# Next Item Recommendation using Sequence Embedding and Pairwise Loss*

Extended Abstract†

Cristobal Eyzaguirre
Pontificia Universidad CatÃşlica de Chile
Santiago, Chile
ceyzaguirre4@uc.cl

Rolf Skog
Pontificia Universidad CatÃşlica de Chile
Santiago, Chile
rpskog@uc.cl

## ABSTRACT

In this paper we propose an alternative to current state of the art next item recommendation systems that, through a novel combination of an existing model and loss function can be tuned to achieve similar results. Furthermore, the model proposed can be extended easily to include other sources of information that may be valuable in the context of recommendation (a known weakness of some current methods). When using contextual information not harnessed by other models we obtain better results than even those of the state of the art at the time of writing this paper. Historical information of past interactions can be used in the model in a straightforward manner by the simple concatenation of the embedded items in a users recent history. Finally, an alternative way to include longer histories using self-attention is proposed, evaluated and discussed.

**ACM Reference Format:**
Cristobal Eyzaguirre and Rolf Skog. 2018. Next Item Recommendation using Sequence Embedding and Pairwise Loss: Extended Abstract. In *Proceedings of (Recsys),* Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, Article 4, 5 pages. https://doi.org/10.475/123_4

## 1 INTRODUCTION

Recommendation systems are a key part of many technologies people use day to day to aid the decision between the ever growing amount of possible choices. The interactions users have with the systems can be harnessed to better the accuracy of future recommendations creating a positive feedback loop, provided the information is useful to the models learning process.

Two main tendencies can be observed in today's technologically connected world: people are increasingly using more and more computational systems, and these in turn acquire more and more information about its users. The increase of interactions includes *return users*, users that use the system on a periodical basis and thus have rich interaction histories with the platforms offerings. These interaction histories, if properly exploited, permits models to predict future interactions based on what has occurred in the past as the past, and thus current recommender systems attempt to capture and represent a users history to achieve better performance.

Recent breakthroughs in the area of Next-Item Recommendation using Deep Learning have increased accuracy of the predictions by using self-attention to exploit the users previous transaction history in order to predict the next. Most notably, the current state of the art [9] in over 10 datasets achieves Hit Rates that are up to 45 percent compared to those of its predecessors. Previous attempts to encode time the time sensitive data of past interactions relied on recurrent neural networks or convolutional neural networks were easily surpassed by this new architecture. However, the architecture used isn't easily extensible to use contextual data that is currently available in real world applications such as the users gender or age.

This failure to acknowledge one of the richest sources of information that is present in almost every dataset and real world use-case is the main motivation for this paper, where we seek to replicate the results obtained by the state of the art using a more extensible and flexible model. Both the main architecture and the loss functions existed prior to the writing of this paper, although to the best of our knowledge they hadn't been used together in any context, much less the context of next item recommendation. The main contributions of this publication stand as follows:

- Proposing a novel combination of existing architectures and loss functions that achieves competitive results.

- Extending the model to use contextual information and showing how this can easily be done in the proposed architecture.

We experiment with adding the self-attention module present in Zhangs work to the proposed model to better model longer sequences.

## 2 RELATED WORK

Most all modern literature on recommendation systems applies to this work, as it attempts to unify state of the art sequence-aware systems with contextual information and embeddings, on which extensive literature exists. We focus on deep learning models, on these applied to recommendation, and finally con deep learning sequence aware systems. The above categories are treated with healthy focus on models more currently relevant.

Deep learning has revolutionized many areas and recommendation isn't the exception. Simple multi-layer perceptrons have been applied to the area where their inclusion of non-linearity's aid in finding more complex relations [10]. Convolutional neural networks provide infrastructure not only for text and image feature extraction (most important in the context of content based recommendation) but also for sequence modeling where sliding

*Produces the permission block, and copyright information
†The full version of the author's guide is available as `acmart.pdf` document


*Recsys, December 2018, Santiago, Chile*

ACM ISBN 123-4567-24-567/08/06.
https://doi.org/10.475/123_4

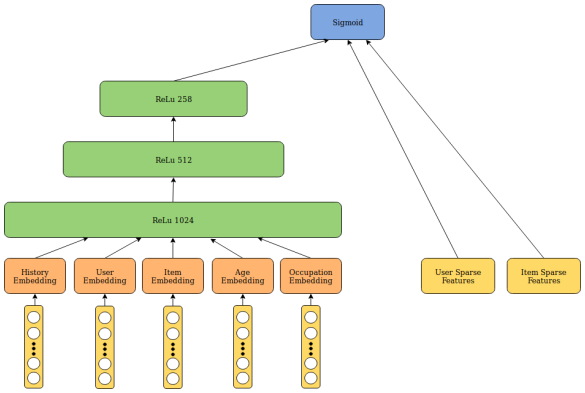


**Figure 1: Deep and Wide implemented architecture**

windows can be used to encode relations maintaining positional information [7]. To the same end, recurrent neural networks can be used to encode positional data and have been extensively used to obtain sequence aware representations in many areas including recommender systems [4].

The deep learning model for english to french translation, Transformer [8], has popularized the use of self attention used in conjunction with time embeddings to represent sequences and model interactions between elements present in the series. The current state of the art for next item recomendation, AttRec [9] builds upon this using the self-attended histories to model short-term user preferences which are later compared to items (as is the norm for metric learning models) and combined with the long-term user preference to obtain an items score for a given user $u$ at time $t$.

## 3 MODEL

In this section we describe the two models evaluated during the experiments. To do this we first show the model's architecture, with a brief explanation of each components function and inputs. We then describe the optimization problem and goal that our models try to solve, so as to correctly serve recommendations.

The two models explored have the same base structure, except for a self attention module to input user watch history. This was done to investigate the effect of self attention on the models performance.

### 3.1 BPR Deep and Wide

*3.1.1 Model Architecture.* The model follows the architecture design proposed by [2] in which the has two separate parts.

The deep section tries to model complex interactions between objects. It first uses a dense embedding layer to lower dimensional and learn interactions between the different fields, especially users and items. The results of each embedding are then concatenated and feed into a normal deep forward network. For our model each user and item embedding has 100 factors while the context embeddings are half the size. The layers size follow Cheng et al., 2016 implementation.

The wide portion of the model learns low order interactions. The features use as input vectors where all the sparse features available for items and users, as well as first order cross product between users and candidate item's features.

*3.1.2 Optimization.* For training the neural network we propose the use of pairwise learning, specifically using BPR[6]. Bayesian Personalized Ranking has shown success when trying to recommend items as it looks to maximize the distance between a liked item and a disliked one. In our implementation we modeled disliked items as all the unseen items for a user, so as to easily implement the model in implicit feedback domains.

The loss function is as follows:

$$\mathcal{L}(\Theta) = \sum_{(u,i,j)\in D} -log\sigma(y_+ - y_-) + \|\Theta\|^2 \quad (1)$$

We use a sigmoid function to soften the difference between a liked and disliked item. During each train iteration we sample a positive and negative item for a user. We then do two forward passes over our model, one for each item. The output of each pass is then fed into the loss function and its result back-propagated to update the weights. The optimizer used for this implementation is ADAM [5].

*3.1.3 Implementation details.* By having to run two forward passes for each iteration, a loss in train efficiency would be noted. This loss could be ignored if the convergence speed is also improved, as noted by Rendle et al., 2009, when doing random samples of pairs. This efficiency loss is only a problem during train, as the serving process is done only with one item, so if improved accuracies are obtained a trade-off could be achieved.

For modeling user's watch history we propose using a concatenated vector of item embeddings. In this way different sequence lengths can be implemented with almost no change in the model design, as the architecture allows for easy adding of dense embeddings. Another advantage is the use of the same embedding for history and items. This will speed up the learning of the embedding and also find different interactions between items previously undiscovered.

### 3.2 Attention module

Having trained the model to achieve similar results to the current state of the art without leveraging additional contextual information we proceed to include a self attention module similar to that described in Zhang et al.. The main intuition to as why we choose to do so is considering more elements of a users history using mere concatenation of the embedded items entails an increase in model complexity while providing increasingly less information since users tastes change little in the short term.

Users histories are combined and reduced to a single vector that we hypothesize to be an approximation of the users short term interests.

*3.2.1 Architecture.* This mean of the last items consumed is built following the specifications on [9] which in turn was inspired by those of the Transformer [8]. The output of this module effectively replaces the history embedding input to the deep section of the model described in the section above and its inputs are those of the same model.

Let $X_t^u \in R^{Nxd}$ be the $d$ dimensional embeddings of the history of the last $L$ items consumed by user $u$ at time $t$. Two diferent matrices ($W_Q$ and $W_K$) are used to compute projections of $X_t^u$ into

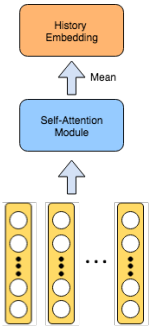


**Figure 2: Attention Module.**

queries and keys respectively. The ReLU activation function is applied to the projections.

$$Q' = ReLU(X_t^u W_Q) \tag{2}$$

$$K' = ReLU(X_t^u W_K) \tag{3}$$

A similarity score is assigned to each pair of elements in the user history by calculating the cross product. We can now scale this value using the embedding dimensionality and a attention probability distribution can be calculated over the projected history items by applying the softmax function.

$$s_t^u = softmax(\frac{Q'K'^T}{\sqrt{d}}) \tag{4}$$

With the probabilities in hand we can now obtain the attended output of the module by multiplying these values with the history matix $X_t^u$.

As in the paper Zhangs publication we embedd a representation of time by adding a time embedding to both query and key before their the ReLU activation function and mask the diagonal of the dot product before the application of softmax.

Although a better combination of the obtained self-attended history elements could maybe be obtained fromby using attention, we follow Zhang in calculating the mean value of the resulting vectors to obtain a $d$ dimensional representation of the last L elements in the users history that can be fed to the Deep and Wide model.

## 4 EXPERIMENTS

In our experiments we want to find what changes to our proposed models yield best results. For this we train the models with slight variations and evaluate them along with their baselines.

### 4.1 Dataset

We used the movielens 100-k dataset to evaluate the models. This decision was made because it is a widely used dataset in the recommender systems domain, making the results obtained in our experiments easy to compare to those of other researchers. The size of the dataset is also small enough to continuously retrain the models even in low grade hardware specs. The dataset includes rating users (identified by userid's) give to items (identified by itemid's) where the rating ranges from 1 to 5 points and is accompanied by a timestamp that indicates the time at which the user rated the item in question. Using the timestamp we can proceed to identify the last and second to last items the user interacted with and use these for evaluation and testing purposes respectively. We consider an item as "liked" when the rating given to the item is greater or equal to 4. When sampling negative items we consider the union of the sets of all items not consumed by the user and those with rating of 3 stars or lower.

**Table 1: Dataset Description**

| #Users | #Items | Interactions | Density |
|---|---|---|---|
| 943 | 1,682 | 100000 | 6.30% |

### 4.2 Evaluation Metrics

We split our dataset in two, the last seen item for each user is put in the test set. The train set is then split using cross-validation techniques for hyperparameter tuning.

We want to evaluate if the next liked item is going to appear in our recommendations. For this we utilize Hit Ratio, a metric that counts the occurrences of items in the test set in the recommendations.

$$HR@K = \frac{1}{M} \sum_{u \in U} 1(R_{u,g_u} < K) \tag{5}$$

where $u$ is an item in the test set and $R_{u,g_u}$ is the rank predicted by the model.

We also want to measure how close to the optimum place most of the next items are. The Mean Reciprocal Rank can be used to see the average rank of the recommended items.

$$MRR = \frac{1}{M} \sum_{u \in U} \frac{1}{R_{u,g_u}} \tag{6}$$

### 4.3 Baselines

The focus of our research if to propose modifications to a couple of state of the art recommender systems. Because of this we test our system against a couple classical models, as well as the implementation of the models proposed by their creators. The complete baselines are the following:

(1) POP: Ranks the items according to their popularity across all users. It recommends the most popular unseen items to each user.
(2) BPRMF: It optimizes the matrix factorization using a BPR approach. This means that it tries to maximize the difference between positive and negative items, but it doesn't include sequence or contextual information.
(3) Deep and Wide: Model propose by Cheng et al., 2016, using pointwise loss and batch learning.
(4) AttRec: the current state of the art in the datasets we use, it computes the euclidean distance between the long term user preference (a learnt embedding) and the short term intent calculated from the recent interaction history using self-attention.

**Table 2: Experiment Evaluation Results**

| Model | MRR | HR@50 |
|---|---|---|
| POP | 0.0388 | 0.2142 |
| BPRMF | 0.0616 | 0.3754 |
| AttRec | 0.0981 | 0.5273 |
| Deep and Wide | 0.0275 | 0.1918 |
| BPR Loss Deep and Wide | 0.1314 | 0.6082 |

**Table 3: Different dense embedding for BPR Deep and Wide**

| Information | MRR | HR@50 |
|---|---|---|
| Only user-item | 0.0204 | 0.1677 |
| No sequence and user context | 0.0296 | 0.2208 |
| sequence and no user context | 0.1175 | 0.5711 |
| All | 0.1314 | 0.6082 |

**Table 4: Convergence Epochs**

| Model | Convergence Epochs |
|---|---|
| Deep and Wide | 6 |
| BPR Loss Deep and Wide | 11 |

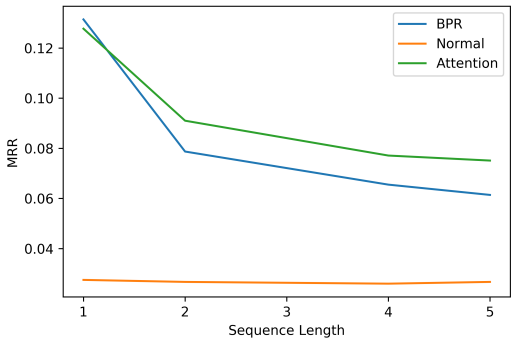


**Figure 3: Plot showing MRR for different sequence lengths**

## 4.4 BPR Deep and Wide experiments

For this model we investigate the effect of different modifications in its input parameters. Mainly we searched for the effects of adding different sequence lengths to the history embedding, searching for the optimal value. For the best results we compare encoding the different length sequences in two manners: the simple concatenation of all the embedded items in the users recent history, and the mean value of the self attended representations.

Finally, we investigated the potential gains to be had from adding context and content about the users and items respectively was also investigated.

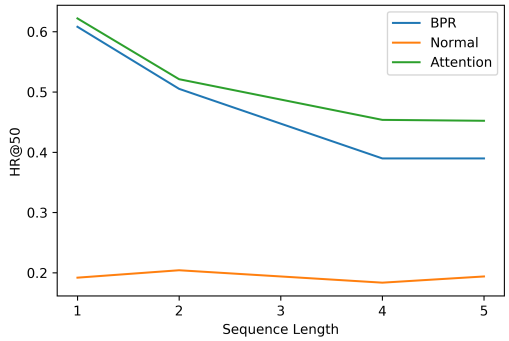


**Figure 4: Plot showing HR@50 for different sequence lengths**

# 5 ANALYSIS

## 5.1 BPR on Deep and Wide

Our proposed change for the Deep and Wide model is to change its training method to a BPR approach. In table 2 we can see that on movielens 100k the accuracy of the deep and wide model was definitely surpassed by using a pairwise training method. This can be explained by a variety of reasons. One of them is that when we want to rank different items, as is the case in next item recommendation, a pairwise approach have been shown to generally obtain better results.

On the other hand the results can also be explained if we take into account the size of the dataset. When we sampled positive items with negative items, we were also increasing the size of the dataset, as the amount of pairs was a lot larger than the amount of ratings. This increase in data size and availability is crucial in deep neural networks, as with more information better training can be obtained. This can also be seen in the amount of epochs needed to reach convergence, because not a lot of information was available the normal model reached convergence quickly, while the BPR version continued to learn.

## 5.2 Sequence Lengths effects

As stated in the investigation motivations, we tested our models with different history sequence lengths to explore the effect of item recommendation. Our initial hypothesis was that by adding longer users watch history the models would increase in accuracy as shown by Zhang et al., 2018. In the BPR Deep and Wide model we can observe in figures 1 and 2 that the evaluation metrics sharply drop as longer watch sequences are used. It should be noted that for every different length the model was tuned to fit achieve better result, so as to discard lack of hyper-parameter optimization as the root cause.

The Deep and Wide results contrast with the other attention based models, as these improve their accuracies with longer sequences, due to the ability to discriminate between useful and useless history items which motivated us to include the same module proposed by Zhang et al. to encode the $Lxd$ history matrix into a single $d$ dimensional vector using self attention. The results are

shown in green alongside the vanilla Deep and Wide and our version that uses the BPR loss function, where we can observe that using self attention increases the accuracy of the model when using longer sequences, but that the best results are still those obtained when considering user histories of length 1. Better results than those displayed can be had by further training the models since we trained all for only 10 epochs and that wasn't enough to obtain convergence on any of the models training processes. This limitation in training time resulted resulted from our lack of resources to train the model thoroughly and stability issues in the (free to use) platform we used for training.

We theorize that the decrease in model accuracy was caused because each new item considered for the history didn't add much useful information. This in turn increased the dimensionality of the models dense embeddings in a disproportionate amount when combining the elements in the history via concatenation. In the case where we used self-attention the logical cause for the decrease in results is that the increase in model complexity was high enough that either the dataset wasn't large enough for the model to overcome its competition, or limiting the training to 10 epochs severly limited its performance. In consequence the model wasn't able to differ between important inputs, like the first history or the user-item combination, and the less important sequence values. Therefore we see less accurate predictions. This being said, we would have observed the same results if we added more useless contextual features, but these weren't available in the dataset.

### 5.3 Contextual information

As observed in table 2, the BPR Deep and Wide model reaches larger hit rates and mean reciprocal ranks than AttRec. Some of this differences can be due to the presence of contextual information. AttRecs only uses item and user embeddings, so by adding available contextual information more complex interactions in the embeddings can be learned. This is an advantage of the Deep and Wide model, as it can easily aggregate user and item information. This would lead to better performance in richer datasets, although differences can already be seen only using genre, age and occupation as seen in the movielens dataset.

## 6 CONCLUSIONS

During this paper we tried to show the effect of modelling user watch sequence as a dense embedding. We successfully implemented different tow different ways of modeling this, a concatenated embedding and a self-attention module. The main result discovered was the great improvement in performance to the model when only the last seen item was used. This shows the importance of correctly modelling histories, as they can greatly improve a model. On the other hand we weren't able to sustain or improve performance with longer sequences, probably to a lack of tuning, though using self-attention improved performances compared to the concatenated vectors.

The other important result was the effectiveness of using pairwise loss for deep learning models. These recommender systems normally require huge amounts of data to be trained effectively, but this can result in harder to implement models. By using BPR we were able to train a complex model in a small dataset and achieve state of the art results. By adjusting the loss function to correctly model the task at hand, better recommendations can me made as the neural network learns to rank items instead of predicting their potential rating.

## 7 FUTURE WORK

The main problem with our research was the lack of variety in the datasets. To understand completely the effect of our modifications more rigorous testing must be done. For this we would train our models in larger datasets of different domain, especially on implicit feedback data.

Our results on the effect of sequence lengths was also counterintuitive, we thought that by adding more information performance should increase. Because of this more research should be done on how to correctly model user histories, as we showed that these were important when trying to recommend items. On this note further improvements could be done on the self-attention module, like adding an attention layer to the output vectors, instead of the mean.

## REFERENCES


[1] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*. ACM, 89–96.

[2] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. ACM, 7–10.

[3] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247* (2017).

[4] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[5] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[6] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 452–461.

[7] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 565–573.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[9] Shuai Zhang, Yi Tay, Lina Yao, and Aixin Sun. 2018. Dynamic Intention-Aware Recommendation with Self-Attention. *arXiv preprint arXiv:1808.06414* (2018).

[10] Shuai Zhang, Lina Yao, Aixin Sun, Sen Wang, Guodong Long, and Manqing Dong. 2018. NeuRec: On Nonlinear Transformation for Personalized Ranking. *arXiv preprint arXiv:1805.03002* (2018).