IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx xx, 2022, date of current version October 11, 2022.

Digital Object Identifier

EXPLORATION IN SEQUENTIAL RECOMMENDER SYSTEMS VIA GRAPH REPRESENTATIONS

DMITRII KISELEV [1,3] *orcid* **and ILYA MAKAROV** [2,3]
[1]HSE University, Moscow, Russia
[2]NITU MISIS, Moscow, Russia
[3]Artificial Intelligence Research Institute, Moscow, Russia

Corresponding authors: Dmitrii Kiselev (e-mail: dkiseljov@hse.ru).

ABSTRACT Temporal graph networks are powerful tools for solving the cold-start problem in sequential recommender systems. However, graph models are still susceptible to feedback loops and data distribution shifts. The paper proposes a simple yet efficient graph-based exploration method for the mitigation of the aforementioned issues. It adopts the counter-based state exploration from reinforcement learning to the bipartite graph domain. We suggest a method that biases model predictions using Rooted PageRank towards locally unexplored items. The method shows competitive quality on the popular recommender systems benchmarks. We, also, provide an extensive qualitative analysis of experiment results and recommendations for the use of our method in production applications.

INDEX TERMS exploration, fine-tuning, graph neural networks, gnn, graphs, interactive recommender systems, intrinsic motivation, online adaptation, pretraining, self-supervised, recommender systems, recsys

I. INTRODUCTION

Nowadays, recommender systems drive a lot of businesses from e-commerce to social media [1], [2]. Such models allow to improve user experience by reducing users' time spent finding options most relevant to them. Classic models aim to find such options based on prior user interactions. However, in most cases recommender models operate in a very dynamic environment [3]: people create a lot of new content, clothes are anchored to the season, and preferred music depends on mood and situation. Thus, historical interactions could be rendered irrelevant in the present.

Moreover, recommender systems are interactive models. They affect user behavior because they restrict the visible set of items. On the other hand, the interactive nature of models allows us to receive almost immediate feedback about items that may possibly fit the changed interests of the user, and adapt to these changes in an online fashion. Multi-armed and contextual bandits and reinforcement learning (RL) based approaches were applied to solve this issue [4]. Both of them use exploration strategies to propose unseen or uncertain states. The state of the recommender system can be represented as a set of items previously interacted with by a user, or as a user-item interaction graph for all users at the given moment.

Recently, graph neural networks (GNN) have been adopted to build recommender systems [5]. In comparison to the classic matrix factorization methods, they consider high-order proximity between users and items [6]. Furthermore, modern graph-based methods can work with the user, item and context information [7], [8] and temporal information [9], [10]. Such models show high performance for user-item interaction graphs even for previously unseen nodes [11].

The exploration-exploitation trade-off is well-studied for classic analytical models like multi-armed and contextual bandits. Nevertheless, such strategies are hard to apply to an arbitrary recommender model [4]. Therefore, the goal of this paper is to develop a new exploration technique specifically for graph-based recommender systems.

The main contribution of the paper is the novel exploration strategy that is based on the ideas of the Rooted PageRank [12]. It estimates the local item popularity and drives the model to recommend the most uncertain items at the moment. The proposed method shows competitive quality on the problem of online model adaptation.

The paper is structured as follows. Firstly, we explain background knowledge. Then, we describe the proposed methods in detail. Next, the experiment methodology is presented. Finally, we discuss obtained results and conclude

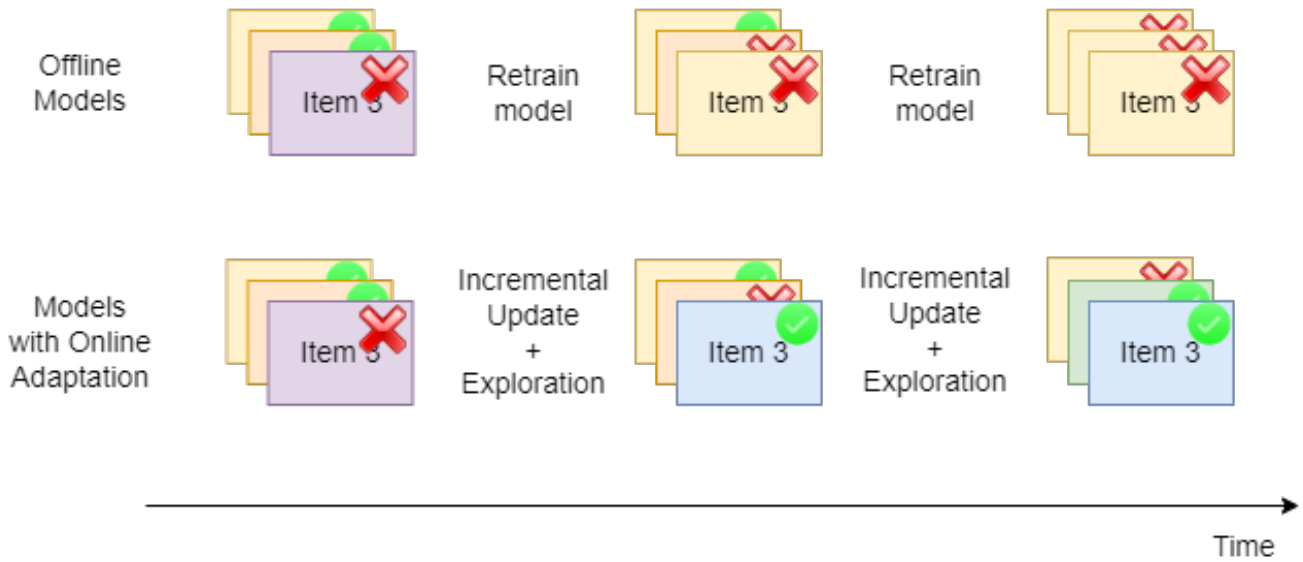

FIGURE 1: Difference between classic offline models and models with online adaptation. Similar color represents similar items. Classic offline models (the top row) require full retraining to adapt to changes in user behavior. Moreover, such retraining could lead to concentration on a specific topic. On the other hand, online models (the bottom row) benefit from exploration with faster adaptation to any behavioral changes and provide more diverse recommendations.

our findings.

II. RELATED WORK

The section describes the previous experience in the relevant fields for the study. We start by explaining modern methods for exploration to handle data distribution shifts and feedback loop problems. Next, we describe the temporal graph networks as an efficient method for sequential recommendations and cold-start problems.

A. EXPLORATION TECHNIQUES

The exploration problem can be considered under two major perspectives: optimism under uncertainty and intrinsic motivation. The general idea of the first group of methods is to estimate distributions of predictions and discriminate between high-mean (good performance) and high-variation (lack of data). The second group applies self-supervised learning to bias the model towards unexplored regions.

A classic approach to the problem of exploration is the multi-armed bandit model. One of the simplest strategies is ϵ-greedy [13]. It exploits model prediction with decreasing probability $1 - \epsilon$ and chooses items uniformly at random with probability ϵ. Due to its simplicity, this model can be used with an arbitrary estimator. More complex methods like contextual bandits are usually based on the LinUCB [13], [14]. It is hard to apply to an arbitrary model because it requires an analytical solution of an underline estimator. Also, it learns different models for different arms which leads to high computational complexity.

The application of optimism under uncertainty requires the estimation of outcome distribution. One of the most common methods for this task is Variational Autoencoders (VAE) [15], [16]. It learns the mean and variance of input in some latent space. We can use it to sample vectors for users and items. To reduce the complexity of the problem, we can limit it to just the output of the encoding of state.

Intrinsic motivation is a popular technique from reinforcement learning [4], [17]. The general idea is to use some additional intrinsic reward for our model to add exploration. One of the main approaches is to estimate the novelty of the state, for example, by counting the number of visits to it [17]. The extension of such an approach is the random network distillation strategy [18]. Instead of counting it uses the idea of the inability to properly predict the value of some complex function in unfamiliar regions. This idea could be also applied to the direct algorithm task. If it has a high prediction error of the next state, then it is not sufficiently explored [19]. Another group of methods preserve the buffer of previous actions and states to estimate novelty relative to them [20]–[22].

B. DYNAMIC GRAPH REPRESENTATION LEARNING

Dynamic graphs can be split into two general groups: discrete-time dynamic graphs (DTDG) and continuous-time dynamic graphs (CTDG) [23]. The first group of methods works with a series of discrete snapshots of a full graph over the fixed timedelta. The difference between two consecutive graphs is defined by the set of all added or removed edges and nodes from the previous graph between two discrete timesteps. The second group of methods treat graph update events (node or edge addition or removal) continuously. This work concentrates on the second type of graph due to its ability to catch each new user-item interaction event in an

online fashion.

Most dynamic graph embedding methods can be generalized by the TGN framework [11]. It introduces a flexible set of modules: memory and memory update procedures, message function, message aggregator and final embedding layer. The main difference between methods lies in memory update procedures and embedding layers. TGAT [24] use no memory and apply an attention mechanism to aggregate node neighborhood. JODIE [9] uses time weighting of memory state to receive embedding. CAW [25] employs an anonymization technique in order to learn final node and edge embeddings, and does not include a memory module. A method described in [26] takes the memory and embedding layer from the original TGN and augments it with CAW vectors. APAN [27] extends the memory to preserve all events from k-hop neighborhood of a vertex.

C. GRAPH NEURAL NETWORKS FOR RECOMMENDER SYSTEMS

There are a lot of applications of GNNs for specific domains in recommender systems like knowledge-aware [28], [29] or social recommendations [30]–[32]. However, our work is concentrated on a classic sequential formulation of recommender systems. In this case, data is usually represented as a user-item interaction graph [33]–[36]. Neural Graph Collaborative Filtering [6] enhances Neural Collaborative Filtering [37] applying GCN [38] to preserve the high-order proximity. LightGCN [7] removes the non-linearities between the GCN layers to ease computational complexity and improve scalability. PinSAGE [8] proposes importance sampling for items to reduce the number of node encodings in aggregations. Additionally, they utilize the Personalized PageRank [12] to sample the hard negative examples. MCCF [39] apply the attention mechanism to aggregate the neighbors' information. XSimGCL [40] propose simple augmentation strategies for graph-based recommendations to improve overall model quality. Some works [41]–[43] preserve temporal structure by building graph differently. They define nodes by items and connect them if a consecutive positive event was performed. Authors of [44], [45] propose to build a hyper-graph of user sessions to account for inter-session correlations. The approach described in [46] applies the methodology of graph embedding to continuous-time dynamic graphs described above in order to solve the recommender systems problem.

III. MODEL

The general idea of the paper is to utilize a user-item interaction graph structure for exploration purposes. In this section, we will describe how the graph is constructed, and how it is encoded to provide recommendations and establish exploration. The code of model and experiment pipeline is available on Github [1].

[1] https://github.com/mkiseljov/graph-based-exploration-access

A. GRAPH CONSTRUCTION

Following the TGN, we treat the graph updates continuously considering each event independently. These events have the following information:

- Source node (user)
- Destination node (item)
- Node (both user and item) features
- Edge features (if presented)
- Timestamp

The event defines the addition of edge between nodes. So, it is important to carefully design how such events occur. Our goal is to build an interactive recommender system that is able to adapt to user decisions. However, the item recommendations from the novel model may not ideally fit the historically shown items. So, we propose to use for learning not all historical edges to the graph but only edges with positive feedback from users during our simulation.

We can summarize the process of graph construction during the experiments in the following steps:

1) Initialize empty graph
2) Recommend some items to the user
3) Add edges to the graph if the user positively reacts (click, like, purchase and so on) to the item.
4) Repeat steps 2-3 while historical data is presented

B. NODE ENCODER

To encode nodes, we use a modified TGN model because it shows high performance in temporal link prediction task and generalizes a lot of other temporal graph embedding methods. The TGN model can be decomposed into two parts. The first part is about how it tracks the memory of the node: it receives the batch of messages (node pairs), encodes it (with identity by default), preserves only the last message for each node and updates the memory using GRU. The second part is how TGN predicts novel links. It takes a pair of nodes and builds embeddings for each one based on the current node features, its memory and time from the last message. Then, it updates the embedding with graph-layer (aggregation of embeddings over node neighbourhood). Finally, received node embeddings are concatenated and edge existence probability is calculated using a fully-connected network.

The first modification of the TGN is that we apply graph layer only to the user nodes. The TGN model aims to encode a small number of node pairs to perform binary classification tasks. However, in the recommender systems, we need to score a large number of items. So, in the case of direct application of TGN, we will need to calculate the graph embeddings for all items online. This requires too much time and computational resources.

The second modification is the replacement of the binary classification output layer with the top-k items recommendation. We find top-k items balancing the exploration and exploitation scores. The exploitation score is calculated as a simple dot product between node embeddings. The exploration score is described in the subsection below. This

modification is essential to fit the recommender systems' requirements.

C. EXPLORATION MODULE

Representation of recommender systems as a graph problem gives us a convenient view of the user-item interactions. We consider that k-hop neighborhood depicts a set of user interests. In these terms, we can state exploration as a problem of reaching items outside this set.

The exploration strategy is inspired by Rooted PageRank (RPR) [12] and its application to hard negative sampling in the PinSAGE model [8]. Rooted PageRank can be considered as a similarity measure between root and other nodes. The authors of PinSAGE propose to use moderately similar nodes as hard negative examples. Such nodes are close enough to the original root but not selected by the user. We employ this idea to perform exploration.

The RPR selects the root node (user node in our case) and samples random walk from it until restart from the root with predefined probability. Then, it calculates node occurrences. However, we want to preserve the causality in the user actions. So, we sample temporal random walks instead of ordinary ones. Also, recommender graphs usually are power-law graphs and, therefore, dense. Thus, we need to sample only a few nodes via random walks, otherwise, the collected statistics will be close to the uniform distribution. Overall, the proposed exploration strategy can be summarized in the following steps:

1) Select the user as a root node
2) Sample several (100) fixed size (length 3) temporal random walks starting from the root node
3) Count the occurrence
4) Bias TGN predictions towards nodes with the smallest occurrence count

D. TRAINING OBJECTIVE

Our goal is to validate how the model and exploration strategies work in interactive environments. However, real data is already biased towards some recommender strategies. Also, historically shown items may not match the model proposition to the user. To overcome these issues we propose to utilize causal inference techniques [47].

We employ two counterfactual evaluation methods aimed to solve the mentioned problems with historical data. Firstly, we utilize the Replay [48] method. The general idea of it is to skip the observations where recommendations do not intersect with the historical slate for the given user. This is essential because if we penalize the model for the wrong prediction since it was not presented in historical data, then we will converge to the data collection strategy, e.g. previous recommendation system. However, this method works only for data sampled uniformly at random. Inverse probability weighting (IPW) [49] reduce estimate bias by making observation distribution in the treatment group similar to the general sample. Basically, it resamples observations according to the inverse probability of its observation. Such a procedure flattens occurrence distribution to the uniform. To provide unbiased estimates we fuse both methods for counterfactual evaluation. We use it to weight binary cross-entropy (BCE) loss between the observations.

1) Select user-item interactions batch by the time
2) Predict top-k items following the exploration and exploitation strategies for the users in the batch
3) Select the items from predictions that were historically shown (any event positive or negative) to the user
4) Calculate BCE loss for the selected user-item pair (omit other pairs from loss calculation)
5) Weight the user-item pair loss with IPW
6) Calculate weighted by IPW average of the user-item pairs' losses
7) Propagate loss

E. CONNECTION TO THE EXISTING METHODS

The proposed exploration method is most closely related to the buffer-based intrinsic motivation strategies. Such methods follow the idea of novelty estimation but with respect to the recent states in the buffer. [20] gives a reward if the agent is far away from states in the buffer. [21] trains the discriminator between current and previous states. If it fails to discriminate, then we do not have enough information about the new state. Some models use sampling from a buffer to measure the state novelty, which requires saving a large number of previous states.

These methods of exploration are applied for the classic reinforcement learning scenarios where the state represents the position of the agent within some environment. However, we aim to apply methods within the recommender systems framework. Here, we can consider the state as the current user-item interaction graph. Let us note that in such a definition the difference between consecutive states is represented only by the addition of one or more links between the user and items. Thus, we can consider the difference between states as some measure over links. The Rooted PageRank can be considered as a similarity measure between nodes that estimate how popular some node is in the local region of the root one. Moreover, temporal random walks give us access to the several previous states of the graph because each sampling step accounts only for edges drawn before. Thus, the proposed method is the application of the existing buffer-based exploration methods from RL to the graph-based recommender systems with novel similarity function and adaptation to the different state representation.

IV. EXPERIMENT SETTINGS

Basically, our goal is to evaluate how different exploration strategies affect the performance of online model adaptation to the new observations. In the next subsection, we describe how we formulate a training and evaluation pipeline for this task. Next, we explain the metrics. Finally, we present the datasets.

A. TRAINING AND VALIDATION

We split datasets into three parts:

1) Pretrain part. We use it to pretrain the TGN encoder using a standard binary classification pipeline with random negative sampling.
2) Validation for pretrain part. We use it to estimate pretraining phase performance and apply an early-stopping strategy to eliminate the over-fitting issue.
3) Online simulation part. We use it to run interactive recommender experiments following the logic described in the subsection III-D

The balance between the parts is $50\% - 10\% - 40\%$. Such balance was selected to save sufficient time horizon for an online simulation experiment while preserving the ability to pretrain the model.

On all parts, we sample batches in time order. The batch size is set to 200, and hyperparameters for the TGN encoder are set to default provided by the authors of [11].

B. METRICS

To estimate the quality of our models we calculate hit rate@k ($HR@k$) for slated recommendations [50]. We use k equal to 10 for both datasets. The hit rate shows whether our recommendations were valid for a specific user at a specific point of time. As described before, we de-bias all metrics with replay and IPW procedures. We do not use any ranking metrics because in our data we consider only binary feedback for a single clicked element.

C. DATA

TABLE 1: Datasets statistics

	MovieLens	LastFM-1b
Number of observations	1000210	6799895
Number of users	6040	18437
Number of items	3706	9783
Average user degree	95	179
Average item degree	163	336
Positive label ratio	0.5752	0.4841

Our problem supposes the temporality of the data and the existence of entity features, so we select the following datasets.

Movielens-1m [51]. The dataset contains one million user ratings for specific movies. Preprocessing consists only of creating an item feature vector as the one-hot encoding of genres. User features were represented by gender and age group. This dataset is a commonly used benchmark for recommender systems. Generally, it has rating event time, so, we are able to train sequential models on them. However, the temporal structure in this dataset is corrupted due to the logic of the rating assignment. Users can rate items at a random point in time, so the order of watches was not preserved. To use a similar classification approach in all datasets we transform labels to the binary scale. If the rating is less than 4, we suppose that film was not liked by the user, so it is assigned with a zero label. Otherwise, we believe that ratings 4 and 5 represent the positive emotions of the user.

LastFM-1b [52]. The dataset contains one billion listening events of different users. Due to the size of the dataset, we consider the albums as items to reduce the prediction space. Also, we do not consider the problem of repeated consumption in the scope of this paper, so we remove the user-item event repetitions saving only the first event. We select a random subsample of users and remove inactive ones to further reduce the size of the dataset. As an implicit label, we take the indicator of whether the number of listen events for a specific item is greater than 4 or not. The reason is two-fold: repeated consumption means that the user possibly likes this item and at the 4 repetitions we achieve the almost ideal balance between targets in our subsample.

D. BASELINES FOR EXPLORATION

To evaluate our model we select two baselines. The first two strategies allow us to understand whether complex exploration strategies advance model performance. VAE was chosen as a method that is able to infer the distributions and apply optimism under uncertainty logic. RND strategy also follows the idea of the self-supervised novelty estimation that is under our consideration.

Dot product. The basic scenario is to use only a dot product over node embeddings after TGN encoding without any extra exploration strategies. It shows the performance of the modified TGN.

Epsilon-greedy [17]. Another straightforward strategy is an ε-greedy. It recommends uniformly at random items with probability equal to ε. Otherwise, it behaves similarly to the dot product.

Variational Autoencoder (VAE). [15], [16] It implements the ideas of optimism under uncertainty and is close to the idea of Thompson Sampling. This strategy applies the VAE to the TGN node embeddings. Two different networks (to recover the mean and standard deviation of latent representations) compress the original node embeddings to the vectors with lower dimensions (half of the original size). Then these vectors are used to sample new latent vectors for nodes. Finally, dot product encoding is applied to the latent node embeddings. While training the reconstruction loss is also propagated to train the VAE part.

Random Network Distillation (RND) [18]. We choose this method as a common baseline for intrinsic motivation. The fully-connected network with randomly initialized weights takes concatenation of TGN node embeddings as input. The student network with similar architecture is used to reconstruct predictions of the original random network. The error between random and student networks is taken as the measure of state novelty. The method use this novelty estimate to bias the dot product encoder predictions.

V. EXPERIMENT RESULTS

In this section, we provide the results of the experiments and analyze them. We present the Hit rate@k metric (the higher,

the better, lies between 0 and 1). Bold text shows the best quality and underlined shows the second-best option.

TABLE 2: Performance comparison of proposed explorer with other baselines

	MovieLens	LastFM-1b
	HR@10	HR@10
Dot	**0.7226**	**0.4835**
ε-greedy	0.7123	0.4767
VAE	0.6697	0.4149
SSL PageRank	<u>0.7173</u>	<u>0.4809</u>
SSL RND	0.6552	0.4695

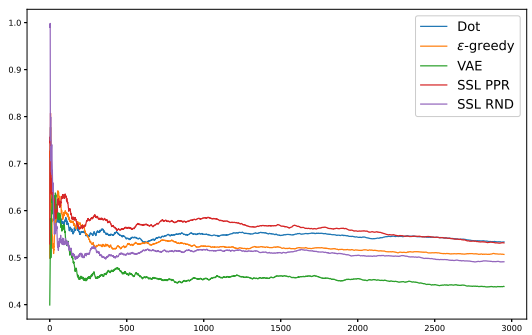

FIGURE 2: Dynamics of hit rate for over batches for LastFM-1b

On both datasets, the dot-product (full exploitation) predictor outperforms the exploration strategies. However, the self-supervised exploration with Rooted PageRank (SSL PageRank) shows the best scores compared to other exploration strategies. It means that penalizing for local item popularity could be enough for exploration purposes. PageRank strategy aims to find locally non-popular points to create a link. Such points are still close to the user but could be slightly different in their representation. This scenario means that after the network embedding procedure, the user vector will be smoothly changed.

PageRank-based exploration does not introduce the additional loss to the training objective in comparison to the Random Network Distillation (SSL RND) and Variational Autoencoder Approaches (VAE). SSL RND adds the MSE loss to train the distilled network for exploration purposes. However, the network takes the node representations from the TGN encoder. So, this intrinsic loss has an effect on the initial encoder weights and biases the model towards worse optima. The logic for VAE is similar, in that it tries to reconstruct the node embeddings encoded by TGN and also propagates the reconstruction loss for it.

The ε-greedy strategy selects the item uniformly at random when explores. However, due to the power-law nature of recommender datasets, such a procedure leads to a high probability of selecting the wrong item. So, it shows worse performance in comparison with the default strategy. In our method, short temporal random walks samples pretty close items for the user. It helps to overcome the issue.

Figure 2 shows the dynamics of the HR@10 metric over the batches. The figure is truncated to the first three thousand batches to align plots for all exploration techniques. The difference in the number of batches is induced by the replay procedure which skips the part of samples. One can find that after some point performance of the SSL PPR model starts decreasing in contrast with other methods whose dynamics are still positive. The main reason for that is the quality of sampled temporal random walks. Before the graph becomes too dense random walks hit the same item a lot. However, in the dense graphs distribution of occurrences becomes close to uniform. So, in this case, the model is unable to properly estimate the local popularity of the item.

Experiment results show that graph-based exploration methods efficiently adopt self-supervised intrinsic motivation ideas from reinforcement learning and perform competitively to other exploration strategies. The quality of the proposed method becomes more substantial when nodes have a relatively low degree and high diameter because it allows omitting over-smoothing over the k-hop similarity.

VI. CONCLUSION

The paper provides a new exploration strategy: Rooted PageRank for local popularity estimation. We apply it to study the online adaptation of sequential recommender system models.

Explained results show the importance of exploration techniques for the online model adaptation. Models benefit from different types of exploration if the temporal structure is properly presented. The relative performance of exploration methods depends on data properties. For the graphs with few positive edges, the Rooted PageRank approach proves more effective. In this case, it samples less diverse nodes and can better estimate the local popularity.

Proposed strategy's performance is competitive to the other exploration strategies for recommender systems. In the future, we aim to apply the proposed technique to the repeated consumption scenarios. Further, we aim to study proposed exploration strategies in more complex model pipelines like multi-stage recommender systems. Also, it is important to study the proposed technique for the heterogeneous graphs when items and users have different types.

REFERENCES

[1] G. Adomavicius, J. C. Bockstedt, S. P. Curley, and J. Zhang, "Effects of online recommendations on consumers' willingness to pay," *Information Systems Research*, vol. 29, no. 1, pp. 84–102, 2018.
[2] D. Jannach and M. Jugovac, "Measuring the business value of recommender systems," *ACM Trans. Manage. Inf. Syst.*, vol. 10, no. 4, dec 2019.
[3] S. Wang, L. Hu, Y. Wang, L. Cao, Q. Z. Sheng, and M. Orgun, "Sequential recommender systems: challenges, progress and prospects," *arXiv preprint arXiv:2001.04830*, 2019.
[4] T. Yang, H. Tang, C. Bai, J. Liu, J. Hao, Z. Meng, and P. Liu, "Exploration in deep reinforcement learning: A comprehensive survey," *arXiv preprint arXiv:2109.06668*, 2021.

[5] S. Wang, L. Hu, Y. Wang, X. He, Q. Z. Sheng, M. A. Orgun, L. Cao, F. Ricci, and P. S. Yu, "Graph learning based recommender systems: A review," *arXiv preprint arXiv:2105.06339*, 2021.

[6] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, 2019, pp. 165–174.

[7] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 2020, pp. 639–648.

[8] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, 2018, pp. 974–983.

[9] S. Kumar, X. Zhang, and J. Leskovec, "Predicting dynamic embedding trajectory in temporal interaction networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. Association for Computing Machinery, 2019, pp. 1269–1278.

[10] I. Makarov, A. Savchenko, A. Korovko, L. Sherstyuk, N. Severin, D. Kiselev, A. Mikheev, and D. Babaev, "Temporal network embedding framework with causal anonymous walks representations," *PeerJ Computer Science*, vol. 8, no. e858, pp. 1–27, 2022.

[11] E. Rossi, B. Chamberlain, F. Frasca, D. Eynard, F. Monti, and M. Bronstein, "Temporal graph networks for deep learning on dynamic graphs," *arXiv preprint arXiv:2006.10637*, vol. 2006.10637, 2020.

[12] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web." Stanford InfoLab, Tech. Rep., 1999.

[13] A. Slivkins, "Introduction to multi-armed bandits," *CoRR*, vol. abs/1904.07272, 2019.

[14] W. Chu, L. Li, L. Reyzin, and R. Schapire, "Contextual bandits with linear payoff functions," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 208–214.

[15] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[16] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *International conference on machine learning*. PMLR, 2014, pp. 1278–1286.

[17] A. Aubret, L. Matignon, and S. Hassas, "A survey on intrinsic motivation in reinforcement learning," *arXiv preprint arXiv:1908.06976*, 2019.

[18] Y. Burda, H. Edwards, A. Storkey, and O. Klimov, "Exploration by random network distillation," 2018.

[19] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International conference on machine learning*. PMLR, 2017, pp. 2778–2787.

[20] N. Savinov, A. Raichuk, R. Marinier, D. Vincent, M. Pollefeys, T. Lillicrap, and S. Gelly, "Episodic curiosity through reachability," *arXiv preprint arXiv:1810.02274*, 2018.

[21] J. Fu, J. D. Co-Reyes, and S. Levine, "Ex2: Exploration with exemplar models for deep reinforcement learning," *arXiv preprint arXiv:1703.01260*, 2017.

[22] Y. Kim, W. Nam, H. Kim, J.-H. Kim, and G. Kim, "Curiosity-bottleneck: Exploration by distilling task-specific novelty," in *International Conference on Machine Learning*. PMLR, 2019, pp. 3379–3388.

[23] C. D. Barros, M. R. Mendonça, A. B. Vieira, and A. Ziviani, "A survey on embedding dynamic graphs," *arXiv preprint arXiv:2101.01229*, vol. 2101.01229, 2021.

[24] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Inductive representation learning on temporal graphs," *arXiv preprint arXiv:2002.07962*, vol. 2002.07962, 2020.

[25] Y. Wang, Y.-Y. Chang, Y. Liu, J. Leskovec, and P. Li, "Inductive representation learning in temporal networks via causal anonymous walks," *arXiv preprint arXiv:2105.02315*, vol. 2101.05974, 2021.

[26] I. Makarov, A. Savchenko, A. Korovko, L. Sherstyuk, N. Severin, D. Kiselev, A. Mikheev, and D. Babaev, "Temporal network embedding framework with causal anonymous walks representations," *PeerJ Computer Science*, vol. 8, p. e858, 2022.

[27] X. Wang, D. Lyu, M. Li, Y. Xia, Q. Yang, X. Wang, X. Wang, P. Cui, Y. Yang, B. Sun *et al.*, "APAN: Asynchronous propagation attention network for real-time temporal graph embedding," *arXiv preprint arXiv:2011.11545*, vol. 2011.11545, 2021.

[28] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: a survey," *ACM Computing Surveys (CSUR)*, 2020.

[29] I. Makarov, M. Makarov, and D. Kiselev, "Fusion of text and graph information for machine learning problems on networks," *PeerJ Computer Science*, vol. 7, no. e526, pp. 1–26, 2021.

[30] I. Makarov, O. Bulanov, and L. E. Zhukov, "Co-author recommender system," in *International Conference on Network Analysis (NET'16)*, National Research University Higher School of Economics. Berlin, Germany: Springer, May 26–28 2016, pp. 251–257.

[31] I. Makarov, O. Bulanov, O. Gerasimova, N. Meshcheryakova, I. Karpov, and L. E. Zhukov, "Scientific matchmaker: Collaborator recommender system," in *Proceedings of the 6th International Conference on Analysis of Images, Social Networks and Texts (AIST'17)*, ser. LNCS, Polytechnic University. Berlin, Germany: Springer, July 27–29 2017, pp. 404–410.

[32] I. Makarov, D. Kiselev, N. Nikitinsky, and L. Subelj, "Survey on graph embeddings and their applications to machine learning problems on graphs," *PeerJ Computer Science*, no. e357, pp. 1–62, 2021.

[33] I. Makarov, O. Gerasimova, P. Sulimov, and L. E. Zhukov, "Recommending co-authorship via network embeddings and feature engineering: The case of national research university higher school of economics," in *Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries (JCDL'18)*, University of North Texas. New York, USA: ACM, June 03–06 2018, pp. 365–366.

[34] I. Makarov, O. Gerasimova, P. Sulimov, K. Korovina, and L. E. Zhukov, "Joint node-edge network embedding for link prediction," in *Proceedings of the 7th International Conference on Analysis of Images, Social Networks and Texts (AIST'18)*, ser. LNCS, Polytechnic University. Berlin, Germany: Springer, July 05–07 2018, pp. 20–31.

[35] I. Makarov, O. Gerasimova, P. Sulimov, and L. E. Zhukov, "Co-authorship network embedding and recommending collaborators via network embedding," in *Proceedings of the 7th International Conference on Analysis of Images, Social Networks and Texts (AIST'18)*, ser. LNCS, Polytechnic University. Berlin, Germany: Springer, July 05–07 2018, pp. 32–38.

[36] ——, "Dual network embedding for representing research interests in the link prediction problem on co-authorship networks," *PeerJ Computer Science*, vol. 5, no. e172, pp. 1–20, 2019.

[37] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 173–182.

[38] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, vol. 1609.02907, 2017.

[39] X. Wang, R. Wang, C. Shi, G. Song, and Q. Li, "Multi-component graph convolutional collaborative filtering," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 04, 2020, pp. 6267–6274.

[40] J. Yu, X. Xia, T. Chen, L. Cui, N. Q. V. Hung, and H. Yin, "Xsimgcl: Towards extremely simple graph contrastive learning for recommendation," *arXiv preprint arXiv:2209.02544*, 2022.

[41] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, and X. Zhou, "Graph contextualized self-attention network for session-based recommendation." in *IJCAI*, vol. 19, 2019, pp. 3940–3946.

[42] P. Gupta, D. Garg, P. Malhotra, L. Vig, and G. M. Shroff, "Niser: normalized item and session representations with graph neural networks," *arXiv preprint arXiv:1909.04276*, 2019.

[43] C. Ma, L. Ma, Y. Zhang, J. Sun, X. Liu, and M. Coates, "Memory augmented graph neural networks for sequential recommendation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 04, 2020, pp. 5045–5052.

[44] J. Wang, K. Ding, Z. Zhu, and J. Caverlee, "Session-based recommendation with hypergraph attention networks," *CoRR*, vol. abs/2112.14266, 2021.

[45] X. Xia, H. Yin, J. Yu, Q. Wang, L. Cui, and X. Zhang, "Self-supervised hypergraph convolutional networks for session-based recommendation," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 5, 2021, pp. 4503–4511.

[46] M. Zhang, S. Wu, X. Yu, Q. Liu, and L. Wang, "Dynamic graph neural networks for sequential recommendation," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[47] L. Yao, Z. Chu, S. Li, Y. Li, J. Gao, and A. Zhang, "A survey on causal inference," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 15, no. 5, pp. 1–46, 2021.

[48] L. Li, W. Chu, J. Langford, and X. Wang, "Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms," in *Proceedings of the fourth ACM international conference on Web search and data mining*, 2011, pp. 297–306.

[49] D. G. Horvitz and D. J. Thompson, "A generalization of sampling without replacement from a finite universe," *Journal of the American statistical Association*, vol. 47, no. 260, pp. 663–685, 1952.
[50] X. Wang, Y. Guo, and C. Xu, "Recommendation algorithms for optimizing hit rate, user satisfaction and website revenue," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.
[51] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *Acm transactions on interactive intelligent systems (tiis)*, vol. 5, no. 4, pp. 1–19, 2015.
[52] M. Schedl, "The lfm-1b dataset for music retrieval and recommendation," in *Proceedings of the 2016 ACM on international conference on multimedia retrieval*, 2016, pp. 103–110.

DMITRII KISELEV received a Master degree in Applied Mathematics and Informatics from HSE University. Now he is pursuing his Ph.D. in Computer Science at his alma mater HSE University, Moscow, Russia. Since 2018 he has been a part-time lecturer at HSE University, School of Data Analysis and Artificial Intelligence. Now is a researcher in the field of application of graph neural networks to the Industrial AI at Artificial Intelligence Research Institute, Moscow, Russia.

Author contribution: initial idea, model and experiment design, paper preparation.

ILYA MAKAROV received the Specialist degree in Mathematics from the Lomonosov Moscow State University, Moscow, Russia, and Ph.D. in Computer Science at the University of Ljubljana, Ljubljana, Slovenia.

Since 2011 he was a full-time lecturer at HSE University, School of Data Analysis and Artificial Intelligence. He was School Deputy Head in 2012-2016. Now he is senior research fellow Artificial Intelligence Research Institute, Moscow, Russia and at Samsung-PDMI Joint AI Center, St. Petersburg Department of Steklov Institute of Mathematics, Russian Academy of Sciences, St. Petersburg, Russia. He is also a lecturer at Moscow Institute of Physics and Technology, and Machine Learning Engineer and Head of Data Science Tech Master program in NLP at National University of Science and Technology MISIS.

Author contribution: paper revision, help with experiment and model design, research supervision