Comparison of Evaluation Protocols for Sequential Recommender Systems

MS. UMBERTO DI FABRIZIO, Politecnico di Milano
Tutor: Ph.D MASSIMO QUADRANA, Politecnico di Milano
Prof. PAOLO CREMONESI, Politecnico di Milano

Recommender Systems are used to predict the user appreciation for some future items. One trend is to exploit the recent history of a user actions in order to enhance the performances of the recommender. Such sequential recommender systems (SRS) can be evaluated using different protocols, namely set or sequential. To this day, there has been no evaluation of the result obtained using one protocol over the other. In this work the protocols are compared with respect to the ranking among six classes of SRS using both precision and recall metrics, moreover the sensitivity of each protocol in comparing the recommenders is discussed. The results show that, even though there is almost no difference in the results produced by the two protocols, the set one may be the favorable choice given his highest sensitivity and faster run time.

Additional Key Words and Phrases: Sequential Recommendation, Evaluation protocols

1. INTRODUCTION

Recommender Systems are used to predict the user appreciation for some future items. Traditionally RSs attempt to make such predictions based on item-similarity or user-similarity as well as hybrid techniques, yet most often the behavior of a user is strictly related to the last actions performed e.g. last songs listened, last items purchased, last apps opened. For this reason it seems reasonable to exploit the user profile/context i.e. the last actions performed by an user, in order to make more reliable and accurate predictions. In the literature there are several systems that try to exploit the recent history of an user's actions, namely Sequential Recommender Systems (SRS).
There are several approaches to build a SRS (which will be discussed in the next section) as well as several protocols to assess the performances of such items and it is of undeniable importance the assessment of different SRS accordingly to the same protocol.
The objective of this work is to implement several classes of sequential recommender systems in order to compare their performance adopting two evaluation protocols: set and sequential. In the next sections the extent to which the two protocols may produce different results when assessing multiple recommenders is analyzed.

2. CLASSES OF SEQUENTIAL RECOMMENDER SYSTEMS

The classes of SRS that are used in this work are taken from the state-of-the-art SRS:

(1) **Popularity based** recommender is non-personalized and doesn't exploit the recent history of a user, yet it has been included for comparison purposes. The popularity recommender always predicts the top N most popular items.
(2) **Frequent Pattern Mining**[Mobasher et al. 2002] recommenders incorporate history in the system by leveraging on the ability to mine frequent sequences in the dataset and in so doing they can learn which sequences of items are more frequent, thus they can recommend the most probable next-item. The algorithm implemented adapts the SPMF library[Fournier-Viger et al. 2016] and a prefix tree structure to keep sequences in memory. Once a pattern in the prefix tree is matched the children of the last node in the matched path are recommended as next items.
(3) **Markov Chain** recommenders [Shani et al. 2005] build a state graph where each state is a sequence of length k and k is the order of the markov model. In the simplest approach there is an edge between two states if the dataset has at least an

occurrence of length k+1 made by concatenating the sequence in the first state with the last item of the next state. This definition is augment as proposed in [Shani et al. 2005] by applying skipping, clustering and model mixing. Although a markov model represents a clear method to exploit the sequence's history, it is computationally expensive in terms of both memory and running time, for this reason it has been constrained to a first order model.

(4) **Prod2Vec** recommenders[Grbovic et al. 2015] are based on the theory of n-grams developed by the NLP community which takes into account the context in which a word is found and creates a distributed representation of that word. Word2Vec models have been used virtually in all types of applications[Nzw0301 2017], in the case of recommender systems items (=words) are projected into a vector space by taking into account their context (=the k items before and after the one considered in the sequence). Similarity among items is calculated by applying the cosine similarity among their distributed representations. In this case the gensim[Rehurek and Sojka 2010] library has been adapted to produce recommendations.

(5) **Supervised** recommenders try to model the historical data in such a way that classical atemporal machine learning algorithms can be used to make recommendations. In this work the *data expansion* approach suggested in [Zimdars et al. 2001] has been used, together with a decision tree in order to obtain the final SRS.

(6) **Factorizing Personalized Markov Chains** represent a hybrid approach exploiting both the power of Markov Chains and Factorization models which have been extensively and successfully used in the literature of recommender system. This class of SRS has been built based on the paper[Rendle et al. 2010] and by adapting the code at [Khesui 2017].

3. EXPERIMENT SET UP

The dataset is composed of 10k unique items, 380k sessions and 28k users, collected from the LastFM website. In order to make this work feasible the number of unique items among the sequences has been reduced to 500, which has been chosen since the dataset size is manageable but the recommendation task is not trivial. For each of the recommenders described in the previous section the two protocols of evaluation are executed for $k \in [1, 5, 10]$ where k is the number of items in the user profile accordingly to the first-k approach: for each sequence only the first k elements are used to recommend the future elements. In the case of set evaluation the prediction list is compared with the ground-truth whilst for the sequence evaluation the first k element are used to make the first recommendation which is evaluated only w.r.t. the first element of the ground-truth,then the user profile length is increased by one and the items are recommended again, this process goes on until the ground-truth has length 0. In this way it is also useful to understand for which values of k the recommendations are more accurate. Table I summarizes the statistics for the datasets adopted for each value of k. For each recommender precision and recall are evaluated on a test set obtained by a random holdout split procedure 80/20, and the metrics are averaged on 5 runs.
The parameters used to run each recommender have been set by empirical evaluation, yet it is worth to notice that the scale of the metrics didn't vary significantly, appendix A provides the parameters choice.

4. RESULTS

In this section the result of the comparison of the two protocols are presented and discussed.
For each k (i.e. length of user profile) precision and recall are the metrics adopted to evaluate the performances of the recommenders. Figures 1 2 3 show the results for the precision metric for the considered values of k. In the graphs the x-axis represents

Table I: Dataset statistics

k	Train Size	Test Size	Average sequence length
1	102193	25549	4.65
5	23973	5994	10.30
10	6947	1737	17.34

the result for the set evaluation while the y-axis the result of the sequential evaluation. The decision to adopt such a visualization technique is twofold: firstly the more a classifier is far from the diagonal the more its performances vary between the two protocols, secondly it is quick to realize the effect of the different evaluation protocols infact any recommender that appears in the bottom right or the upper left has very different metric scores between the two protocols. By analyzing the figures it is evident that the set evaluation usually leads to higher scores for precision yet recall has the same magnitude for the two protocols, more importantly if the recommenders can be aligned in a monotonic increasing order it means that the two protocols provide the same ranking when ordering the recommenders.

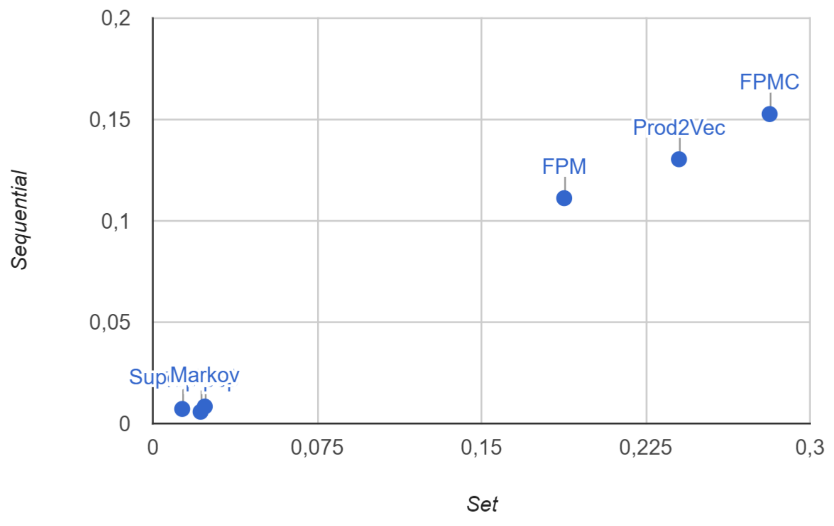

Fig. 1: Precision scores for the set evaluation and the sequential evaluation, with $k = 1$.

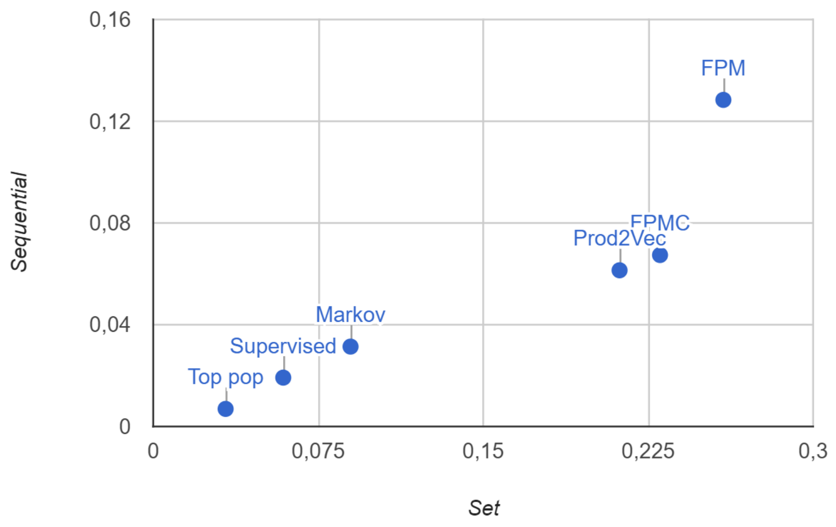

Fig. 2: Precision scores for the set evaluation and the sequential evaluation, with $k = 5$.

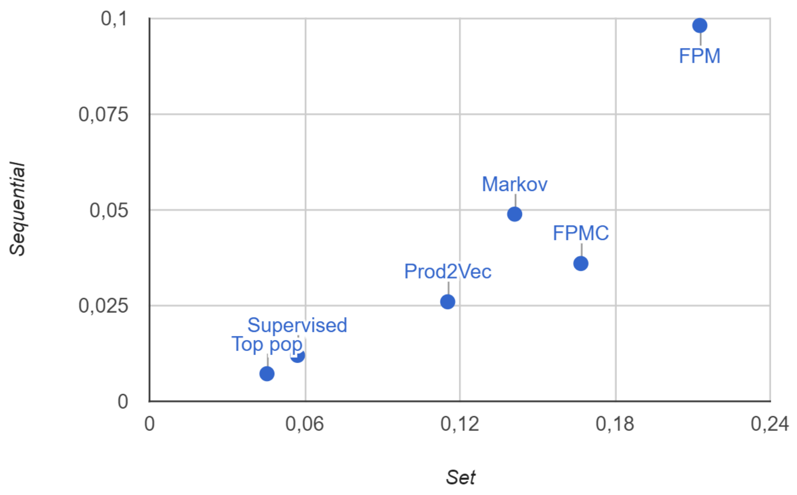

Fig. 3: Precision scores for the set evaluation and the sequential evaluation, with $k = 10$.

Since the interest is to verify the correct order of the ranking of the recommenders for both the evaluation protocols, the following measure has been adopted in order to evaluate the difference of the rankings produced by the evaluation protocols :

$$r = \frac{\frac{1}{L}\sum_i(\frac{1}{d_i+1}) - \frac{1}{L}}{1 - \frac{1}{L}}$$

where L is the number of recommenders and d_i is the distance in terms of positions in the rankings between the two evaluation protocols, $r = 1$ when there is perfect agreement in the rankings, 0 when there's total disagreement. It is possible to see that for all the scores of precision 1 2 3 and recall 4 5 6 $r = 1$ except for fig 3 where $r = 0.8$ because the Markov model is better than FPMC for the sequential evaluation but worst for the set evaluation.
Anyway comparing the rankings is not the only interest, infact it is also worth to compare the sensitivity of the two protocols, in other words one evaluation protocol may differentiate between recommenders much more that the other, for instance in fig 2 the difference of precision among FPM and FPMC is 0.02 for the set evaluation while 0.06 for the sequential protocol.
It is important to evaluate the sensitivity of the two protocol because even though they may give the same ranking, one protocol may produce a much larger gap between two models than the other and this may cause misleading conclusions.
It is possible to capture the sensitivity given two recommenders i, j, by computing the absolute difference in the metric between i, j using one protocol and subtracting the absolute difference of the metric between i, j using the other protocol. In so doing if the metric score of the set evaluation among two recommenders differs of 0.02 and the same metric using the sequence evaluation differs of 0.06 it is possible to say that the sequential evaluation is more sensitive. This idea can be extended by adding up the sensitivity value among two recommenders for each pair of recommenders in the evaluation, leading to the formula:

$$sensitivity = \frac{\sum_{i,j} |set_i - set_j| - |sequential_i - sequential_j|}{2}$$

where set_x is the metric score for recommender x calculated using the set protocol whilst $sequential_x$ is the score calculated with the sequential protocol. Notice that the sensitivity value does not say anything about the ranking among the recommenders but addresses the magnitude of difference among them w.r.t. the metric used. Using together the r value and $sensitivity$ score it is possible to say a) whether the two evaluation protocols produce the same ranking among the recommenders b) the scale of difference in the metric score among them.

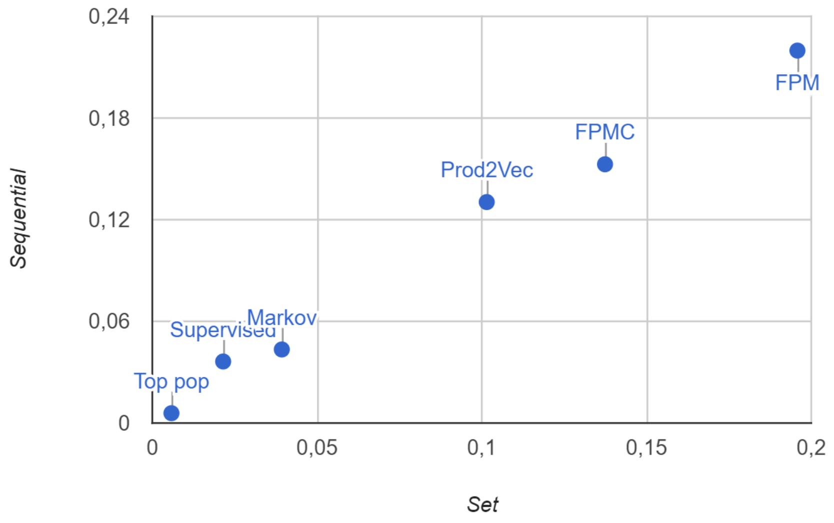

Fig. 4: Recall scores for the set evaluation and the sequential evaluation, with $k = 1$.

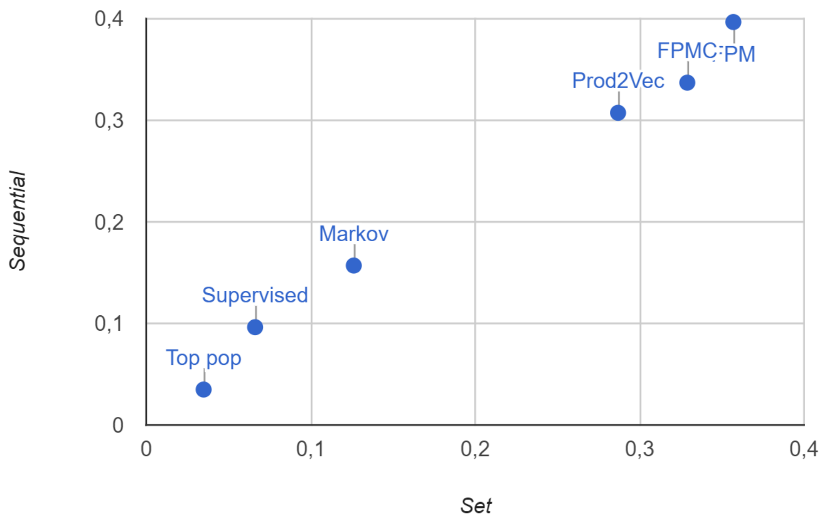

Fig. 5: Recall scores for the set evaluation and the sequential evaluation, with $k = 5$.

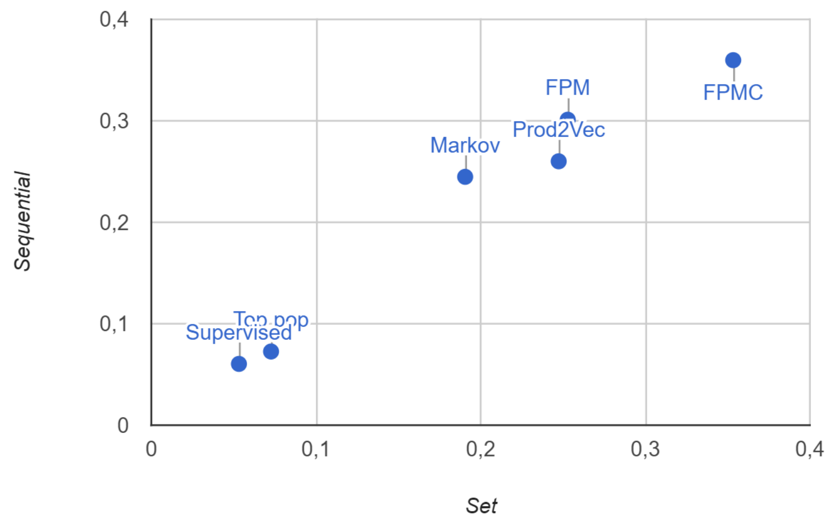

Fig. 6: Recall scores for the set evaluation and the sequential evaluation, with $k = 10$.

The sensitivity for the precision score for each k is $(0.9534, 0.9856, 0.618)$ meaning that the set evaluation usually gives much more difference in the evaluation of the recommenders than the sequential one. With respect to the recall instead the sensitivity for each k is $(-0.1463, -0.1227, -0.0994)$ meaning that the magnitude of the scores is almost the same yet slightingly stronger for the sequential evaluation.

5. SCALING UP

The analysis done for 500 unique items in the dataset has been extended to 5000 unique items, in so doing the validity of the work is extended to real-world scenarios. The main issue with such a large dataset (statistics provided in Table II) is that not all the models presented scale to such a large dataset (with the available CPU and memory).

Table II: Dataset statistics for 5000 unique items

k	Train Size	Test Size	Average sequence length
1	272292	68074	7.28
5	127816	31955	11.73
10	49537	12385	18.30

6. CONCLUSION

In this work two evaluation protocols (i.e. set and sequential) for SRS are compared with respect to the ranking produced among the recommenders using both precision and recall. Six classes of state-of-the-art SRS are implemented in order to perform such evaluation. The analysis takes into account several values of k (i.e. length of user profile) and assesses the difference among the two evaluation protocols in terms of r value and sensitivity. By analyzing the r value it is possible to conclude that the rankings produced by the two protocols are the same except for one case where the r value is a little lower than 1. The analysis using the sensitivity value highlights the fact that the precision metric calculated using the set protocol usually gives much bigger gaps between the SRS whilst for the recall is close to 0. The conclusion that can be drawn from this work is that when one has to choose between the two protocols, and the use of the recommender (i.e. application scope) does not enforce one choice over the other, the set evaluation is advised because it gives a clearer picture of the difference among recommenders performance, moreover it is much faster to evaluate than the sequential one.

7. APPENDIX

7.1.

The code repo is available at :https://umbertoDifa@bitbucket.org/umbertoDifa/seq_rec_evaluation.git

7.2.

— **Popularity**
Top n = 5. Number of items recommended.

— **Frequent Pattern Mining**:
Min context = 1. The minimum context used while applying the all-kth algorithm[Nakagawa and Mobasher 2003].
Max context = 10. The maximum context for the all-kth algorithm.
Min support = 0.002. Minimum support for frequent sequence mining.
Min confidence = 0.1. Minimum confidence to recommend an item.
Top n = 5. Number of items recommended.

— **Markov Chain**
From k = 1. Minimum order or the model.
To k = 11. Maximum order of the model.
Top n = 5. Number of items recommended.

— **Prod2Vec**
Min count = 10. Minimum occurrences of an item, in order to be considered.
Size = 300. The dimensionality of the feature vectors..
Window = 5. The maximum distance between the current and predicted item within a sequence.
Top n = 5. Number of items recommended.

— **Supervised**
History length = 1. Length of history to consider acconrding to the data expansion model.[Zimdars et al. 2001]
Top n = 5. Number of items recommended.

— **Factorizing Personalized Markov Chains**
Latent factors = 92. Number of latent factors.
Learning rate = 0.01. Learning rate
Regularization = 0.001. Regularization term
Epochs = 40. Number of epochs
Negative samples = 10. Number of negative samples
Top n = 5. Number of items recommended.

References

Philippe Fournier-Viger, Jerry Chun-Wei Lin, Antonio Gomariz, Ted Gueniche, Azadeh Soltani, Zhihong Deng, and Hoang Thanh Lam. 2016. The SPMF Open-Source Data Mining Library Version 2. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 36–40.

Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. 2015. E-commerce in Your Inbox: Product Recommendations at Scale. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15)*. ACM, New York, NY, USA, 1809–1818. DOI:http://dx.doi.org/10.1145/2783258.2788627

Khesui. last access March 2017. https://github.com/khesui/FPMC

B. Mobasher, Honghua Dai, Tao Luo, and M. Nakagawa. 2002. Using sequential and non-sequential patterns in predictive Web usage mining tasks. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*. 669–672. DOI:http://dx.doi.org/10.1109/ICDM.2002.1184025

Miki Nakagawa and Bamshad Mobasher. 2003. Impact of site characteristics on recommendation models based on association rules and sequential patterns. In *Proceedings of the IJCAI*, Vol. 3.

Nzw0301. last access March 2017. https://gist.github.com/nzw0301/333afc00bd508501268fa7bf40cafe4e

Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer.

Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing Personalized Markov Chains for Next-basket Recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW '10)*. ACM, New York, NY, USA, 811–820. DOI:http://dx.doi.org/10.1145/1772690.1772773

Guy Shani, David Heckerman, and Ronen I. Brafman. 2005. An MDP-Based Recommender System. *J. Mach. Learn. Res.* 6 (Dec. 2005), 1265–1295. http://dl.acm.org/citation.cfm?id=1046920.1088715

Andrew Zimdars, David Maxwell Chickering, and Christopher Meek. 2001. Using Temporal Data for Making Recommendations. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI '01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 580–588. http://dl.acm.org/citation.cfm?id=647235.720264