# Lightweight Model for Session-Based Recommender Systems with Seasonality Information in the Fashion Domain

NICOLA DELLA VOLPE, Politecnico di Milano, Italy
LORENZO MAINETTI, Politecnico di Milano, Italy
ALESSIO MARTIGNETTI, Politecnico di Milano, Italy
ANDREA MENTA, Politecnico di Milano, Italy
RICCARDO PALA, Politecnico di Milano, Italy
GIACOMO POLVANESI, Politecnico di Milano, Italy
FRANCESCO SAMMARCO, Politecnico di Milano, Italy
FERNANDO B. PÉREZ MAURERA, Politecnico di Milano, Italy
CESARE BERNARDIS, Politecnico di Milano, Italy
MAURIZIO FERRARI DACREMA, Politecnico di Milano, Italy

This paper presents the solution designed by the team "Boston Team Party" for the ACM RecSys Challenge 2022. The competition was organized by Dressipi and was framed under the session-based fashion recommendations domain. Particularly, the task was to predict the purchased item at the end of each anonymous session. Our proposed two-stage solution is effective, lightweight, and scalable. First, it leverages the expertise of several strong recommendation models to produce a pool of candidate items. Then, a Gradient-Boosting Decision Tree model aggregates these candidates alongside several hand-crafted features to produce the final ranking. Our model achieved a score of 0.18800 in the public leaderboard. To aid in the reproducibility of our findings, we open-source our materials.

CCS Concepts: • **Information systems** → **Learning to rank**; • **Theory of computation** → **Boosting**.

Additional Key Words and Phrases: Recommender Systems, RecSys Challenge, Neural Networks, Boosting, Feature Engineering

**ACM Reference Format:**
Nicola Della Volpe, Lorenzo Mainetti, Alessio Martignetti, Andrea Menta, Riccardo Pala, Giacomo Polvanesi, Francesco Sammarco, Fernando B. Pérez Maurera, Cesare Bernardis, and Maurizio Ferrari Dacrema. 2022. Lightweight Model for Session-Based Recommender Systems with Seasonality Information in the Fashion Domain. In . ACM, New York, NY, USA, 9 pages. https://doi.org/XXXXXXX.XXXXXXX

## 1 INTRODUCTION

Recommender systems are software applications which aim to help users in making choices from a large pool of possibilities and have become a crucial part of many modern online services. Historically, these systems have mainly been based on the assumption that information about individual users' long-term preference is available[13]. However, in many real-world applications, such as e-commerce, this long-term information is often unavailable, as users are not always logged-in or they are using the service for the first time[22]. In these cases, instead of long-term profiles, the


Manuscript submitted to ACM

interactions of the user during the ongoing session can be used to personalize recommendations. Such a setting is usually termed a "session-based recommendation" problem, which is the focus of the *ACM RecSys Challenge 2022*[1], organized by Dressipi, the global leader in fashion-AI.

The challenge consists in accurately predicting which fashion item will be purchased at the end of a session based on the items visited during the same session, historical data about previous interactions, and several attributes of the items. The proposed final solution is an ensemble which leverages a Gradient-Boosted Decision Trees (GBDT) model to combine the performances of several strong recommendation algorithms. The source code of our final model and the respective documentation are publicly available on GitHub. [2]

## 2 PROBLEM FORMULATION AND EDA

As part of this challenge, Dressipi published a dataset of past anonymous online retail sessions, sampled over 18 months - between January 2020 and June 2021. The total number of available items is around 24k, while the number of sessions is 1.1M. In particular, each session, which correspond to one user's activity on one day, is composed by a sequence of views, i.e., the items seen by the user during the session and result into a purchase that happened at the end of the session. Each view and purchase is enriched with a timestamp, including both date and time, which has been assumed to be real and significant as no further information was provided in the official data description.

### 2.1 Problem Formulation

The task is a traditional top-100 recommendation scenario, i.e., to produce a recommendation list of length 100 for each session. Each list must contain the purchased item from a subset of possible candidates. The evaluation metric is the Mean Reciprocal Rank (MRR) metric [16, 21]. The evaluation is performed over the month of June 2021 - referenced as the test month, for which only the views are available. Additionally, the test sessions have been randomly sampled as to only keep the first 50 to 100% of the their viewed items in order. This poses a problem for sequential and recurrent architectures which are trained to predict the next item in the succession until the purchased one, as the real progression is not always available and the assumption of sequentiality of the model fails.

Each test session must be independently considered at prediction time, i.e., the prediction process for a single test session cannot exploit information about other test sessions. Moreover, test sessions must exclusively be used at inference time as inputs and cannot be employed to train models.

### 2.2 Data Analysis

Analyzing data in depth is crucial for this type of competition. For the case at hand, the analysis was carried out on four fronts: anomaly detection, feature analysis, session analysis, and seasonalities.[3]

The majority of purchases are made in the evening while the months with more purchases are November 2020 and May 2021. The average duration of a session in terms of number of viewed items was found to be around 4.7, with a minimum of 1 and a maximum of 100. A small number of sessions - around 1.38% for the test month - displays an interesting trend: they only contain views of articles which were never viewed or purchased before. These "cold" sessions required special care, as they could not easily be compared to other sessions. Additionally, only 0.02% of the

[1] https://www.recsyschallenge.com/2022/
[2] https://github.com/KingPowa/Rec_Sys_2022_Boston_Team.git
[3] The notebook containing the most relevant data analysis is located in the "Notebook" section of the GitHub repository

Table 1. Data splits used in our experiments.

| Split | Date Interval | Num. of Sessions |
|---|---|---|
| Training | 01/01/2020 - 31/04/2021 | 918 382 |
| Validation | 01/05/2021 - 31/05/2021 | 81 618 |
| Leaderboard and Final Test | 01/06/2021 - 30/06/2021 | 50 000 |

items have no interaction at all in the dataset. The lack of information about users and their characteristics meant it was not possible to distinguish among different types of customers, e.g., a spendthrift customer from a stingier one.

### 2.3 Notation

For the rest of the paper the following notation is used: $\mathcal{I}$ set of possible items, $i$ a generic item; $\mathcal{A}(i) = \{(c_{i_0}, v_{i_0}), ...\}$ a set of attributes of item $i$, where $c_{i_j}$ is a *category* index and $v_{i_j}$ is its *value*; $\mathcal{F}$ set of possible seasons, $f$ a generic season; $\mathcal{S}$ set of possible sessions, $s = \{i_0, i_1, ..., i_n\}$ a generic session; $n_s$ number of items in session $s$; $Q(i, \cdot)$ number of interactions for item $i$; $Q(i, f)$ number of interactions for item $i$ in season $f$; $URM$ a binary matrix in which each cell $URM(s, i)$ has a value of 1 if session $s$ contains *at least* one view or purchase associated with item $i$, 0 otherwise.

## 3 DATA SPLITTING AND PRE-PROCESSING

The full dataset consists of 18 months of data, from January 2020 to June 2021. In particular, the first 1M sessions compose the training split and the last 100k compose the two test splits (leaderboard and final). Since the test splits span all and only the entire month of June 2021, the straightforward idea that allowed to have a local test set similar to the real ones, was to use all and only the sessions in the month of May 2021 as a validation set. Details of each split are shown in Table 1.

### 3.1 Interaction Weighting

In order to provide the models with a more meaningful representation of the data, the following interaction weighting strategies were developed:

- *Views-purchases distinction*: views are weighted with a value $\alpha \in (0, 1)$.
- *Cyclic decay*: a sinusoid with frequency of $\frac{1}{365}$ is used to give a bigger weight to those interactions that are in a period of the year that is closer to the reference timestamp[4].
- *Exponential decay*: an exponential function is used to reduce the weight of the interactions as the distance of these increments with respect to the reference timestamp[4].

The rationale behind the deploying of the first strategy is that the majority of the models used in the final ensemble were not suitable to make a distinction between views and purchases. This is due to their usage of the $URM$, which also limits the models to take advantage of the temporal correlation between the session views. Additionally, it has to be taken into account that the fashion realm is a field in which the tastes of the customers are in continuous evolution, with different periods that often present far different tendencies on the purchased products [5]. This could lead the models, in some cases, to learn *noisy patterns* for the reference period. This two latter issues justify the deployment of the second and the third strategies.

[4]the used reference was typically the starting date of the test set

## 4 FEATURES

The feature engineering phase was of the utmost importance to achieve the best performance of our models. Given the dataset provided by Dressipi, both an elaboration of the basic features and the extraction of some custom attributes were performed. In the following sections the most important features are presented.

### 4.1 Item Features

A very specific taxonomy was available in the dataset as content data, where each item $i$ was decorated with an attribute set $\mathcal{A}(i)$. This representation was devoid of semantics, in fact both category and value were represented by integers thus it was not possible to infer any specific meaning from them except of the presence or absence of the specific couple. A further complication was given by the possibility for an item to have multiple values for the same category, formally:

$$\exists i_j, i_k : \{(c_{i_j}, v_{i_j}), (c_{i_k}, v_{i_k})\} \in \mathcal{A}(i) \ \wedge \ c_{i_j} \neq c_{i_k} \ \wedge \ v_{i_j} = v_{i_k}$$

The set of unique couples $(c_{i_j}, v_{i_j})$ was 904. In order to manage such number of couples a *multi-label-encoding* (MLE) [7] approach was adopted. Each of the available tuples was translated into a feature, leading to represent each item with a vector of 904 elements expressing the presence (1) or absence (0) of the specific pair.

### 4.2 Booster Features

Additional features were needed to enhance the ranking process, e.g., features related to seasonality traits. At the same time, these features needed to have low correlation and a tractable dimension. The term *booster features* is used to refer to this set of features, which was used in the booster architecture (described in Section 5.2). Firstly the dimensionality of the basic attributes was reduced adopting a *Variational Autoencoder* [12] with a latent space of size 32. This model was trained to minimize the reconstruction error of the *MLE vector*, using as regularizing factor the *Kullback–Leibler Divergence* [9], as optimizer *Adam* [11], and a train-validation split of 80% of items for the train set and 20% of items for the validation set. Once obtained the model, it was possible to retrieve the *embeddings* for each item. We then explored way to enrich session representation in the booster, as noted at the start of the section. A first, naive approach consisted in aggregating through a sum the embeddings of the items belonging to the same session. Another type of representation was instead obtained through a *Recommender Variational Autoencoder* (RecVae) [19] model, optimised on the top-N recommendations task. After a training procedure, each different session was used as an input of the network, and the output of the encoder was extracted and used as the additional representation of the sessions. Additionally, for each session-item candidate, the *scores* provided by our basic models were inserted, range-normalized in an intra-session fashion. As final attributes, the seasonal tendencies of each item $i$ were added, encoded as a continuous value ranging from 0 to 1, considering distinctly in the computation the concept of view and purchase. These attributes express the tendency of an item to be purchased or viewed in a specific season or subset of seasons (values close to 1) or to be all-seasonal (values close to 0). This seasonal tendency was computed leveraging the concept of *entropy* of a random variable [18]. The seasonal tendency $ss_{tend\_total}$ can be expressed as:

$$ss_{tend-total} = 1 - \sum_{f \in \mathcal{F}} ss_{tend}(f) \log_{|\mathcal{F}|} ss_{tend}(f)$$

where $ss_{tend}(f)$ indicates the specific season tendency, computed as the portion of purchases or views over the total:

$$ss_{tend}(f) = \frac{Q(i, f)}{Q(i, \cdot)}$$

## 5 PROPOSED SOLUTION

The solution consisted in two steps: first, multiple recommender models were optimized to perform top-N recommendations, then for each session a selection of the best candidate items of each model was merged and ranked through Gradient Boosting. Finding the optimal hyperparameters of each model is a fundamental step to maximize recommendation quality. For all algorithms, *Bayesian Optimization* was performed using the Optuna [1] framework, followed by the optimization of the interaction weighting parameters as described in Section 3. Additionally, each parameter combination for LightGBM was evaluated using 3-fold cross validation.

### 5.1 Candidate Selection

Several baseline models were tested to produce the initial candidates. K-Nearest Neighbors (KNN) models were fast to train and provided satisfying results. A hybrid item-based KNN taking advantage of both collaborative and content based information (ItemKNN-CFCBF) and a user-based collaborative KNN (UserKNN-CF) were built. For the latter, extra care was taken to ignore the similarity information among sessions belonging to the test set, in order to comply with the rules of the competition. A purely content-based item KNN was also trained specifically on the sessions in which the collaborative information was lacking. Graph-based models were also tested with a special focus on RP3Beta [2]. Another group of baseline models that were trained are autoencoders, specifically EaseR [20], RecVAE [19], and MultVAE [14]. Finally, a simple non-personalized algorithm which recommends the items with the highest number of interactions was added (TopPop). Each baseline model is able to produce a *relevance score* for each recommended item. The nature and scale of this metric depends on the structure and inner mechanics of the specific algorithm: for collaborative and content-based models, the score is given by computing the dot product between the session profile and the similarity matrix calculated at training time; for GRU4Rec, RecVAE and MultVAE the score is the predicted preference of the items, i.e., the likelihood for each item of being the next in the session; for the TopPop recommender, the score is simply the popularity of the item, i.e., the number of unique interactions with the item across the dataset.

*5.1.1 GRU4Rec.* Recurrent Neural Networks (RNNs) are well suited to model complex dynamics of sequential event sessions [15]. For this reason, GRU4Rec [6] played a key role in our experiments. Such approach exploits the Gated Recurrent Unit (GRU) to incrementally learn the succession of events in variable-length series and to predict the subsequent ones. The best configuration found led to a final architecture composed by a single *GRU layer* with 122 units, preceded by an *embedding layer* of 98 units with a minimum *dropout* of 0.05 to prevent the network to overfit the training data. The introduction of the embedding layer slightly improved the performance, whereas adding other GRU layers provided far worse results, as well as increasing the layer dimension. The model was trained for only 6 epochs by using *BPR-max* loss function [17]. Being a pairwise ranking loss function, it considers the difference in score between relevant and non-relevant items, therefore it is very suitable for recommendation tasks. Furthermore, it seemed to be the only loss function able to keep the network stable during the training phase. The cold-start problem for items needed to be addressed. At inference time the accepted inputs for the model are sessions including only items already seen during the training phase. For this reason, all the new items that were not in the training data were excluded from the test sessions.

### 5.2 Ranking

Gradient-Boosted Decision Trees (GBDTs) are particularly popular in the Recommender Systems field because of their capacity to work on heterogeneous features and to aggregate results from different sources [8]. In our case, GBDTs

Table 2. Performance and best interaction-weighting hyper-parameters of the base models in the local validation set. Models are presented in descending order based on their MRR. Interaction weighting is not applied to GRU4Rec as it is does not uses the URM.

| Model | MRR Validation | View-Purchase Weight | Uses Cycling Decay | Exponential Decay Weight |
|---|---|---|---|---|
| **GRU4Rec** | 0.17953 | - | - | - |
| **RP3Beta** | 0.15768 | 0.2 | No | - |
| **EaseR** | 0.15518 | 0.5 | Yes | 182 |
| **UserKNN CF** | 0.14962 | 0.2 | Yes | 182 |
| **ItemKNN CF+CBF** | 0.14886 | 0.5 | Yes | 182 |
| **RecVAE** | 0.14748 | 0.5 | Yes | 182 |
| **MultVAE** | 0.13004 | 0.5 | No | 365 |

were trained to solve a *Learning To Rank* (LETOR) task over the recommendations produced by many different baseline models for the purpose of finding their optimal ordering. The *LambdaMART* algorithm [3] was used to maximize the Mean Average Precision (MAP) metric with a cutoff value of 100 to match the scoring conditions of the competition. MAP was chosen since it is strongly correlated with MRR and it is already provided as a metric by the two tested state-of-the-art implementations of GBDTs which were tested: XGBoost [4] and LightGBM [10]. It is worth mentioning that both libraries work with tabular data, therefore the following description will refer to tables, rows and columns. Training the boosting model required five main steps:

(1) The best N baseline algorithms were trained on the training split and used to produce sixty candidate items each together with their relevance scores; the candidates of all models were combined by performing an inner join, discarding duplicate recommendations but keeping all scores on separate columns.
(2) The candidates table was checked against the validation data table to ensure that the actual purchased item for each session was present, otherwise the session was excluded from the training of the GBDT as to make sure each session had at least one positive sample the boosting model could use to learn; finally, the ground truth was inserted as a binary value for each row.
(3) The GBDT model was trained on the candidates table, augmented with session and item attributes, using the optimal hyperparameters found as described in Section 5.
(4) The baseline models were all trained again, this time including the validation month; the candidates were recalculated and then processed as described in step one.
(5) The previously trained GBDT model was used to perform prediction over the new candidates table, again augmented with session and item attributes.

The final prediction was obtained by simply reordering the candidates of each session according to the score assigned by the GBDT model.

## 6 RESULTS

The baseline models provided an initial measure of the complexity of the task. Table 1 presents the overall performance of these models with respect to the local validation set. From the table, neural (GRU4Rec) and graph based models (RP3Beta) stood out with respect to the others. The top performer was GRU4Rec, which exhibited also an interesting behaviour with respect to sessions of length 1. In fact, in our local validation set, it showed promising results (MRR $\sim$ 0.23) in these specific sessions, although they can be considered among the most complex to predict. Unfortunately it was not possible to translate such a result in the test set due to its structure (spurious sessions of length 1 created by the random cutoff). The use of our interaction weighting strategies was effective as it generally led to accuracy increases

Table 3. Performance of the boosters both in the local validation set and in the public test set

| Model | 3-Fold CV MAP (Validation) | MRR (Public Leaderboard) |
|---|---|---|
| LightGBM | 0.49115 | 0.18800 |
| XGBoost | 0.46390 | 0.18347 |

with respect to the traditional use of the URM. The most accurate and easy-to-implement interaction weighting strategy was *views-purchases distinction*. Combining this strategy with *cyclic* and *exponential decay* generally improved the accuracy further. Our most accurate URM-based models (EaseR, UserKNN, ItemKNN, MultVAE, RecVAE) used the three strategies: views-purchases distinction, cyclic, and exponential decay. RP3Beta was the exception, as it only used views-purchases distinction and did not achieved higher accuracy by including time-based decays. Interestingly, the best weight for a given interaction-weighting strategy vary per model. For instance, RP3Beta and EaseR use two different weights: 0.2 and 0.5, respectively. These differences in each strategy between the models suggest that tuning these strategies on each model is required.

Regarding the second stage of the pipeline, which involved the usage of GBDTs as stated in Section 5.2, our final choice has been LightGBM, due to its faster training and superior accuracy with respect to XGBoost as shown in Table 2. It was also computed the feature importance of each attributes involved in the final model, in order to check its relevance.[5] The results show a balance between features, meaning that all of them contributed mostly equally in final ranking, with a slight preference for the scores of the baseline models. This suggests that the GBDTs model interpreted the scores as knowledgeable experts in specific contexts, selecting which ones to trust more according to the recommendation scenario of interest.

In the first evaluation round our approach got a score of about 0.1845 which put us in 29th place in the final leaderboard. Unfortunately, due to a technical mistake in our submission, we achieved in the second evaluation (after the re-opening of the challenge) a lower score and position in the leaderboard.

## 7 CONCLUSIONS

The ACM RecSys Challenge 2022 consisted in predicting the purchased item of anonymous sessions composed by the sequence of interactions with items. Starting from the available data, two set of features related to the items and to sessions were extracted both with ad-hoc models and feature engineering. A GBDT model was used to provide final recommendations. This model combines the candidates selected the baseline recommendation models alongside their features. Our two-stage model was able to efficiently and effectively recommend accurate fashion items, resulting in a simple and scalable solution. Our results indicate that all features, including but not limited to session and seasonal features, are important to provide accurate recommendations in this domain. Also, weighting interactions on baseline models contributes positively in the accuracy of them. These weighting strategies differentiate types of interactions and include seasonal aspects into these models.

.

## ACKNOWLEDGMENTS

We would like to thank Prof. Paolo Cremonesi for his support.

[5]Results ommited due to space limitations.

## REFERENCES


[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis (Eds.). ACM, 2623–2631. https://doi.org/10.1145/3292500.3330701

[2] Cesare Bernardis, Maurizio Ferrari Dacrema, and Paolo Cremonesi. 2018. A novel graph-based model for hybrid recommendations in cold-start scenarios. *CoRR* abs/1808.10664 (2018), 2 pages. arXiv:1808.10664 http://arxiv.org/abs/1808.10664

[3] Christopher J. C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report. Microsoft Research. http://research.microsoft.com/en-us/um/people/cburges/tech_reports/MSR-TR-2010-82.pdf

[4] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi (Eds.). ACM, 785–794. https://doi.org/10.1145/2939672.2939785

[5] Yashar Deldjoo, Fatemeh Nazary, Arnau Ramisa, Julian J. McAuley, Giovanni Pellegrini, Alejandro Bellogín, and Tommaso Di Noia. 2022. A Review of Modern Fashion Recommender Systems. *CoRR* abs/2202.02757 (2022), 35 pages. arXiv:2202.02757 https://arxiv.org/abs/2202.02757

[6] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). 10 pages. http://arxiv.org/abs/1511.06939

[7] Mayoore S. Jaiswal, Bumsoo Kang, Jinho Lee, and Minsik Cho. 2019. MUTE: Data-Similarity Driven Multi-hot Target Encoding for Neural Network Design. *CoRR* abs/1910.07042 (2019), 9 pages. arXiv:1910.07042 http://arxiv.org/abs/1910.07042

[8] Dietmar Jannach, Gabriel de Souza Pereira Moreira, and Even Oldridge. 2020. Why Are Deep Learning Models Not Consistently Winning Recommender Systems Competitions Yet?: A Position Paper. In *RecSys Challenge '20: Proceedings of the Recommender Systems Challenge 2020, Virtual Event Brazil, September, 2020*. ACM, 44–49. https://doi.org/10.1145/3415959.3416001

[9] James M. Joyce. 2011. Kullback-Leibler Divergence. In *International Encyclopedia of Statistical Science*, Miodrag Lovric (Ed.). Springer, 720–722. https://doi.org/10.1007/978-3-642-04898-2_327

[10] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). 3146–3154. https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html

[11] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). 15 pages. http://arxiv.org/abs/1412.6980

[12] Diederik P. Kingma and Max Welling. 2019. An Introduction to Variational Autoencoders. *Found. Trends Mach. Learn.* 12, 4 (2019), 307–392. https://doi.org/10.1561/2200000056

[13] Sara Latifi, Noemi Mauro, and Dietmar Jannach. 2021. Session-aware recommendation: A surprising quest for the state-of-the-art. *Inf. Sci.* 573 (2021), 291–315. https://doi.org/10.1016/j.ins.2021.05.048

[14] Dawen Liang, Rahul G. Krishnan, Matthew D. Hoffman, and Tony Jebara. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*, Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis (Eds.). ACM, 689–698. https://doi.org/10.1145/3178876.3186150

[15] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-Aware Recommender Systems. *ACM Comput. Surv.* 51, 4 (2018), 66:1–66:36. https://doi.org/10.1145/3190616

[16] Dragomir R. Radev, Hong Qi, Harris Wu, and Weiguo Fan. 2002. Evaluating Web-based Question Answering Systems. In *Proceedings of the Third International Conference on Language Resources and Evaluation, LREC 2002, May 29-31, 2002, Las Palmas, Canary Islands, Spain*. European Language Resources Association, 4 pages. http://www.lrec-conf.org/proceedings/lrec2002/sumarios/301.htm

[17] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009*, Jeff A. Bilmes and Andrew Y. Ng (Eds.). AUAI Press, 452–461. https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1630&proceeding_id=25

[18] Claude E. Shannon. 1948. A mathematical theory of communication. *Bell Syst. Tech. J.* 27, 3 (1948), 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

[19] Ilya Shenbin, Anton Alekseev, Elena Tutubalina, Valentin Malykh, and Sergey I. Nikolenko. 2020. RecVAE: A New Variational Autoencoder for Top-N Recommendations with Implicit Feedback. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, James Caverlee, Xia (Ben) Hu, Mounia Lalmas, and Wei Wang (Eds.). ACM, 528–536. https://doi.org/10.1145/3336191.3371831

[20] Harald Steck. 2019. Embarrassingly Shallow Autoencoders for Sparse Data. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia (Eds.). ACM, 3251–3257. https://doi.org/10.1145/3308558.3313710

[21] Ellen M. Voorhees and Dawn M. Tice. 2000. The TREC-8 Question Answering Track. In *Proceedings of the Second International Conference on Language Resources and Evaluation, LREC 2000, 31 May - June 2, 2000, Athens, Greece*. European Language Resources Association, 8 pages.

http://www.lrec-conf.org/proceedings/lrec2000/html/summary/26.htm

[22] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z. Sheng, Mehmet A. Orgun, and Defu Lian. 2022. A Survey on Session-based Recommender Systems. *ACM Comput. Surv.* 54, 7 (2022), 154:1–154:38. https://doi.org/10.1145/3465401