Caser+ and CosRec+: Closing the Gap Between CNNs and Attention Models in SRS[⋆]

Federico Siciliano[1,*,†], Antonio Purificato[1,†], Filippo Betello[1,†], Nicola Tonellotto[2] and Fabrizio Silvestri[1]

[1] *Department of Computer, Control and Management Engineering, Sapienza University of Rome, Italy*
[2] *Information Engineering Department, University of Pisa, Italy*

Abstract

Sequential Recommender Systems (SRSs) have predominantly shifted toward neural-based models. Despite significant advances, Convolutional Neural Network (CNN)-based SRSs have been increasingly overshadowed by more powerful attention-based approaches. In this paper, we introduce a novel adaptation of two popular CNN-based SRSs, Caser and CosRec. We enhance their training by adjusting the convolution and pooling operations to process the entire input sequence simultaneously rather than focusing only on the most recent item. Experimental results show that these modified CNN-based models achieve improvements of up to +65% in NDCG@10 over their original versions. Code is available at https://github.com/antoniopurificato/recsys_conv_conf.

Keywords

Recommender Systems, Convolutional Neural Networks, Sequential Recommendation

1. Introduction

Sequential Recommender Systems (SRSs) have evolved into indispensable tools that shape our everyday interactions with online content [2]. The rapid growth of e-commerce and entertainment services has only amplified the demand for personalized recommendations that boost user engagement and improve content discovery [3, 4, 5]. Because people's preferences are shaped by their past behaviors and often change over time, capturing this temporal nature is a core challenge for SRSs [6, 7, 8]. As the field of sequential recommendation has progressed, its advances have been closely tied to breakthroughs in deep learning techniques [9, 10, 11]. Convolutional models gained popularity as a means to capture local dependencies in user histories [12, 13]. More recently, however, the introduction of transformer-based solutions [14] has driven a surge of interest in attention-powered SRSs [15, 10]. Despite these strides, convolutional architectures have never achieved the dominance seen with transformer-based counterparts. Nevertheless, CNN-based methods, such as [12], continue to serve as strong baseline comparisons in contemporary work [16].

This paper introduces an enhanced training paradigm for two well-known CNN-based SRSs—Caser and CosRec—yielding new variants that we refer to as Caser+ and CosRec+. The key idea is to train these models to predict a series of future elements, rather than merely the most recent one. We make several targeted modifications to the original architectures to enable this. These changes enable multiple predictions per sequence, leading to more stable training and better convergence.

Our empirical evaluation confirms the effectiveness of these improvements. Caser+ yields a 750% relative gain in NDCG@10 over its predecessor, and CosRec+ improves NDCG@10 by 65%. Moreover, CosRec+ even surpasses SASRec, a leading attention-driven SRS, by 53%.

IIR2025: 15th Italian Information Retrieval Workshop, 3th - 5th September 2025, Cagliari, Italy

[⋆] This work is an extended abstract of "Are Convolutional Sequential Recommender Systems Still Competitive? Introducing New Models and Insights" accepted at the International Joint Conference on Neural Networks 2025 [1].

[*] Corresponding author.

[†] These authors contributed equally.

siciliano@diag.uniroma1.it (F. Siciliano); purificato@diag.uniroma.it (A. Purificato); betello@diag.uniroma1.it (F. Betello); nicola.tonellotto@unipi.it (N. Tonellotto); fsilvestri@diag.uniroma1.it (F. Silvestri)

0000-0003-1339-6983 (F. Siciliano); 0009-0009-3933-380X (A. Purificato); 0009-0006-0945-9688 (F. Betello); 0000-0002-7427-1001 (N. Tonellotto); 0000-0001-7669-9055 (F. Silvestri)

2. Methodology

2.1. Background

Sequential recommendation aims to predict the next item i_{l+1} based on a preceding sequence $(i_1, \ldots, i_l)$. Directly training a model to output only the last element i_{l+1} can be inefficient for longer histories [17]. A more effective strategy, as adopted by sequence-to-sequence architectures [10], is to predict each successive interaction: i_2 from (i_1), then i_3 from (i_1, i_2), and so forth [18]. In the neural recommendation paradigm, each item is projected into a continuous embedding space [19], producing an input representation as an $l \times h$ matrix. Classic recurrent-based recommenders like GRU4Rec [9] process one timestep at a time. The hidden state at time t feeds into both the next recurrent cell and the output layer, allowing information from $(i_1, \ldots, i_t)$ to accumulate and influence all future predictions. Attention-based solutions like SASRec [10] use self-attention in order to evaluate all positions in the sequence simultaneously. Masking restricts each timestep t so that it only sees past interactions $(i_1, \ldots, i_t)$.

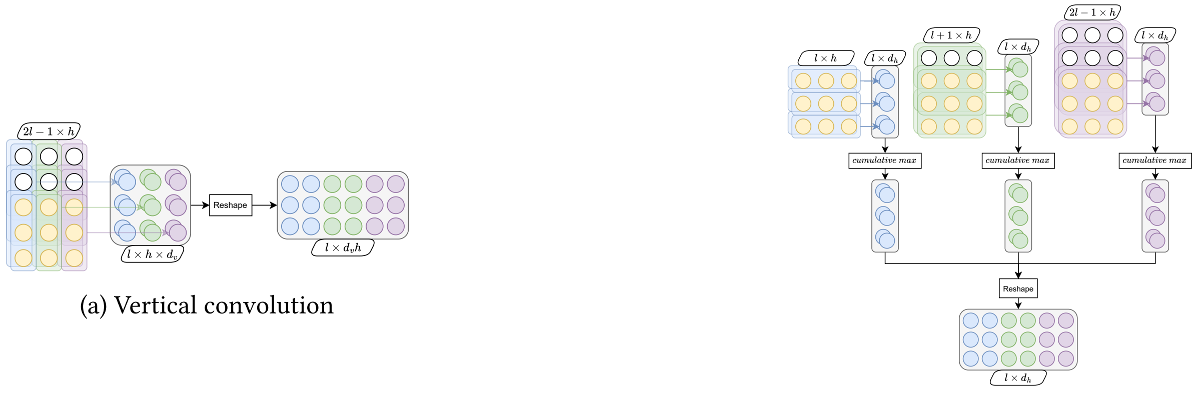
CNNs, which originated in image processing, slide a convolutional filter across the input to extract local patterns. Here we focus on two CNN-based recommenders: Caser [12] and CosRec [13].

2.2. Caser and CosRec

Caser applies two kinds of convolution. First, its vertical filters cover all l timesteps but only one embedding dimension per filter. This yields a vector of size $h \times d_v$, where d_v denotes the number of vertical kernels.Second, horizontal convolutions use multiple filters with different temporal extents $j \in \{1, \ldots, l\}$. A kernel of shape $j \times h$ captures local patterns across j consecutive items. Each horizontal filter produces a $(l - j + 1)$-long feature map, then a max pooling compresses this into a single d_h-dimensional vector. Concatenating all l pooled vectors yields a representation of length $l \times d_h$, which is then merged with the vertical features. The resulting vector of size $(h \times d_v) + (l \times d_h)$ feeds into a fully-connected layer to generate the final score for every potential next item.

CosRec follows a different design by first forming all possible pairs of embeddings. Specifically, it constructs a 3D tensor of shape $l \times l \times 2h$, where each slice encodes the concatenated embeddings of an item pair. This tensor is then passed through two convolutional blocks: each block contains a 1×1 and a 3×3 convolution, followed by batch normalization and ReLU. With no padding, each block shrinks the spatial dimensions by 2 on each axis, resulting in an $(l - 4) \times (l - 4) \times d_c$ tensor at the end of the pipeline. Finally, global average pooling across the first two dimensions produces a d_c-dimensional summary vector, which is processed by a dense layer to obtain the output predictions.

2.3. Caser+ and CosRec+



(a) Vertical convolution

(b) Horizontal convolution

Figure 1: Visual depiction of Caser+ functioning.

Table 1
Results of the proposed models and SASRec in terms of Precision@K (P@K), Recall@K (R@K), NDCG@K and MAP@K, with K $\in \{10, 20\}$. **Bold** denotes the best model for a dataset by the metric, underlined the second best. † indicates a statistically significant result of the new model w.r.t. its original version and * means statistically significant w.r.t. SASRec, based on Wilcoxon test with **p**-value **< 0.05**.

Dataset	Model	P@5	R@5	NDCG@5	P@10	R@10	NDCG@10	P@20	R@20	NDCG@20
FS-TKY	Caser	0.0065	0.0327	0.0252	0.0048	0.0475	0.0300	0.0031	0.0619	0.0337
	Caser+	0.0599†	0.2996†	0.2280†	0.0383†	0.3833†	0.2551†	0.0242†	0.4841†	0.2807†
	CosRec	0.0753	0.3764	0.3140	0.0450	0.4496	0.3378	0.0265	0.5299	0.3581
	CosRec+	**0.1156**†*	**0.5780**†*	**0.5439**†*	**0.0626**†*	**0.6260**†*	**0.5594**†*	**0.0326**†*	**0.6519**†*	**0.5660**†*
	SASRec	0.0783	0.3915	0.3526	0.0431	0.4312	0.3653	0.0235	0.4709	0.3754
FS-NYC	Caser	0.0026	0.0129	0.0093	0.0019	0.0194	0.0114	0.0013	0.0259	0.0130
	Caser+	0.0504†	0.2521†	0.1716†	0.0359†	0.3592†	0.2065†	0.0229†	0.4571†	0.2312†
	CosRec	0.1025	0.5125	**0.4736**	0.0553	0.5531	**0.4872**	0.0303	0.6066	**0.5007**
	CosRec+	**0.1029***	**0.5146***	0.4409	**0.0582**†*	**0.5822**†*	0.4627	**0.0321**	**0.6420**†*	0.4777
	SASRec	0.0720	0.3598	0.3212	0.0412	0.4117	0.3379	0.0233	0.4658	0.3516
ML-1M	Caser	0.0112	0.0558	0.0374	0.0091	0.0914	0.0487	0.0078	0.1555	0.0648
	Caser+	0.0201†	0.1003†	0.0676†	0.0156†	0.1556†	0.0854†	0.0117†	0.2344†	0.1052†
	CosRec	0.0203	0.1015	0.0688	0.0163	0.1629	0.0885	0.0122	0.2447	0.1091
	CosRec+	0.0226	0.1131	0.0784†	0.0176†	0.1765†	0.0988†	0.0130†	0.2603†	0.1198†
	SASRec	**0.0323**	**0.1613**	**0.1132**	**0.0242**	**0.2419**	**0.1392**	**0.0172**	**0.3434**	**0.1649**

To enable Caser for full sequence-to-sequence learning, we modify both its vertical and horizontal convolutional components. For the vertical filters, we introduce left-padding of $(l-1)$ so that the l-sized kernels initially cover just the first item, then the first two, and so on. This adjustment yields an output tensor of shape $l \times h \times d_v$, allowing a sequential processing of the input, as depicted in Fig. 1a.

The horizontal convolutions require a similar treatment. Each horizontal kernel, spanning $j \times h$ where $j \in \{1, \ldots, l\}$, is left-padded with j placeholder elements. This setup allows every filter to produce a $l \times d_h$ matrix, from which we compute a cumulative maximum across the temporal axis instead of reducing along that axis. This preserves the intended max-pooling behavior at each timestep while retaining the full sequence length. Finally, we concatenate the vertical and horizontal outputs, resulting in a combined representation of shape $l \times (h \times d_v + l \times d_h)$, as illustrated in Fig. 1b.

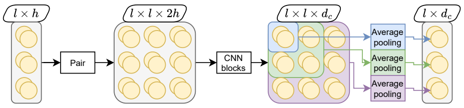


Figure 2: Visual depiction of CosRec+ functioning.

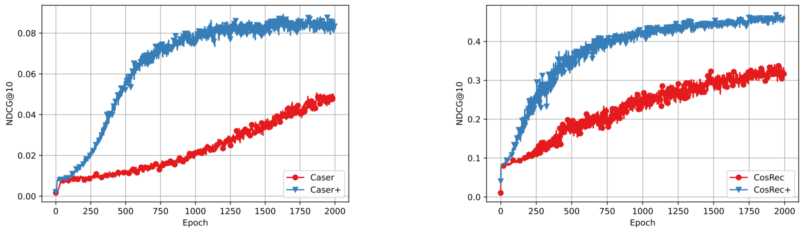
For CosRec, our adjustment follows a similar padding logic. Within each convolutional block, padding is introduced so that all intermediate tensors remain of shape $l \times l \times d_c$. Specifically, the 1×1 convolution requires no padding, while the 3×3 convolution is padded so that the output resolution stays constant across layers. Next, we redefine the average pooling strategy. Instead of a global average across the entire 2D space, we accumulate averages progressively. Starting with the top-left corner, we compute the mean of the 1×1 submatrix. Then we move on to the 2×2 top-left submatrix, and so on up to the full $l \times l$ matrix. This yields a condensed $l \times d_c$ output. A summary of this process is given in Fig. 2.

3. Results

Our setup mirrors that of [12]: interactions are treated as implicit feedbacks, users with fewer than five interactions are removed, and a leave-one-out split is used. We use three well-known datasets—MovieLens 1M (ML-1M) [20], Foursquare Tokyo (FS-TKY), and Foursquare New York City (FS-NYC) [21]. To address RQ2, we also compare against the attention-based SASRec [10]. All experiments were conducted using the EasyRec toolkit [22].

3.1. Comparison w.r.t. Caser & CosRec

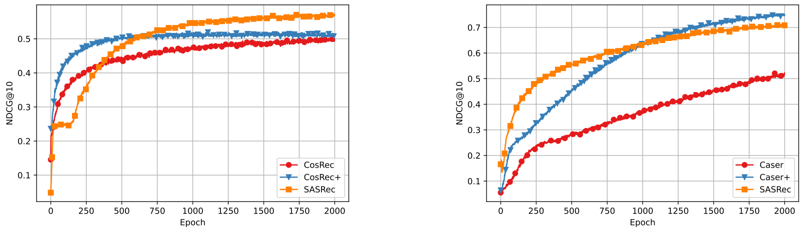
We train all models for 2000 epochs and show their results in Table 1. The modified architectures yielded better scores than the baseline models across all metrics. For instance, on the FS-TKY dataset, resp. ML-1M dataset, Caser+ achieves an improvement of 0.2251, resp. 0.0367, in NDCG@10 w.r.t. Caser. Similarly, CosRec+ obtains an improvement of 0.2216, resp. 0.0103, in NDCG@10 w.r.t. CosRec.



(a) Caser and Caser+ on ML-1M.

(b) CosRec and CosRec+ on FS.

Figure 3: Test performance during training epochs for the original models and their enhanced versions.



(a) CosRec, CosRec+ and SASRec on ML-1M.

(b) Caser, Caser+ and SASRec on FS-TKY.

Figure 4: Test performance during training epochs for the original models, their enhanced version and SASRec.

In Fig. 3, across all epochs, the enhanced models consistently outperform their respective baselines. Notably, CosRec+ reaches convergence in approximately 1000 epochs and achieves an NDCG@10 of 0.471, while the original CosRec struggles to surpass 0.30. This is especially important in low-resource settings where only a limited number of training epochs can be run.

3.2. Comparison with SASRec

From Table 1, on FS-TKY, CosRec+ demonstrates a clear advantage over SASRec across nearly all metrics, achieving up to a 0.1941 increase in NDCG@10. On ML-1M, SASRec still holds the edge overall, but the gaps have noticeably narrowed—our models trail by at most 0.0404 in NDCG@10.

Fig. 4a shows that while SASRec eventually surpasses both CosRec and CosRec+, the CNN-based models produce higher test performance during the first 250 and 500 epochs, with NDCG@10 reaching 0.3524 for SASRec, 0.4010 for CosRec, and 0.4746 for CosRec+. Similarly, Fig. 4b illustrates that although SASRec converges faster on FS-TKY, Caser+ overtakes it after epoch 1000.

4. Conclusions

This work demonstrates that appropriately modifying convolution-based sequential recommenders can substantially enhance their performance. Although our findings are not yet definitive, they suggest that CNN-based SRSs can surpass attention-based approaches on certain datasets and under specific conditions. In future work, we plan to conduct a more extensive hyperparameter search to determine whether these revised convolutional architectures can achieve even greater improvements.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

Acknowledgments

This work was partially supported by projects FAIR (PE0000013) and SERICS (PE00000014), under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU, and project NEREO (Neural Reasoning over Open Data), funded by the Italian Ministry of Education and Research (PRIN) Grant no. 2022AEFHAZ.

References

[1] F. Siciliano, A. Purificato, F. Betello, N. Tonellotto, F. Silvestri, Are convolutional sequential recommender systems still competitive? introducing new models and insights, in: 2025 International Joint Conference on Neural Networks (IJCNN), IEEE, 2025.

[2] G. Adomavicius, A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE transactions on knowledge and data engineering 17 (2005) 734–749.

[3] F. Betello, F. Siciliano, P. Mishra, F. Silvestri, Finite rank-biased overlap (frbo): A new measure for stability in sequential recommender systems, in: Proc. of the 14th Italian Information Retrieval Workshop, volume 3802, 2022, pp. 78–81.

[4] F. Betello, F. Siciliano, P. Mishra, F. Silvestri, Investigating the robustness of sequential recommender systems against training data perturbations, in: European Conference on Information Retrieval, Springer, 2024, pp. 205–220.

[5] A. Sbandi, F. Siciliano, F. Silvestri, Mitigating extreme cold start in graph-based recsys through re-ranking, in: Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, 2024, pp. 4844–4851.

[6] F. Betello, The role of fake users in sequential recommender systems (2024).

[7] A. Purificato, F. Silvestri, Eco-aware graph neural networks for sustainable recommendations, in: International Workshop on Recommender Systems for Sustainability and Social Good, Springer, 2024, pp. 111–122.

[8] A. Bacciu, F. Siciliano, N. Tonellotto, F. Silvestri, Integrating item relevance in training loss for sequential recommender systems, in: Proceedings of the 17th ACM Conference on Recommender Systems, 2023, pp. 1114–1119.

[9] B. Hidasi, A. Karatzoglou, L. Baltrunas, D. Tikk, Session-based Recommendations with Recurrent Neural Networks, in: Proc. ICLR, 2016.

[10] W.-C. Kang, J. McAuley, Self-attentive sequential recommendation, in: 2018 IEEE international conference on data mining (ICDM), IEEE, 2018, pp. 197–206.

[11] A. Purificato, G. Cassarà, F. Siciliano, P. Liò, F. Silvestri, Sheaf4rec: Sheaf neural networks for graph-based recommender systems, ACM Transactions on Recommender Systems (2023).

[12] J. Tang, K. Wang, Personalized top-n sequential recommendation via convolutional sequence embedding, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 565–573. URL: https://doi.org/10.1145/3159652.3159656. doi:10.1145/3159652.3159656.

[13] A. Yan, S. Cheng, W.-C. Kang, M. Wan, J. McAuley, Cosrec: 2d convolutional neural networks for sequential recommendation, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 2173–2176. URL: https://doi.org/10.1145/3357384.3358113. doi:10.1145/3357384.3358113.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[15] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, P. Jiang, Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer, in: Proceedings of the 28th ACM international conference on information and knowledge management, 2019, pp. 1441–1450.

[16] X. Du, H. Yuan, P. Zhao, J. Qu, F. Zhuang, G. Liu, Y. Liu, V. S. Sheng, Frequency enhanced hybrid attention network for sequential recommendation, in: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2023, pp. 78–88.

[17] M. Quadrana, P. Cremonesi, D. Jannach, Sequence-aware recommender systems, ACM Comput. Surv. 51 (2018). URL: https://doi.org/10.1145/3190616. doi:10.1145/3190616.

[18] G. Di Teodoro, F. Siciliano, N. Tonellotto, F. Silvestri, A theoretical analysis of recommendation loss functions under negative sampling, in: 2025 International Joint Conference on Neural Networks (IJCNN), IEEE, 2025.

[19] S. Zhang, L. Yao, A. Sun, Y. Tay, Deep learning based recommender system: A survey and new perspectives, ACM Comput. Surv. 52 (2019). URL: https://doi.org/10.1145/3285029. doi:10.1145/3285029.

[20] F. M. Harper, J. A. Konstan, The movielens datasets: History and context, ACM Trans. Interact. Intell. Syst. 5 (2015). URL: https://doi.org/10.1145/2827872. doi:10.1145/2827872.

[21] D. Yang, D. Zhang, V. W. Zheng, Z. Yu, Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns, IEEE Transactions on Systems, Man, and Cybernetics: Systems 45 (2015) 129–142. doi:10.1109/TSMC.2014.2327053.

[22] F. Betello, A. Purificato, F. Siciliano, G. Trappolini, A. Bacciu, N. Tonellotto, F. Silvestri, A reproducible analysis of sequential recommender systems, IEEE Access (2024).