Item-Item Collaborative Filtering via Graph Regularization

Melle Koper
Supervisor(s): Maosheng Yang, Elvin Isufi
EEMCS, Delft University of Technology, The Netherlands
22-6-2022

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

Abstract

A recommendation algorithm aims to predict the quality of a user's future interaction with certain items based on their previous interactions. As research progresses, these algorithms are becoming increasingly more complicated with the use of machine learning and neural networks. This paper looks into a more simple solution. The recommendation domain can be represented as a graph, meaning different graph regularization techniques can be used to solve the same problem. After running experiments comparing the Item-Item Tikhonov Regularizer and the Sobolev Regularizer to a baseline, the item-item standard Collaborative Filtering method, it is clear that the Graph Regularization techniques outperform the baseline. Given that it has been shown that Collaborative Filtering is a relatively competitive method in this field, outperforming it means that Graph Regularizers are a viable and potentially competitive method for solving the recommender problem.

1 Introduction

A recommendation algorithm analyses the previous interactions between users and items, these interactions are good indicators of future choices. This allows an algorithm to predict or recommend future items to a user. [1] The purpose of a recommender system is to filter information for specific users to avoid an information overload, an increasingly important task as the internet grows with more information.[2] Consider a simple example, watching a movie on Netflix. In order to select a movie, one does not look at all the movies in the catalog. Netflix filters these movies based on the users preferences in order to make the seemingly trivial task of selecting what to watch easier. Selecting a movie is an easy task when selecting from a handful. However, as the catalog increases the task becomes increasingly monumental. Recommender systems aim to make tasks like these easier for the user and can be applied to more than just movies.

In an effort to improve the accuracy and precision of these recommender systems, researchers are drifting to increasingly more complex solutions, which may be the wrong direction. [2] Previous research leans towards that simpler recommendation techniques seem to outperform more complex ones. [3] Large companies like Netflix are major researchers in this field and have therefore published many articles on the subject. In an effort to increase the performance of these recommender systems, Netflix specifically is implementing and testing deep learning techniques with good results. [4] They however conclude that the benefits are very limited, and the classic baselines perform surprisingly well in comparison. [4] Netflix therefore recognizes that deep learning is not necessarily the future for recommendation systems, leading them to conduct research into various other simple algorithms that don't include hidden layers. [5] In the past, research has been focused on the deep learning aspect of recommender systems, however more recent research is showing that simple techniques could be the way forward.

Another field in which relatively simple techniques are used is graph signal processing. Research in this field mainly focuses on the application of these techniques, which include but are not limited to transportation, sensor and neuronal networks. [6] Graph signal processing techniques allow for the ability to denoising and filtering data. [6] Graph regularization techniques have more recently been used to estimate the amount of new coronavirus cases in 2019. [7] In this case, the domain of the problem was structured as a graph allowing for graph signal processing to be applied. [7]

Inspired by previous research suggesting that simple recommendation techniques can outperform more complex ones, this paper aims to analyze the performance of Item-Item graph regularization techniques when compared to a standard baseline. The main experiment will focus on the comparison of the Tikhonov Regularizer and a standard Item-Item Collaborative Filter. [6] [1] This paper is split into 6 parts, the first of which being the introduction. The second part of the paper describes the methodology by which the algorithms will be compared. Part three gives a detailed explanation of the three algorithms implemented. Part four discusses the training of each method and final results of the experiment. In part 5, the discussion, compares the results to an other paper from within the research team, draws conclusions, as well as discusses any future work that is of interest. Finally the last section discusses the responsibility of this research paper, with a focus on the reproducibility of the results.

2 Methodology

To compare the item-item Tikhonov Recommender to the basic Collaborative Filtering method, we will conduct various experiments. This section will elaborate on the domain in which the two methods are compared. As well as the metrics used to evaluate the algorithms.

2.1 Experimental Setup

In the experiment, we compare Item-Item Tikhonov Regularization and Item-Item Sobolev Regularization to the Item-Item Collaborative Filter as a baseline. Each recommender is run using gridsearch on a random 80-20 train test split for 5 rounds. The data on which the recommenders are trained is the MovieLens100K data set. The algorithms are analysed on seven metrics, Root Mean Squared Error, Precision@5, Precision@10, Recall@5, Recall@10, NDCG@5 and NDCG@10. These metrics are collected each round and the algorithms are compared on their best average of all possible hyper parameter combinations.

2.2 Data set

Following the article mentioned above, the experiments will take place using the MovieLens 100K data set [8]. This data set was released in 1998. It consists of 100,000 ratings ranging from 1 to 5 from 943 users on 1682 movies. [8] The data set ensures that each user has rated at least 20 movies. These features allow the MovieLens 100K data set to be a stable benchmark to test recommender systems. The MovieLens data set is also advantageous as not much pre-processing

is necessary to conduct experiments. [9] This allows for results to be more easily compared across papers, as less data pre-prossesing means more similar data sets.

For the purpose of this experiment, the data set has been altered slightly. The data set ensures that each user has rated at least 5 items, however, it does not ensure each item has been rated at least 5 times. [8] Since these experiments are about item-item collaborative filtering techniques, we decided to remove any item rated less than five times in an effort to have relatively similar results when compared to user-user algorithms.

2.3 Metrics

The article *Are we really making much progress?* only takes into account classification and ranking metrics. The metrics used are Precision and Recall. In addition to these rankings, the experiments in this paper will also take into account Root Mean Squared Error and Hit Rate.

Precision, Recall and Hit Rate analyse the top n recommendation problem, where a recommender system suggests the top n items that a user might like. Root Mean Squared Error however, is used to analyse the numerical value of the ratings themselves.

Precision and Recall

Precision is the percentage of relevant movies recommended by the algorithm. While Recall is the percentage of ground truths recommended as positive.

$$\text{Precision(n)} = 100 * \frac{|S(n) \cap G|}{|S(n)|} \tag{1}$$

$$\text{Recall(n)} = 100 * \frac{|S(n) \cap G|}{|G|} \tag{2}$$

Where S(n) is the set of recommended items, and G is the set of actual top n items. [1]

Normalized Discounted Cumulative Gain (NDCG)

NDCG is the ratio of the discounted cumulative gain (DCG) relative to the ideal discounted cumulative gain (IDCG).[1]

$$\text{DCG} = \frac{1}{m} \sum_{u=1}^{m} \sum_{j \in I_u} \frac{g_{uj}}{log_2(v_j + 1)} \tag{3}$$

In this equation g_{ui} is the utility of user u watching item j.

3 Base Line and Tikhonov Recommender

This paper compares two algorithms, the item-item Tikhonov Recommender, and the item-item Collaborative filtering method. Both of which need a similarity matrix.

3.1 Similarity Matrix

A similarity matrix for item item collaborative filtering and the Tikhonov recommender is a matrix containing the similarity measure between each item pair. The similarity matrix for both methods is constructed using the Pearson Correlation Coefficient. The equation below is the Pearson Correlation between two items i and j.

$$w_(i,j) = \frac{\sum_{u \in U_{ij}} (r_u i - \bar{r_i})(r_{uj} - \bar{r_j})}{\sqrt{\sum_{u \in U_{ij}} (r_u i - \bar{r_i})^2 \sum_{u \in U_{ij}} (r_{uj} - \bar{r_j})^2}} \tag{4}$$

[10]

Where U_{ij} is the set of users that have rated both item i and item j. The two methods differ in how they use this matrix.

3.2 Standard Collaborative Filtering

The baseline used in these experiments is a basic version of the K Nearest Neighbors Collaborative Filtering method. Once the similarity matrix above is created, the algorithm only keeps the k largest similarities in each column. Thus creating a matrix containing only the highest similarities per item. With this new similarity matrix, the ratings for missing values are estimated. Each rating is estimated by taking the weighted average of the sum of all ratings most similar to the item in question.

$$\hat{r}_u i = \frac{\sum_{j \in N_u(i)} w_{ij} r_{uj}}{\sum_{j \in N_u(i)} |w_{ij}|} \tag{5}$$

[10]

Where $N_u(i)$ is the set of items rated by user u that are in the neighborhood of item i. The w_{ij} represents the Pearson similarity for items i and j.

In the case an item has no similar items, the average rating of the user is assigned as the predicted rating for that item.

3.3 Tikhonov Collaborative Filtering

The Tikhonov Collaborative Filtering is very similar to the standard version, however after constructing the neighborhood, Tikhonov uses various matrix multiplication techniques to infer the missing data.

The Optimization Problem

In essence, the Tikhonov regularizer is an optimization problem. [6] In the case of Item-Item Tikhonov Regularization it would take the form of the following equation.

$$\mathbf{min}_{\boldsymbol{X} \in R^{UxI}} ||\boldsymbol{Y} - \boldsymbol{X}||_F^2 + \mu \mathbf{tr}(\boldsymbol{X}^T \boldsymbol{L} \boldsymbol{X}) \tag{6}$$

Where L is the Laplacian matrix, $\boldsymbol{X} \in R^{UxI}$ is full ratings matrix and $||.||_F$ is the Frobenius norm.

For the purposes of the recommender problem, this equation is more suitable in its discrete matrix form:

$$\boldsymbol{X}^* = (\boldsymbol{I} + \mu \boldsymbol{L})^{-1} \boldsymbol{Y} \tag{7}$$

[11]

Where X^* is the predictions matrix, I is the identity matrix, μ is a hyper parameter, L is the Laplacian matrix and Y is the original utility matrix.

Similarity Matrix as a Graph

In order for the Tikhonov Regularization to work on the domain of the recommendation algorithm, it is necessary to transfer the domain into a graph processing one. Luckily this is very possible as the similarity matrix itself can be seen as a

graph. Take for example this graph from the paper *Graph Filters for Processing and Learning from Network Data* below [11]:

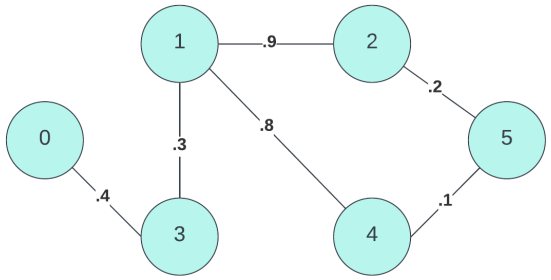

Figure 1: Undirected Weighted graph where the values around the links are the edge weights

This Graph is represented by the following weighted matrix.

$$\boldsymbol{W} = \begin{bmatrix} 0 & 0 & 0 & .4 & 0 & 0 \\ 0 & 0 & .9 & .3 & .8 & 0 \\ 0 & .9 & 0 & 0 & 0 & .2 \\ .4 & .3 & 0 & 0 & 0 & 0 \\ 0 & .8 & 0 & 0 & 0 & .1 \\ 0 & 0 & .2 & 0 & .1 & 0 \end{bmatrix}$$

In this case the similarity matrix would simply represent the weighted matrix. This means each similarity value between two items would act as the edge connecting the two. In the case of using the similarity matrix in the Tikhonov regularizer, it would have to be symmetric.

Constructing the Laplacian Matrix

The Laplacian matrix is defined as:

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A} \tag{8}$$

Where D is the degree matrix and A is the adjacency matrix (similarity matrix). [11].

In order for the Laplacian to be constructed properly, a symmetric adjacency matrix is required. To achieve this, when constructing the neighborhoods, if item j is in i's neighborhood, item i must be in j's. This means that not all neighborhoods are constructed with the top k most similar items, but it does ensure that the adjacency matrix is symmetric.

As the adjacency matrix is symmetric, it is undirected. Meaning the degree matrix is a the diagonal matrix where:

$$d_u = \sum_{v=1}^{N} A_{uv} \tag{9}$$

Where d_u is the u-th element on the diagonal, and represents the degree of node u. The degree of a node is the weights of all connected edges to said node. [11]

The Pipeline

The Tikhonov Recommender takes as input a matrix of user-item ratings, which to get optimum results are then centered around the mean. The ratings are centered based on their respective item means, so for each rating, the item mean is subtracted.[1] The next step is to create a similarity matrix for each item-item pair in the ratings matrix. From this similarity matrix, the symmetric adjacency matrix is constructed as described previously, from which the Laplacian is formed.

Once the Laplacian is found, the actual Tikhonov regularization can occur. All ratings are then predicted using the regularizer. After being left with the predicted ratings, their respective item means are added back in order to counter act the earlier mean centering.

Much like the baseline recommender, in the case an item cannot be given a rating for a specific user, that rating is the assigned the user average rating.

3.4 Sobolev regularizer

A variation of the Tikhonov Regularizer, is the Sobolev regulizer. Much like Tikhonov, the Sobolev regularizer is another graph signal regularization technique that can be used to solve the recommendation problem. The Sobolev norm is defined as:

$$||\boldsymbol{x}||_{\beta,\epsilon} = ||(\boldsymbol{L} + \epsilon \boldsymbol{I})^{\frac{\beta}{2}} \boldsymbol{x}||_2 \tag{10}$$

[7]

It is then possible to find a vector form when the Laplacian is symmetric:

$$||\boldsymbol{x}||^2_{\beta,\epsilon} = \boldsymbol{x}^T (\boldsymbol{L} + \epsilon \boldsymbol{I})^{\beta} \boldsymbol{x} \tag{11}$$

[7]

From the vector form, one can produce the matrix form used in this recommender system:

$$\boldsymbol{X^*} = (\boldsymbol{I} + \mu * (\boldsymbol{L} + \epsilon * \boldsymbol{I})^{\beta})^{-1} \boldsymbol{Y} \tag{12}$$

The construction of this recommender is fairly similar to the Tikhonov, as the Laplacian is built in the same manner, the large difference is that the Sobolev regularizer has three hyper parameters, the size of the neighborhoods, ϵ and β. Just like the Base line recommender and the Tikhonov Recommender, if the Sobolev Recommender is unable to predict a rating for a user-item pair, the algorithm will give the predicted rating the value of the avererage user rating.

4 Training and Results

This section covers the training of the algorithms as well as the final best results for each respective algorithm.

4.1 Training

The results of the training have been split up per metric. Each of the three methods have been trained on 5 random train test splits and are analyzed based on RMSE, Precision, and NDCG.

Base Line CF

First we looked at the standard Collaborative Filtering Method. The Baseline was tested with neighborhood sizes ranging from 5 to 40 with a step size of 5. No other parameters were tested as the amount of neighbours is the sole hyper parameter.

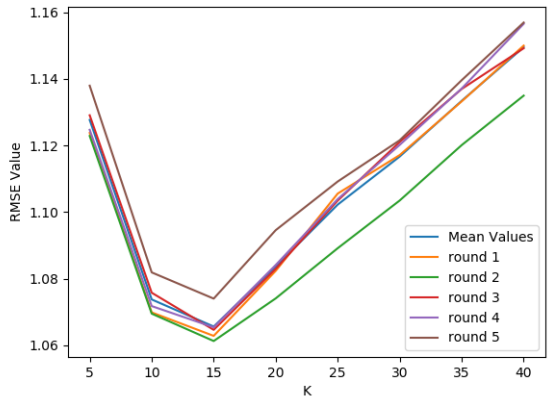

Figure 2: RMSE scores for Collaborative Filtering

Figure 2 displays the RMSE value for all rounds conducted in the experiment, as well as the mean results of the five rounds. When optimizing for RMSE, it is clear that the standard Collaborative filtering performs best at 15 neighbours, with the average RMSE of 1.066.

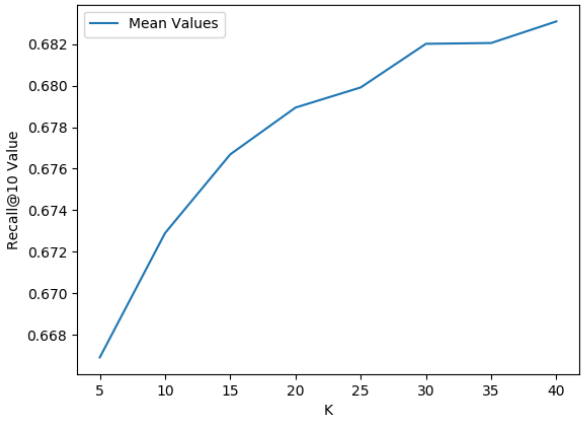

Figure 3: Mean Recall@10 scores for Collaborative Filtering.

The highest mean recall@10 score achieved when training the base line algorithm is 0.683, which is achieved when K is equal to 40. From figure 3 it seems like the mean value is reaching a plateau. This is also evident in the figures for mean precision and mean NDCG.

The precision curve is very similar to that of the recall one, however the values are slightly lower. But much like recall, the best results for Precision@10 are achieved when k is 40. The highest value achieved is 0.579.

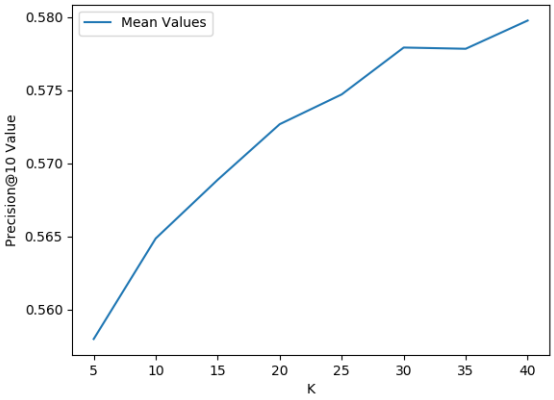

Figure 4: Mean Precision@10 scores for Collaborative Filtering

The NDCG training curve follows the same pattern as the others, with its peak being at k = 40, resulting in a mean NDCG@10 of 0.869.

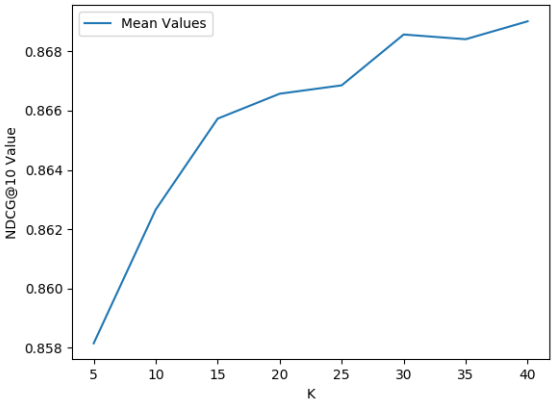

Figure 5: Mean NDCG@10 scores for Collaborative Filtering

It is evident that when solely focusing on RMSE the ideal value for k is 15, however this is not the case when optimizing for the other metrics. One would assume that a low RMSE would result in a more positive score in the other metrics, however the other metrics do not neceserally rely on how accurately a rating is predicted, but lean more towards if the predictions are correctly labeled as positive or negative.

Tikhonov Recommender

The Tikhonov Regularizer was tested with values for μ ranging from 0 to 1 with a step size of .01. Each value for mu was subsequently tested with each different neighborhood sizes (k). The values of k tested ranged from 5 to 40 with a step size of 5.

The Tikhonov recommender seemed to have a smooth RMSE training curve when looking at just one round, however this was not the case when taking the average of five rounds.

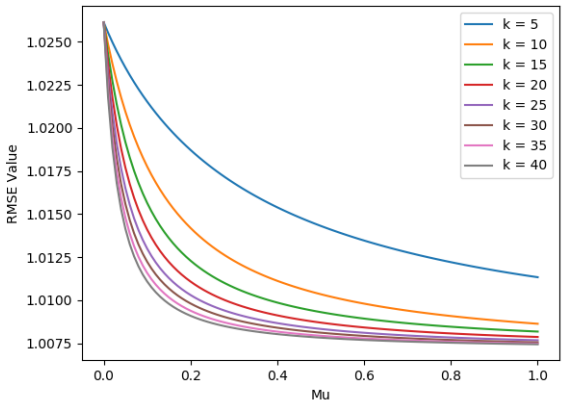

Figure 6: RMSE training curve in round 4 of the experiments

From figure 6, the training curve is smooth and predictable, however this is not the case every round. Tikhonov seems to be somewhat unpredictable when testing combinations between mu and k on different train test splits. This results in the graph of the mean values to look like this:

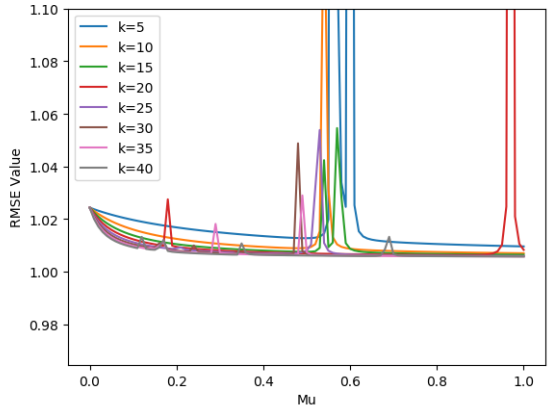

Figure 7: Mean RMSE per k of 5 rounds, with mu ranging from 0 to 1 with step sizes of 0.01

As can be seen, there are a lot of spikes in the data, these however are the result of a bad prediction in a specific round. It is my belief that if one increases the amount of rounds these spikes will be less noticeable. It is however important to remark the inconsistent nature of the training curve.

It is also apparent that the lowest RMSE values achieved are when μ approaches 1. The reason we did not test μ values greater than one is because this curve levels out and the more desirable RMSE score will only differ slightly than what is achieved here. When disregarding the spikes, the largest RMSE value at the start of the curve is 1.0226 and the lowest RMSE value achieved is 1.0043. One could continue the curve, but the gains in RMSE would be minimal.

What is also interesting to point out is that for all values of K, the mean RMSE scores seem to converge, meaning that the initial similarity matrix does not have as large of an impact on the results as μ increases. The amount of neighbours does however have an impact on how fast the RMSE values plateau.

Much like the RMSE training curves, the curve for Recall@10 is also very volatile.

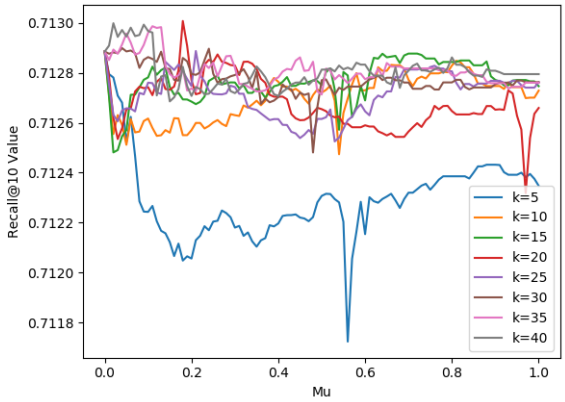

Figure 8: Mean Recall@10 per k of 5 rounds, with mu ranging from 0 to 1 with step sizes of 0.01

Even though the data is very volatile and it is hard to see a pattern, suspect that if more rounds are conducted the learning curve will become more smooth and the results should move towards some upper bound asymptotically, much like can be seen with the RMSE and its lower bound.

The training curve for Precision@10 is also unstable, but relatively more predictable than recall:

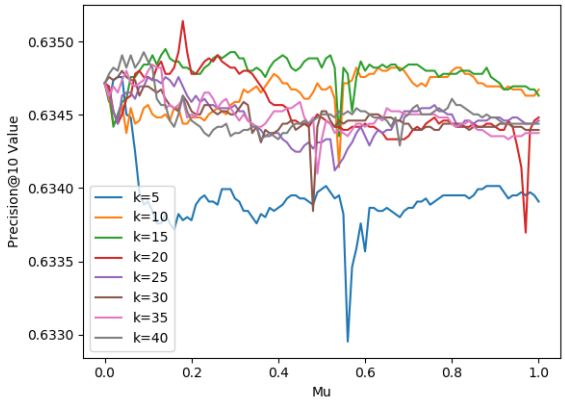

Figure 9: Mean Precision@10 per k of 5 rounds, with mu ranging from 0 to 1 with step sizes of 0.01

Much like recall, the training curve should average out when conducting experiments with more rounds. However currently, the results are too reliant on the random train test splits creating various spikes in the data.

Even though both these plots seem rather unstable, it is not necessarily the fault of the recommender, but also the nature of the precision and recall metric. Unlike RMSE, the punishment and reward for recommendations are higher for precision and recall. Let's say the recommender recommends a list of 10 movies of which only 5 are relevant, it's precision will be 50% but if 6 are relevant the precision jumps from 50% to 60% making the curve far less smooth.

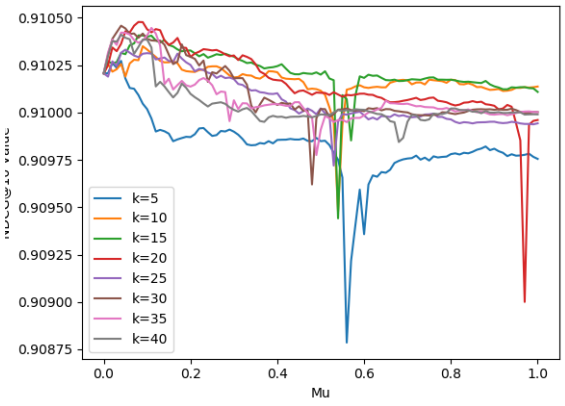

Figure 10: Mean NDCG@10 per k of 5 rounds, with mu ranging from 0 to 1 with step sizes of 0.01

Much like precision and recall, the NDCG training curve contains multiple spikes, the scale in which the NDCG values change however is incredibly small, meaning that changing the value for K or μ does not affect the score by a significant amount.

Sobolev Recommender

Finally, the Sobolev Regularizer was tested on values of μ ranging from 0 to 1 with a step size of 0.1. In addition, Sobolev was tested with the ϵ values of [0, 0.25, 0.5, .75, 1.0] and values β of [1, 1.5, 2]. Just like the baseline and the Tikhonov Regularizer, Sobolev was tested for the neighborhood sizes from 5 to 40 with a step size of 5.

Due to the fact that Sobolev has many hyper parameters, a plot would become very difficult to read, therefore table 1 is an attempt at representing the training curve based on the size of the neighborhood.

	k = 5	k = 10	k = 15	k = 20	k = 25	k = 30	k = 35	k= 40
RMSE	1.010	1.007	1.006	1.006	1.006	1.006	1.006	1.006
Precision@5	.688	.688	.688	.688	.688	.688	.688	.688
Precision@10	.635	.635	.635	.635	.635	.635	.635	.634
Recall@5	.520	.520	.520	.520	.520	. .520	.520	.520
Recall@10	.713	.713	.713	.713	.713	.713	.713	.713
NDCG@5	.889	.889	.889	.889	.889	.889	.889	.889
NDCG@10	.910	.910	.910	.911	.910	.910	.910	.910

Table 1: Optimal Metrics based on Neighbors in Sobolev

As can be seen, the size of the neighborhoods does not seem to have a large affect on the resulting metrics, which only differ past the fourth and fifth decimals. This means that the other hyper parameters create the scenario for which regardless of k, the metrics converge to the same value, much like in the Tikhonov Regularizer.

4.2 Final Results

After running grid search on 5 random 80-20 train test split, the average of each metric was taken.

	RMSE	Precision@5	Precision@10	Recall@5	Recall@10	NDCG@5	NDCG@10
Item-Item CF	1.066	0.600	0.580	0.480	0.683	0.834	0.869
Item-Item TK	1.006	0.688	0.635	0.521	0.713	0.889	0.910
Item-Item Sobolev	1.006	0.688	0.635	0.520	0.713	0.889	0.911

Table 2: Final highest performing metric for each algorithm rounded to the nearest thousanth.

Comparing Tikhonov to CF

When solely comparing the two based on which one scores the lowest RMSE, it is obvious that Tikhonov wins out. 1.005 is less than 1.065, this simple comparison however does not do justice to the nuances of both algorithms. Even though Tikhonov performs better in its extreme cases, from the results gathered it also shows that Tikhonov is fairly unpredictable. Meaning that if one has unfavorable data the resulting predictions will be considerably off the mark. This means that the answer to which algorithm performs more accurately is not straight forward.

When comparing the algorithms in Precision@10 and Recall@10, the results are a little more straightforward. Just like RMSE the values for Tikhonov are very volatile, however the worst scores achieved by Tikhonov are better than the best achieved by the baseline. This means that even though Tikhonov is less predictable, from these experiments it shows that it was more favorable when compared to standard Collaborative Filtering.

Comparing Sobolev

The Sobolev Recommender performs very similarly to Tikhonov. In fact when comparing results, the metrics only differ starting at the 4th decimal place. Take for example Precision@10, for Tikhonov it is 0.6247, Sobolev performs slightly better with a precision of 0.6349. The similarity between the results can be explained by the similarity of the methods themselves. It is also of note to mention that Sobolev can be further optimized for this problem domain, as the grid search performed in this experiment did not cover all hyper parameter possibilities.

5 Discussion

The research conducted aimed to compare the performance of graph regularizers when used for item-item collaborative filtering. To achieve this, we ran a series of experiments. We completed 5 rounds where the data was split up into random train test splits. From these 5 rounds the average metrics were taken for RMSE, Precision, Recall and NDCG.

From the experiments performed in this paper, it is clear that the graph regularizers outperform the standard Collaborative Filtering method in all metrics.

When comparing Item-Item Tikhonov to Item-Item Collaborative Filtering, it is clear that Tikhonov wins out on all metrics tested in this experiment. This trend is not present when comparing User-User Tikhonov to User-User Collaborative Filtering as done by another member of the research team. [12]

	RMSE	Recall@5	Recall@10	Recall@20	Precision@5	Precision@10	Precision@20	NDCG@5	NDCG@10	NDCG@20
CF	1.092	0.521	0.715	0.864	0.640	0.692	0.593	0.914	0.892	0.932
Tikhonov	1.020	0.516	0.711	0.861	0.682	0.636	0.593	0.885	0.908	0.928
Sobolev	1.021	0.517	0.711	0.864	0.684	0.635	0.593	0.884	0.907	0.928

Figure 11: Serenic Monte's results from his experiments

As can be seen in table 11, User-User Tikhonov and Sobolev do not outright outperform User-User Collaborative Filtering. The only metric in which graph regularizers significantly outperform CF is in RMSE. When directly comparing Item-Item Tikhonov to User-User Tikhonov, it is evident that the two perform very similarly. The main difference in the results is the performance of the respective baselines.

In conclusion, this paper has shown that graph regularizers, more specifically the Tikhonov Regularizer and Sobolev Regularizer, are viable options for solving the recommendation problem as they both outperformed the proven Item-Item Collaborative Filtering method. In the future the unpredictable nature of Tikhonov can be researched. Future research can also be done analysing the performance of graph regularizers with more sophisticated techniques for constructing the k nearest neighbors similarity matrix. In addition, in order to improve this research one can run an experiment with more rounds and optimizing the algorithm for time efficiency. The results of running experiments with more rounds will most likely smooth the Tikhonov Training curve and make it simpler to draw conclusions. Finally, this paper lightly touched on the Sobolev Regularizer, a similar concept to Tikhonov, however it allows for more variables to be tuned. It would be interesting to see the performance of this algorithm when the time and resources are invested in more thoroughly training this algorithm.

6 Responsible Research

As the recommendation problem is arguably not in an ethical gray area, the main responsibility as a researcher is to ensure the results achieved in this experiment are reproducible. According to Monya Baker, as written in her article *Is there a Reproducibility Crisis?* the two main factors leading to unreproducble results is pressure to publish and selective reporting. [13]

The first pitfall is hard to avoid as there is a hard deadline on this paper as it is written for the CSE 3000 Research Project course given by the TU Delft. The second major factor however has been taken into account. In an effort to reduce selective reporting all relevant data is included in this paper and all code used to collect said data will be submitted along side this report. This avoids the need to hide any negative results. Negative results also don't affect our reporting in this paper, as there are no outside pressure, financial or otherwise, on us as researchers for this algorithm to perform as a recommender system.

In addition to sharing the code itself, all experiments run are described in detail, such that any one attempting to reproduce the research done here knows what metrics, hyper parameter values and databases were used. The choice in the MovieLens 100k database is also an attempt at making reproducibility easier, as this is a robust and very frequently used database. It is therefore unlikely for this database to lose support in the foreseeable future, enabling anyone whom enquires to use it.

References

[1] C. C. Aggarwal, *Recommender Systems - The Textbook*. Springer, 2016.

[2] F. Isinkaye, Y. Folajimi, and B. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian Informatics Journal*, vol. 16, no. 3, pp. 261–273, 2015.

[3] M. Ferrari Dacrema, P. Cremonesi, and D. Jannach, "Are we really making much progress? a worrying analysis of recent neural recommendation approaches," 09 2019.

[4] H. Steck, L. Baltrunas, E. Elahi, D. Liang, Y. Raimond, and J. Basilico, "Deep learning for recommender systems: A netflix case study," *AI Magazine*, vol. 42, pp. 7–18, Nov. 2021.

[5] H. Steck, "Embarrassingly shallow autoencoders for sparse data," *CoRR*, vol. abs/1905.03375, 2019.

[6] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013.

[7] J. H. Giraldo, A. Mahmood, B. Garcia-Garcia, D. Thanou, and T. Bouwmans, "Reconstruction of time-varying graph signals via sobolev smoothness," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 8, pp. 201–214, 2022.

[8] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 5, December 2015.

[9] A. Tousch, "How robust is movielens? A dataset analysis for recommender systems," *CoRR*, vol. abs/1909.12799, 2019.

[10] A. N. Nikolakopoulos, X. Ning, C. Desrosiers, and G. Karypis, "Trust your neighbors: A comprehensive survey of neighborhood-based methods for recommender systems," 09 2021.

[11] E. Isufi, B. Das, A. Natali, M. Sabbaqi, and M. Yang, "Graph filters for processing and learning from network data," tech. rep., Delft University of Technology, 2021.

[12] S. Monté, "Tikhonov and sobolev regularisers compared to user-based knn collaborative filtering." unpublished.

[13] M. Baker, "Is there a reproducibility crisis?," *Nature*, vol. 533, pp. 452–454, 05 2016.