Adapting Sequential Recommender Models to Content Recommendation in Chat Data using Non-Item Page-Models

Albin Zehe[1], Elisabeth Fischer[1], Jonas Kaiser[1], Toni Wagner[2] and Andreas Hotho[1]

[1]Data Science Chair, CAIDAS, University of Würzburg, Am Hubland, 97074 Würzburg, Germany
[2]vAudience, John-Skilton-Straße 22, 97074 Würzburg, Germany

Abstract

Most research in sequential recommender models has focused on sequences that are purely made of items (e.g, movies, page clicks), excluding additional elements in the sequence that may provide more information for the next relevant item. Recently, it has been proposed to include non-item pages (e.g., list pages or blog posts), in order to represent the users' intent more clearly. In this paper, we transfer the same modelling principle to sequences made of items and text messages. This enables us to adapt arbitrary sequential recommender methods to a new application area: We can use any sequential recommendation model to recommend links to relevant content in a chat setting, using the history of previous messages and mentioned items as context. We evaluate our models on four different datasets and show that we can identify content relevant to the conversations well when using pre-trained embeddings for the messages in the conversations.

Keywords

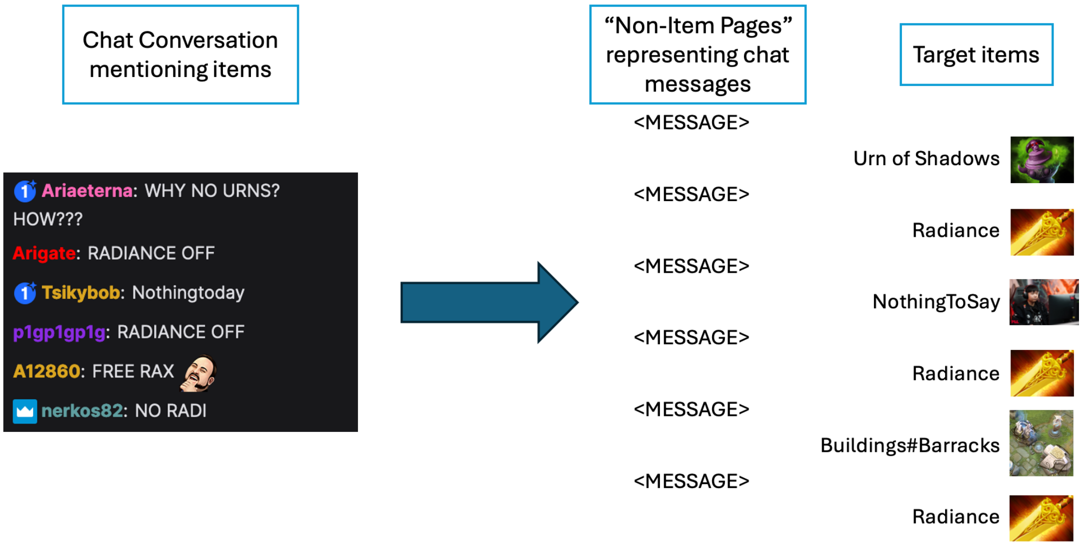
Sequential Recommendation, Non-Item Pages, Content Enrichment

1. Introduction

Recommender systems play a vital role in many application settings, ranging from online shops over streaming services to websites that want to suggest other relevant content based on their users' browsing history. Although sequential recommendation has been extensively studied and several model architectures [1, 2, 3] and several ways to incorporate additional information about users or items [4, 5, 6] have been proposed, they are not directly applicable in settings where the context is not given in the form of click sequences. Recently, however, applications based on chat or dialogue data have become increasingly relevant. Here, context is provided in the form of text messages rather than previous clicks. The use cases for this kind of recommender are manifold, ranging from chatbots that can observe chat rooms and enrich them with additional information about the topics being discussed to conversational recommender systems, where the bots actively engage in the conversation and try to find the users' interest by asking questions. Although the setting is, at first glance, very different from traditional sequential recommenders, we show that we can adapt arbitrary sequential recommender models to this task using a modeling approach for non-item pages proposed in Fischer et al. [7, 8]. The main idea of this approach is to incorporate additional information into sequential recommender models by including non-item pages in the sequences. These non-item pages can be, for example, list pages, category pages, or blog posts describing articles of a certain category. They are inserted into the sequence and represented by an embedding vector like regular items, but are never the target of the recommender. In the same way, we can build a sequence consisting of chat messages and items (representing relevant content in the sequence), treating the chat messages like non-item pages. In this paper, we evaluate the suitability of sequential recommender models in their basic version and when using non-item page modeling on four datasets of conversations about movies, website content

Sixth Knowledge-aware and Conversational Recommender Systems (KaRS) Workshop @ RecSys 2024, September 14–18 2024, Bari, Italy.

zehe@informatik.uni-wuerzburg.de (A. Zehe); elisabeth.fischer@informatik.uni-wuerzburg.de (E. Fischer); jonas.kaiser@informatik.uni-wuerzburg.de (J. Kaiser); toni.wagner@vaudience.ai (T. Wagner); hotho@informatik.uni-wuerzburg.de (A. Hotho)



Ariaeterna: WHY NO URNS? HOW???

Arigate: RADIANCE OFF

Tsikybob: Nothingtoday

p1gp1gp1g: RADIANCE OFF

A12860: FREE RAX

nerkos82: NO RADI

Figure 1: A schematic visualisation of our task and approach. Our goal is to identify items that are mentioned in a conversation, possibly by very noisy names (e.g. "RADI" referring to the item **Radiance** and "RAX" referring to **Buildings#Barracks**). To this end, we represent each message in a conversation as a "non-item" and train a model to recommend pages with information about the items referred to in the messages.

or sports events. In all of these datasets, our goal is to detect which items (e.g., players, movies) are currently discussed in the conversation and to recommend relevant links to the users. Figure 1 gives a visualization of our task and approach.

Our contributions in this paper are as follows: (a) We adapt sequential recommender models to our task of providing additional content relevant to the conversation. (b) We introduce four datasets for this task, partially derived from existing datasets, enabling further research. (c) We compare our approach with baseline recommender models, showing that it is effective in finding relevant content for conversations.

2. Related Work

As neural networks have gained widespread popularity, they have become a popular choice for modeling and learning user behavior sequences. Sequential recommender models have been developed based on recurrent neural networks [1] or convolutional neural networks [2, 9, 10]. Numerous architectures have since adapted the attention mechanism [11] for sequential recommendation [3, 12, 13, 14, 15].

All of these works utilized only the item sequences itself, but the inclusion of additional item or user information has also been investigated by several studies. Item features have been modeled in RNNs [4, 16], CNNs [5], and transformer models [6, 17, 18, 19, 20]. Some approaches merge additional item information directly in the attention layer [21, 22]. There are also works that study the inclusion of user representations in a similar way for different architectures [2, 23, 24, 25, 26]. Information that is not directly related to a user or an item, yet still part of the sequence, has not been investigated as extensively. Some works for the Coveo challenge [27] are an exception, as they include interactions that are not tied to specific items. These non-item interactions are included in sequential recommender models by [28, 29, 30], but they are represented the same way as item interactions. An explicit modeling and a formal definition of non-item pages was first introduced for transformer-based models by Fischer et al. [7]. A more extensive study [8] expands the setup to a wider set of sequential recommender models and allows the integration of non-items with generic embedding representations (see Section 3.1).

Providing relevant content items based on textual information can also be viewed as a form of Entity Linking (EL). Usually, the task involves the recognition of entities explicitly mentioned in texts and their disambiguation [31]. Since many conversations feature nondirect references to entities, where context clues must be leveraged to identify the target, many EL approaches face difficulties recognizing and resolving such references. This special setting – Implicit Entity Linking – has been investigated primarily in the setting of Twitter posts [32, 33, 34]. These approaches, however, rely heavily on manual feature selection rather than current deep learning models. Additionally, due to the setting, they often only consider limited conversational context directly. In this work, we investigate how sequential recommendation models can be adapted to this task. Applying these models to implicit entity linking is beneficial, since it enables us to directly transfer any progress in sequential recommendation to implicit entity linking.

Our setting is also related to conversational recommendation [35]. However, while the goal there is to lead a conversation with a user, ask questions designed to narrow down the space of possible items, and finally provide a recommendation, our goal is to detect the items that are mentioned in a conversation to recommend pages providing additional content relevant to the conversation.

Our model can be used as one component of a conversational recommender system, for example to identify the movies/items that the user mentions and provide them to the recommender model in charge of determining new suggestions.

3. Background

3.1. Sequential Item Recommendation with Non-Item Pages

Non-Item Pages have been proposed recently as a way to incorporate sequence elements other than the items that are potential targets for recommendation into sequential recommender systems [7, 8]. In essence, they can be of arbitrary nature, representing additional content related to the items, for example, a page listing items of a specific category, a search query, or a blog post about a set of items. [8] propose different ways of representing these non-item pages, depending on their nature. The two representations relevant for our setting are: (1) Unique Page-ID (UPID): A unique identifier can always be assigned to any non-item page and integrated into the sequence as an item. Possible drawbacks are high sparsity and a growing vocabulary. (2) Page Embedding (PE): A single placeholder id is used to represent all non-items. The information about the non-item is included by adding an embedding representation of the non-item to the id in the embedding layer.

In this work, we use this non-item page modeling to include chat messages as information about the current topic in a conversation.

3.2. Task Description

Our task is to enrich conversations in chat systems with content that is relevant to the current conversation. We model this task as a sequential recommendation problem, treating chat messages as non-item pages and relevant content as items, for example, in the form of links to Wiki pages.

More formally, we closely follow the setup of [8] and introduce our notation as follows: Our work aims to solve the task of recommending items (i.e., relevant content) in a conversation (modeled as a session) given a sequence of chat messages and previously mentioned items. We define the item set as $\mathscr{V} = \{v_1, v_2, \dots, v_{|\mathscr{V}|}\}$ and the set of non-items as $\mathscr{M} = \{m_1, m_2, \dots, m_{|\mathscr{M}|}\}$. The set of conversations is defined as $\mathscr{C} = \{c_1, c_2, \dots, c_{|\mathscr{C}|}\}$. For each conversation c, we can denote the sequence of interactions as $s_c = [i_1, i_2, \dots, i_n] \subseteq \{\mathscr{V} \cup \mathscr{M}\}^n$. Using these notations, our goal is to solve the task of predicting the next item $i_{n+1} \in \mathscr{V}$ for every interaction i_n in each sequence $s_c \in \mathscr{S}$.

4. Methodology

We model our task as a sequential recommendation problem with sequences consisting of items and possibly chat content. This enables us, in principle, to use any sequential recommendation architecture. In this section, we describe our model architectures, our different ways of representing conversations as sequences and our subsequence sampling.

4.1. Representing Conversations as Sequences of (Non)-Items

Here, we discuss our representation of conversations with mentioned items as sequences for a recommender model. Assume this short conversation as an example, where items are marked in **bold**:

User 1 I like **The Godfather**

User 2 Do you like **The Matrix**?

We compare different variants of modeling this as a sequence of items and possibly non-items:

Items Only As a baseline, we apply standard sequential recommendation models directly to the sequence of items mentioned in the conversations, ignoring the chat messages. In this model, the conversation above becomes $s_{\text{items}} = (\textbf{The Godfather}, \textbf{The Matrix})$. This can work on some datasets that are close to the traditional sequential recommender setting (e.g., in the `ReDial-Mention` dataset, cf. Section 5.2). However, we expect that this setting does not carry enough information in other datasets and that the content of the conversation is required to adequately identify the items that are referred to in the conversation.

Unique Token ID (UTID) Therefore, we propose to switch to a non-item modeling setting to include chat content. As a first step towards this, we split the messages into tokens corresponding to words and introduce a new ”word-item” m_i to the set of non-items $\mathscr{M}$ for each unique token occurring in the dataset. We then map each occurrence of a token to the corresponding word-item and include these word-items in the sequence. Now, our sequence becomes $s_{\text{UTID}} = (\text{I}, \text{like}, \text{The}, \text{Godfather}, \textbf{The Godfather}, \text{Do}, \text{you}, \text{like}, \text{The}, \text{Matrix}, ?, \textbf{The Matrix})$. This means that the model only has to identify the target items that have been referred to in the conversation rather than predict which item could be brought up next. Note that the modelling here is still very close to a traditional sequential recommender, with the only difference being that our models are only trained to predict actual items from $\mathscr{V}$, that is, word-items from $\mathscr{M}$ are never the target for recommendation, as we always build our sequences to end with actual items. This setting is identical to the Unique Page-ID (UPID) setting in [8].

Token Embedding (TE) Since the previous setting introduces a very high number of items to the vocabulary (each word in the texts is mapped to a separate item), making it more difficult to train the models, we explore the Page Embedding setting from [8] next: Here, we map all words to a single placeholder token `T`, leading to a set of non-items with only one entry $\mathscr{M} = \{\texttt{T}\}$. Each occurrence of this token is then associated with a pre-computed embedding $\hat{r}_m$ for the word it corresponds to. These embeddings can, for example, be extracted from language models or simple word embeddings. With this approach, our example conversation is represented by the id sequence $s_{\text{TE}} = (\texttt{T}, \texttt{T}, \texttt{T}, \texttt{T}, \textbf{The Godfather}, \texttt{T}, \texttt{T}, \texttt{T}, \texttt{T}, \texttt{T}, \texttt{T}, \textbf{The Matrix})$. The actual content of the tokens will only be represented by the corresponding embeddings.

Message Embedding (ME) While using Token Embeddings solves the problem of producing a very large vocabulary, it still leads to a high sequence length, since each word is included in the conversation as a separate element. However, the non-item page modeling also allows us to see entire messages as non-items. Therefore, in our final setting, we map each *message* to a placeholder token `M`, again leading to a singleton set of non-items $\mathscr{M} = \{\texttt{M}\}$. As in the previous setting, this placeholder token is shared for all messages and the content of the message is represented by a precomputed embedding $\hat{r}_m$, for example from a language model. In this setting, our sequence becomes $s_{\text{ME}} = (\texttt{M}, \textbf{The Godfather}, \texttt{M}, \textbf{The Matrix})$.

4.2. Sequential Recommender Models

We utilize a set of popular recommender models which all leverage the sequence of item ids. While UTID representations can be used without model changes, we need to adjust the embedding layer to include additional embedding representations for non-items. The common setup for the embedding layer, with embedding size d, looks as follows:

(i) an id embedding $E_{\mathscr{V}} \in \mathbb{R}^{|\mathscr{V}| \times d}$

(ii) an optional embedding $E_G \in \mathbb{R}^{N \times d}$ to encode the position of the items in the sequence, with N as the maximum input sequence length.

To include any additional precomputed embedding representation $\hat{r}_i \in \mathbb{R}^{|R|}$ for an interaction i, we add

(iii) a linear layer $l_R(x) = Wx + b$ with the weight matrix $W \in \mathbb{R}^{R \times d}$ and bias $b \in \mathbb{R}^d$ which scales the representation to the embedding size d.

The final output of the embedding layer is created by summing up all layers. For each sequence step n we embed the id of interaction i_n with $e_n = E_{\mathscr{V}}(i_n)$, the position with $g_n = E_G(n)$ and the scaled precomputed representation $\hat{r}_i$ with $r_n = l_R(\hat{r}_{i_n})$. We compute $h_n^0 = e_n + g_n + r_n$ as the input for the following layers of the sequential recommender. This allows us to add additional latent representations for any interaction, for non-items as well as items. Specifically, it also enables the use of the Token Embedding and Message Embedding strategies (cf. Section 4.1), which use only one shared non-item id and represent the content of the non-items using pre-computed embeddings.

4.3. Subsequence Sampling

In our setting, not only predicting the last item in a sequence is of relevance, but rather predicting all items given the chat context up to this point. Therefore, we generate a set S_i of subsequences from each sequence s_i in our datasets in the following way: We select all items $v_j \in \mathscr{V}$ in s_i and, for each of these items, generate a subsequence $S_{i,j} = s_i[1 : j]$ containing the sequence up to item v_j. Should a subsequence exceed the maximum sequence length set for the model, we truncate it from the left side. We evaluate our models on each of these subsequences to get an accurate estimate of their performance.

Since some of our datasets are small, we optionally also employ the same subsequence generation scheme for the training data, training on each subsequence and therefore increasing the amount of training samples.

5. Datasets

We use four datasets in our experiments:

1. `ReDial-Mention`: conversations between two users, where the task is to recommend links to the movies mentioned in the conversation,

2. `HandballSynth`: synthetic conversations in streams of Handball games from the European Championship, where the goal is to link the profiles of mentioned players and teams,

3. `Bestatter`: conversations between users and a chatbot, with the task of recommending pages with additional information about the discussed topic and

4. `Twitch`: chat sessions in e-sports streams, aiming to recommend wiki pages with information about the players, teams, in-game characters and other entities mentioned by the users.

Descriptive statistics for all annotated datasets are provided in Table 1.[1]

[1] All datasets are available from our repository at https://github.com/LSX-UniWue/non-items-recbole/tree/kars-workshop-24.

5.1. Dataset Construction

Generally, we construct samples for our task from conversations in the following way: Each conversation is converted into a sequence s consisting of items $v \in \mathscr{V}$ and potentially text messages $m \in \mathscr{M}$. For *Items Only*, we build a sequence from every item v mentioned in the conversation, discarding text messages. In all other variants, we build a sequence by representing each message m either as the sequence of its words or as a single token `M` as described in Section 4. If one or multiple items v are relevant to a message m, we append the items directly after m as targets for our recommender in the order in which they were discussed in m.

5.2. `ReDial-Mention`

We adapt the ReDial dataset for conversational recommendation [36] to our task. ReDial consists of conversations between two users, one “seeker”, who is looking for movie recommendations, and one “recommender”, who tries to find fitting movies to suggest to the seeker. This dataset is suitable for our task because it contains conversations annotated with movies that are mentioned in the messages. We construct samples for our task from ReDial as described above. ReDial contains movie IDs in the messages, which we replace by the full movie title to which they refer. Additionally, we crawled information such as the plot summary and director from the OMDb API[2], from which we build our item embeddings.

`ReDial-Mention-Noise` Since the first version of the ReDial dataset always mentions the exact movie title in the messages, potentially making the task too easy, we construct an additional noisy version of the dataset. To this end, we employ a pre-trained Large Language Model, specifically Llama 3.1 8b[3] to get modified versions of the movie titles. We query the model for 20 alternative titles for each movie with this prompt template:

> Return a name for this movie as it would likely be used in a conversation. For example, if the movie is called “The Matrix”, you might return “Matrix”. For “The Lord of the Rings: The Fellowship of the Ring”, you might return “The first Lord of the Rings movie”. Return only one possible name, nothing else:
>
> { Movie Title }

We keep the list of all 20 alternative titles, including duplicates. Since this sometimes yields titles that are too noisy or even wrong, we apply an additional filtering step, where we query the same model with this prompt template:

> You will be passed a pair of original movie title and a noisy version of it. Your task is to determine if the noisy version is a valid reference that could be used in a conversation. This can be either the original title, a slightly modified version or a description.
>
> If you think the noisy version is a valid alternative, return “yes”. If you think the noisy version is not a valid alternative, return “no”. Do not return anything else.
>
> The original title is: { Movie Title }. The noisy title is: { Noisy Title }.

We reconstruct the dataset by sampling an alternative title from this list for each mention of a movie when building the message representation.

5.3. `HandballSynth`

We generate a synthetic dataset for a controlled scenario with accurate information on all intended references. It simulates a chat with live messages from ongoing handball matches, where entities (e.g., players and coaches) are discussed and we want to recommend pages with additional information about

[2] https://www.omdbapi.com/

[3] https://ollama.com/library/llama3.1:8b

Table 1
Statistics of the preprocessed datasets. We report the number of unique sessions, items, messages and the overall number of interactions with items and messages as well as session length.

Dataset							Session length		
Name	Variant	|Sessions|	|Items|	|Messages|	$\|i_{Items}\|$	$\|i_{Messages}\|$	Min	Avg.	≈90%ile
HandballSynth	Items Only	7	74	0	554	0	63	79.1	90
	UTID|TE	7	75	1 096	554	8 997	1 152	1 364.4	1 735
	ME	7	75	649	554	1 188	213	248.8	280
ReDial-Mention	Items Only	6 866	5 553	0	45 131	0	4	6.5	10
	UTID|TE	6 904	5 560	27 394	45 362	770 925	20	118.2	165
	ME	6 904	5 560	81 890	45 362	102 552	7	21.4	30
	ME-Noisy	6 904	5 560	82 474	45 362	102 552	7	21.4	30
Bestatter	Items Only	1 476	243	0	9 868	0	2	6.6	20
	UTID|TE	2 122	266	4 330	10 514	4 612 854	21	2 178.7	8 930
	ME	2 122	266	926	10 514	24 5420	2	120.6	485
Twitch	Items Only	49	355	0	2 034	0	6	41.5	75
	UTID|TE	49	356	6 335	2 034	21 870	123	487.8	640
	ME	49	356	3 817	2 034	4 614	59	135.6	175

the entities that are mentioned by the users. We used the pre-trained Large Language Model Claude 3.5 Sonnet[4] to generate conversations based on play-by-play information from live-ticker data[5].

The LLM was prompted to create artificial chat messages as reactions to the events in the ticker. In order to keep the generated messages realistic, we instructed the model to generate emotional, short and informal messages, as well as occasional off-topic comments. We included the relevant team and player data[6] in the prompt and instructed the model to discuss specific entities in roughly every third message. This information is also used to build the embeddings for the items. Additionally, the model includes the link to the website containing the entity's information – obtained from the sources included in the prompt – with every generated reference. We remove these links from the messages and use them as target items as before.

5.4. Bestatter

We use a dataset of user conversations from a service chat bot on a German website for information about funerals[7]. The bot provides users with information for funeral related inquiries. In addition to general information on the topic, the bot also recommends links to information on the website, external web sources, as well as relevant contact information.

From these conversations, we construct a dataset for our task as before. As target items, we extract links mentioned by the bot, as well as contact information such as email addresses and phone numbers and replace their occurrences in the dialogues with a corresponding placeholder string.

We generate item representations by embedding the text contents of the linked websites for links. For contact information we construct the representation for the item by embedding the local context of the reply message containing the email address or phone number, including up to 250 character before and after the mentioned contact information.

5.5. Twitch

We collected a dataset of messages from Twitch.tv[8], a popular streaming website, primarily used for e-sports streams. Along with each stream, Twitch provides a chat where users can comment on the content of the stream. The purpose of this data set is to recommend links that provide background

[4] https://www.anthropic.com/news/claude-3-5-sonnet
[5] https://ticker.ehf.eu/
[6] https://www.eurohandball.com/
[7] https://www.bestatter.de/
[8] https://www.twitch.tv/

information on entities disscussed in the chat. Messages on Twitch.tv generally have an extremely noisy language with a high amount of spelling errors and slang [37], making this dataset especially challenging.

Dataset Generation We downloaded chat messages from several streams of the popular e-sports game Dota 2 using TwitchDownloader[9] and manually annotated them with relevant links from Liquipedia[10] and Wikipedia. Liquipedia provides information about both in-game characters and mechanics as well as real-world entities (players, teams) and tournaments in the context of Dota 2.

In total, we selected seven broadcasts from between September 2023 and June 2024, featuring both official streams from tournaments of different sizes as well as personal streams by individual, well known streamers. From these chat logs we randomly sampled segments of 100 consecutive messages for annotation. The messages where annotated independently by two annotators, with 1900 messages judged by both and an additional 3000 messages annotated by either one alone. Using the messages annotated by both participants, we observe a moderate inter annotator agreement of Cohen's kappa $\kappa = 0.47$. To further ensure quality of the annotations, an additional curator merged the two sets of annotations, resolving any discrepancies in the process.

Annotators were instructed to go through the chat log segments one message at a time, providing links to any entity directly mentioned or implied within that message. These links will serve as targets for our recommendation task.

In case of a direct mention of an entity, we can map them directly to their wiki page. However, many elements of the game do not have their own article. In this case, we link to the parent wiki page: For example, if a named ability is mentioned, we link the wiki page of the character with this ability. Apart from direct mentions, memes and in-jokes are common within the community. These often serve as a reference to a particular player or incident, from which we again derive appropriate link targets.

Again, we use the content of the linked pages to compute embeddings for the target items. As with the other datasets introduced above, sequences are constructed from the annotated messages.

Sequences are constructed from the annotated messages as above.

6. Experiments

Evaluation Metrics For all sequences, we ensure that the last interaction in a sequence is an actual item $v \in \mathcal{V}$ for training and testing. We calculate the *Hit Rate@k (HR@k)* for $k \in \{1, 5\}$ and the *Normalized Discounted Cumulative Gain@5 (NDCG@5)*, as we are only interested in recommending a small number of items at once. We calculate our metrics only on the top k recommended items, ignoring non-items. Metrics are computed for all subsequences, as described in Section 4.3.

Models For each dataset, we evaluate our setup on the following popular sequential recommender architectures:

- GRU4Rec[38]: a simple recurrent neural network with Gated Recurrent Units.
- CASER [2]: a convolutional network with horizontal and vertical convolution filters.
- NARM [3]: a neural network combining recurrent neural networks with attention.
- NextITNet [10]: a model built upon CASER, using dilated convolutions.
- SASRec [12]: a popular baseline utilizing self-attention for sequential recommendation. Here, it is used with a cross entropy loss.

[9] https://github.com/lay295/TwitchDownloader
[10] https://liquipedia.net/

- BERT4Rec [13]: an adaption of BERT [39] with bidirectional transformer layers and masked training [40].
- CORE [15]: an attention-based model, which couples the representation space for both encoding and decoding by representing the session as a linear combination of the items.
- LightSANs [14]: an attention-based model with a low-rank decomposed self-attention and a decoupled positional encoding.

Model Hyper-Parameters For the configuration of our experiments, we follow the setup in [8]. All of our models are trained with the Adam optimizer, and all models have a hidden and/or embedding size of 64 and an inner size of 256. The number of heads and layers is set to 2 for transformer layers and the dropout to 0.2. GRU4Rec also has 2 layers. The mask ratio for BERT4Rec is set to 0.2, while the fine-tuning ratio is set to 0.1. For CASER we set dropout = 0.4, Markov chain length = 5, vertical filters = 4 and horizontal filters = 8. For NextItNet, we set the kernel size = 3, the convolutional filter width = 5 and the dilations = $[1, 4]$. We use [212,10,42,404,6] as fixed seeds and report the average and standard deviation over 5 runs. We set the maximum sequence length to the ≈90% -quantile; see Table 1. We train all models for 50 epochs with a batch size of 64 on a cluster of L40-GPUs. We utilize the code from [8] and provide our own code and configurations in our repository[11].

Embedding Models We compute non-item embeddings from the text of the messages and item embeddings from the content of the linked websites to provide the models with additional information about the targets. All content is embedded using Sentence Transformers [41, 42] models. For the English datasets (`ReDial-Mention`, `ReDial-Mention-Noise`, `Twitch`), we use a monolingual English model.[12] For the German datasets (`HandballSynth`, `Bestatter`), we use a multilingual model.[13] Both models provide embeddings of size 384.

7. Results

The main results for all datasets are provided in Tables 2 to 5. We selected the two best-performing models, SASRec and LightSANs, and refer to the appendix for results of the other models. All results in this section are for models trained and evaluated on subsequences (cf. Section 4.3). We include results for models that are not trained, but only tested on subsequences, in the appendix. These models consistently perform worse, which is expected due to the lower amount of training data available in this setting. We compare our different sequence representation strategies (Items Only, UTID, TE, ME; cf. Section 4.1), as well as training and evaluating the models with or without item embeddings (non-items are always represented with embeddings). We can summarize our findings as follows:

Overall Performance Overall, our models yield reasonable to very good performance on all datasets. Message Embeddings (ME) perform best consistently. This is expected since the message embeddings provided by the Sentence Transformers models should capture the content of the messages well, while also keeping the sequence length manageable. This makes it relatively easy for the model to determine which entities are referred to in the messages. The models trained on word-level sequences perform worse, but still manage to provide good recommendations in most cases. Interestingly, using item embeddings only has a small influence on the results. This may be caused by the rather long content of the linked items: Sentence Transformer models are primarily trained to represent sentences and short paragraphs well, while we use them to represent entire websites. We leave this investigation, as well as the development of better representations for the item content, for future work.

[11] https://github.com/LSX-UniWue/non-items-recbole/tree/kars-workshop-24

[12] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

[13] https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2

Table 2
Average HR@k with $k \in \{1, 5\}$ and NDCG@5 with standard deviation on the `ReDial-Mention` (top) and `ReDial-Mention-Noise` (bottom) dataset over five random seeds for LightSANs and SASRec. ItemEmb indicates the use of Item Embeddings.

Model	ItemEmb	Metrics	Items Only	UTID	TE	ME
			Clean			
LightSANs	no	HR@1	0,046± 0,002	0,299± 0,003	0,310± 0,003	**0,619**± 0,007
		HR@5	0,144± 0,004	0,468± 0,010	0,487± 0,003	**0,755**± 0,013
		NDCG@5	0,097± 0,002	0,390± 0,007	0,406± 0,002	**0,698**± 0,010
	yes	HR@1	0,041± 0,003	0,308± 0,004	0,315± 0,009	**0,624**± 0,005
		HR@5	0,132± 0,004	0,485± 0,006	0,490± 0,016	**0,771**± 0,004
		NDCG@5	0,088± 0,003	0,404± 0,004	0,411± 0,013	**0,708**± 0,002
SASRec	no	HR@1	0,046± 0,001	0,541± 0,055	0,533± 0,033	**0,631**± 0,006
		HR@5	0,144± 0,002	0,733± 0,036	0,729± 0,022	**0,792**± 0,004
		NDCG@5	0,097± 0,001	0,650± 0,045	0,644± 0,027	**0,724**± 0,005
	yes	HR@1	0,042± 0,003	0,521± 0,006	0,505± 0,017	**0,619**± 0,007
		HR@5	0,132± 0,003	0,716± 0,004	0,706± 0,017	**0,764**± 0,005
		NDCG@5	0,088± 0,002	0,630± 0,004	0,618± 0,017	**0,702**± 0,005
			Noisy			
LightSANs	no	HR@1	0,046± 0,002	0,271± 0,007	0,276± 0,007	**0,594**± 0,008
		HR@5	0,144± 0,004	0,442± 0,010	0,454± 0,006	**0,743**± 0,004
		NDCG@5	0,097± 0,002	0,364± 0,008	0,372± 0,006	**0,680**± 0,005
	yes	HR@1	0,041± 0,003	0,280± 0,009	0,278± 0,007	**0,595**± 0,009
		HR@5	0,132± 0,004	0,457± 0,012	0,446± 0,013	**0,750**± 0,004
		NDCG@5	0,088± 0,003	0,376± 0,011	0,369± 0,010	**0,684**± 0,005
SASRec	no	HR@1	0,046± 0,001	0,536± 0,050	0,515± 0,020	**0,598**± 0,007
		HR@5	0,144± 0,002	0,720± 0,038	0,710± 0,016	**0,765**± 0,005
		NDCG@5	0,097± 0,001	0,640± 0,045	0,625± 0,017	**0,694**± 0,003
	yes	HR@1	0,042± 0,003	0,489± 0,035	0,498± 0,039	**0,590**± 0,005
		HR@5	0,132± 0,003	0,686± 0,023	0,697± 0,026	**0,750**± 0,005
		NDCG@5	0,088± 0,002	0,600± 0,028	0,610± 0,031	**0,681**± 0,005

`ReDial-Mention` The Message Embedding-models are able to identify the movies mentioned in the conversations very well. Word-level models (UTID, TE) still perform somewhat well but worse than the ME models. Item Only models are not able to solve the task well, since they are missing the content of the conversations.

`ReDial-Mention-Noise` Results on the noisy version of the dataset follow exactly the same trends as on `ReDial-Mention`. Since the titles of the movies are no longer contained exactly in the messages, the task becomes harder and, consequently, the performance drops slightly. However, the models are still able to identify the mentioned movies well if given access to the content of the conversation.

`HandballSynth` The general trends also hold on this dataset. Here, the Items Only strategy yields acceptable results for HR@5, which is likely caused by the somewhat low number of entities in the dataset. However, for both HR@1 and HR@5, including the conversation's content into the models still leads to a clear improvement.

`Bestatter` We were unable to train word-level models in the `Bestatter` dataset, as word-level modeling leads to very long sequences on this dataset (cf. Table 1). Again, the Items Only strategy works relatively well here. In addition to the low number of entities in the dataset, as in `HandballSynth`, the conversations in this dataset are also somewhat schematic: Many of the conversations follow the

Table 3
Average HR@k with $k \in \{1, 5\}$ and NDCG@5 with standard deviation on the `HandballSynth` dataset over five random seeds for LightSANs and SASRec. ItemEmb indicates the use of Item Embeddings.

Model	ItemEmb	Metrics	Items Only	UTID	TE	ME
LightSANs	no	HR@1	0,164± 0,043	0,369± 0,011	0,346± 0,018	**0,618**± 0,006
		HR@5	0,309± 0,030	0,479± 0,019	0,492± 0,007	**0,638**± 0,006
		NDCG@5	0,242± 0,031	0,425± 0,013	0,422± 0,013	**0,629**± 0,004
	yes	HR@1	0,148± 0,025	0,369± 0,011	0,346± 0,009	**0,618**± 0,006
		HR@5	0,449± 0,017	0,487± 0,033	0,500± 0,000	**0,638**± 0,006
		NDCG@5	0,308± 0,011	0,429± 0,017	0,424± 0,003	**0,629**± 0,005
SASRec	no	HR@1	0,078± 0,018	0,351± 0,039	0,339± 0,025	**0,618**± 0,006
		HR@5	0,273± 0,009	0,477± 0,033	0,500± 0,029	**0,631**± 0,006
		NDCG@5	0,175± 0,011	0,416± 0,030	0,424± 0,026	**0,625**± 0,007
	yes	HR@1	0,203± 0,025	0,351± 0,039	0,323± 0,040	**0,615**± 0,000
		HR@5	0,455± 0,000	0,487± 0,042	0,513± 0,033	**0,636**± 0,007
		NDCG@5	0,336± 0,011	0,421± 0,034	0,422± 0,028	**0,626**± 0,004

Table 4
Average HR@k with $k \in \{1, 5\}$ and NDCG@5 with standard deviation on the `Bestatter` dataset over five random seeds for LightSANs and SASRec. ItemEmb indicates the use of Item Embeddings.

Model	ItemEmb	Metrics	Items Only	ME
LightSANs	no	HR@1	0,519± 0,042	**0,854**± 0,006
		HR@5	0,828± 0,008	**0,942**± 0,001
		NDCG@5	0,686± 0,019	**0,906**± 0,003
	yes	HR@1	0,544± 0,017	**0,882**± 0,007
		HR@5	0,823± 0,003	**0,944**± 0,000
		NDCG@5	0,695± 0,005	**0,919**± 0,003
SASRec	no	HR@1	0,509± 0,056	**0,845**± 0,008
		HR@5	0,794± 0,020	**0,941**± 0,000
		NDCG@5	0,659± 0,034	**0,901**± 0,003
	yes	HR@1	0,533± 0,027	**0,872**± 0,011
		HR@5	0,826± 0,010	**0,943**± 0,002
		NDCG@5	0,691± 0,010	**0,914**± 0,005

same structure, meaning that items mentioned in the future can often be inferred from past items without including information about the conversation.

`Twitch` This is the most challenging dataset due to the very noisy nature of messages on twitch.tv. Consequently, the scores are lower than for other datasets, but the models still yield reasonable performance when using Message Embeddings. Otherwise, the results follow the same trends as on the other datasets.

8. Conclusion

We have shown that we can adapt arbitrary sequential recommender models to the task of recommending relevant content in chat conversations between users. To this end, we have adapted the recently proposed non-item page modeling to represent the content of chat messages in multiple ways. We have evaluated these models on several datasets, showing that our modeling is successful. We find that the models perform best when given access to pre-trained message embeddings to represent the content of the chat messages.

Our modeling can be applied in a variety of settings, ranging from the direct application of showing relevant content in chat rooms to being used as an intermediate component in a conversational

Table 5
Average HR@k with $k \in \{1, 5\}$ and NDCG@5 with standard deviation on the `Twitch` dataset over five random seeds for LightSANs and SASRec. ItemEmb indicates the use of Item Embeddings.

Model	ItemEmb	Metrics	Items Only	UTID	TE	ME
LightSANs	no	HR@1	0,163± 0,009	0,201± 0,006	0,193± 0,020	**0,438**± 0,020
		HR@5	0,402± 0,013	0,293± 0,011	0,286± 0,013	**0,575**± 0,018
		NDCG@5	0,288± 0,008	0,249± 0,010	0,242± 0,012	**0,515**± 0,008
	yes	HR@1	0,141± 0,015	0,210± 0,015	0,199± 0,011	**0,419**± 0,013
		HR@5	0,351± 0,012	0,301± 0,010	0,284± 0,026	**0,554**± 0,032
		NDCG@5	0,253± 0,007	0,255± 0,007	0,242± 0,017	**0,492**± 0,017
SASRec	no	HR@1	0,156± 0,007	0,207± 0,016	0,202± 0,014	**0,422**± 0,009
		HR@5	0,379± 0,026	0,327± 0,010	0,338± 0,008	**0,573**± 0,016
		NDCG@5	0,275± 0,015	0,270± 0,009	0,272± 0,008	**0,507**± 0,010
	yes	HR@1	0,157± 0,020	0,202± 0,011	0,215± 0,011	**0,412**± 0,028
		HR@5	0,397± 0,015	0,322± 0,022	0,346± 0,025	**0,563**± 0,018
		NDCG@5	0,286± 0,013	0,264± 0,015	0,280± 0,015	**0,495**± 0,012

recommender system.

Our findings imply that any improvements in sequential recommender models can be directly transferred to our task of recommending relevant content to chat conversations. In particular, we see the opportunity to adapt recommender models with access to knowledge graphs to our task. There is potential to improve our results further by providing better content representations for both recommended items and messages (especially in the case of Twitch.tv, where standard embedding models may struggle with the noisy language), which we see as a promising direction for future work.

Acknowledgments

This work is partially funded by the German Federal Ministry of Education and Research (BMBF) under grant number 01IS22051B in the KILiMod project.

References

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

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

[4] B. Hidasi, M. Quadrana, A. Karatzoglou, D. Tikk, Parallel recurrent neural network architectures for feature-rich session-based recommendations, in: Proceedings of the 10th ACM Conference on Recommender Systems, RecSys '16, ACM, New York, NY, USA, 2016, pp. 241–248. doi:10.1145/2959100.2959167.

[5] T. X. Tuan, T. M. Phuong, 3d convolutional networks for session-based recommendation with content features, in: Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys '17, ACM, New York, NY, USA, 2017, pp. 138–146. doi:10.1145/3109859.3109900.

[6] E. Fischer, D. Zoller, A. Dallmann, A. Hotho, Integrating keywords into bert4rec for sequential recommendation, in: KI 2020: Advances in Artificial Intelligence: 43rd German Conference on AI, Bamberg, Germany, September 21–25, 2020, Proceedings, Springer-Verlag, Berlin,

Heidelberg, 2020, p. 275–282. URL: https://doi.org/10.1007/978-3-030-58285-2_23. doi:10.1007/978-3-030-58285-2_23.

[7] E. Fischer, D. Schlör, A. Zehe, A. Hotho, Enhancing sequential next-item prediction through modelling non-item pages, in: 2023 IEEE International Conference on Data Mining Workshops (ICDMW), 2023, pp. 128–136. doi:10.1109/ICDMW60847.2023.00024.

[8] E. Fischer, D. Schlör, A. Zehe, A. Hotho, Modeling and analyzing the influence of non-item pages on sequential next-item prediction, 2024. URL: https://arxiv.org/abs/2408.15953. arXiv:2408.15953.

[9] C. Xu, P. Zhao, Y. Liu, J. Xu, V. S. S.Sheng, Z. Cui, X. Zhou, H. Xiong, Recurrent convolutional neural network for sequential recommendation, in: The World Wide Web Conference on - WWW'19, ACM Press, 2019. doi:10.1145/3308558.3313408.

[10] F. Yuan, A. Karatzoglou, I. Arapakis, J. M. Jose, X. He, A simple convolutional generative network for next item recommendation, 2018. arXiv:1808.05163.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, in: Advances in neural information processing systems, 2017, pp. 5998–6008.

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

[13] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, 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. doi:10.1145/3357384.3357895.

[14] X. Fan, Z. Liu, J. Lian, W. Zhao, X. Xie, J.-R. Wen, Lighter and better: Low-rank decomposed self-attention networks for next-item recommendation, 2021, pp. 1733–1737. doi:10.1145/3404835.3462978.

[15] Y. Hou, B. Hu, Z. Zhang, W. X. Zhao, Core: Simple and effective session-based recommendation within consistent representation space, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 1796–1801. URL: https://doi.org/10.1145/3477495.3531955. doi:10.1145/3477495.3531955.

[16] Q. Liu, S. Wu, D. Wang, Z. Li, L. Wang, Context-aware sequential recommendation, 2016. arXiv:1609.05787.

[17] G. de Souza Pereira Moreira, S. Rabhi, J. M. Lee, R. Ak, E. Oldridge, Transformers4rec: Bridging the gap between nlp and sequential / session-based recommendation, in: Proceedings of the 15th ACM Conference on Recommender Systems, RecSys '21, Association for Computing Machinery, New York, NY, USA, 2021, p. 143–153. URL: https://doi.org/10.1145/3460231.3474255. doi:10.1145/3460231.3474255.

[18] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, Q. V. Le, Xlnet: generalized autoregressive pretraining for language understanding (2019).

[19] K. Clark, M. Luong, Q. V. Le, C. D. Manning, ELECTRA: pre-training text encoders as discriminators rather than generators, CoRR abs/2003.10555 (2020). URL: https://arxiv.org/abs/2003.10555. arXiv:2003.10555.

[20] A. Jagatap, N. Gupta, S. Farfade, P. M. Comar, Attribert - session-based product attribute recommendation with bert, in: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '23, Association for Computing Machinery, New York, NY, USA, 2023, p. 3421–3425. URL: https://doi.org/10.1145/3539618.3594714. doi:10.1145/3539618.3594714.

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

[22] Y. Xie, P. Zhou, S. Kim, Decoupled side information fusion for sequential recommendation, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '22, Association for Computing Machinery, New York, NY, USA, 2022, p. 1611–1621. URL: https://doi.org/10.1145/3477495.3531963. doi:10.1145/3477495.3531963.

[23] L. Wu, S. Li, C.-J. Hsieh, J. Sharpnack, Sse-pt: Sequential recommendation via personalized transformer, in: Proceedings of the 14th ACM Conference on Recommender Systems, RecSys '20, Association for Computing Machinery, New York, NY, USA, 2020, p. 328–337. URL: https://doi.org/10.1145/3383313.3412258. doi:10.1145/3383313.3412258.

[24] Q. Chen, H. Zhao, W. Li, P. Huang, 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, 2019. doi:10.1145/3326937.3341261.

[25] Q. Zhang, L. Cao, C. Shi, Z. Niu, Neural time-aware sequential recommendation by jointly modeling preference dynamics and explicit feature couplings, IEEE Transactions on Neural Networks and Learning Systems 33 (2022) 5125–5137. doi:10.1109/TNNLS.2021.3069058.

[26] E. Fischer, A. Dallmann, A. Hotho, Personalization through user attributes for transformer-based sequential recommendation, in: H. J. Corona Pampín, R. Shirvany (Eds.), Recommender Systems in Fashion and Retail, Springer Nature Switzerland, Cham, 2023, pp. 25–43. doi:https://doi.org/10.1007/978-3-031-22192-7_2.

[27] J. Tagliabue, C. Greco, J.-F. Roy, B. Yu, P. J. Chia, F. Bianchi, G. Cassani, Sigir 2021 e-commerce workshop data challenge, 2021. arXiv:2104.09423.

[28] G. de Souza P. Moreira, S. Rabhi, R. Ak, M. Y. Kabir, E. Oldridge, Transformers with multi-modal features and post-fusion context for e-commerce session-based recommendation, 2021. arXiv:2107.05124.

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

[30] E. Fischer, D. Zoller, A. Hotho, Comparison of transformer-based sequential product recommendation models for the coveo data challenge, SIGIR Workshop On eCommerce (2021).

[31] I. Guellil, A. Garcia-Dominguez, P. R. Lewis, S. Hussain, G. Smith, Entity linking for english and other languages: a survey, Knowledge and Information Systems (2024) 1–52.

[32] S. Perera, P. N. Mendes, A. Alex, A. P. Sheth, K. Thirunarayan, Implicit entity linking in tweets, in: H. Sack, E. Blomqvist, M. d'Aquin, C. Ghidini, S. P. Ponzetto, C. Lange (Eds.), The Semantic Web. Latest Advances and New Domains, Springer International Publishing, Cham, 2016, pp. 118–132.

[33] H. Hosseini, T. T. Nguyen, J. Wu, E. Bagheri, Implicit entity linking in tweets: An ad-hoc retrieval approach, Applied Ontology 14 (2019) 451–477.

[34] H. Hosseini, E. Bagheri, Learning to rank implicit entities on twitter, Information Processing & Management 58 (2021) 102503.

[35] D. Jannach, A. Manzoor, W. Cai, L. Chen, A survey on conversational recommender systems, ACM Computing Surveys 54 (2021) 1–36. URL: http://dx.doi.org/10.1145/3453154. doi:10.1145/3453154.

[36] R. Li, S. E. Kahou, H. Schulz, V. Michalski, L. Charlin, C. Pal, Towards deep conversational recommendations, in: Advances in Neural Information Processing Systems 31 (NIPS 2018), 2018.

[37] K. Kobs, A. Zehe, A. Bernstetter, J. Chibane, J. Pfister, J. Tritscher, A. Hotho, Emote-controlled: Obtaining implicit viewer feedback through emote based sentiment analysis on comments of popular twitch.tv channels, ACM Transactions on Social Computing 3 (2020) 1–34. URL: https://doi.org/10.1145%2F3365523. doi:10.1145/3365523.

[38] Y. K. Tan, X. Xu, 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.

[39] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423.

[40] W. L. Taylor, "cloze procedure": a new tool for measuring readability., Journalism & Mass Communication Quarterly 30 (1953) 415–433.

[41] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2019. URL: https://arxiv.org/abs/1908.10084.

[42] N. Reimers, I. Gurevych, Making monolingual sentence embeddings multilingual using knowledge distillation, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2020. URL: https://arxiv.org/abs/2004.09813.

Appendix

Table 6
Average HR@5 with standard deviation on the `ReDial-Mention` dataset for all models over five random seeds. SubSeq indicates training on subsequences, ItemEmb the use of Item Embeddings.

Model	SubSeq	ItemEmb	Items Only	UTID	TE	ME
BERT4Rec	no	no	**0,074**± 0,005	0,035± 0,003	0,037± 0,003	0,046± 0,012
		yes	**0,072**± 0,005	0,035± 0,003	0,036± 0,003	0,037± 0,002
	yes	no	0,082± 0,003	0,274± 0,027	0,039± 0,006	**0,419**± 0,007
		yes	0,090± 0,005	0,275± 0,030	0,038± 0,006	**0,369**± 0,024
Caser	no	no	0,028± 0,003	0,062± 0,009	0,046± 0,006	**0,066**± 0,008
		yes	0,045± 0,005	0,061± 0,006	0,042± 0,009	**0,093**± 0,011
	yes	no	0,080± 0,003	0,362± 0,017	0,379± 0,010	**0,495**± 0,010
		yes	0,074± 0,004	0,359± 0,012	0,370± 0,011	**0,504**± 0,011
Core	no	no	0,135± 0,003	**0,681**± 0,008	0,677± 0,010	0,443± 0,027
		yes	0,066± 0,004	**0,656**± 0,012	0,645± 0,012	0,575± 0,016
	yes	no	0,169± 0,002	**0,860**± 0,002	0,844± 0,003	0,851± 0,002
		yes	0,115± 0,004	**0,860**± 0,001	0,848± 0,004	0,857± 0,002
GRU4Rec	no	no	0,028± 0,003	0,382± 0,050	**0,409**± 0,007	0,331± 0,009
		yes	0,046± 0,002	**0,441**± 0,032	0,441± 0,016	0,334± 0,021
	yes	no	0,092± 0,003	**0,812**± 0,005	0,803± 0,003	0,703± 0,003
		yes	0,095± 0,004	**0,819**± 0,004	0,812± 0,003	0,709± 0,006
LightSANs	no	no	0,117± 0,003	0,213± 0,006	0,218± 0,006	**0,618**± 0,002
		yes	0,096± 0,003	0,214± 0,008	0,215± 0,004	**0,616**± 0,002
	yes	no	0,144± 0,004	0,468± 0,010	0,487± 0,003	**0,755**± 0,013
		yes	0,132± 0,004	0,485± 0,006	0,490± 0,016	**0,771**± 0,004
NARM	no	no	0,012± 0,006	0,234± 0,017	**0,243**± 0,019	0,181± 0,019
		yes	0,056± 0,003	0,216± 0,012	**0,226**± 0,019	0,124± 0,018
	yes	no	0,006± 0,000	0,752± 0,007	**0,757**± 0,007	0,660± 0,007
		yes	0,087± 0,005	0,757± 0,004	**0,760**± 0,004	0,672± 0,006
NextItNet	no	no	0,021± 0,005	0,036± 0,005	0,038± 0,007	**0,040**± 0,003
		yes	0,038± 0,009	0,038± 0,006	0,035± 0,007	**0,049**± 0,012
	yes	no	0,064± 0,004	0,036± 0,003	0,034± 0,001	**0,548**± 0,006
		yes	0,077± 0,004	0,035± 0,001	0,034± 0,001	**0,546**± 0,006
SASRec	no	no	0,117± 0,003	0,258± 0,003	0,244± 0,005	**0,617**± 0,001
		yes	0,097± 0,002	0,251± 0,010	0,257± 0,010	**0,613**± 0,003
	yes	no	0,144± 0,002	0,733± 0,036	0,729± 0,022	**0,792**± 0,004
		yes	0,132± 0,003	0,716± 0,004	0,706± 0,017	**0,764**± 0,005

Table 7

Average HR@5 with standard deviation on the `ReDial-Mention`-Noise dataset for all models over five random seeds. SubSeq indicates training on subsequences, ItemEmb the use of Item Embeddings.

Model	SubSeq	ItemEmb	Items Only	UTID	TE	ME
BERT4Rec	no	no	**0,074**± 0,005	0,035± 0,003	0,037± 0,003	0,046± 0,012
		yes	**0,072**± 0,005	0,035± 0,003	0,036± 0,003	0,037± 0,002
	yes	no	0,082± 0,003	0,274± 0,027	0,039± 0,006	**0,419**± 0,007
		yes	0,090± 0,005	0,275± 0,030	0,038± 0,006	**0,369**± 0,024
Caser	no	no	0,028± 0,003	0,062± 0,009	0,046± 0,006	**0,066**± 0,008
		yes	0,045± 0,005	0,061± 0,006	0,042± 0,009	**0,093**± 0,011
	yes	no	0,080± 0,003	0,362± 0,017	0,379± 0,010	**0,495**± 0,010
		yes	0,074± 0,004	0,359± 0,012	0,370± 0,011	**0,504**± 0,011
Core	no	no	0,135± 0,003	**0,681**± 0,008	0,677± 0,010	0,443± 0,027
		yes	0,066± 0,004	**0,656**± 0,012	0,645± 0,012	0,575± 0,016
	yes	no	0,169± 0,002	**0,860**± 0,002	0,844± 0,003	0,851± 0,002
		yes	0,115± 0,004	**0,860**± 0,001	0,848± 0,004	0,857± 0,002
GRU4Rec	no	no	0,028± 0,003	0,382± 0,050	**0,409**± 0,007	0,331± 0,009
		yes	0,046± 0,002	**0,441**± 0,032	0,441± 0,016	0,334± 0,021
	yes	no	0,092± 0,003	**0,812**± 0,005	0,803± 0,003	0,703± 0,003
		yes	0,095± 0,004	**0,819**± 0,004	0,812± 0,003	0,709± 0,006
LightSANs	no	no	0,117± 0,003	0,213± 0,006	0,218± 0,006	**0,618**± 0,002
		yes	0,096± 0,003	0,214± 0,008	0,215± 0,004	**0,616**± 0,002
	yes	no	0,144± 0,004	0,468± 0,010	0,487± 0,003	**0,755**± 0,013
		yes	0,132± 0,004	0,485± 0,006	0,490± 0,016	**0,771**± 0,004
NARM	no	no	0,012± 0,006	0,234± 0,017	**0,243**± 0,019	0,181± 0,019
		yes	0,056± 0,003	0,216± 0,012	**0,226**± 0,019	0,124± 0,018
	yes	no	0,006± 0,000	0,752± 0,007	**0,757**± 0,007	0,660± 0,007
		yes	0,087± 0,005	0,757± 0,004	**0,760**± 0,004	0,672± 0,006
NextItNet	no	no	0,021± 0,005	0,036± 0,005	0,038± 0,007	**0,040**± 0,003
		yes	0,038± 0,009	0,038± 0,006	0,035± 0,007	**0,049**± 0,012
	yes	no	0,064± 0,004	0,036± 0,003	0,034± 0,001	**0,548**± 0,006
		yes	0,077± 0,004	0,035± 0,001	0,034± 0,001	**0,546**± 0,006
SASRec	no	no	0,117± 0,003	0,258± 0,003	0,244± 0,005	**0,617**± 0,001
		yes	0,097± 0,002	0,251± 0,010	0,257± 0,010	**0,613**± 0,003
	yes	no	0,144± 0,002	0,733± 0,036	0,729± 0,022	**0,792**± 0,004
		yes	0,132± 0,003	0,716± 0,004	0,706± 0,017	**0,764**± 0,005

Table 8

Average HR@5 with standard deviation on the `HandballSynth` dataset for all models over five random seeds. SubSeq indicates training on subsequences, ItemEmb the use of Item Embeddings.

Model	SubSeq	ItemEmb	Items Only	UTID	TE	ME
BERT4Rec	no	no	0,135± 0,024	**0,138**± 0,019	0,128± 0,029	0,115± 0,071
		yes	**0,140**± 0,039	0,138± 0,019	0,128± 0,029	0,115± 0,071
	yes	no	0,135± 0,012	0,062± 0,033	0,162± 0,028	**0,172**± 0,017
		yes	0,140± 0,011	0,062± 0,033	0,159± 0,030	**0,169**± 0,014
Caser	no	no	0,036± 0,050	**0,090**± 0,000	0,044± 0,039	0,072± 0,038
		yes	0,010± 0,017	**0,051**± 0,043	0,021± 0,030	0,023± 0,032
	yes	no	0,148± 0,020	0,149± 0,028	0,136± 0,050	**0,228**± 0,042
		yes	0,203± 0,033	0,113± 0,042	0,136± 0,025	**0,282**± 0,091
Core	no	no	**0,127**± 0,037	0,054± 0,049	0,036± 0,049	0,062± 0,041
		yes	0,018± 0,041	**0,054**± 0,049	0,036± 0,049	0,000± 0,000
	yes	no	0,138± 0,030	0,131± 0,070	**0,162**± 0,028	0,126± 0,019
		yes	**0,158**± 0,038	0,156± 0,067	0,149± 0,058	0,123± 0,056
GRU4Rec	no	no	0,047± 0,034	0,018± 0,040	0,082± 0,047	**0,100**± 0,091
		yes	0,031± 0,028	0,003± 0,006	**0,077**± 0,043	0,072± 0,064
	yes	no	0,156± 0,033	0,197± 0,084	**0,554**± 0,091	0,538± 0,016
		yes	0,221± 0,036	0,213± 0,066	0,492± 0,111	**0,518**± 0,063
LightSANs	no	no	0,013± 0,000	0,074± 0,021	0,090± 0,000	**0,172**± 0,007
		yes	0,052± 0,000	0,082± 0,017	0,090± 0,000	**0,169**± 0,006
	yes	no	0,164± 0,043	0,369± 0,011	0,346± 0,018	**0,618**± 0,006
		yes	0,148± 0,025	0,369± 0,011	0,346± 0,009	**0,618**± 0,006
NARM	no	no	0,039± 0,037	0,056± 0,053	**0,059**± 0,055	0,056± 0,053
		yes	**0,065**± 0,046	0,046± 0,053	0,059± 0,042	0,051± 0,047
	yes	no	0,120± 0,017	0,328± 0,090	0,379± 0,064	**0,428**± 0,035
		yes	0,218± 0,035	0,367± 0,067	0,403± 0,055	**0,421**± 0,017
NextItNet	no	no	0,068± 0,038	0,036± 0,049	0,018± 0,040	**0,090**± 0,000
		yes	0,086± 0,007	0,072± 0,040	0,036± 0,049	**0,090**± 0,000
	yes	no	0,127± 0,050	**0,180**± 0,000	0,154± 0,035	0,174± 0,007
		yes	0,148± 0,059	**0,180**± 0,000	0,167± 0,029	0,154± 0,035
SASRec	no	no	0,016± 0,006	0,059± 0,042	0,090± 0,000	**0,177**± 0,017
		yes	0,073± 0,046	0,064± 0,048	0,090± 0,000	**0,177**± 0,017
	yes	no	0,078± 0,018	0,351± 0,039	0,339± 0,025	**0,618**± 0,006
		yes	0,203± 0,025	0,351± 0,039	0,323± 0,040	**0,615**± 0,000

Table 9
Average HR@5 with standard deviation on the `Bestatter` dataset for all models over five random seeds. SubSeq indicates training on subsequences, ItemEmb the use of Item Embeddings.

Model	SubSeq	ItemEmb	Items Only	ME
BERT4Rec	no	no	0,224± 0,050	**0,324**± 0,022
		yes	0,159± 0,047	**0,322**± 0,011
	yes	no	0,393± 0,036	**0,433**± 0,014
		yes	0,287± 0,036	**0,402**± 0,021
Caser	no	no	0,194± 0,040	**0,488**± 0,025
		yes	0,190± 0,037	**0,541**± 0,007
	yes	no	0,373± 0,032	**0,623**± 0,021
		yes	0,353± 0,032	**0,699**± 0,007
Core	no	no	0,170± 0,017	**0,620**± 0,014
		yes	0,291± 0,032	**0,653**± 0,009
	yes	no	0,160± 0,018	**0,766**± 0,003
		yes	0,492± 0,006	**0,791**± 0,029
GRU4Rec	no	no	0,236± 0,025	**0,594**± 0,008
		yes	0,228± 0,049	**0,621**± 0,013
	yes	no	0,411± 0,029	**0,799**± 0,028
		yes	0,490± 0,024	**0,847**± 0,026
LightSANs	no	no	0,309± 0,030	**0,620**± 0,031
		yes	0,289± 0,013	**0,626**± 0,012
	yes	no	0,519± 0,042	**0,854**± 0,006
		yes	0,544± 0,017	**0,882**± 0,007
NARM	no	no	0,206± 0,017	**0,568**± 0,003
		yes	0,217± 0,030	**0,590**± 0,020
	yes	no	0,403± 0,043	**0,738**± 0,012
		yes	0,434± 0,025	**0,783**± 0,016
NextItNet	no	no	**0,222**± 0,039	0,194± 0,077
		yes	0,180± 0,035	**0,194**± 0,077
	yes	no	**0,387**± 0,029	0,056± 0,000
		yes	**0,363**± 0,024	0,057± 0,002
SASRec	no	no	0,235± 0,056	**0,585**± 0,020
		yes	0,288± 0,045	**0,641**± 0,024
	yes	no	0,509± 0,056	**0,845**± 0,008
		yes	0,533± 0,027	**0,872**± 0,011

Table 10
Average HR@5 with standard deviation on the `Twitch` dataset for all models over five random seeds. SubSeq indicates training on subsequences, ItemEmb the use of Item Embeddings.

Model	SubSeq	ItemEmb	Items Only	UTID	TE	ME
BERT4Rec	no	no	**0,242**± 0,014	0,148± 0,038	0,205± 0,031	0,228± 0,025
		yes	**0,232**± 0,020	0,148± 0,035	0,205± 0,031	0,228± 0,025
	yes	no	**0,348**± 0,008	0,198± 0,017	0,232± 0,011	0,225± 0,046
		yes	**0,336**± 0,040	0,203± 0,013	0,237± 0,014	0,197± 0,042
Caser	no	no	0,126± 0,007	**0,132**± 0,010	0,123± 0,034	0,117± 0,014
		yes	0,091± 0,019	0,135± 0,020	**0,141**± 0,026	0,098± 0,052
	yes	no	**0,332**± 0,057	0,150± 0,047	0,153± 0,065	0,200± 0,052
		yes	**0,267**± 0,043	0,151± 0,053	0,103± 0,052	0,188± 0,085
Core	no	no	**0,368**± 0,039	0,138± 0,019	0,139± 0,019	0,147± 0,028
		yes	**0,182**± 0,031	0,148± 0,024	0,138± 0,017	0,166± 0,010
	yes	no	0,485± 0,007	0,373± 0,020	0,346± 0,058	**0,494**± 0,045
		yes	0,341± 0,103	0,414± 0,039	0,368± 0,014	**0,438**± 0,024
GRU4Rec	no	no	0,128± 0,024	0,111± 0,048	0,122± 0,026	**0,149**± 0,015
		yes	0,115± 0,036	0,120± 0,034	0,130± 0,015	**0,141**± 0,014
	yes	no	0,360± 0,014	0,332± 0,029	0,462± 0,011	**0,511**± 0,011
		yes	0,384± 0,024	0,332± 0,029	0,455± 0,008	**0,486**± 0,025
LightSANs	no	no	**0,188**± 0,016	0,118± 0,022	0,103± 0,019	0,178± 0,014
		yes	0,121± 0,019	0,111± 0,035	0,111± 0,017	**0,186**± 0,017
	yes	no	0,402± 0,013	0,293± 0,011	0,286± 0,013	**0,575**± 0,018
		yes	0,351± 0,012	0,301± 0,010	0,284± 0,026	**0,554**± 0,032
NARM	no	no	**0,167**± 0,013	0,097± 0,034	0,085± 0,051	0,108± 0,031
		yes	**0,178**± 0,023	0,114± 0,035	0,088± 0,064	0,149± 0,016
	yes	no	0,353± 0,040	0,345± 0,047	0,352± 0,029	**0,396**± 0,045
		yes	0,359± 0,036	0,356± 0,043	0,369± 0,024	**0,420**± 0,048
NextItNet	no	no	0,105± 0,035	0,108± 0,035	**0,164**± 0,029	0,096± 0,027
		yes	0,101± 0,024	0,117± 0,031	**0,160**± 0,022	0,119± 0,009
	yes	no	**0,287**± 0,045	0,231± 0,021	0,234± 0,018	0,219± 0,035
		yes	0,211± 0,071	**0,244**± 0,005	0,236± 0,021	0,227± 0,007
SASRec	no	no	**0,203**± 0,015	0,118± 0,023	0,113± 0,018	0,181± 0,015
		yes	0,141± 0,014	0,106± 0,031	0,102± 0,017	**0,181**± 0,009
	yes	no	0,379± 0,026	0,327± 0,010	0,338± 0,008	**0,573**± 0,016
		yes	0,397± 0,015	0,322± 0,022	0,346± 0,025	**0,563**± 0,018