Sequence and Time Aware Neighborhood for Session-based Recommendations

Diksha Garg, Priyanka Gupta, Pankaj Malhotra, Lovekesh Vig, Gautam Shroff
diksha.7@tcs.com,priyanka.g35@tcs.com,malhotra.pankaj@tcs.com,lovekesh.vig@tcs.com,gautam.shroff@tcs.com
TCS Research
New Delhi, India

ABSTRACT

Recent advances in sequence-aware approaches for session-based recommendation, such as those based on recurrent neural networks, highlight the importance of leveraging sequential information from a session while making recommendations. Further, a session-based k-nearest-neighbors approach (SKNN) has proven to be a strong baseline for session-based recommendations. However, SKNN does not take into account the readily available sequential and temporal information from sessions. In this work, we propose *Sequence and Time Aware Neighborhood* (STAN), with vanilla SKNN as its special case. STAN takes into account the following factors for making recommendations: i) position of an item in the current session, ii) recency of a past session w.r.t. to the current session, and iii) position of a recommendable item in a neighboring session. The importance of above factors for a specific application can be adjusted via controllable decay factors. Despite being simple, intuitive and easy to implement, empirical evaluation on three real-world datasets shows that STAN significantly improves over SKNN, and is even comparable to the recently proposed state-of-the-art deep learning approaches. Our results suggest that STAN can be considered as a strong baseline for evaluating session-based recommendation algorithms in future.

KEYWORDS

Session-based Recommendation;Sequence and Time Aware Recommendations;Nearest Neighbors

ACM Reference Format:
Diksha Garg, Priyanka Gupta, Pankaj Malhotra, Lovekesh Vig, Gautam Shroff. 2019. Sequence and Time Aware Neighborhood for Session-based Recommendations. In *42nd Int'l ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR'19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3331184.3331322

1 INTRODUCTION

Recommender systems aim to rank or score relevant items of interest for a user, and are utilized in a variety of areas including e-commerce, music streaming applications, and so on. However, in many real-world (web) applications, demographic profiles of users may not be available and the recommendations are based on the user interactions with the system within a session without any additional information about the user. In recent years, several algorithms for session-based recommendation (SR) have been proposed (e.g. [5–7, 10, 14]). The goal of SR algorithms is to recommend items that a user is likely to click on next by using the sequence of user actions (e.g. item/product clicks) so far. SR algorithms using deep neural networks architectures such as those based on recurrent neural networks (RNNs) (e.g. [5, 6]), graph neural networks (e.g. [14]), attention networks (e.g. [10]), etc. have yielded state-of-the-art algorithms for SR. It has been shown that instead of relying only on similar items, better recommendations can be made by modeling the whole session via sequential models like RNNs [6]. Further, nearness of sessions in time (recency) has been shown to be useful while determining similar sessions to current session [7]. In a related scenario of collaborative filtering, the importance of incorporating time and gradual forgetting to adapt to evolving user interests has been well-studied (e.g. [2]) under Time-Aware Recommender Systems (TARS) [1].

Simple heuristics-based SR approaches using k-nearest-neighbors [4] such as Session-KNN (SKNN) [7, 11] have been shown to be very effective as well and provide a strong baseline. However, unlike most recent deep learning approaches, SKNN does not incorporate sequential information from a session for making recommendations. In this work, we propose Sequence and Time Aware Neighborhood (STAN) approach for SR that extends and generalizes SKNN to incorporate sequential and temporal information by introducing simple decay factors at three different levels: i) gradual item-forgetting within current session, ii) gradual item-forgetting in neighboring past sessions based on the context of current session, and iii) gradual session-forgetting based on time gap w.r.t. current session. Empirical evaluation on three real-world datasets shows that STAN significantly improves over SKNN while being comparable to state-of-the-art deep learning approaches. Our results suggest that STAN can be considered as a strong baseline for evaluating novel SR algorithms in future.

2 RELATED WORK

Recently, nearest neighbor approaches have been considered for SR: Item-KNN (as used in [6]) considers only the last item in a current session and recommends most similar items in terms of their co-occurrence in other sessions. SKNN approach and its variants [7, 11] compare the entire current session with the past sessions in the training data to determine the items to be recommended. Contextual KNN [3] shows an improvement over SKNN by using contextual information to identify importance of different items in current session (e.g. most recent clicked item is most relevant) and

SIGIR '19, July 21–25, 2019, Paris, France

ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331322

diffusion-based similarity to find relevant similar sessions which are related to last click of current session. In the related task of collaborative filtering (CF), item-based CF approaches, e.g. [9], find nearest neighbor of items but do not take position of items in a session or sequence information into account. Our approach can be considered as a generalization of nearest neighbor approaches that simultaneously takes into account multiple factors such as recency of item in current session, session-similarity in terms of number of common items, and recency of a past session w.r.t. the current session to find neighboring sessions and subsequent item relevance, while allowing adaptation to the application domain by using controllable weight factors.

Time-awareness in kNN-based approaches has been studied in the past under Time-Aware Recommender Systems (TARS) [1] in context of CF, and have been found to improve over vanilla item-based kNN methods. Considering drifting user interests and temporal recency has been found to be useful in CF, e.g. in [2, 13]. Using tunable exponential decay on old ratings to emphasize more on recent ratings and enable gradual forgetting is known to be useful [2, 13]. While most of these approaches have been studied in context of CF, we show how gradual forgetting can improve neighborhood-based approaches for SR by simultaneously incorporating it at three stages: i) gradual item-forgetting within current session, ii) gradual item-forgetting in neighboring past sessions based on the context of current session, iii) gradual session-forgetting based on time gap w.r.t. current session.

Incorporating sequential information in SR algorithms is important as user preferences for items can drift and evolve over time within a session. Recent deep learning approaches for SR also consider dealing with time and sequence information explicitly by using sequential neural network architectures, e.g. GRU4Rec [5, 6], STAMP [10], and time aware models, e.g. TA4Rec [12]. Similarly, NARM [8] and SR-GNN [14] model session as well as item representations while considering sequential information in session. While deep neural network architectures attempt to capture sequence and time information in the network weights, our approach proposes useful heuristics to incorporate such information in kNN based approaches. The efficacy of our approach can help develop new deep neural network architectures in future.

3 PRELIMINARY: SESSION-KNN

Let $\mathcal{S}$ denote all past sessions, and s denote the current session for which recommendations are to be made. Let $\mathcal{I} = \{i_1, i_2, \ldots, i_m\}$ denote the set of all the m items observed in the set $\mathcal{S}$. Any session $s_j \in \mathcal{S}$ is a chronologically ordered tuple of $l(s_j) \in \mathbb{N}$ item-click events: $s_j = (i_{j,1}, i_{j,2}, \ldots, i_{j,l(s_j)})$, where $i_{j,k} \in \mathcal{I}$ and $l(s_j)$ denotes the length of the session. Further, let $t(s_j)$ denote the timestamp of the most recent item $i_{j,l(s_j)}$ in s_j. Given s, the goal of SR is to determine a relevance score for any item $i \in \mathcal{I}$ such that an item with higher score is more likely to be clicked next.

SKNN first determines the neighborhood set $\mathcal{N}(s) \subseteq \mathcal{S}$ of N most similar past sessions by computing the cosine similarity between s and all sessions $s_j \in \mathcal{S}$, where sessions s_j and s are represented as binary vectors $\vec{s}_j$ and $\vec{s}$ in the m-dimensional space of items (value of 1 for dimensions corresponding to the items in the session, 0 otherwise) [7]. Cosine similarity between s_j and s is given by:

$$sim(s, s_j) = \frac{\vec{s}.\vec{s_j}}{\sqrt{l(s).l(s_j)}} \tag{1}$$

The score of an item i for session s is then given by:

$$score_{SKNN}(i, s) = \sum_{n \in \mathcal{N}(s)} sim(s, n) I_n(i), \tag{2}$$

where the indicator function $I_n(i)$ returns 1 if n contains item i and 0 otherwise. It is to be noted that $score_{SKNN}(i, s)$ does not take into account the position of i in either s or n, nor does it consider the recency of items in s while computing similarity with a past session s_j.

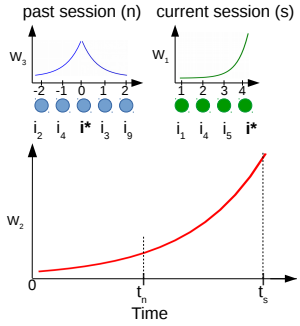


Figure 1: Illustration of decay factors in STAN.

4 STAN

As depicted in Fig. 1, STAN extends SKNN by incorporating the following factors related to the sequence of items in sessions and nearness of sessions in time:

(1) ***Factor-1****: Recent items in s are more important.* When comparing similarity of s with any session $s_j \in \mathcal{S}$, more recent items in session s should be given higher importance. We assign a value

$$w_1(i, s) = \exp\left(\frac{p(i, s) - l(s)}{\lambda_1}\right) \tag{3}$$

to item i in vector $\vec{s}$, where $\lambda_1 > 0$ and $p(i, s)$ denotes the position/index of i in s. Therefore, more recent items in s get a higher weightage, and the resulting real-valued vector $\vec{s_w}$ (note: $\vec{s}_j$ is still binary-valued) is used to find the similarity between s and s_j as follows:

$$sim_1(s, s_j) = \frac{\vec{s_w}.\vec{s_j}}{\sqrt{l(s).l(s_j)}} \tag{4}$$

(2) ***Factor-2****: Recency of session s_j w.r.t s.* Sessions farther from s are assigned lower weightage as in TARS [1].

$$w_2(s_j|s) = \exp\left(-\frac{t(s) - t(s_j)}{\lambda_2}\right), \tag{5}$$

where $\lambda_2 > 0$, $t(s) > t(s_j)$, and $w_2(a|b)$ is read as weight-factor for a w.r.t. b.

The neighborhood $\mathcal{N}(s)$ is then found by taking the top N most similar sessions using the similarity measure:

$$sim_2(s, s_j) = sim_1(s, s_j).w_2(s_j|s) \tag{6}$$

(3) ***Factor-3****: Position of a recommendable item in $n \in \mathcal{N}(s)$.* Once the neighborhood of s is obtained, any item in a session $n \in \mathcal{N}(s)$

is a potential candidate for recommendation. Consider the item i^* that occurs in both s and n, and is most recent w.r.t. s. Then, items in n that are closer to i^* are assigned higher weightage compared to items that were clicked farther from i^* according to:

$$w_3(i|s,n) = \exp\Big(-\frac{|p(i,n) - p(i^*,n)|}{\lambda_3}\Big) I_n(i), \tag{7}$$

where $\lambda_3 > 0$, such that the relevance of item i from neighboring session n for the current session s is given by:

$$rel(i|s,n) = sim_2(s,n).w_3(i|s,n) \tag{8}$$

Finally, the score of item i for session s over all neighbors is given by:

$$score_{STAN}(i,s) = \sum_{n \in \mathcal{N}(s)} rel(i|s,n) \tag{9}$$

The speed of forgetting items (sessions) can be adjusted via λ_is such that lower values of λ_is imply more emphasis on recent items (sessions). In the limit where $\forall \lambda_i \to \infty$, we get vanilla SKNN as in Eq. 2.

5 EXPERIMENTAL EVALUATION

We consider the task of incrementally predicting the immediate next item given the prior sequence of items in the session on three benchmark datasets: i) Yoochoose (YC): RecSys'15 Challenge dataset[1], ii) Diginetica (DN): CIKM Cup 2016 dataset[2], and iii) RetailRocket(RR)[3]. We omit detailed statistics of datasets for brevity - we use same evaluation protocol and train-test splits as used in [14] for YC and DN[4], and in [11] for RR. Considering the large number of sessions in YC, the recent 1/4 and 1/64 fractions of the training set are used to form two datasets: YC-1/4 and YC-1/64, respectively, as done in [14]. We use same evaluation metrics Recall@K (R@K) and Mean Reciprocal Rank@K (MRR@K) as in [14] and [11] for respective datasets. Further, we use 10% of train set as hold-out validation set for tuning parameters λ_is. We use neighborhood of $N = 500$ sessions and recommendation list of length $K = 20$ in all experiments.

Existing benchmarks and baselines considered. We compare STAN with standard neighborhood methods Item-KNN [6, 9] and SKNN [7, 11], four deep learning approaches (GRU4Rec, NARM, STAMP, and SR-GNN), a matrix factorization approach (BPR-MF) that optimizes pairwise ranking objective function via gradient descent, and a sequential prediction method based on Markov chains (FPMC), as benchmarked in [14] for YC and DN, and in [11] for RR. Further, to evaluate the effect of λ_is, we consider special cases of STAN where we consider only one of the three factors at a time to obtain three variants of SKNN: i) SKNN+w_1 with $\lambda_2, \lambda_3 \to \infty$, ii) SKNN+$w_2$ with $\lambda_1, \lambda_3 \to \infty$, and iii) SKNN+$w_3$ with $\lambda_1, \lambda_2 \to \infty$. Note, results for SKNN correspond to STAN with $\lambda_1, \lambda_2, \lambda_3 \to \infty$.

Selecting the decay factors. Let L denote the average session length over the training sessions. The parameters λ_1, λ_3 are chosen from the set $\{\frac{L}{8}, \frac{L}{4}, \frac{L}{2}, L, 2L\}$. All datasets have timestamps in seconds, we therefore, choose $\frac{\lambda_2}{T}$ from the set $\{2.5, 5, 10, 20, 40, 80, 100\}$,

[1]http://2015.recsyschallenge.com/challege.html
[2]http://cikm2016.cs.iupui.edu/cikm-cup
[3]https://www.dropbox.com/sh/dbzmtq4zhzbj5o9/AACldzQWbw-igKjcPTBI6ZPAa?dl=0
[4]Splitting as per https://github.com/CRIPAC-DIG/SR-GNN.

Table 1: R and MRR for a recommendation list length of $K = 20$. All benchmarks for YC and DN are taken from [14], and for RR from [11]. Underlined numbers are overall best models, numbers in bold are best non-deep learning results.

	YC-1/64		YC-1/4		DN		RR	
Method	**R**	**MRR**	**R**	**MRR**	**R**	**MRR**	**R**	**MRR**
BPR-MF	31.31	12.08	3.40	1.57	5.24	1.98	35.70	30.30
FPMC	45.62	15.01	-	-	26.53	6.95	32.00	27.30
Item-KNN	51.60	21.81	52.31	21.70	35.75	11.57	24.00	10.70
SKNN [7, 11]	63.77	25.22	62.13	24.82	48.06	16.95	64.57	43.59
SKNN+w_2	63.86	25.19	64.13	25.25	48.63	17.09	64.57	43.51
SKNN+w_3	64.75	25.53	63.01	25.14	48.80	17.30	65.47	45.12
SKNN+w_1	68.11	28.47	64.86	27.66	49.91	18.07	64.73	43.63
STAN (ours)	**69.45**	**28.74**	**70.07**	**28.89**	**50.97**	**18.48**	**65.66**	**45.18**
GRU4REC[6]	60.64	22.89	59.53	22.60	29.45	8.33	-	-
NARM[8]	68.32	28.63	69.73	29.23	49.70	16.17	-	-
STAMP[10]	68.74	29.67	70.44	30.00	45.64	14.32	-	-
SR-GNN[14]	70.57	30.94	71.36	31.89	50.73	17.59	-	-

Table 2: Optimal values of λ_1, λ_2 and λ_3.

Dataset	λ_1	λ_2/T	λ_3	**Dataset**	λ_1	λ_2/T	λ_3
YC-1/64	1.04	5	4.17	**DN**	1.25	80	2.5
YC-1/4	1.02	5	2.05	**RR**	3.54	20	3.54

Table 3: Performance analysis of STAN and SKNN at varying session lengths L.

	YC-1/64				DN			
	L<5		L>=5		L<5		L>=5	
Method	**R**	**MRR**	**R**	**MRR**	**R**	**MRR**	**R**	**MRR**
STAN	**71.77**	**32.41**	**66.53**	**24.09**	**51.56**	**19.67**	**50.18**	**16.88**
SKNN	69.72	31.45	56.25	17.30	50.33	19.33	44.99	13.73

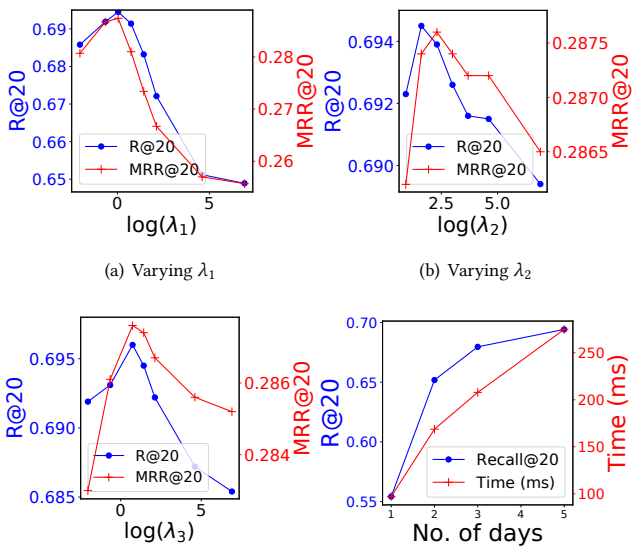
where $T = 24 \times 3600$. The best parameters are obtained using grid search to maximize R@20 on the validation set.

5.1 Results and Observations

We make the following key observations:

(1) From Table 1, we observe that *STAN shows consistent and significant improvement over all non-deep learning approaches on all datasets*. Further, amongst the three variants of SKNN, SKNN+w_1 is consistently the best across datasets and shows significant improvement on all dataset except for RR. STAN outperforms all variants of SKNN as it considers all three factors simultaneously. As shown in Table 2, the optimal values obtained for the corresponding parameters vary across datasets, highlighting the diversity of datasets considered, and the ability of STAN to adapt to diverse application domains by adjusting λ_is.

(2) Figs. 2(a)-2(c) show the effect of the three factors in STAN (same effect was observed across datasets):
i) As in observation 1, Fig. 2(a) further reiterates that λ_1 plays a significant role in giving relevant recommendations, as R and MRR vary significantly with varying λ_1. Giving all importance to only the most recent item while ignoring the rest in the current session ($\lambda_1 \ll 1$, as in Item-KNN) or giving equal importance to all items in the current session ($\lambda_1 \to \infty$, as in SKNN) may be sub-optimal. Items in the current session other than the most recent ones are also important but not as much as the more recent items.
ii) Similarly, Fig. 2(b) suggests that using only the most recent past sessions ($\lambda_2 \ll 1$) or giving equal weightage to all past sessions



(a) Varying λ_1 (b) Varying λ_2

(c) Varying λ_3 (d) Varying past data size

Figure 2: (a)-(c) Influence of the three factors in STAN on R@20 and MRR@20 measured by varying one parameter while keeping the other two constant at their optimal values on YC-1/64. (d) Effect of dataset size on computational efficiency (average time per recommendation) and R@20.

($\lambda_2 \rightarrow \infty$) can degrade the recommendation performance. A gradually decreasing weightage given to past sessions controlled via λ_2 works best.

iii) Fig. 2(c) suggests that suitable weightage to items in a neighboring session depending on its position w.r.t. to the item(s) common with the current session impacts recommendation performance. Gradually decreasing importance to items farther from the most recent common item with the current session, as controlled by λ_3, works best compared to taking only the immediate previous and next items ($\lambda_3 \ll 1$) or giving equal weightage to all items in neighboring sessions ($\lambda_3 \rightarrow \infty$).

(3) Despite being simple and intuitive, the performance of STAN is competitive to state-of-the-art deep learning models including NARM [8], STAMP [10] and SR-GNN [14]. Our results suggest that STAN can be considered as a strong baseline for future research in SR algorithms.

(4) Fig. 2(d) shows the trade-off between computational efficiency and recommendation quality by varying the number of days (past sessions) considered from past. Using suitable item-session inverted indices for efficiency for all datasets considered, we found that average time per recommendation to be highest for YC (equal to 274 ms)[5] amongst all datasets (on a 2.70GHZ processor with 32GB RAM), which may be considered viable. We leave efficiency for future investigation, which can be achieved by using, e.g., locality sensitive hashing.

[5]corresponding to YC-1/64, this amounts to reduction in R@20 of 0.9% when considering the bigger YC-1/4

(5) We observe than STAN performs better than vanilla SKNN for short as well as long sessions as shown in Table 3. More importantly, the performance gap increases for longer sessions indicating the advantage of various weight factors to enable better attention on, e.g. more recent intent of the user. Similar behavior has been observed, and accordingly leveraged, in e.g. [10, 14].

6 CONCLUSION AND FUTURE WORK

We have proposed Sequence and Time Aware Neighborhood approach (STAN) for session-based recommendations (SR). Despite the simple techniques employed to incorporate sequence and time information using decay factors at three different places while determining session-level similarity and item relevance, STAN shows significant improvement over existing neighborhood-based methods, and is even comparable to state-of-the-art deep learning approaches. Our results suggest that STAN can be considered as a strong baseline for future research in SR algorithms: insights drawn from our intuitive approach can even be useful to come up with novel deep learning architectures for SR.

REFERENCES

[1] Pedro G Campos, Fernando Díez, and Iván Cantador. 2014. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction* 24, 1-2 (2014), 67–119.
[2] Yi Ding and Xue Li. 2005. Time weight collaborative filtering. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, 485–492.
[3] Huifeng Guo, Ruiming Tang, Yunming Ye, Feng Liu, and Yuzhou Zhang. 2018. An Adjustable Heat Conduction based KNN Approach for Session-based Recommendation. *arXiv preprint arXiv:1807.05739* (2018).
[4] Negar Hariri, Bamshad Mobasher, and Robin Burke. 2012. Context-aware music recommendation based on latent topic sequential patterns. In *Proceedings of the sixth ACM conference on Recommender systems*. ACM, 131–138.
[5] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 843–852.
[6] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *Proceedings of the 4th International Conference on Learning Representations (ICLR-2016)*.
[7] Dietmar Jannach and Malte Ludewig. 2017. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 306–310.
[8] Jing Li, Pengjie Ren, Zhumin Chen, Zhaochun Ren, Tao Lian, and Jun Ma. 2017. Neural attentive session-based recommendation. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1419–1428.
[9] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 1 (2003), 76–80.
[10] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1831–1839.
[11] Malte Ludewig and Dietmar Jannach. 2018. Evaluation of Session-based Recommendation Algorithms. *arXiv preprint arXiv:1803.09587* (2018).
[12] Yu Sun, Peize Zhao, and Honggang Zhang. 2018. TA4REC: Recurrent Neural Networks with Time Attention Factors for Session-based Recommendations. In *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–7.
[13] João Vinagre and Alípio Mário Jorge. 2012. Forgetting mechanisms for scalable collaborative filtering. *Journal of the Brazilian Computer Society* 18, 4 (2012), 271–282.
[14] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based Recommendation with Graph Neural Networks. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*.