Enhancing Sequential Next-Item Prediction through Modelling Non-Item Pages

Elisabeth Fischer, Daniel Schlör, Albin Zehe, Andreas Hotho
Data Science Chair
Center for Artificial Intelligence and Data Science (CAIDAS)
Julius-Maximilians-Universität (JMU)
Würzburg, Germany
Email: {elisabeth.fischer, schloer, zehe, hotho}@informatik.uni-wuerzburg.de

***Abstract*—By analyzing the sequence of historical interactions between users and items, sequential recommendation models can learn and infer user intent and make predictions about the next item of interest. Next to these item interactions, in most systems there are also interactions with pages not related to specific items, for example navigation pages, account pages, and pages for a specific category or brand, which may provide additional insights into the user interests. However, while there are various approaches to integrate additional information about items and users, the topic of non-item pages has been less explored. In order to fill this gap, we propose various approaches of representing these non-item pages (e.g, based on their content or a unique id) to use them as an additional information source for the task of sequential next-item prediction in transformer-based models. We create a synthetic dataset with non-item pages highly related to the subsequent item to show that the models are generally capable of learning from these non-item page interactions, and subsequently evaluate the improvements gained by including non-item pages contained in two real-world datasets. We adapt two state-of-the-art models capable of integrating item attributes and investigate the abilities of bi- and uni-directional transformers to extract user intent from additional non-item pages. Our results show that non-item pages are a valuable source of information, but representing such a page well is the key to successfully leverage them. The inclusion of non-item pages, when represented appropriately, increases the performance of state-of-the-art transformer models for next-item prediction.**

***Index Terms*—sequential recommendation, next-item prediction, non-item pages, transformer models**

I. Introduction

Sequential recommender models are useful building blocks for providing personalized recommendations to users. By analyzing the sequence of historical interactions of the user with items, sequential recommendation models can learn and infer user preferences and make predictions about the next item of interest. Recently, especially Deep Learning approaches based on recurrent neural networks (RNNs) [1], convolutional neural networks (CNNs) [2], and transformer-based models [3], [4] have received attention, typically modeling each interaction with an item as one step in the sequence. A significant amount of research has been conducted that incorporates additional item information, for example, textual and visual item features [5], [6], item attribute embeddings [7], [8] and categorical and numerical item features [9]. Besides click sequences of item pages, most systems for which recommenders are used also contain other types of information, including knowledge about users and often also pages that do not directly represent an item. While there are many papers exploring strategies to incorporate user-specific information [2], [10]–[12], the aspect of leveraging non-item information in sequences to further improve recommendation has been less explored. However, in many applications, interactions with these pages that are not items themselves or cannot even be associated with a particular item (hereafter referred to as non-item pages, cf. Section II-A) still provide useful information in the context of sequential recommendation. Examples of these pages include search results or blog posts that contain descriptions of entire item categories, which can be very helpful in detecting the user's interest when browsing through a website. Figure 1 shows a case where including these non-item pages can be useful: When only looking at the item history, we see that the user switches from pants to hiking shoes without any explanation. Including non-item pages, we see that the user is interested in hiking equipment, which explains their next step of looking at hiking shoes and enables the recommender to specifically suggest other hiking shoes as opposed to, for example, high heels.

We argue that ignoring this information, which is available in most real-world settings (cf. Section II-A), limits the potential of recommender systems to infer the user's intent. To address this limitation, we propose incorporating non-item pages and present various modeling approaches for non-item pages for sequential next-item prediction in transformer-based models. As a proof of concept for our approach in a controlled setting, we create a synthetic dataset with artificial non-item pages, which are highly related to the following item. Finally, we evaluate the effectiveness of our modeling on two real-world datasets from the e-commerce domain. Specifically, we consider three types of non-item pages: (1) a list of items, (2) a single non-item page, and (3) a list of non-item pages. We compare different types of non-item page representations (based on their content or a unique id) and choose two state-of-the-art transformer-based models to work with click sequences enriched by these representations. We adapt KeBERT4Rec, a bi-directional transformer model integrating item attributes based on BERT4Rec, and the corresponding variant of the uni-directional model SASRec. In summary, our proposed

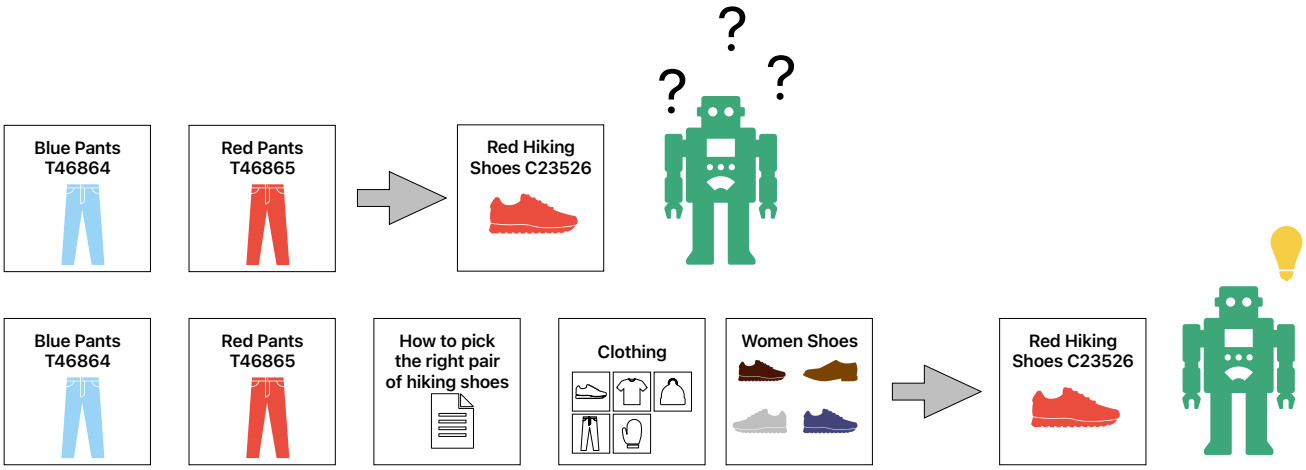

Fig. 1: Exemplary click sequence in an online store. The inclusion of non-item pages provides an explanation for the shift in user interest, enabling a more accurate prediction of their preferences.

modeling approaches and evaluation addresses the gap in current research regarding the modeling of non-item pages in sequential recommendation systems.

To sum up our contributions, we:

1) propose various modeling approaches for non-item pages,
2) adapt KeBERT4Rec and SASRec^{cross} to use arbitrary vector representations of pages and
3) systematically evaluate the use of non-item pages on one synthetic and two real-world datasets.

Our results show that non-items can reveal valuable information about the user's intent, but also the importance of finding good representations for non-item pages. Furthermore, both model adaptions, P-Bert4Rec and P-SASRec^{c}, are capable of learning correlations between item and non-item pages in different representations.

II. Background

A. Non-Item Pages

In this section we describe the types of non-item pages that we aim to include in our models in more detail.

In almost any setting where sequential recommendation is usually applied, the users' navigation does not consist exclusively of items or item pages. For example, web-shops often have additional pages, some of which are not relevant for inferring the users' intent (e.g., payment information or legal notes), while others are strongly related to items and therefore to the users' intent. Examples of these pages include search results (where a user has either typed out a search query or selected a set of filters like color or size), category pages (e.g., "women shoes" or "men shirts"), or blog pages containing additional content provided by the shop's owner (e.g., pages comparing different types of hiking shoes regarding their suitability for travelling in mountains or other areas).

These pages often provide even clearer hints towards the user's intent ("I want to go hiking in the mountains") than could be inferred only from their item clicks. Similar types of non-item pages exist also in other settings: for example, streaming services often provide info pages for actors (listing all of the movies or shows that the actor stars in), category pages (listing all content for a specific genre like horror, drama, ...) or tags (e.g., "high suspense"). While these kinds of pages are often not included in public datasets, they usually exist within the respective companies and can be used to improve recommendations.

In this paper, we use three types of non-item pages:

1) a list of items: e.g., a search result page
2) a single non-item page: e.g., a blog post with information about articles
3) a list of non-item pages: e.g., a category page for all kinds of clothing, which lists category pages for pants, shoes etc.

All of these pages can be represented by different strategies, for example based on their content (the text in the case of blog posts and the contained items in case of list pages) or separate unique ids. We propose possible representations in detail in Section IV-B.

B. Transformer-based models

As we want to study the general impact of including non-item pages, we now present state-of-the-art transformer models partially able to work with non-item pages. Through the attention mechanism, these architectures are capable of extracting relevant parts within a sequence and therefore suitable when dealing with possibly noisy user interactions. BERT4Rec [4] and SASRec [3] are both well-known bi- and uni-directional transformer models, but they do not allow attribute integration. Therefore, we use KeBERT4Rec proposed by Fischer et al. [7]

and the corresponding SASRec^{cross} model from [12], which is based on SASRec but trained with a cross entrophy loss. Both models, SASRec^{cross} and KeBERT4Rec, are depicted in Figure 2. They both consist of

(i) a common *embedding layer* for learning the representation of the sequence of interactions. This includes an embedding of the item ids, the positional and an embedding of multi-hot encoded item attributes.
(ii) The *transformer layer*, with L layers of transformer blocks to process the information. The transformer blocks are uni-directionally connected for SASRec^{cross} and bi-directional for KeBERT4Rec.
(iii) a *projection layer*, which uses the last output of the transformer and maps the results back to the item space with a linear layer. KeBERT4Rec utilizes a linear layer with GELU activation and reuses the item embedding matrix. SASRec^{cross} uses a linear layer for projecting.

KeBERT4Rec is trained with the cloze task [13], where items and their attributes are randomly masked for training. SASRec^{cross} is, unlike the original SASRec, trained with a cross-entropy loss for each sequence step.

III. Related Work

In this section, we first present an overview on deep sequential recommendation and then focus on architectures handling information apart from the sequence of item interactions themselves. With the increasing popularity of neural networks, neural architectures have been widely employed for modeling and learning sequences of user interactions [1]–[4], [14], [15]. While these works focused on the dependencies on a per-item level, some studies aimed at incorporating additional information along with the items, which can be sorted into two specific groups, either adding additional information for items or adding additional information to model the user.

The inclusion of additional item information has been the goal of several modifications of the models mentioned earlier. Hidasi et al. [5], for example, model textual and visual item features in separate RNNs and combine their output afterward, while Tuan et al. [6] include character-wise item descriptions using CNNs with 3D convolutions. NovaBert [8] uses bi-directional transformers and merges item attributes directly in the attention mechanism as query and value. Fischer et al. [7] allow the integration of categorical item attributes in BERT4Rec by adding attribute embeddings to the item embeddings. Transformers4Rec [9] uses models like XLNet [16] or Electra [17] from the Natural Language Processing domain and integrates categorical and numerical item features. Liu et al. [18] show how the inclusion of context (being time and location) can be utilised in a RNN.

Regarding user information, several approaches have been proposed, for example, CASER [2], SSE-PT [11], the work of Chen et al. [10] and TARN [19], which combines user, item and temporal context.. They treat the user separately from the sequence, while the approach by Fischer et al. [12] integrates the user representation as the first item in a sequence. This is more similar to our usecase, but in contrast to user representations, interactions with non-item pages can appear several times and have specific positions in the sequence.

Some works that participated in the SIGIR Coveo Challenge 2021 [20] are notable exceptions, as they include additional information that is not related to items or users. The Coveo dataset contains page views that are not related to items, as well as search results. Three participants in the challenge, [21]–[23], embed the page views as an item in the sequence for transformer models, while [24] uses the count of page views and the first page view as features for an ensemble model of LSTMs and Matrix Factorization. Moreira et al. [21] also used search queries as context for the whole sequence. The effect of including page events in the sequence is only reported by [23], which can be considered to be most similar to our work, however, since the focus of their work is particularly on the challenge, the benefits of non-item information and different modeling approaches are not systematically explored and cannot be investigated completely from outside the challenge's evaluation framework. In contrast to previous works, we verify our modeling approaches on several datasets and systematically model various types of non-item pages in the sequence to consider the dependencies to items in chronological sequence order.

IV. Methodology

In this section we discuss the problem setting, including our modeling approach for various types of non-item pages as well as the foundations of both transformer-based models and our adaptations.

A. Problem Setting

To formalize the problem setting, we closely follow [7] and introduce our notation as follows: Our work aims to solve the task of recommending items to a user given the sequence of previous item interactions and other page interactions. We define the item set as $\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\}$ and the set of non-items as $\mathcal{LP} = \{p_1, p_2, \ldots, p_{|\mathcal{LP}|}\}$. The set of users is defined as $\mathcal{U} = \{u_1, u_2, \ldots, u_{|\mathcal{U}|}\}$. For each user u, we can denote the sequence of interactions as $s_u = [i_1, i_2, \ldots, i_n]$ with $i \in \{\mathcal{V} \cup \mathcal{LP}\}$. The set of all sessions is defined as $\mathcal{S} = \{s_{u_1}, s_{u_2}, \ldots, s_{u_{|\mathcal{U}|}}\}$. Formally written, our goal is to solve the task of predicting the next item $v_{n+1} \in \mathcal{V}$ for every interaction i_n in each sequence $s_u \in \mathcal{S}$.

If the dataset also contains some kind of categorical attributes (e.g., tags like "category:shoes" or "genre:horror"), we define the set of all attributes as $\mathcal{A}$ and the set of all attribute combinations that occur for at least one page in the dataset as $\mathcal{P} \subset 2^{\mathcal{A}}$.

B. Modeling Non-Item Pages

For each of the three types of non-item pages (lists of items, lists of non-item pages and other non-item pages) we explore three options for representing the page: using or generating a unique id, generating a content-based id for the page, or generating an embedding for the page's content.

a) Unique Page ID (UPID): The simplest solution is to assign a unique id to each non-item page p_n. This is possible for any kind of page, for example by using the URL used to access the page, the title of the page or - as a last resort - by assigning a random id to each unique page. However, this comes with several drawbacks: First, there are instances where the generated unique id is not informative. This is the case if a URL is generic and content is generated on the fly. And even if there is an id, it can also be too specific (e.g. the URL containing a search query). Moreover, ids may be too rare for a model to learn from them, and any content-related meaning will also be lost. The vocabulary size will increase and become unmanageable, and the models will learn to predict non-items, even though the goal is item recommendation. While predictions can be filtered afterwards, valuable capacity of the model will be used for a task irrelevant to next-item prediction.

b) Content-based Page ID (CPID): To tackle these challenges, we utilize the content of non-item pages. We can use categorical attributes to create meaningful and shared ids for similar non-item pages. In the easiest case, a non-item page $p_n \in \mathcal{LP}$ already has some categorical attributes $p_n^a = \{a_1, a_2, \ldots, a_{|p_n^a|}\}$, $a \in \mathcal{A}$, assigned, for example because it is a category list page (all items from category "shoes") or because it has been manually tagged. If this is not the case, for list pages we can construct attributes from the attributes of the items contained in the list. For non-list pages, this strategy is not applicable if there are no tags assigned to the page.

c) Page Embedding (PE): By building ids from attributes, as proposed in the previous paragraph, our vocabulary still grows by size $|\mathcal{P}|$. Therefore, we propose to use one single placeholder id to represent all non item pages and encode the content of a page with an embedding, mitigating the issue of increasing the vocabulary size. This also allows the use of any vector representation for the page. This representation $p_n^e \in \mathbb{R}^m$ (m: dimension of the embedding vector) can be extracted either from existing categorical attributes of a page p_n, from textual content, images, or a search query. In general, this embedding representation is possible for any kind of non-item page.

C. Adapting KeBERT4Rec and SASRec^{cross} for latent page representations

Both KeBERT4Rec and SASRec^{cross} can be used to include non-items with either unique ids or non-items represented by categorical attributes. But, since we also want to use arbitrary page representations, we need to adapt both models. The original embedding layer consists of three different embeddings.

(i) an embedding $E_\mathcal{V} \in \mathbb{R}^{|\mathcal{V}| \times d}$ of the identifier,
(ii) an embedding $E_O \in \mathbb{R}^{N \times d}$ to encode the position of the items in the sequence, with N as the maximum input sequence length.
(iii) a multi-hot encoding of the item attributes $E_A \in \{0,1\}^{|\mathcal{A}|}$ and a linear layer for resizing E_A to the hidden size d: $l_A(x) = Wx + b$ with the weight matrix $W \in \mathbb{R}^{|\mathcal{A}| \times d}$ and bias $b \in \mathbb{R}^d$.

We modify the embedding layer as follows to include page representations[1]. When we create unique ids for each p, the embedding of the id will be $E_\mathcal{V} \in \mathbb{R}^{|\mathcal{V} \cup \mathcal{LP}| \times d}$ instead. When including the page information as categorical attributes, the embedding is defined as $E_\mathcal{V} \in \mathbb{R}^{|\mathcal{V}+1| \times d}$, as we add only one general token for all pages p. Assuming that there is a latent representation available for p called $\hat{r}_p \in \mathbb{R}^{|R|}$, we add one linear layer $l_R(x) = Wx + b$ with the weight matrix $W \in \mathbb{R}^{R \times d}$ and bias $b \in \mathbb{R}^d$ to scale the latent representation to the hidden size d. The original embeddings are summed up to create the final output of the embedding layer, so we follow this idea to include l_R. For each timestep t, we use the id embedding $e_t = i_t E_\mathcal{V}$ of interaction i_t, the positional embedding $o_t = tE_O$, the keyword embedding $a_t = l_A(E_A(p_t^a))$ and the scaled latent representation $r_t = l_R(\hat{r}_{p_t})$ to compute $h_t^0 = e_t + o_t + a_t + r_t$, which is then used as the input to the transformer layer. The other layers of the models remain unchanged. For training of KeBERT4Rec, we make sure to mask attributes and representations whenever an interaction is masked. We refer to our models as P-Bert4Rec and P-SASRec^c.

V. Datasets

This section introduces the datasets used in our experiments. The statistics for all datasets can be found in Table I.

A. Synthetic Dataset SynDS

To assess the effectiveness of our proposed approaches in enabling transformer models to learn from non-item pages, it is essential to have a dataset that guarantees the presence of valuable information within non-item pages. To ensure this, we create a synthetic dataset SynDS, based on the ML-20m dataset. The ML-20m[2] consists of movie ratings from users and is a popular benchmark for sequential recommendation. For each movie metadata in the form of genres is available. We use the preprocessing steps from [4] and split the data into 80% for training, 10% for validation and 10 % for testing.

As the dataset does not contain any non-item interaction, we create them based on the movie genres. We create three dataset variants, containing different levels of information as follows:

1) Prev-SynDS: In our first experiment, we want to explore the benefit of highly informative non-item pages. Therefore, for each movie interaction in a sequence, we add a non-item page tagged with the movie's genres immediately prior to that movie. This simulates that the user first navigates to a category page listing all movies of a particular genre and then selects a specific movie afterwards. Each added non-item page contains highly relevant information for the recommendation task, as the genre for the next movie is now already known. For example, instead of the movie sequence "Alien → Jaws → Shrek" the enriched sequence would be "Horror|Sci-Fi →

[1]Our code is available at https://anonymous.4open.science/r/non-item-transformers-8301/

[2]https://grouplens.org/datasets/movielens/20m/

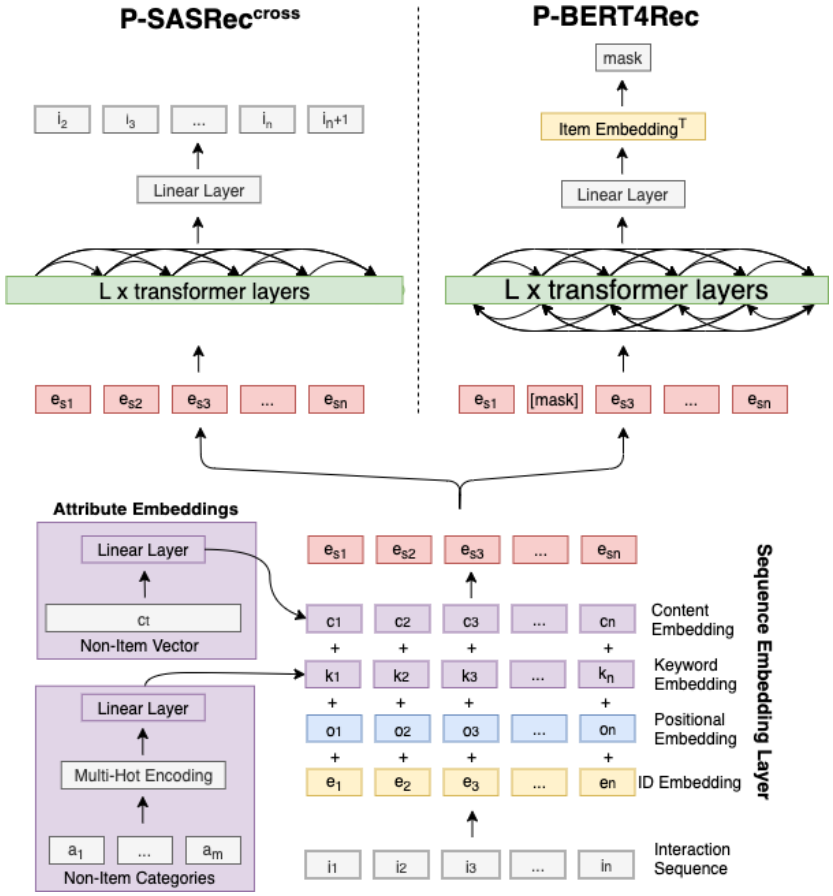

Fig. 2: The modified architectures of KeBERT4Rec and SASRec^{cross}. Any vector representation of a non-item page can be included in the embedding layer by scaling it with a linear layer first and adding it with the other embeddings. SASRec^{cross} contains only forward connections in the transformer layer, while BERT4Rec works bi-directionally.

Alien → Action|Horror → Jaws → Adventure|Children → Shrek".

2) Random-SynDS: Covering the case of non-item-pages containing no useful information at all, we generate a second dataset with completely random non-item pages to explore the susceptibility of the models to noise potentially introduced by adding non-item pages. We therefore add a non-item page tagged with random genres before each movie interaction. For the example above, the sequence could look like "Musical|Children → Alien → Comedy|Adventure → Jaws → Documentary → Shrek".

3) Group-SynDS: Besides both extreme variants - Prev-SynDS contains a strong prior and Random-SynDS contains only noise - we also explore a more realistic setting by only adding a non-item-page when the genre of the following movie is from different genres than the movies before. For example, instead of "Horror|Sci-Fi → The Thing → Horror|Sci-Fi → Alien → Action|Horror → Jaws" as in Prev-SynDS, we generate "Horror|Sci-Fi → The Thing → Alien → Action|Horror → Jaws". This reduces the number of non-item interactions to about 19 million.

For each dataset, we can represent non-item pages either by using the genre combination as an id or by an page embedding with the genres as keywords.

Note that the results obtained on these datasets are not comparable to the original ML-20m anymore, as we leak information about the following movie. The datasets are only intended to show that the models are generally capable of leveraging the information encoded in the additional non-item pages in a setting where we can ensure that the pages contain valuable information, or in the case of Random-SynDS that the models do not suffer from additionally introduced noise.

B. Coveo

Published for the SIGIR 21 challenge, the Coveo[3] dataset [20] is an extensive e-commerce dataset that captures browsing and search interactions. The captured browsing behavior consists of interactions with a product page or the view of a web page. A view can be related to a product, but is often only represented by its hashed URL. The dataset also provides search requests, including the embedded search query as a vector of length 45 and the list of retrieved items.

[3]https://www.coveo.com/en/ailabs/sigir-ecom-data-challenge

Moreover, the dataset includes metadata such as hierarchical categories, price, and preprocessed text and image embeddings for products. As our focus in this paper is on incorporating non-item pages, we are not directly utilizing them. However, we use the categories of items to build representations for the lists of retrieved items from search requests. For preprocessing, we drop all duplicate item interactions which includes page views linked to a product. We condense all interactions with the same item or page in a row into one single interaction and ensure that each sequence contains a minimum of two items afterwards. Furthermore, each item and page view must occur at least 5 times in the dataset.

As we want to examine both types of non-item pages in the dataset separately, we split the data in two sets: For *Coveo-Pageview*, we exclusively use item interactions and page views as non-item pages. As there is no further information available, we can only represent a page view with separate ids, namely their hashed URL. For *Coveo-Search*, we reduce the data set to sessions containing at least one search event. For each search event, we use the list of retrieved items as a non-item page. We create several representations for search pages: First we compute the three most frequent categories of the products in the retrieved list. As a second categorical representation we use the categories of the first product in the result list. Both can now be represented by a category-based id or by embedding the categories. We also use the product id of the first result directly as a representation, therefore using an item to represent the non-item page and not increasing the vocabulary at all. As all these representations depend on the quality of the retrieved item list, we also use the search query embedding vector, which is an direct expression of a user's interest. We split both Coveo-Pageview and Coveo-Search into 70% for training and 15% for test and validation each by time.

C. Fashion

The Fashion dataset is composed of interactions with product and list pages from a major online fashion store, collected over 20 days. A product page shows a single item, while a list page displays multiple items of particular (sub-)categories with additional filters applied by the user. Categories and filters are available as attributes for each page, and while the information is partially overlapping, filters usually contain more detailed information.

For example, a list page with the category "shoes" can have more specific filters, such as "type:shoes", "color:black" and "material:leather". Multiple interactions with the same product or page in a row are condensed into a single interaction. Furthermore, we ensure a minimum sequence length of 3, a maximum of 200 and ensure an item occurs at least 5 times in the data as in [4]. The dataset is split by time into 70% for training, 15% for validation, and 15% for testing.

VI. Experiments

In this section we introduce the research questions we aim to answer. We provide an explanation of the experimental setup, present the conducted experiments, and subsequently discuss our results.

In our experiments, we aim to address the following four research questions:

- Can transformer models benefit from the inclusion of non-item pages for predicting the next item?
- Are there benefits from including non-item pages completely lacking content information in a real-world dataset?
- Can we find good representations for utilizing search requests as non-item interactions?
- Can we find good representations to integrate list pages as non-item pages?

We train all our models with the AdamOptimizer and, unless otherwise specified, use the parameters as in [4]. We set transformer heads = 2, transformer layers = 2, dropout = 0.2, hidden size = 32 for all experiments. For evaluation, we ensure that the target is an actual item instead of a page and filter all suggestions of pages before computing the metrics and report the *Hit Rate (HR)* and the *Normalized Discounted Cumulative Gain (NDCG)* at k = 1,5,10.

RQ1: Can transformer models benefit from the inclusion of non-item pages for predicting the next item?

Our first objective is to investigate whether transformer models can effectively utilize non-item pages. To accomplish this, we created the SynDS datasets based on ML-20m with three variants: Prev-SynDS with non-item pages contains the exact genre information for the next movie, Random-SynDS contains non-item-pages with pure noise and Group-SynDS includes fewer non-item pages with the genre information for the following movies condensed.

For all of the artificial pages, we can now generate unique ids by concatenating categories, or assign one placeholder id and encode the categories as attributes. We train P-Bert4Rec and P-SASRec^c for both cases as well as a model utilizing the sequence with items only. As a second baseline, we also include a most popular baseline given the correct genres of the target movie. We train all models for 100 epochs with a batch size of 64 and a maximum sequence length of 200. Table II illustrates the results.

The genre-specific most popular baseline performs nearly half as good as the transformer based models on items only, which shows that genres are indeed a good indicator for the next movie.

The results of our first experiment including meaningful non-item pages for each movie interaction in Prev-SynDS compared to the models trained on pure item sequences are therefore not surprising, as the genre of a movie is available at the preceding non-item page. Still, the increase of up to 0.62 for P-SASRec^c and P-Bert4Rec in HR@10 shows both models are capable of successfully utilising the information from non-item pages. Another observation is that P-SASRec^c performs better than P-Bert4Rec, with the advantage becoming more prominent when including non-item interactions. This falls in line with previous findings of Fischer et al. [12].

TABLE I: Statistics of the preprocessed datasets ML-20m, Coveo-Search, Coveo-Pageview and Fashion, showing the number of users, items, pages, attributes and their combinations, filters and their combinations, interactions with items and pages and the average session length without and with pages.

dataset	$\lvert\mathcal{U}\rvert$	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{LP}\rvert$	$\lvert\mathcal{A}_{cat}\rvert$	$\lvert\mathcal{P}_{cat}\rvert$	$\lvert\mathcal{A}_{fil}\rvert$	$\lvert\mathcal{P}_{fil}\rvert$	$\#i_{\mathcal{V}}$	$\#i_P$	Avg. sess. len/with i_P
SynDS	138,493	26,744	1,329	20	1,329	-	-	20 m	20 m	144.4/288.8
Coveo-PV	1,318,922	29,268	76,080	-	-	-	-	8,6 m	7.1 m	6.6/11.9
Coveo-S	134,881	27,013	166	112	166	-	-	895,760	231,133	6.6/8.3
Fashion	199,474	8,412	27,863	204	7,207	5,647	27,863	4.3 m	1.3 m	21.7/28.2

TABLE II: Hitrate and NDCG for P-Bert4Rec and P-SASRec^c on the SynDS dataset.

Model	Metric	Genre-POP	Items Only	Prev-CPID	Prev-PE	Random-CPID	Random-PE	Group-CPID	Group-PE
P-Bert4Rec	HR@1	0.010	0.019	**0.393**	0.308	0.022	0.003	0.397	0.016
	HR@5	0.031	0.078	**0.662**	0.605	0.083	0.014	0.652	0.040
	HR@10	0.050	0.133	**0.753**	0.719	0.135	0.027	0.745	0.061
	NDCG@5	0.020	0.049	**0.537**	0.465	0.052	0.008	0.533	0.028
	NDCG@10	0.026	0.066	**0.566**	0.503	0.069	0.012	0.563	0.035
P-SASRec^c	HR@1	0.010	0.027	**0.426**	0.384	0.024	0.025	0.426	0.371
	HR@5	0.031	0.103	**0.699**	0.681	0.086	0.097	0.681	0.659
	HR@10	0.050	0.164	**0.785**	0.780	0.143	0.158	0.768	0.754
	NDCG5	0.020	0.066	**0.572**	0.543	0.055	0.061	0.563	0.525
	NDCG@10	0.026	0.085	**0.600**	0.575	0.073	0.080	0.591	0.556

The integration of non-item pages as separate ids (CPID) yields better results than the integration via attributes (PE) for P-Bert4Rec with a difference of 3.4% for HR@10, but with 0.5% less for P-SASRec^c. Using CPID is generally less flexible than PE and uses more space, but shows better performance. One reason could be the relatively small number of non-item pages (≈ 1.300), so it might be easier for the model to learn a fixed number of ids instead of a more complex embedding space.

In case the non-item-pages do not contain useful information, transformer models are mostly able to ignore the irrelevant information, as shown by the performance on the Random-SynDS. For P-SASRec^c the performance is slightly lower compared to the items-only baseline, for P-Bert4Rec with CPID it is slightly better, but within a similar range.

P-Bert4Rec is struggling with a big performance drop when adding noise for content embedded non-item-pages, as well as when adding the weaker signal in Group-PE. P-SASRec^c on the other hand can leverage the information.

Overall, these results show that both models in general can extract the signal embedded in non-item pages, and both representations, as tokens and embedded categories, are valid approaches for integrating non-item pages, although the combination of P-Bert4Rec and page embeddings might not work.

RQ2: Are there benefits from including non-item pages completely lacking content information in a real-world dataset?

As we have seen that embedding pages via ids works in an artificial setting, we endeavor to verify this with the Coveo-Pageview dataset on real-life data. In this dataset, there is only a unique hashed URL for non-product interactions available and no other way to create a representation. Using the URL as an id gives us the results shown in table III. We train all models

TABLE III: Hitrate and NDCG for P-Bert4Rec and P-SASRec^c on the Coveo-Pageview dataset.

Model	Metric	Items Only	URL-UPID
P-Bert4Rec	HR@1	0.094	**0.097**
	HR@5	0.330	**0.331**
	HR@10	0.413	**0.419**
	NDCG@5	0.216	**0.217**
	NDCG@10	0.243	**0.246**
P-SASRec^c	HR@1	**0.146**	0.133
	HR@5	0.351	**0.357**
	HR@10	0.429	**0.443**
	NDCG5	**0.253**	0.248
	NDCG@10	**0.278**	0.276

for 50 epochs with a batch size of 64 and a maximum sequence length of 30. For P-Bert4Rec we see only small improvements and mixed results for P-SASRec^c. Only HR@5 and HR@10 could improve through the use of non-item pages. One reason for this might be the number of non-item pages, which is more than twice as many as the number of items in the Coveo-Pageview dataset (table I) and therefore their usage is very sparse. The growing vocabulary might also hinder the models, as already discussed in section IV-B in more detail. As we do not have much information on the types of non-item pages in the dataset, it is also possible that the non-item pages contain too little useful information and too many random and noisy pages. Overall, there is a small benefit of using the non-item pages, as the models can still extract some behavior patterns, but the improvements are limited.

RQ3: Can we find good representations for utilizing search requests as non-item interactions?

Next, we investigate content-based page representations. In the Coveo-Search dataset, we have non-item interactions

TABLE IV: Hitrate and NDCG for P-Bert4Rec and P-SASRec^c on the Coveo-Search dataset.

Model	Metric	Items Only	FreqCat-CPID	FirstCat-CPID	FirstItem-CPID	Query-PE	FreqCat-PE	FirstCat-PE
P-Bert4Rec	HR@1	0.083	0.086	0.086	**0.089**	0.088	0.000	0.000
	HR@5	0.297	0.309	0.307	0.315	**0.321**	0.004	0.005
	HR@10	0.378	0.392	0.386	0.398	**0.401**	0.012	0.010
	NDCG@5	0.193	0.202	0.200	0.205	**0.208**	0.002	0.002
	NDCG@10	0.220	0.229	0.226	0.232	**0.234**	0.004	0.004
P-SasRec^c	HR@1	0.123	0.119	0.119	0.118	**0.124**	0.122	0.122
	HR@5	**0.309**	0.297	0.297	0.301	**0.309**	0.304	0.304
	HR@10	0.379	0.365	0.365	0.382	**0.388**	0.376	0.376
	NDCG5	**0.220**	0.212	0.212	0.212	**0.220**	0.216	0.216
	NDCG@10	0.243	0.234	0.234	0.238	**0.245**	0.240	0.240

TABLE V: Hitrate and NDCG for P-Bert4Rec and P-SASRec^c on the Fashion dataset.

Model	Metric	Items Only	Cat-CPID	Filter-CPID	FirstItem-CPID	Cat-PE	Filter-PE
P-Bert4Rec	HR@1	0.164	0.186	0.169	0.134	0.180	**0.188**
	HR@5	0.428	0.439	**0.446**	0.422	0.439	**0.446**
	HR@10	0.517	0.521	**0.534**	0.510	0.524	0.533
	NDCG@5	0.303	0.320	0.314	0.285	0.316	**0.324**
	NDCG@10	0.303	0.346	0.343	0.314	0.343	**0.352**
P-SASRec^c	HR@1	0.215	0.212	0.215	0.204	0.223	**0.224**
	HR@5	0.450	0.455	0.458	0.437	0.472	**0.475**
	HR@10	0.520	0.529	0.534	0.512	0.548	**0.555**
	NDCG5	0.341	0.342	0.345	0.329	0.356	**0.359**
	NDCG@10	0.364	0.366	0.370	0.353	0.381	**0.384**

caused by search requests. A search request represents a clear expression of the user's intent and should contain valuable information. We compare three different CPID tokens from the list of retrieved items to represent a search interaction:

(a) the category, (b) the product id of the first item and (c) the concatenation of the three most frequent categories in the list. In case of the product id, no new ids are added to the vocabulary. For both categories we also test the attribute-based approach with multi-hot embeddings (PE) based on the categories instead of unique ids. Besides these trials, we also evaluate using the provided query embedding, as it directly represents the search intent and not the feedback from the search engine. We use a batch size of 64, a maximum sequence length of 30 and train for 100 epochs.

Table IV shows our results. The category-based CPID embeddings perform very similarly to each other, but yield only a small improvement for P-Bert4Rec and even lower the performance for P-SASRec^c in the HR@10. The first item id embedding shows more notable improvements for P-Bert4Rec for HR@10 (+0.01) and NDCG@10 (+0.012), but P-SASRec^c still performs slightly worse, except for HR@10. Using the query embedding, we get the highest overall scores for both models. While the improvements for P-Bert4Rec are evident (+2.3% for HR@10), the performance for P-SASRec^c is very close to the base model. Here, the advantage of using more flexible page embeddings becomes evident, as the CPID embeddings cannot represent the query directly.

With categories embedded as attributes, the P-Bert4Rec model completely fails, but also P-SASRec^c drops in performance. It seems that a simplified category representation is not an adequate estimate for a search request. Although the results for P-Bert4Rec are surprisingly low in comparison to P-SASRec^c, P-Bert4Rec already showed a huge performance drop in our first experiment (section VI) when using a content embedding. P-Bert4Rec seems to be more receptive when adding information, but could also suffer more from a non-optimal or weak representation in return. In conclusion, the query embedding is intuitively the closest representation of the real intent of a search interaction and using it as a representation does improve the performance for both models.

RQ4: Can we find good representations to integrate list pages as non-item pages?

We further explore the potential benefits of integrating non-item pages with content based page ids and page embeddings based on categories for list pages. In the Fashion dataset categories and filters are available as direct descriptions of a list pages' content. We test both CPID and PE-based approaches for categories and filters. The first item on a list page is also used as a representation. Again, in this special case, the vocabulary will not grow.

Looking at the results in table V, we can again verify P-SASRec^c as the better performing model overall. With the FirstItem-UPID all metrics for both models drop, which suggests that this is not a helpful representation, similar to the results in VI. Looking at the category-CPID we see improvements in all metrics for P-Bert4Rec, with up to 4.6% in the NDCG@10. Using the filters as CPID, the gain on the NDCG is slightly lower, but we get the overall best scores for HR@5 (44.6%) and HR@10 (53.4%).

For P-SASRec^c we also see improvements, albeit smaller, except for HR@1. Integrating the category and filter as at-

tributes further improves the performance of both models. The best overall results are achieved by using the filters, which improves the predictions for both models by about 2% for each metric. In the case of the Fashion dataset, categorical representations seem to work well. Filters might be more useful than category information as they contain more detailed information. Interestingly, the representation as attributes works better than the encoding as id for this dataset. This might be related to the higher number of attributes and their combinations, which differs between the datasets.

VII. Conclusion

The effect of the inclusion of item and user properties for sequential item recommendation has been studied, but non-item interactions have been an afterthought, although they are present in most recommendation setups. In our paper, we close this gap by giving an overview on the various types of non-item interactions and proposing different modeling approaches for them. We adapt KeBERT4Rec and SASRec^{cross} to include arbitrary content-based representations. Evaluation on one artificially constructed and two real-life datasets show that non-item interactions can indeed improve sequential recommendation. P-Bert4Rec profits slightly more from the inclusion of non-items, while P-SASRec^{c} shows the best overall performance, but both transformer-based models are able to leverage non-items and handle noise. Selecting meaningful non-item pages with content related to the item is one important step in leveraging non-item pages, but our studies also highlight the importance of choosing a good representation for non-item interactions for each dataset and scenario individually. For future work, it would be compelling to test our approach in more advanced sequential models. We could also investigate other ways to fuse non-item embeddings with the sequence.

References

[1] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," in *ICLR (Poster)*, Y. Bengio and Y. LeCun, Eds., 2016.

[2] 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*, 2018, pp. 565–573.

[3] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 197–206.

[4] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management - CIKM 19*. ACM Press, 2019.

[5] B. Hidasi, M. Quadrana, A. Karatzoglou, and D. Tikk, "Parallel recurrent neural network architectures for feature-rich session-based recommendations," in *Proceedings of the 10th ACM Conference on Recommender Systems*, ser. RecSys '16. New York, NY, USA: ACM, 2016, pp. 241–248.

[6] T. X. Tuan and T. M. Phuong, "3d convolutional networks for session-based recommendation with content features," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, ser. RecSys '17. New York, NY, USA: ACM, 2017, pp. 138–146.

[7] E. Fischer, D. Zoller, A. Dallmann, and A. Hotho, "Integrating keywords into BERT4Rec for sequential recommendation," in *KI 2020: Advances in Artificial Intelligence*, 2020.

[8] C. Liu, X. Li, G. Cai, Z. Dong, H. Zhu, and L. Shang, "Non-invasive self-attention for side information fusion in sequential recommendation," *arXiv preprint arXiv:2103.03578*, 2021.

[9] G. de Souza Pereira Moreira, S. Rabhi, J. M. Lee, R. Ak, and E. Oldridge, "Transformers4rec: Bridging the gap between nlp and sequential/session-based recommendation," in *Fifteenth ACM Conference on Recommender Systems*, 2021, pp. 143–153.

[10] Q. Chen, H. Zhao, W. Li, P. Huang, and W. Ou, "Behavior sequence transformer for e-commerce recommendation in alibaba," in *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. ACM, Aug. 2019.

[11] L. Wu, S. Li, C.-J. Hsieh, and J. Sharpnack, "Sse-pt: Sequential recommendation via personalized transformer," in *Fourteenth ACM Conference on Recommender Systems*, 2020, pp. 328–337.

[12] E. Fischer, A. Dallmann, and A. Hotho, "Personalization through user attributes for transformer-based sequential recommendation," in *Recommender Systems in Fashion and Retail*, H. J. Corona Pampín and R. Shirvany, Eds. Cham: Springer Nature Switzerland, 2023, pp. 25–43.

[13] W. L. Taylor, ""cloze procedure": a new tool for measuring readability." *Journalism & Mass Communication Quarterly*, vol. 30, pp. 415–433, 1953.

[14] C. Xu, P. Zhao, Y. Liu, J. Xu, V. S. S.Sheng, Z. Cui, X. Zhou, and H. Xiong, "Recurrent convolutional neural network for sequential recommendation," in *The World Wide Web Conference on - WWW'19*. ACM Press, 2019.

[15] 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, pp. 1419–1428.

[16] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," *Advances in neural information processing systems*, vol. 32, 2019.

[17] K. Clark, M. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: pre-training text encoders as discriminators rather than generators," *CoRR*, vol. abs/2003.10555, 2020. [Online]. Available: https://arxiv.org/abs/2003.10555

[18] Q. Liu, S. Wu, D. Wang, Z. Li, and L. Wang, "Context-aware sequential recommendation," 2016.

[19] Q. Zhang, L. Cao, C. Shi, and Z. Niu, "Neural time-aware sequential recommendation by jointly modeling preference dynamics and explicit feature couplings," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 10, pp. 5125–5137, 2021.

[20] J. Tagliabue, C. Greco, J.-F. Roy, B. Yu, P. J. Chia, F. Bianchi, and G. Cassani, "Sigir 2021 e-commerce workshop data challenge," 2021.

[21] G. de Souza P. Moreira, S. Rabhi, R. Ak, M. Y. Kabir, and E. Oldridge, "Transformers with multi-modal features and post-fusion context for e-commerce session-based recommendation," 2021.

[22] S. Ishihara, S. Goda, and H. Arai, "Adversarial validation to select validation data for evaluating performance in e-commerce purchase intent prediction," 2021.

[23] E. Fischer, D. Zoller, and A. Hotho, "Comparison of transformer-based sequential product recommendation models for the coveo data challenge," *SIGIR Workshop On eCommerce*, July 2021.

[24] Y. Sakatani, "Session-based recommendation using an ensemble of lstm- and matrix factorization-based models," 2021.