Scaling SVD recommender systems

Team members:

Nikita Borovkov

Filipp Furaev

Introduction

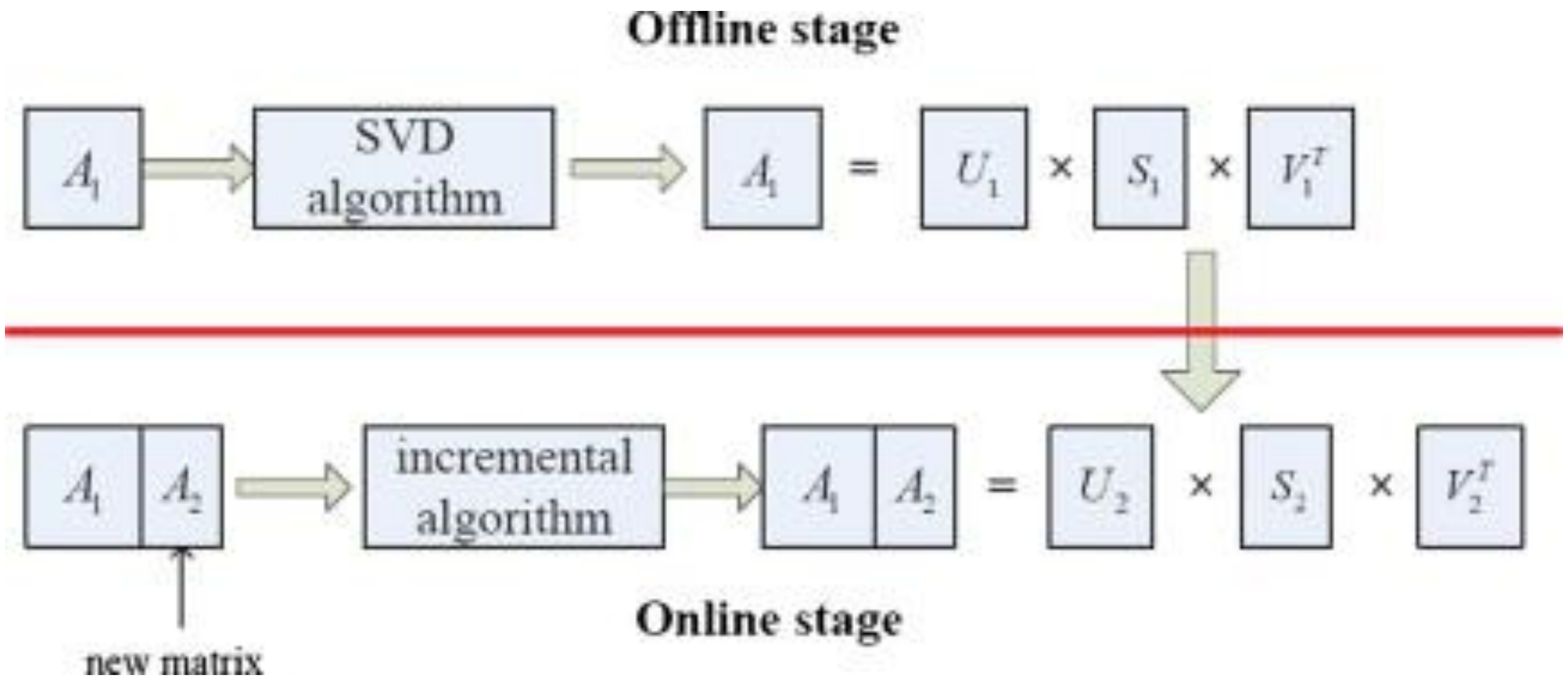
- Variety of products (e. g. films)
- People use and rate them, producing information
- This information can be used to make personalized recommendations
  - Make choice easier
  - Lets companies increase their income
- A widely used approach to recommender systems is using Singular Value Decomposition (SVD)

Problem statement

- Amounts of information are rapidly increasing
- Many high quality recommendations per second are required
- Usage of SVD-based models requires high computing powers:
  - Computing SVD of user-product matrix (m x n), where m is the number of users and n is the number of products and each element of the matrix is user rating
  - SVD itself has $O(n^3)$ complexity, where n is the size of the matrix
- Recommendation requires precomputed SVD
- What happens if a new film (or user) is added to the matrix?
- Recomputation of SVD would take too much time to make new recommendations

Solution

- No recomputing SVD every time
- Instead of rebuilding the model entirely we can iteratively build SVD, thus scaling the model
- Reviewed approaches:
  - Folding-in SVD
  - Incremental SVD



X. Zhou, J. He: SVD-based incremental approaches for recommender systems

Prediction generation using SVD

- User-product matrix is reduced into three SVD component matrices with k features U_k, S_k , V_k
- Computing cosine similarities of between m pseudo-customers $U_k\sqrt{S_k}$ and n pseudo-products $\sqrt{S_k}V_k^T$
- Prediction for the i-th customer and for the j-th product:

$$P_{i,j} = \overline{r_j} + \left(U_k\sqrt{S_k}\right)_{[i,:]}\left(\sqrt{S_k}V_k^T\right)_{[:,j]}$$

where $\overline{r_j}$ is average rating for item j

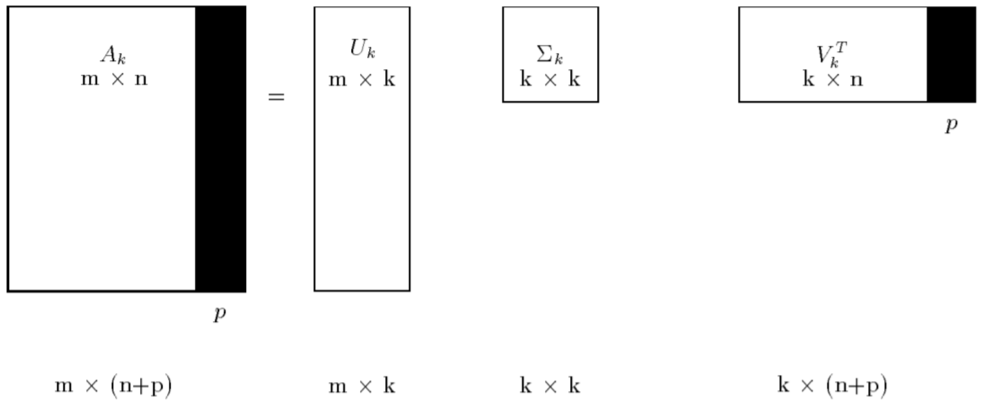
- Once SVD is done prediction requires $O(1)$ time, since k is a constant

Folding-in SVD algorithm

- Updating films:

$$new\ V_k^T\ columns = \Sigma_k^{-1} U^T c,$$

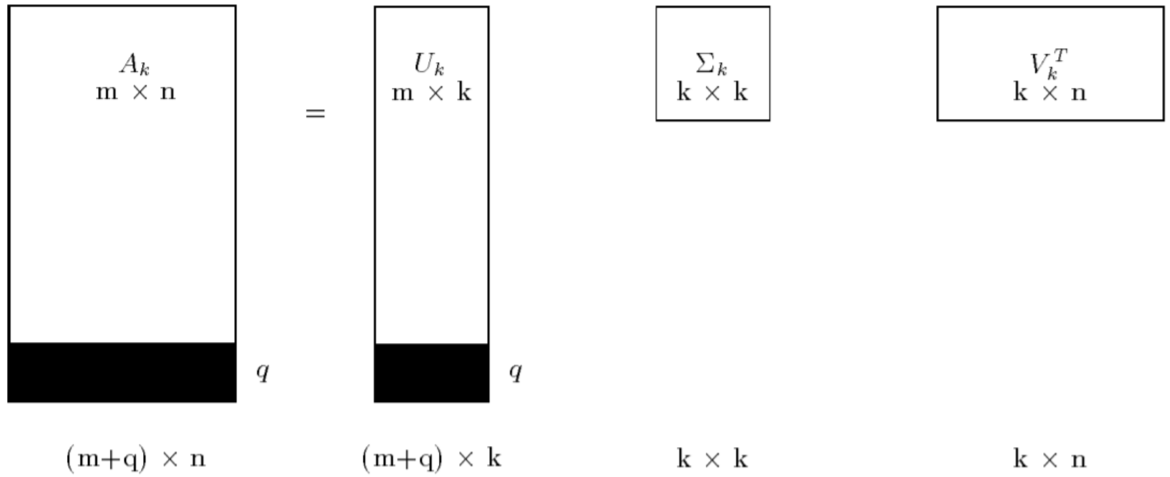
$$Where\ c - new\ films\ columns$$



- Updating users:

$$new\ U_k\ rows = r V_k^T \Sigma_k^{-1}$$

$$Where\ r - new\ users\ rows$$



M.W. Bery. S.T. Dumais. Using Linear Algebra for Intelligent Information Retrieval

Incremental SVD algorithm: Updating films

Let $B = (\, A_k \mid D)$, SVD (B) = $U_B \Sigma_B V_B^T$, then

$$U_k^T B \begin{pmatrix} V_k & O \\ O & I_p \end{pmatrix} = (\Sigma_K \mid U_k^T D\,) = \mathrm{F},$$

where p = number of new films; size of F is k x (k + p).

If $(\Sigma_K \mid U_k^T D\,) = \mathrm{F}$, SVD (F) = $U_F \Sigma_F V_F^T$, then it follows that

$$U_B = U_k U_F, V_B = \begin{pmatrix} V_k & O \\ O & I_p \end{pmatrix} V_F, and\ \Sigma_B = \Sigma_F$$

B. Sarwar, G. Karypis: Incremental Singular Value Decomposition Algorithms for Highly Scalable Recommender Systems

Incremental SVD algorithm: Updating users

Let $C = \left(\frac{A_k}{T}\right)$, SVD (C) = $U_C \Sigma_C V_C^T$, then

$$\begin{pmatrix} U_k^T & O \\ O & I_q \end{pmatrix} C V_k = \left(\frac{\Sigma_K}{T V_k}\right) = \mathrm{H},$$

where q = number of new users; size of H is (k + q) x k.

If $\left(\frac{\Sigma_K}{T V_k}\right) = \mathrm{H}$, SVD (H) = $U_H \Sigma_H V_H^T$, then it follows that

$$U_C = \begin{pmatrix} U_k & O \\ O & I_q \end{pmatrix} U_H, V_C = V_k V_H, and\ \Sigma_C = \Sigma_H$$

B. Sarwar, G. Karypis, Incremental Singular Value Decomposition Algorithms for Highly Scalable Recommender Systems

Experiments

- MovieLens 100K Dataset:
  - 100000 ratings
  - 1000 users
  - 1700 movies
- Only the users which rated at least 20 movies are included in the data
- We compute SVD for different number of films in the initial system, scale it using the mentioned algorithms and measure the quality of prediction on the test set (20% of the initial dataset ratings)
- Mean absolute error (MAE):

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N}$$

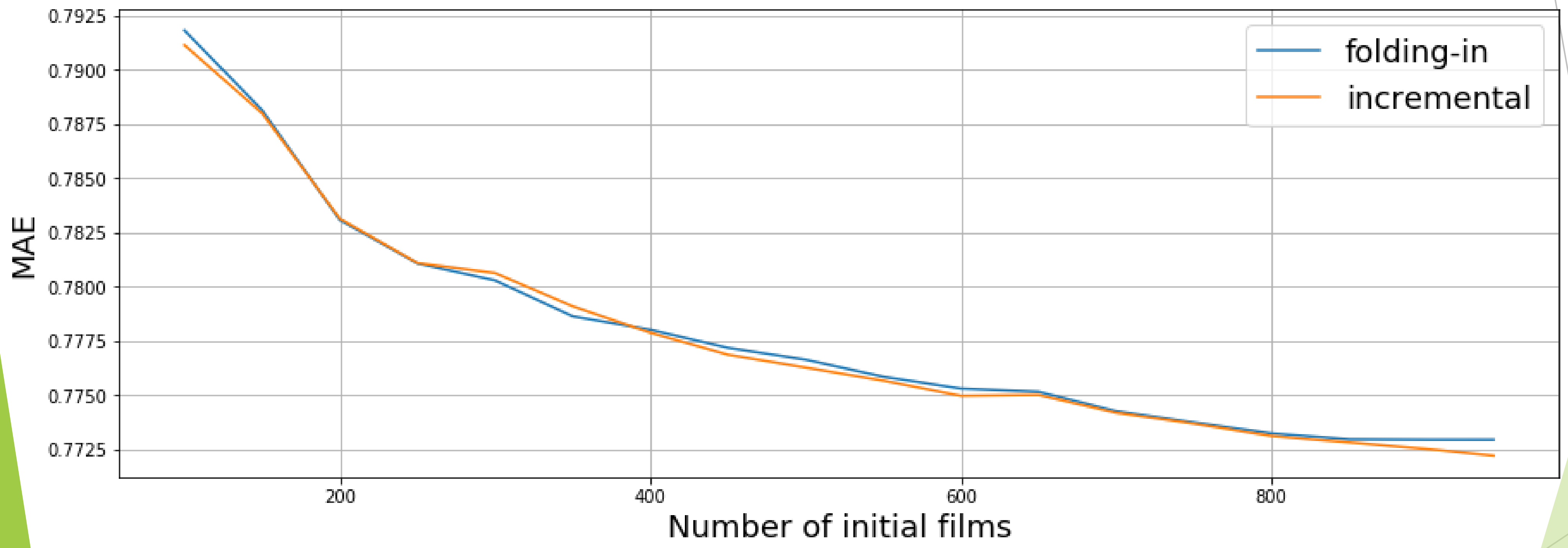
where p_i is true rating, q_i is predicted rating

- Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (p_i - q_i)^2}{N}}$$

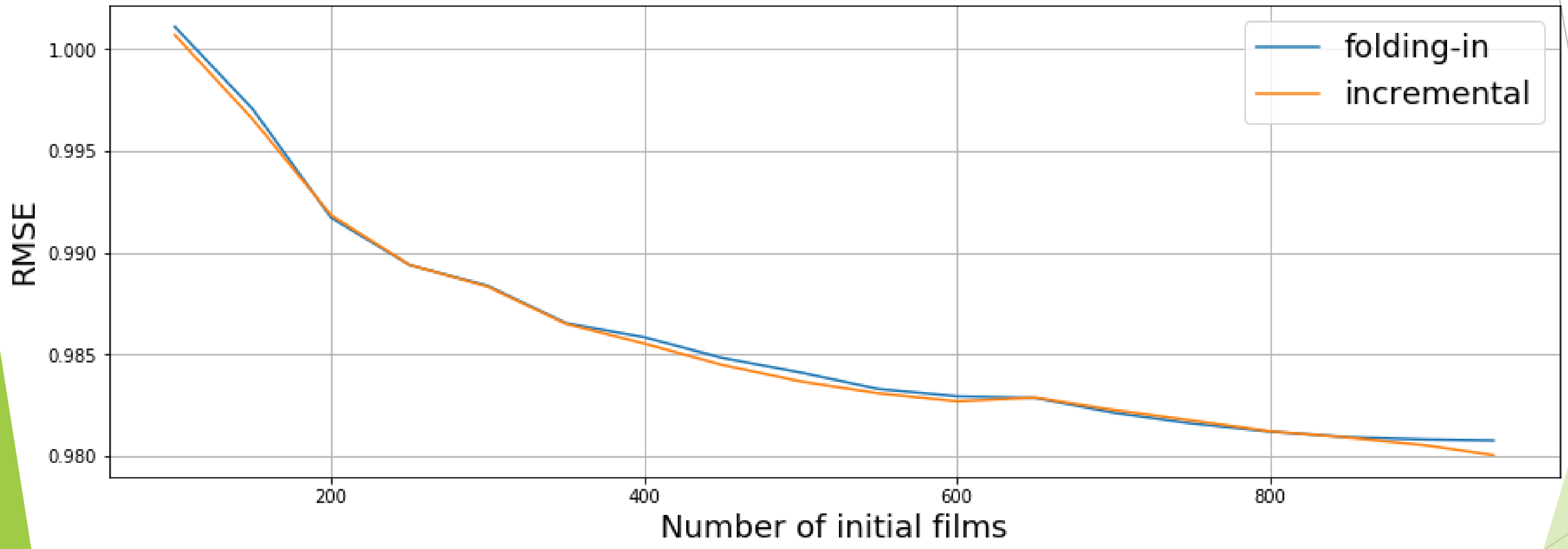
- Also, the time is measured

Results: MAE

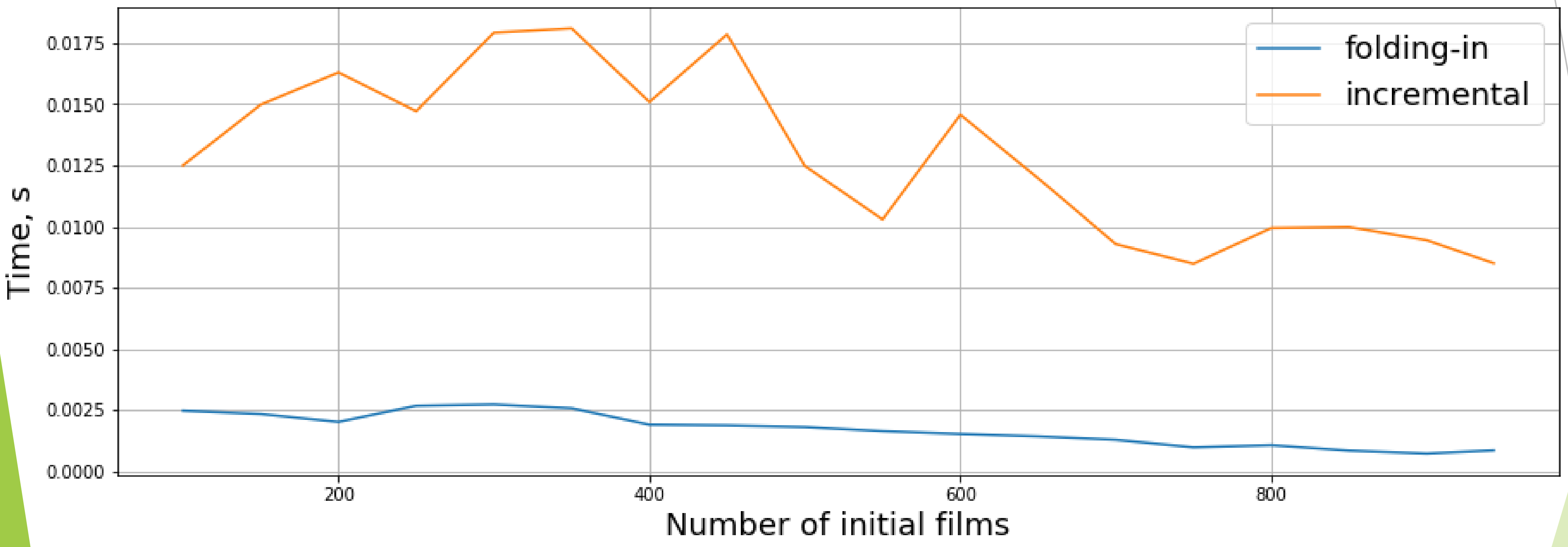


0.7925
0.7900
0.7875
0.7850
0.7825
0.7800
0.7775
0.7750
0.7725
MAE
200
400
600
800
Number of initial films
folding-in
incremental

Results: RMSE



Results: time



Conclusion

- Tested three different methods for scaling SVD recommended systems: naive algorithm for updating SVD, incremental SVD algorithm, folding-in algorithm and made different tests to compare them (RMSE, MAE, time measuring)
- In comparison to computing SVD every time a new film appears, quality of prediction using scaling methods is worse, but the difference is not significant
- The quality of prediction using incremental SVD and folding-in SVD almost the same
- Folding-in SVD algorithm requires less time to compute than incremental SVD, therefore, it is more suitable for online stage of a recommender system

Thanks for attention

Team contribution

- Nikita Borovkov
  - Pipeline implementation
  - Preparing presentation
- Filipp Furaev
  - Folding-in and incremental algorithms implementation
  - Preparing presentation

References

- M.W. Bery. S.T. Dumais. Using Linear Algebra for Intelligent Information Retrieval: https://pdfs.semanticscholar.org/0265/769b0fbf86bb0e700573c80e388bb54c3f7a.pdf
- B. Sarwar, G. Karypis. Incremental Singular Value Decomposition Algorithms for Highly Scalable Recommender Systems: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.3.7894&rep=rep1&type=pdf
- X. Zhou, J. He: SVD-based incremental approaches for recommender systems: https://www.sciencedirect.com/science/article/pii/S0022000014001706?via%3Dihub