM
* results and complete printed output for each dataset:
  + Recall@20, NDCG@20
  + configuration settings
  + results of each training epoch including train time, test time, loss, f1, precision, recall, NDCG
  + early stop at epoch number
  + number of best epoch
  + best recall and best NDCG
* exact evaluation measures

**Analysis of experiment setup:**

* Data:
  + ml-1m:
    - MovieLens 1M: 1 mio. movie ratings
    - Preprocessed data taken from another repo:  
      <https://github.com/Wenhui-Yu/LCFN>
    - Preprocessing consists only of parsing integer-values from .dat-file, selecting specific columns (user-id, movie-id) and reshaping the data matrix
    - Split size: train:test ~ 88:12
* early stop triggered:
  + if recall has not improved for *early\_stop\_count* of times
  + *early\_stop\_count* is defined in dataset-specific config-files (between 15-250)
* Sampling (main.py line 238)
  + for sampling data for each batch during training
  + no seeds for numpy random sampling defined  
    (maybe therefore slightly different results with our own run)
* choice of main measures (Recall, NDCG):
  + paper: “Recall@20 and NDCG@20 are chosen as the evaluation metrics as they are popular in the evaluation of GCN-based CF models.”
  + Recall@20:   
    How many of the top 20 relevant values are among the top 20 recommendations (focuses on the number of relevant items found)

<> Precision@20: How many of the recommended items are actually relevant (focuses on the average relevance of all recommendations)

* + NDCG@20:  
    Measures the overall relevance of the top 20 recommendations, where the results are normalized across all recommender-queries
* Evaluation done only with a single train-test-split; therefore, the performance results are not generalizable:
  + The test split might be especially lucky or unlucky, so that the results are not representative
  + Without averaging over multiple measurements (e.g. by cross-validation), no expectation of performance can be determined, since the measurements are only single possible results
  + During training the test set is used to evaluate after each step to determine when to stop training. This introduces a bias from the test set.
  + The main purpose of the paper was to compare the model to the LightGCN approach, which was evaluated under conditions as similar as possible. For this the applied evaluation strategy is well suited.
* Comparison to other approaches done with significance testing:
  + Exact resulting p-values given
  + no code or details about implementation of tests
* total training time, number of training epochs needed and time per epoch were compared to other approaches
* ablation study:
  + parameters and were used to disable certain features and investigate the resulting performance
    - reduced to matrix factorization training with BCE loss, no graph information, no item collaborations
    - no graph convolution
    - no learning of item-item relationships
    - Model performance:
  + Learning of item-item relationships: better performance with “augmenting positive user-item pairs” than with “optimizing between item-item pairs”
  + User-user co-occurrence information does not improve recommender performance
* parameter analysis:  
  influence of model parameters was investigated
  + K: number of selected neighbours
  + and

**Reproducibility**

* Performance analysis:
  + Easy to execute since implementation given as python-file and settings given as config-files
  + Reproducing results for ml-1m: successful, very close to results in paper

reproduced recall off by -0.32%

reproduced NDCG off by -0.11%

* + Possible causes for differing results: no seed given for random sampling of data for batches during training
  + Reproducing amazon: unable to run, because not enough RAM available
* Significance testing for comparison to other approaches: Too little information
  + type of test
  + measure used to compare   
    (probably recall and NDCG)
  + number of runs
  + which parameter was changed between runs and how   
    (probably different dataset splits)
* Efficiency comparison:
  + Epochs for ml-1m:

Small difference might be due to the random sampling too

difference in epochs needed when reproducing: -7 epochs or -5.15%

* + Training time: not reproducible for us as same hardware is not available
* Ablation study:
  + **TODO**: run on amazon dataset with changed parameter settings:
* Parameter analysis:
  + **TODO**: run on amazon dataset with different parameter settings:  
    (evaluated for each parameter separately)
    - 𝐾 in [5, 10, 20, 50]
    - 𝜆 in [0.2 … 1.4] (with 0.2 interval)
    - 𝛾 in [0.1, 0.5, 1, 1.5, 2, 2.5, 3, 3.5]

**Recall@*k*:**

For recommender systems.

Measure regarding (relevance of) the top ***k*** results recommended by the system.

e.g. amount of relevant items in the top *k* recommendations

**NDCG**

*Normalized discounted cumulative gain*

Measure for recommender systems or search algorithms based on the weighted relevance of recommended items

<https://en.wikipedia.org/wiki/Discounted_cumulative_gain>