INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2026, VOL. 72, NO. 3, PP. 1–9
Manuscript received March 9, 2026; revised July 2026. doi: 10.24425/ijet.2026.1722

The impact of data augmentation on the performance of session-based recommender systems

Urszula Kużelewska

***Abstract*—In response to the challenges related to the large volume of data that application users encounter while interacting with internet services, recommender systems have emerged as a valuable solution. The content provided by recommenders is tailored to the specific choices of each user.**

Despite the proposal of novel models in the existing literature, there is also an emergence of studies addressing alternative factors that may enhance recommendation performance. Data augmentation is one of the steps involved in data pre-processing that affects the final accuracy.

The objective of this paper is to examine the most efficient methods of data augmentation on a range of session-based recommender systems. Furthermore, novel strategies for data enhancement are proposed and verified.

***Keywords*—session-based recommender system, data augmentation**

I. Introduction

THE domain of recommender system design has emerged as a response to the challenges related to the large amount of data that Internet service and application users encounter in their daily lives. Recommender systems employ a range of advanced Artificial Intelligence algorithms to analyse user preferences and past behaviour, resulting in selecting the most suitable resources. The content provided is personalised, as it is adapted to a particular user's choices, thus facilitating a time-consuming decision-making process [1], [2].

Collaborative filtering methods in recommendation generation involve searching for mutual similarities among users' behaviour. In the traditional approach, all user interactions are repeatedly collected and stored in user-item matrices. In the majority of real-world applications, such as online shops and news portals, users interact with the system for a comparatively brief period and remain anonymous. No long-term interests are captured for them [3].

Systems that focus solely on short-period user interactions are known as session-based recommender systems. Sessions are defined as sequences of actions (items) organised with a reference to time. The objective of such systems is to predict the subsequent action for the session that is currently being processed. The system analyses the similarity among user sessions with a purpose to generate recommendations [4].

The earliest approaches to sequential data analysis were based on frequent patterns and Markov chains [5], [6], [7]. The following, including SKNN (Session-based kNN) [8], VSKNN (Vector Multiplication Session-Based kNN) [4], STAN (Sequence and Time Aware Neighborhood) [9], VSTAN and CT (Context Trees) [10], applied the nearest neighbours calculations among sessions. A large group of methods are those involving deep learning. The examples are: GRU4REC [11], NARM (Neural Attentive Recommendation Machine) [12], STAMP (Short-Term Attention/Memory Priority Model for Session-based Recommendation) [13], NextItNet [14]. Recent publications related to session-based recommender systems consider graph modelling [15] or a hybrid combination [16].

Although new models based on neural networks or a mixture of methods are proposed in the literature, new works addressing other factors related to improving recommendation performance also appear. Data augmentation is one of the steps involved in data pre-processing that affects the final accuracy. Furthermore, it addresses the issue of data sparsity, which is a common challenge in the domain of recommendation generation. Many models, particularly those based on neural networks, require a large amount of training data to produce high-quality results. Consequently, in multiple domains, notably computer vision, data augmentation is becoming a vital component [17]. It includes the field of recommender systems. However, although the number of articles is increasing and new techniques are proposed, the impact of data augmentation on recommendations, in particular session-based, has not been adequately examined [18], [19].

The following data augmentation strategies have already been successfully applied in the context of recommenders: noise injection, redundancy injection, item masking, synonym replacement, item insertion, item replacement, cropping, deletion, reordering, subset-splitting and slide-window [18], [19]. The most attention was devoted to the application of these techniques in sequential recommender systems and contrastive learning. It is worth noting that the results described in [18] demonstrated that the performance of all selected methods has been improved when data were augmented, regardless of the modification strategy.

The aim of this paper is to examine the most efficient methods of data augmentation on selected session-based rec-

The work was supported by a grant from the Bialystok University of Technology WZ/WI-IIT/3/2023 and funded with resources for research by the Ministry of Education and Science in Poland.

U. Kużelewska is with Faculty of Computer Science, Bialystok University of Technology, Wiejska 45a, 15-351 Bialystok, Poland (e-mail: u.kuzelewska@pb.edu.pl).

ommender systems. In addition, new strategies for data enhancement are to be proposed and verified.

The contributions of this work is as follows:

- Evaluation of the most efficient augmentation methods' performance belonging to the category of data manipulation integrated with the common session-based recommender systems.
- Comparison of the obtained results with respect to the input data characteristics, such as data sparsity, session length.
- Propositions of 2 new augmentation methods, which one of them is based on reducing the impact of popularity on final recommendations, while the second method focuses on frequent patterns that occur in similar sessions.

The next section (Section II) of this article summarises the recent research devoted to data augmentation in the field of recommender systems with a particular focus on those belonging to the category of session-based. The subsequent section (Section III) details the most efficient approaches to data augmentation, in addition to the methods proposed in this work. The Section IV outlines the experimental setup including characterising input data and the evaluation procedure. It also presents the results obtained from these experiments. The last section concludes the paper.

II. Related work

For many years, the recommendation systems domain has been focused on enhancing the accuracy of generated propositions. This has been achieved by proposing and developing new recommendation methods [20]. Then, other issues, related to general recommender system's performance, arose interest of researchers [21]. Examples are articles devoted to the impact of data characteristics on recommendation lists accuracy [22], and data augmentation [23].

Despite its potential to improve the accuracy and effectiveness of generated propositions, data augmentation has not yet been fully explored in the domain of recommendation systems and particularly in a session-based approach [17].

In one of the first such works [24], the authors used simple data augmentation to improve the performance of an RNN-based model. The idea was to generate a series of shorter subsessions from one long session. This solution achieved an accuracy improvement of 12.8%.

Another article [25] employed a technique of merging basket data from various shops' baskets and then generating subsessions.

The first work related to sequential recommenders was [26], in which the authors proposed more advanced methods than just subset selection. The following methods were employed: Noise Injection, Redundancy Injection, Item Masking and Synonym Replacement. The first one involved inserting a random item not present in the session into this session also on a random position. Alternatively, inserting a random item but present in the session into this session was named as Redundancy Injection. Item Masking is a technique that involves the removal of k items from the original session, selected at random. In contrast, Synonym Replacement involves the replacement of k items from the original item sequence with their synonyms, chosen for their high degree of similarity. The most effective strategies were those involving injections, which resulted in improvements ranging from several percent to as much as 28%.

It should be noted that there are also methods in existence that, despite their effectiveness in improving performance, are not without their limitations. A CNN-based solution to collaborative filtering, CASER [27], is able to capture short-term preferences in item sequences by splitting original sequences into several shorter ones. This approach consequently generated additional values in the input data. However, it is important to note that the component of data augmentation is strictly integrated with the particular recommendation method and cannot be used as a separate approach.

Another example is provided by methods belonging to the isolated category of self-supervised learning with their data augmentation techniques adapted only to them [28].The approach involves constructing a graph structure, incorporating negative samples (so called contrastive learning).

III. Data augmentation

It is well-known that insufficient data can result in unsatisfactory outcomes when using machine learning methods. Furthermore, the issue of small data size is frequently associated with the phenomenon of algorithmic data overfitting [29].

One of the methods that can be employed to increase the size of input data is data augmentation [26], [30]. In the field of computer vision, data augmentation has become a standard preprocessing step, leading to significant improvements in generalisation [31]. Nevertheless, it is important to note that there is no universally optimal augmentation strategy. The algorithm is usually selected empirically for a particular problem [31].

This section outlines the methods employed in the field of recommender systems for data augmentation. As outlined in the article, this section focuses on the separated algorithms to be applied in the pre-processing step in session-based approaches. It is important to note that these algorithms do not affect the overall recommender architecture. It covers the techniques that have been presented in the literature as well as the ones proposed in this paper.

A. Methods of augmentation presented in the literature

The augmentation methods operating on data samples in the pre-processing step are based on subset selection (the earliest) and direct manipulation.

The subset selection methods are the following:

- Subset split (SR) - the each session of data is divided into separate subsequences. The length of subsequences can be different.
- Sliding window (SW) - the each session of data is divided into separate subsequences of the same length by sliding the window.

The direct manipulation approach assumes the following solutions [26], [30]:

- Noise injection (NI) - the data is extended by adding a randomly generated item that was not present in the original data. This item is treated as a negative sample.
- Redundancy injection (RI) - the data is extended by randomly selecting and adding an item from the original data. This item is treated as a positive sample.
- Item masking (IM) - the data is reduced by randomly selecting and excluding k items from the original item sequence.
- Synonym replacement (SR) - the size of the data set remains unchanged, but the k selected items are replaced by the most similar items (synonyms) from the other sessions. The authors of [26] exploited additional item descriptions and calculated the semantic similarity among the items.
- Item swapping (ISW) - the size of the data set remains unchanged, but the selected item is swapped with the target (the last item in the session). There are several variants of this approach. For example, the exchange is performed on one of the last k items.
- Item shuffling (ISH) - the size of the data set remains unchanged, but the k selected recently viewed items are shuffled.

The subset selection techniques were previously assessed in comparison to the direct manipulation solutions, which were found to generate inferior results. As such, these elements were not included in the scope of the experimental analysis.

B. Proposed methods of augmentation

The novel solutions associated with data augmentation that are proposed in this article are of the direct manipulation nature. One of these models is based on the concept of items' popularity exerting a negative influence on the systems' personalisation accuracy, while the other model applies the principle of session similarity and occurrence of item patterns within them.

- Popularity shifting (PS) - the objective of this approach is to move backwards through a session the items with the highest popularity value. The target item remains in its original position, while the other items in the session are evaluated based on their popularity value in the entire input dataset. Subsequently, instances with a value exceeding the predefined threshold, pop_{th}, are transferred to the end of the session. This approach minimises the adverse impact of the popularity phenomenon on the accuracy of the recommendation lists.
- Similar items pattern (SIR) - the objective is to search for k similar sessions to the target session (i.e. the session to be augmented) and then identify the items that are common in these sessions. It is imperative that the identified items are visited before the target item. This set of items is regarded as a frequent pattern of those that can be considered similar, as they occur in related sessions. The pattern items with the target one form a separate augmented session. The calculation of similarity among sessions is performed using commonly employed metrics, such as the cosine measure.

IV. Experiment

This section presents the results obtained on session-based recommender systems with original and augmented data. The objective of this study is twofold: firstly, to examine how data enhancement impacts data characteristics, and secondly, to verify whether the performance of recommendation algorithms is improved after such an operation. The experiments were conducted on two distinct subsets of each of the two original data sets: Diginetica dataset [32] and Retailrocket [33]. Consequently, the experiments involved four subsets. They contain recordings of user interactions with items in the form of sessions. Each session comprises a session ID and the IDs of the items with which the user interacted. Each interaction is also assigned a timestamp.

A. Experimental setup

Each of the four subsets was divided into two distinct splits, from which a training and testing part was extracted. In the case of Retailrocket, the testing period covered the last day, whereas for Diginetica, which had a more sparse schedule, it covered the last 17 days. The augmentation procedure was applied to every session in the training data; the test set remained unmodified. Recommendations were then generated for each of the two splits and evaluated. The final evaluation values were averaged. This procedure was performed on all the augmented data and the original data. The following systems, which belong to the nearest neighbour and general network approaches, were selected for generating recommendations: SKNN, VSKNN, GRU4REC, STAMP and NARM.

1) Datasets: The datasets used in the experiments were characterised as follows (the respective column names from the tables below are given in brackets): a number of actions (Actions), a number of session (Sessions), a number of items (Items), ratio of sessions to items (Sessions/Items), density of a dataset with respect to sessions (Actions/Sessions), density of a dataset with respect to items (Actions/Items). The values before and after data augmentation are respectively presented in Tables I, II respectively for Diginetica and Retailrocket datasets. In the case of augmented data with the same values, these were placed on the same line, with their labels separated by commas.

2) Augmentation methods used in the experiments: The following baseline methods were selected for data augmentation: Item swapping, Item masking and Item shuffling. Their detailed description and configuration are presented below:

- ISW(1) - the exchange was conducted between the previously visited item and the target item.
- IM(1) - the truncation of visited items was performed at a rate of between 50 and 100 %.
- ISW(2) - the exchange was conducted between one of the k previously visited item and the target item. The k was also determined randomly.
- IM(2) - the truncation of visited and the target items was performed at a rate of between 50 and 100 %. The item that remained in the session, which had been visited recently, was designated a target item.

- ISW(3) - the exchange was carried out between the previously visited item and the second-to-last visited item.
- ISH(1) - the reorganisation of the items under consideration was executed on the final k items that had previously been visited. The number k was selected at random.

Summary of the proposed approaches including their configuration values is provided below:

- PS - the popularity threshold pop_{th} in this solution was set to 30%, which means that the items belonging to the 30% products with the highest overall popularity were selected for the process of selection. The optimal threshold value has been determined empirically.
- SIR - in the procedure applied in the experiments, the value of k was set to 3. Consequently, for each augmented session 3 other similar sessions were calculated. In the absence of common items, or where the number was less than 2, the final result was not formed.

3) Evaluation protocol and performance measures: The metrics employed for the evaluation of recommender systems were those typically utilised for this purpose. The following were included: Hit Rate (HR) and Mean Reciprocal Rank (MRR) to assess recommendation accuracy, as previously outlined by [11]. Furthermore, often additional indices are used in order to evaluate the diversity of the generated lists and resistance of the algorithm to the tendency of recommending popular items [34].

- **HR** and **MRR**: The procedure for calculating both HR and MRR is based on the evaluation of the content of recommendation lists when successive items are incrementally added to the test sessions. It has been adopted from [11] and other works. Then, after generating the propositions, the list is truncated at the particular position and the content is examined in terms of the presence of the items from the test vector. MRR also calculates weights that are inversely related to the order of the predicted items in the test sessions.
- **Coverage**: Coverage [20] indicates the frequency with which items appear in the recommendation lists. Its high values indicate different sets of suggestions for different users, its low values relate to the tendency to recommend the same set to many users, even if they have different preferences.
- **Popularity**: Popularity index [11] measures the bias of the algorithm to include only popular items in the recommendation lists. Its low values are beneficial and are related to the ability of the methods to tackle the long tail problem. This index is calculated by the average popularity score of the top-k items in the recommendation lists. This final score is an average of the individual popularity scores of each recommended item. It is estimated by counting the occurrences of the items in one of the training sessions and then applying min-max normalisation to obtain a score between 0 and 1.

All the algorithms require input parameter values to be specified. These values were optimised for each method on the original datasets individually, in order to maximise the value of MRR@20. It was decided not to recalibrate the augmented data due to insignificant changes in their characteristics.

B. Obtained results

Despite the experiments being conducted on all the prepared and augmented subsets, this section presents only a limited set of results, due to the extensive numbers obtained during the research. The selection of outcomes was based on their significance to the final achievements. The focus was on the results that showed significant improvements compared to the original datasets. However, it is important to note that not all outcomes were selected, despite being enhanced. It is evident that the results obtained with the proposed augmentation techniques were presented in full, despite the fact that the improvement was achieved.

Tables III, IV, V, VI, VII contain the results generated on Diginetica dataset by the algorithms respectively: SKNN, VSKNN, GRU4REC, STAMP and NARM. Note, that the values related to the original data are in the row with a letter "o" ($diginetica(1) - o$). Furthermore, improved results are highlighted in bold and the best value for the particular index is additionally underlined.

In the case of the SKNN algorithm, the augmentation process was found to have a beneficial effect on overall performance, as measured by accuracy, coverage and popularity (see Table III). For the second subset, an enhancement was evident in all performance measures, notwithstanding the setting of a cut-off point. The SIR augmentation procedure was found to be particularly advantageous (diginetica(2)-SIR), with a 9% enhancement in accuracy being observed. The PS approach was also efficient (diginetica(2)-PS) with a 6% enhancement in accuracy.

In the case of the VSKNN algorithm and the Diginetica dataset, the augmentation approach based on item swapping (ISW(2)) was predominant in terms of accuracy (see Table IV). However, PS generated superior results for HR@10, MRR@10 and MRR@20. The SIR approach was only advantageous for HR@20; nevertheless, it was the most valuable for this index. It was evident that both solutions, PS and SIR, were efficient in relation to coverage and popularity values.

The GRU4REC algorithm was found out to be less beneficial for Diginetica subsets during the augmentation process(see Table V). It was observed that a multitude of outcomes remained unimproved (all MRR values and Coverage for Diginetica(1) and HR@5, HR@10 and all MRR values for Diginetica(2)). However, the SIR method was advantageous for all HR indices in the first subset (diginetica(1)-SIR) and for the Popularity index. In contrast, for the subsequent subset, the SIR method was only advantageous for HR@20 (diginetica(2)-SIR). It was established that the most valuable results for HR@5 and Popularity for the initial subset (diginetica(1)-PS) and for HR@20, Coverage and also Popularity for the secondary data (diginetica(2)-PS) were generated.

A similar effect can also be observed when analysing Tables VI and VII with results obtained by the STAMP and NARM algorithms, respectively. Augmentation has been shown to be beneficial for both algorithms, particularly ISW(3) and SIR for

TABLE I. Original and augmented input data characteristics on the dataset Diginetica

dataset	Actions	Sessions	Items	Session/Items	Actions/Sessions	Actions/Items
diginetica(1)-original	201501	27932	36757	0.7599	7.2140	5.4820
diginetica(1) ISW(1), ISW(2),ISW(3), TSH(1),PS	385873	48998	36757	1.3330	7.8753	10.4979
diginetica(1)-IM(1),IM(2)	293644	48998	36757	1.3330	5.9930	7.9888
diginetica(1)-SIR	254215	40756	36757	1.1088	6.2375	6.9161
diginetica(2)-original	201457	31572	36905	0.8555	6.3809	5.4588
diginetica(2) ISW(1), ISW(2), ISW(3) IM(1), TSH(1),PS	379873	53937	36905	1.4615	7.0429	10.2933
diginetica(2)-IM(2)	290471	53937	36905	1.4615	5.3854	7.8708
diginetica(2)-SIR	254626	44891	36905	1.2164	5.6721	6.8995

TABLE II. Original and augmented input data characteristics on the dataset Retailrocket

dataset	Actions	Sessions	Items	Session/Items	Actions/Sessions	Actions/Items
retailrocket(1)-original	320328	20580	47019	0.4377	15.5650	6.8127
retailrocket(1) ISW(1), ISW(2),ISW(3), TSH(1),PS	620757	32732	47019	0.6961	18.9648	13.2023
retailrocket(1)-IM(1)	547805	32732	47019	0.6961	16.7361	11.6507
retailrocket(1)-IM(2)	463700	32732	47019	0.6961	14.1666	9.8620
retailrocket(1)-SIR	374230	28269	47019	0.6012	13.2382	7.9591
retailrocket(2)-original	320624	50336	52387	0.9608	6.3697	6.1203
retailrocket(2) ISW(1), ISW(2),ISW(3), TSH(1),PS	583570	76250	52387	1.4555	7.6534	11.1396
retailrocket(2)-IM(1)	520343	76250	52387	1.4555	6.8242	9.9327
retailrocket(2)-IM(2)	452649	76250	52387	1.4555	5.9364	8.6405
retailrocket(2)-SIR	414911	67215	52387	1.2830	6.1729	7.9201

TABLE III. Selected performance results of SKNN algorithms on diginetica dataset

dataset	HR@5	HR@10	HR@20	MRR@5	MRR@10	MRR@20	Coverage	Popularity
diginetica(1)-o	0.2442	0.3362	0.4542	0.1363	0.1484	0.1566	0.1769	0.1277
diginetica(1)-ISW(2)	**0.2469**	**0.3373**	**0.4564**	**0.1383**	**0.1528**	**0.1585**	**0.1874**	**0.1136**
diginetica(1)-IM(2)	0.2290	0.3313	0.4467	0.1288	0.1421	0.1500	**0.1776**	**0.1198**
diginetica(1)-PS	0.2442	**0.3465**	**0.4575**	**0.1376**	**0.1510**	**0.1587**	**0.1788**	**0.1214**
diginetica(1)-SIR	0.2404	**0.3476**	**0.4559**	**0.1368**	**0.1508**	**0.1582**	**0.1792**	**0.1111**
diginetica(2)-o	0.2469	0.3411	0.4586	0.1359	0.1482	0.1564	0.1800	0.1221
diginetica(2)-IM(1)	**0.2555**	**0.3583**	**0.4637**	**0.1483**	**0.1620**	**0.1693**	**0.2063**	**0.1020**
diginetica(2)-PS	**0.2620**	**0.3615**	**0.4697**	**0.1491**	**0.1622**	**0.1697**	**0.2066**	**0.1010**
diginetica(2)-SIR	**0.2657**	**0.3529**	**0.4653**	**0.1518**	**0.1634**	**0.1712**	**0.2043**	**0.0926**

the first subset of Diginectica and PS for the second subset in the case of STAMP, and SIR for the first subset of Diginectica. In the second subset, the most optimal outcomes were achieved without the augmentation step.

In the context of the SKNN algorithm and the Retailrocket dataset, the augmentation approach that employs item masking (IM(1)) has been shown to be the most effective strategy in terms of accuracy (see Table VIII). This observation pertains to the Retailrocket(1) dataset. However, both PS and SIR generated superior results for HR@5, HR@10 and Coverage and Popularity. The SIR approach was found to be advantageous in the case of Retailrocket(2) data (for HR@5, MRR@5, Coverage and Popularity) with the most accurate results for this issue.

In the case of the VSKNN algorithm, 2 methods of the augmentation processes were identified as demonstrating superior performance in comparison to the remaining approaches and the original method (see Table IX). The solution was found to be IM(2) (retailrocket(1)-IM(2)) and SIR (retailrocket(2)-SIR). The results demonstrated a positive impact on overall performance, as measured by accuracy, coverage and popularity. The PS algorithm produced sub-optimal results for both subsets.

The following table (Table X) contains results generated by GRU4REC algorithm on Retailrocket data. In this case

TABLE IV. Selected performance results of VSKNN algorithms on diginetica dataset

dataset	HR@5	HR@10	HR@20	MRR@5	MRR@10	MRR@20	Coverage	Popularity
diginetica(1)-o	0.3037	0.3692	0.4591	0.2198	0.2284	0.2346	0.2853	0.0974
diginetica(1)-IM(1)	**0.3059**	0.3687	0.4575	0.2194	0.2277	0.2338	**0.2855**	0.0976
diginetica(1)-ISW(2)	**0.3043**	**0.3736**	0.4570	**0.2209**	**0.2301**	**0.2358**	0.2829	**0.0969**
diginetica(1)-PS	0.3032	**0.3720**	0.4570	0.2198	**0.2290**	**0.2348**	**0.2857**	**0.0965**
diginetica(1)-SIR	0.2983	0.3687	**0.4624**	0.2176	0.2270	0.2334	**0.2855**	**0.0853**
diginetica(2)-o	0.3037	0.3703	0.4591	0.2190	0.2279	0.2340	0.2827	0.0978
diginetica(2)-IM(2)	0.3012	**0.3749**	0.4578	**0.2309**	**0.2407**	**0.2463**	**0.2942**	**0.0880**
diginetica(2)-ISH(1)	**0.3043**	**0.3736**	0.4570	**0.2209**	**0.2301**	**0.2358**	**0.2829**	**0.0969**
diginetica(2)-PS	0.3029	0.3690	**0.4605**	**0.2284**	**0.2375**	**0.2437**	**0.2995**	**0.0866**
diginetica(2)-SIR	0.2991	**0.3749**	**0.4610**	**0.2257**	**0.2358**	**0.2415**	**0.2993**	**0.0767**

TABLE V. Selected performance results of GRU4REC algorithms on diginetica dataset

dataset	HR@5	HR@10	HR@20	MRR@5	MRR@10	MRR@20	Coverage	Popularity
diginetica(1)-o	0.2648	0.3200	0.3822	0.2083	0.2158	0.2201	0.3053	0.0782
diginetica(1)-ISW(1)	0.2648	0.3135	**0.3860**	0.1994	0.2057	0.2106	0.3017	**0.0746**
diginetica(1)-IM(1)	0.2615	0.3178	**0.3844**	0.1962	0.2035	0.2082	0.3025	**0.0742**
diginetica(1)-IM(2)	0.2648	**0.3243**	**0.3963**	0.2000	0.2079	0.2128	0.2903	**0.0733**
diginetica(1)-PS	**0.2669**	0.3156	0.3801	0.1979	0.2042	0.2085	0.3045	**0.0752**
diginetica(1)-SIR	**0.2723**	**0.3205**	**0.3893**	0.2052	0.2117	0.2165	0.2926	**0.0682**
diginetica(2)-o	0.2711	0.3174	0.3798	0.2196	0.2254	0.2298	0.3253	0.0632
diginetica(2)-PS	0.2641	0.3110	**0.3819**	0.2073	0.2134	0.2181	**0.3279**	**0.0612**
diginetica(2)-SIR	0.2630	0.3211	**0.3889**	0.2088	0.2163	0.2210	0.3220	0.0530

TABLE VI. Selected performance results of STAMP algorithms on diginetica dataset

dataset	HR@5	HR@10	HR@20	MRR@5	MRR@10	MRR@20	Coverage	Popularity
diginetica(1)-o	0.3032	0.3768	0.4613	0.1920	0.2021	0.2081	0.3193	0.0984
diginetica(1)-ISW(1)	**0.3059**	**0.3822**	0.4607	0.1902	0.2005	0.2060	**0.3256**	**0.0974**
diginetica(1)-IM(1)	0.2972	**0.3801**	0.4597	0.1833	0.1945	0.2000	**0.3304**	**0.0940**
diginetica(1)-ISW(2)	0.2962	0.3747	0.4532	0.1891	0.2000	0.2051	0.3118	0.1005
diginetica(1)-IM(2)	0.2810	0.3633	0.4385	0.1770	0.1877	0.1930	**0.3266**	**0.0927**
diginetica(1)-ISW(3)	**0.3173**	**0.3893**	**0.4640**	**0.2083**	**0.2179**	**0.2232**	0.3116	0.1014
diginetica(1)-PS	0.2826	0.3584	0.4499	0.1782	0.1883	0.1948	0.3172	**0.0883**
diginetica(1)-SIR	**0.3135**	**0.3925**	**0.4710**	**0.1970**	**0.2077**	**0.2133**	0.3163	**0.0854**
diginetica(2)-o	0.3007	0.3803	0.4476	0.1878	0.1984	0.2031	0.3395	0.0897
diginetica(2)-PS	0.3002	0.3776	**0.4513**	**0.1911**	**0.2015**	**0.2067**	0.3318	**0.0831**
diginetica(2)-SIR	0.2975	0.3615	0.4303	0.1825	0.1911	0.1958	0.3225	**0.0806**

augmentation was found out to be very beneficial for accuracy and coverage and popularity. Both solutions, proposed in this article, PS and SIR, generated the optimal outcomes: for Diginetica(1) PS was outperforming the others in respect to HR@5, MRR@5, MRR@10,MRR@20, Coverage, whereas SIR was the best in respect to HR@10 and Popularity. For Diginetica(2) only SIR generated better results than not augmented recommender configuration, but only in respect to HR@5, HR@20 and Popularity. It is important to note that the highest levels of performance were achieved in the Retailrocket subset by the solution based on data augmentation.

As demonstrated in Table XI, the outcomes of the STAMP algorithm and Retailrocket data were highly varied. Although the augmentation procedure has been shown to be beneficial, there was no predominant solution identified. For Retailrocket (1), the respective values were ISW (2) and TSH (1), whereas for Retailrocket (2), the values were IM (1) and ISW (3).

It was not possible to generate the results by NARM algorithm for either the original or augmented data, due to the algorithm requiring an exceptionally long time to run.

V. Conclusion

The purpose of this paper was to present the results of experiments that validate the impact on the performance of session-based recommender systems of the most efficient solutions for data augmentation. These methods, considered as baseline methods, were selected after a thorough analysis of the outcomes presented in the literature.

Furthermore, novel algorithms for data augmentation were proposed in this article, which were examined against recom-

TABLE VII. Selected performance results of NARM algorithms on diginetica dataset

dataset	HR@5	HR@10	HR@20	MRR@5	MRR@10	MRR@20	Coverage	Popularity
diginetica(1)-o	0.1727	0.2258	0.2929	0.1041	0.1113	0.1160	0.3412	0.0623
diginetica(1)-ISW(1)	0.1467	0.2057	0.2740	0.0843	0.0920	0.0968	**0.3493**	0.0623
diginetica(1)-IM(1)	0.1662	0.2241	**0.2951**	0.0917	0.0995	0.1045	0.3307	**0.0619**
diginetica(1)-IM(2)	**0.1814**	**0.2404**	**0.3081**	**0.1082**	**0.1159**	**0.1205**	0.3282	**0.0576**
diginetica(1)-PS	0.1473	0.2117	0.2853	0.0836	0.0921	0.0972	**0.3473**	0.0624
diginetica(1)-SIR	**0.1926**	**0.2512**	**0.3141**	**0.1187**	**0.1266**	**0.1309**	0.3376	**0.0527**
diginetica(2)-o	0.1851	0.2437	0.3147	0.1065	0.1151	0.1200	0.3323	0.0597
diginetica(2)-PS	0.1382	0.1892	0.2517	0.0803	0.0868	0.0912	**0.3638**	**0.0569**
diginetica(2)-SIR	0.1576	0.2222	0.2932	0.0977	0.1065	0.1113	**0.3366**	**0.0499**

TABLE VIII. Selected performance results of SKNN algorithms on retailrocket dataset

dataset	HR@5	HR@10	HR@20	MRR@5	MRR@10	MRR@20	Coverage	Popularity
retailrocket(1)-o	0.1968	0.2730	0.3455	0.12495	0.1352	0.1405	0.1064	0.0695
retailrocket(1)-IM(1)	**0.2009**	**0.2775**	**0.3455**	**0.1265**	**0.1367**	0.1416	**0.1070**	**0.0602**
retailrocket(1)-IM(2)	0.1947	0.2705	0.3351	0.1216	0.1320	0.1366	0.1048	0.0645
retailrocket(1)-PS	**0.1984**	**0.2746**	0.3376	0.1236	0.1340	0.1384	**0.1084**	**0.0600**
retailrocket(1)-SIR	**0.2034**	**0.2738**	0.3413	0.1244	0.1338	0.1386	**0.1076**	**0.0615**
retailrocket(2)-o	0.6300	0.7628	0.8409	0.4270	0.4453	0.4510	0.1233	0.0354
retailrocket(2)-PS	0.6285	0.7510	0.8381	0.4554	0.4432	0.4492	**0.1248**	**0.0258**
retailrocket(2)-SIR	**0.6363**	0.7543	0.8366	**0.4302**	0.4459	0.4520	**0.1242**	**0.0258**

TABLE IX. Selected performance results of VSKNN algorithms on retailrocket dataset

dataset	HR@5	HR@10	HR@20	MRR@5	MRR@10	MRR@20	Coverage	Popularity
retailrocket(1)-o	0.2684	0.3239	0.3749	0.2034	0.2109	0.2144	0.1803	0.0601
retailrocket(1)-ISW(1)	**0.2697**	0.3227	0.3724	0.2027	0.2098	0.2133	**0.1820**	**0.0578**
retailrocket(1)-IM(1)	**0.2688**	0.3165	0.3674	0.2030	0.2095	0.2131	0.1772	0.0609
retailrocket(1)-IM(2)	**0.2742**	**0.3165**	**0.3683**	**0.2058**	**0.2115**	**0.2151**	**0.1832**	**0.0635**
retailrocket(1)-PS	0.1678	0.2129	0.2659	0.1214	0.1277	0.1313	0.0804	0.2182
retailrocket(1)-SIR	0.1773	0.2253	0.2751	0.1249	0.1314	0.1347	0.0809	0.2041
retailrocket(2)-o	0.6136	0.7301	0.8217	0.4290	0.4444	0.4511	0.1323	0.0384
retailrocket(2)-IM(1)	0.6065	0.7145	0.8018	**0.4293**	0.4441	0.4504	0.1330	**0.0378**
retailrocket(2)-PS	0.6016	0.7173	0.8082	0.4252	0.4411	0.4477	0.1323	0.0379
retailrocket(2)-SIR	**0.6186**	**0.7330**	**0.8224**	**0.4322**	**0.4475**	**0.4540**	0.1320	**0.0284**

mendation lists generated without data preprocessing and compared with baseline methods. The experimental findings have demonstrated that the implementation of data augmentation can enhance the efficacy of session-based recommendations. Whilst the enhancement may be negligible or disadvantageous on occasion, augmentation has been demonstrated to facilitate superior generalisation by the model in the majority of instances. The proposed algorithms were found to be particularly efficient in the case of the recommenders based on kNN approach and for sparse data.

As with the other solutions outlined in the literature, this research does not provide a comprehensive solution for data augmentation. Instead, it explores a potential approach for preprocessing data to enhance the capacity of recommender systems. This is also an initial step to determining whether such a process is advantageous and whether it is beneficial in the case of particular algorithms or data. While the results were not as straightforward as expected and the relation was not immediately obvious, additional experiments and modifications are required to explore this.

The direction of future research will be towards the examination of additional data sets, with a specific focus on investigating the correlation between the characteristics of the data and the impact level of the augmentation phase. The development of the proposed methods is also to be studied, as well as the identification of novel solutions.

ACKNOWLEDGMENT

The research was carried out within project no. WZ/WI-IIT/3/2023 at Bialystok University of Technology and financed from the research subsidy of Bialystok University of Technology.

TABLE X. Selected performance results of GRU4REC algorithms on retailrocket dataset

dataset	HR@5	HR@10	HR@20	MRR@5	MRR@10	MRR@20	Coverage	Popularity
retailrocket(1)-o	0.5953	0.6392	0.6632	0.4540	0.4602	0.4619	0.3103	0.0234
retailrocket(1)-IM(1)	0.5907	0.6363	**0.6703**	0.4286	0.4349	0.4374	0.2861	0.0244
retailrocket(1)-IM(2)	**0.5973**	**0.6396**	0.6616	0.4498	0.4557	0.4573	**0.3338**	**0.0222**
retailrocket(1)-PS	**0.6065**	**0.6396**	0.6578	**0.4703**	**0.4749**	**0.4762**	**0.3528**	**0.0219**
retailrocket(1)-SIR	**0.6015**	**0.6442**	0.6632	**0.4588**	**0.4648**	**0.4662**	**0.3120**	**0.0184**
retailrocket(2)-o	0.6982	0.7628	0.8018	0.5260	0.5350	0.5377	0.1929	0.0245
retailrocket(2)-PS	0.6889	0.7621	0.7962	0.5131	0.5228	0.5252	0.1873	0.0238
retailrocket(2)-SIR	**0.6989**	0.7585	**0.8047**	0.5167	0.5248	0.5280	0.1899	**0.0165**

TABLE XI. Selected performance results of STAMP algorithms on retailrocket dataset

dataset	HR@5	HR@10	HR@20	MRR@5	MRR@10	MRR@20	Coverage	Popularity
retailrocket(1)-o	0.0543	0.0688	0.0779	0.0426	0.0445	0.0452	0.0347	0.0626
retailrocket(1)-ISW(1)	**0.0597**	**0.0646**	0.0737	**0.0508**	**0.0514**	**0.0520**	0.0222	0.2458
retailrocket(1)-ISW(2)	**0.0675**	**0.0758**	**0.0924**	**0.0603**	**0.0613**	**0.0625**	**0.0381**	0.1485
retailrocket(1)-TSH(1)	**0.0746**	**0.0829**	**0.0920**	**0.0628**	**0.0639**	**0.0645**	**0.0697**	**0.0398**
retailrocket(1)-PS	**0.0659**	**0.0729**	**0.0795**	**0.0591**	**0.0600**	**0.0605**	0.0325	0.1889
retailrocket(1)-SIR	**0.0634**	**0.0758**	**0.0911**	**0.0556**	**0.0573**	**0.0583**	0.0272	0.0850
retailrocket(2)	0.5959	0.6740	0.7273	0.4383	0.4488	0.4526	0.1425	0.0468
retailrocket(2)-IM(1)	**0.6321**	**0.7081**	**0.7550**	**0.4697**	**0.4803**	**0.4803**	**0.1504**	**0.0460**
retailrocket(2)-IM(2)	**0.5973**	**0.6396**	**0.6616**	**0.4700**	**0.4749**	**0.4762**	**0.1528**	**0.0219**
retailrocket(2)-ISW(3)	**0.6037**	**0.6719**	**0.7337**	0.4329	0.4421	0.4465	**0.1602**	**0.0444**
retailrocket(2)-PS	**0.6001**	**0.6776**	**0.7379**	0.4205	0.4312	0.4354	**0.1475**	**0.0350**
retailrocket(2)-SIR	0.5945	**0.6761**	**0.7379**	0.4338	0.4449	0.4491	**0.1475**	**0.0350**

REFERENCES

[1] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, *Recommender Systems*. Cambridge University Press, 2010, vol. 24.

[2] F. Ricci, L. Rokach, and B. Shapira, *Recommender Systems: Introduction and Challenges*. Springer, 2015, vol. 1-35.

[3] M. Quadrana, P. Cremonesi, and D. Jannach, "Sequence-aware recommender systems," *ACM Comput. Surv.*, vol. 51, no. 4, 2018.

[4] M. Ludewig, N. Mauro, S. Latifi, and D. Jannach, "Empirical analysis of session-based recommendation algorithms," *User Model User-Adap Inter*, vol. 31, p. 149–181, 2021.

[5] G.-E. Yap, X. li, and P. Yu, "Effective next-items recommendation via personalized sequential pattern mining," vol. 7239, 04 2012, pp. 48–64.

[6] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *Proceedings of the 19th International Conference on World Wide Web*. Association for Computing Machinery, 2010, p. 811–820.

[7] B. Mobasher, H. Dai, T. Luo, and M. Nakagawa, "Using sequential and non-sequential patterns in predictive web usage mining tasks," 10 2002.

[8] G. Bonnin and D. Jannach, "Automated generation of music playlists: Survey and experiments," *ACM Comput. Surv.*, vol. 47, no. 2, 2014.

[9] D. Garg, P. Gupta, P. Malhotra, L. Vig, and G. Shroff, "Sequence and time aware neighborhood for session-based recommendations: Stan," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019, p. 1069–1072.

[10] F. Mi and B. Faltings, "Context tree for adaptive session-based recommendation," 2018. [Online]. Available: https://arxiv.org/abs/1806.03733

[11] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," 2016. [Online]. Available: https://arxiv.org/abs/1511.06939

[12] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, p. 1419–1428.

[13] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "Stamp: Short-term attention/memory priority model for session-based recommendation," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, p. 1831–1839.

[14] F. Yuan, A. Karatzoglou, I. Arapakis, J. Jose, and X. He, "A simple convolutional generative network for next item recommendation," 2019, pp. 582–590.

[15] J. Wang, K. Ding, Z. Zhu, and J. Caverlee, *Session-based Recommendation with Hypergraph Attention Networks*, pp. 82–90.

[16] M. Choi, S. Lee, S. Park, and J. Lee, "Linear item-item models with neural knowledge for session-based recommendation," in *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2025, p. 1666–1675.

[17] Y. Zhang and Y. Zhang, "Mixrec: Individual and collective mixing empowers data augmentation for recommender systems," in *Proceedings of the ACM on Web Conference 2025*, 2025, p. 2198–2208.

[18]

[19] P. Zhou, Y.-L. Huang, Y. Xie, J. Gao, S. Wang, J. B. Kim, and S. Kim, "Is contrastive learning necessary? a study of data augmentation vs contrastive learning in sequential recommendation," in *Proceedings of the ACM Web Conference 2024*, 2024, p. 3854–3863.

[20] G. Adomavicius and Y. Kwon, "Improving aggregate recommendation diversity using ranking-based techniques," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 5, pp. 896–911, 2012.

[21] Y. Deldjoo, A. Bellogin, and T. Di Noia, "Explaining recommender systems fairness and accuracy through the lens of data characteristics," *Information Processing & Management*, vol. 58, no. 5, p. 102662, 2021.

[22] "blind version."

[23] Q. Wang, H. Yin, H. Wang, Q. V. H. Nguyen, Z. Huang, and L. Cui, "Enhancing collaborative filtering with generative augmentation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 548–556.

[24] Y. K. Tan, X. Xu, and Y. Liu, "Improved recurrent neural networks for session-based recommendations," in *Proceedings of the 1st workshop on deep learning for recommender systems*, 2016, pp. 17–22.

[25] M. Wölbitsch, S. Walk, M. Goller, and D. Helic, "Beggars can't be choosers: augmenting sparse data for embedding-based product recommendations in retail stores," in *Proceedings of the 27th ACM conference on user modeling, adaptation and personalization*, 2019, pp. 104–112.

[26] J.-y. Song and B. Suh, "Data augmentation strategies for improving sequential recommender systems," *arXiv preprint arXiv:2203.14037*, 2022.

[27] J. Tang and K. Wang, "Personalized top-n sequential recommendation via convolutional sequence embedding," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, 2018, p. 565–573.
[28] M. Jing, Y. Zhu, T. Zang, and K. Wang, "Contrastive self-supervised learning in recommender systems: A survey," *ACM Trans. Inf. Syst.*, vol. 42, no. 2, 2023.
[29] I. Goodfellow, "Deep learning," 2016.
[30] B. Schifferer, J. Liu, S. Rabhi, G. Titericz, C. Deotte, G. De Souza P. Moreira, R. Ak, and K. Onodera, "A diverse models ensemble for fashion session-based recommendation," in *Proceedings of the Recommender Systems Challenge 2022*. Association for Computing Machinery, 2022, p. 10–17.
[31] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of big data*, vol. 6, no. 1, pp. 1–48, 2019.
[32] "Diginetica dataset." [Online]. Available: https://darel13712.github.io/rs_datasets/Datasets/diginetica/
[33] "Reatailrocket dataset." [Online]. Available: https://www.kaggle.com/datasets/retailrocket/ecommerce-dataset/
[34] M. Ludewig and D. Jannach, "Evaluation of session-based recommendation algorithms," *User Modeling and User-Adapted Interaction*, vol. 28, no. 4–5, p. 331–390, 2018.