Master's Thesis of Data Science

Music Streaming Session-based Recommendation with Transformer Architectures

트랜스포머 기반 음악 스트리밍 세션 추천 시스템

August 2022

Graduate School of Seoul National University
Department of Data Science (Data Science Major)

Kihong Seong

Music Streaming Session-based Recommendation with Transformer Architectures

Advisor Hyopil Shin

Submitting a master's thesis of
Data Science

June 2022

Graduate School of Seoul National University
Department of Data Science (Data Science Major)

Kihong Seong

Confirming the master's thesis written by
Kihong Seong
July 2022

Chair	차 상 균	(Seal)
Vice Chair	신 효 필	(Seal)
Examiner	오 민 환	(Seal)
Examiner	이 상 학	(Seal)

Abstract

Recommendation systems have grown in popularity over the last few years, with the rise of big data and development of computing resources. Compared to simple rule based methods or content based filtering methods used for recommendation during the early development stage of recommendation systems, recent methodologies try to implement much more complex models. Latent factor models and collaborative filtering methods were developed to find similarities between users and items without actually knowing their characteristics, and gained popularity. Various item domains, mainly movie and retail, have extensively used these recommendation algorithms.

With the development of deep learning architectures, various deep learning based recommendation systems emerged in recent years. While a lot of them were focused on generating the predicted item ratings when given a big data comprised of user ids, item ids, and ratings, there were some efforts to generate next-item recommendations as well. Next-item recommendations receive a session or sequence of actions by some user, and try to predict the next action of a user. NVIDIA recently used Transformers, a deep learning architecture in the field of Natural Language Processing (NLP), to build a session based recommendation system called Transformers4Rec [1]. The system showed state of the art performances for the usual movie and retail domains.

In the music domain, unfortunately, advanced models for session-based recommendations have been explored to a small extent. Therefore, this thesis will attempt to apply Transformer based architectures to session-based recom-

[1]https://github.com/NVIDIA-Merlin/Transformers4Rec/

mendation for music streaming, by utilizing a dataset from Spotify and framework from NVIDIA. In this thesis, unique characteristics of music data that validates this research's purpose are explored. The effectiveness of Transformer architectures on music data are shown with next-item prediction performances on actual user streaming session data, and methods for feature engineering and data preprocessing to ensure the best prediction results are investigated. An empirical analysis that compares various Transformer architectures is also provided, with models further analyzed with additional feature information.

Keywords: Transformers, Recommendation Systems, Music Streaming Sessions

Student ID: 2020-25184

Contents

List of Figures

List of Tables

1 Introduction

1.1 Research Topic

This thesis's topic is on developing a recommendation system (RecSys) for music streaming session data with Transformer architectures. Unlike traditional RecSys which attempt to recommend the most likeable item to a certain user, our aim will be a session-based RecSys (Ludewig et al. 2020), which uses session data; sequence of actions made by users (items bought, movies watched, music listened, etc.). Based on numerous sequences, a session-based RecSys tries to recommend the next item in a certain user's session (Wang et al. 2021). This is illustrated in Figure 1.1 with songs as items. We have a certain user's previous sequence of items interacted with timestamps ($t-4$ to $t-1$), and attempt to predict the next item at timestamp t. Therefore, it is important for the RecSys to know which parts of the sequence to pay attention to.

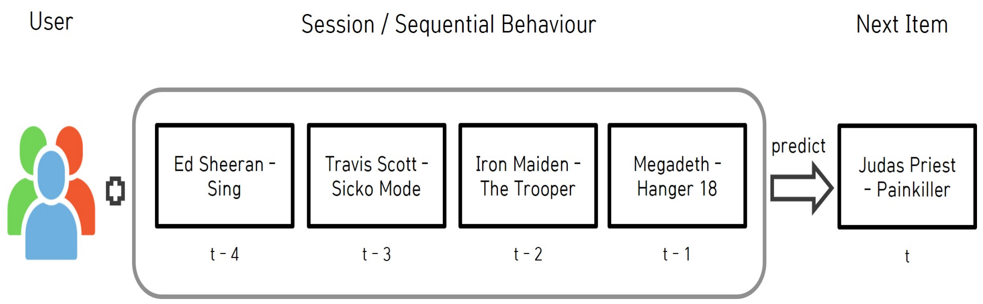
User
Session / Sequential Behaviour
Next Item
Ed Sheeran - Sing
Travis Scott - Sicko Mode
Iron Maiden - The Trooper
Megadeth - Hanger 18
predict
Judas Priest - Painkiller
t - 4
t - 3
t - 2
t - 1
t

Figure 1.1 Session-based Recommendation.

Transformer, a model developed in the Natural Language Processing (NLP) domain, aims to learn which parts of a sentence to pay attention to for various tasks (Vaswani et al. 2017). Among those various tasks, language modeling is a task that seems to align to the next-item recommendation task for session-based recommendation. Language modeling is the task of predicting the next word to appear in a sentence.

Acknowledging the similarities between next-item recommendation and language modeling, this thesis applies Transformer architectures to music streaming session-based recommendation. Traditional words and sentences will be replaced with songs and sequences of songs to feed into Transformer architectures, to predict the next song to appear in a streaming session.

1.2 Purpose of Research

The purpose of this research is to 1) Produce a novel RecSys in the field of session-based recommendation for music streaming sessions, 2) Thus providing a baseline model for deep learning based music streaming session RecSys, and 3) Explore the compatibility between Transformer architectures and music streaming sessions data.

The RecSys in this research is novel in the domain of next-item recommendation for music streaming sessions data, compared to traditional collaborative or content-based approaches. Due to the use of Transformer architectures, item and user information is no longer necessary (although is good to use as side information), and deep neural architectures are utilized. To our knowledge, less research has been done on utilizing music streaming session data and modern Transformer architectures on session-based recommendation. This status quo will be explored later in Section 2. The plausibility of this attempt is demonstrated by the use of Transformer architectures for recommendation of items from different domains (Chen et al. 2019; Fang 2021; Luo et al. 2020).

Not just settling on a novel system, this research created a high performing system that displays significant recommendation performance. In order to do this, our research used specific training methods for music streaming sessions and also utilized metadata of song sessions and feature information.

The idea of simply switching words/sentences in language modeling and songs/streaming sessions in next-item recommendation seems straightforward, but is expected to have many obstacles. The characteristics of language and music obviously do not align. For example, words inside a sentence follow common grammar rules while songs inside a streaming session do not. In addition, there is no guarantee that Transformer architectures, which were successful in capturing information within language data, will be successful in capturing information within music streaming sessions data. This research explored this unsure compatibility and developed the most compatible model by first comparing various Transformer based models, and then further fine-tuning and adding feature information.

1.3 Need for Research

1.3.1 Recent Trends

Music consumption culture has changed significantly over the past few years. After the monumental shift from physical CDs to online downloads, the recent shift from online downloads to online streaming introduced huge changes in the market (Datta, Knox, and Bronnenberg 2017). Major music streaming platforms like Spotify and Apple Music have emerged, and session-based recommendation has gained importance in the music industry to provide more accurate recommendations to users.

In search of improved RecSys, numerous competitions have been held online. A huge amount of data is normally needed to improve recommendation, so the size of music data made available to the public has continuously increased as well. The music streaming sessions dataset released by Spotify (Brost, Mehrotra, and Jehan 2019), which will be used in this research, clearly shows that there is a need for more advanced RecSys, and firms are not afraid to open up their data for this.

1.3.2 Dataset Characteristics

In terms of session-based recommendation, attention based architectures like Transformers and BERT have been previously explored and proven superior performances than its competitors (Sun et al. 2019a). However, these projects were mostly on retail and movie data, which is totally different from music streaming data. Music streaming data has numerous subtle features to be investigated, like user actions while listening, how a song was played, etc. Therefore, whether these subtleties can be explained by Transformer architectures needed research.

To dive deeper into a the uniqueness our dataset, the Music Streaming Sessions Dataset (MSSD), Table 1.1 is shown, with dataset statistics of MSSD and 4 other datasets that were tested with session-based recommendation in BERT4Rec (Sun et al. 2019a). The comparison will mainly be conducted with the MovieLens (ML-1m, ML-20m) dataset, since movies would mostly likely be deemed best similar to music. The differences probably stem from

Datasets	No. Users	No. Items	No. Actions	Avg. Length	Density
MSSD	39,582,988	2,377,710	271,053,259	19.2	< 0.01%
Beauty	40,226	54,542	0.35m	8.8	0.02%
Steam	281,428	13,044	3.5m	12.4	0.10%
ML-1m	6040	3416	1.0m	163.5	4.79%
ML-20m	138,493	26,744	20m	144.4	0.54%

Table 1.1 Datasets Statistics.

the innate difference in average runtime of a song and a movie. While vast majority of movies show a runtime between 80 and 120 minutes (Jarzabek 2018), the average length of songs are known to be around 3 minutes and 30 seconds (Sanchez 2019). Thus, many aspects in consumer behaviour will vary. For instance, watching a full movie and listening to a full song would not always be the same amount of positive feedback, since watching a full movie takes much more commitment. This means that the methods for distinguishing if a user actually gave positive feedback to an item will be different for movie and music datasets. It is actually easier for the MovieLens dataset since they have an explicit feedback information with the ratings user gave to items. For MSSD, we do not have any explicit feedback, just an implicit feedback that the user listened to the songs and extra session information to help predict a user's preference. These session information will be mentioned in later sections, and are information like 'how a user played this track' and 'the length of pause before this track', which are not included in movie datasets.

The length of sessions will also vary, since more songs will be listened to than movies watched. In MSSD however, all sessions are cut to length 20, and since past work also cuts maximum session length to 20 (Sun et al. 2019a), session length itself is not expected to cause differences during experiments. In

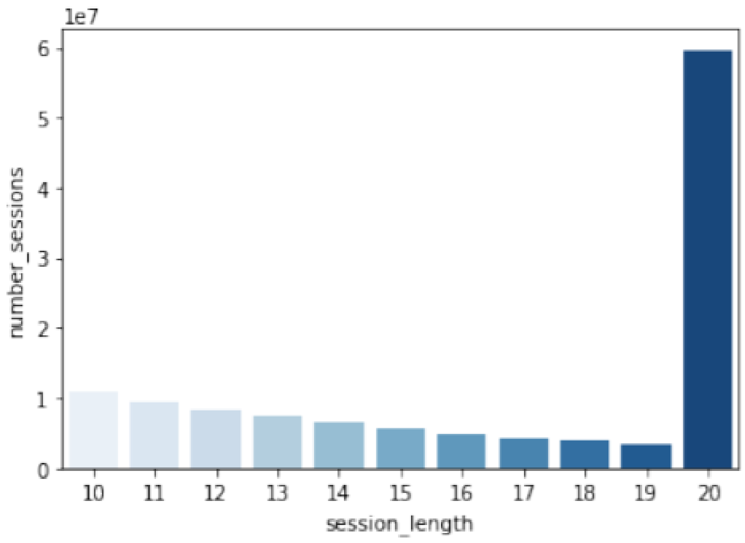

Figure 1.2 Distribution of Session Lengths.

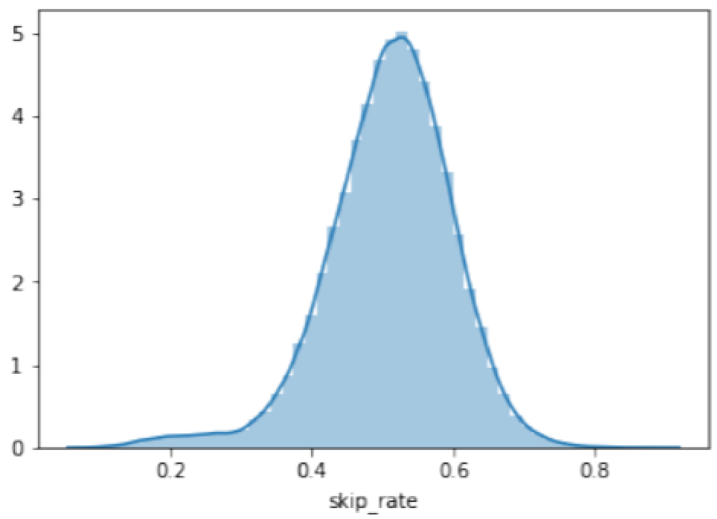

Figure 1.3 Distribution of Skip Rate.

Table 4.1, average length of the movie datasets (ML-1m, ML-20m) is the length before truncation and that of MSSD is the length after truncation. Luckily, from the authors of MSSD, we can see the rough distribution of session lengths in Figure 1.2 (Brost, Mehrotra, and Jehan 2019). It shows that most sessions were truncated since they were over length 20. Apart from length of sessions, we can evidently observe the incomparable number of users, items and interactions, thus resulting in extreme sparsity for MSSD. This is not just compared movie datasets, it is also for other datasets in the field of online reviews (*Beauty*) and eCommerce (*Steam*).

The most unique features about MSSD would be track metadata. Track metadata are numeric audio features like acousticness, energy, tempo, etc. that will be explained deeply in later sections. These are innate content-based characteristics from music that are different from the ones for movies, and information that makes MSSD interesting. In addition, the users in MSSD are from the streaming platform Spotify, and are sometimes exposed to hindrances like advertisements if they are free users, a.k.a. ‘Non-premium’ users. This would result in skipping a track even if the user likes it, and need to be acknowledged. In terms of skipping, this information also plays a big part in explaining user preferences. MSSD contains information on the length a user listened to a track before skipping. From Figure 1.3, provided to us by MSSD authors (Brost, Mehrotra, and Jehan 2019), we can see that the rate of skipping is mainly centered around 0.5 and is indeed relevant. There is also a phenomenon named ‘Sleeping Sessions’. They are sessions where the user mindlessly turns on a playlist and listens to all songs regardless of preference. These sessions need to be filtered out so that they do not disrupt the training process.

Due to the above trends and differences, the need for this research of this thesis certainly seems to exist. This research will match the trend in RecSys with Big Data, and also provide deeper understandings about the relationship between various unique characteristics of streaming data and next-song recommendation.

2 Related Works

2.1 Overview of NLP and RecSys

In this section, we will discuss related works to look at how the field of NLP developed and effects of those developments in the field of session-based recommendation.

Before Transformers, there were several architectures that attempted to model sequences/sessions in the field of NLP. Before Transformers came out, Recurrent Neural Networks, or RNN, had prevalence. It took timestamps of items into account and tried to predict the next item in a sequence by learning information from all the past items. However, it had a vanishing gradient problem as sequences got longer, and failed to successfully model long-term dependencies. Therefore, RNN based architectures that tried to solve RNN's vanishing gradient problem gained popularity afterwards. Some examples were Long Short-Term Memory (LSTM), which utilized a new cell state that would keep long term memory, and Gated Recurrent Units (GRU) (Chung et al. 2014) which made LSTM simpler and used gates that would control the amount of past information to keep and new information to accept.

Some researchers attempted use RNN for recommendation, specifically GRU in session based recommendations, and this produced the best results in the field at that time (Hidasi et al. 2016). There were previous attempts to utilize NLP approaches like Word2Vec (Mikolov et al. 2013) to RecSys, like Prod2Vec (Grbovic et al. 2015) and Doc2Vec (Le and Mikolov 2014). However, these only aimed to learn product representations using the Word2Vec model. GRU4Rec developed a session-based RecSys, and attempted to solve the problem of large vocabulary in RecSys due to high number of items by using a loss function similar to Bayesian Personalized Ranking. This way, the authors succeeded in only comparing scores of two items (positive, negative) and not comparing scores of all items in the vocabulary.

Then Transformers and BERT appeared and changed the field of NLP completely. Transformers was a model based on encoder-decoder architectures and introduced self-attention (Vaswani et al. 2017), building from the Attention paper (Bahdanau, Cho, and Bengio 2016). Self-attention allows words at each position to learn which positions to pay attention to and learn information from. Bidirectional Encoder Representations from Transformer, or BERT, improved Transformers by only taking the encoder part and applying masked language modeling to train the model (Devlin et al. 2019). This way, words at every position learned bidirectional representations, an improvement from unidirectional representations in RNN architectures. GPT-2 (Radford and Narasimhan 2018) was also a model that utilized Transformers but unlike BERT, it only used a stack of Transformer decoder blocks using a causal language modeling approach.

Researchers in the recommendation systems field reacted to this change and came up with a few architectures that used Transformers architecture. There was SASRec (Kang and McAuley 2018), which utilized the Transformers decoder like GPT-2 to produce sequential recommendation. Then BERT4Rec (Sun et al. 2019a) came out, which showed improvements from SASRec by using BERT. The reasoning behind BERT4Rec's superior performance was no different than why BERT showed state of the art performances at that time. It was because BERT utilized bidirectional information while Transformers decoder only utilized unidirectional information.

Recently, modern Transformer architectures like XLNet (Yang et al. 2020) and ELECTRA (Clark et al. 2020) tried to fix BERT's drawbacks like the discrepancy between pre-training and fine-tuning, and showed better performances. They achieved this by introducing different pre-training methods like permutation language modeling and replace token detection. Consequently, it seemed natural to try to implement the new models to session based recommendation systems due to the success of BERT4Rec, and NVIDIA recently released the paper Transformers4Rec that introduced a framework able to utilize modern Transformer architectures like the ones mentioned above to produce session-based recommendation models (Souza Pereira Moreira et al. 2021). By using models in the HuggingFace (HF) library, Transformers4Rec can produce models fit for next-item recommendation. HuggingFace is an open-source library that contains code implementations for various state of the art Transformers architectures like BERT, XLNet and ELECTRA (Wolf et al. 2020). It is contributed by more than 400 developers, and allows continuous model uploads from the general public. Transformers4Rec with XLNet showed great results on movie and retail data, and the authors experimented further with news datasets. The model from this thesis is based on Transformers4Rec, but is modified in order to learn relationships in music streaming data. Our model is presumably one of the initial researches on Transformers for music streaming session-based recommendation, containing unique challenges as mentioned in Section 1.3.

2.2 Past Works on Incorporating Features

There has been past works on incorporating features to RecSys, but most of them were not in the field of session-based recommendation. Early development in RecSys has been in the field of collaborative filtering and matrix factorization, without the use of content-based features (Cremonesi, Koren, and Turrin 2010; Gemulla et al. 2011; Jin, Chai, and Si 2004; Koren 2008). They were focused on learning patterns inside a matrix of users, items and the magnitude of preference based on numeric ratings. The main datasets used were movie datasets, like MovieLens and Netflix. Later on, the systems started to consider context like time and characteristics of users and movies (Karatzoglou et al. 2010; Koren 2009). Although these were attempts to incorporate features, they were not solving next-item recommendation for sessions.

Past researches on session-based recommendation mentioned above mostly involve simple item ID embeddings only, without any feature incorporation. GRU4Rec and BERT4Rec explicitly mentions feature incorporation in the future works (Hidasi et al. 2016; Sun et al. 2019a), and SASRec attempted item embeddings but removed them due to impaired performance (Kang and McAuley 2018). Transformers4Rec is one of the few papers that attempted extensive feature incorporation with item context and achieved a small increase in performance (Souza Pereira Moreira et al. 2021). Another paper that used attention for next-item recommendation attempted context embedding for transaction data (Wang et al. 2018). This thesis aims to join this small group of research on extensive feature incorporation for next-item recommendation, and will specifically be on music streaming sessions data.

3 Methodology

3.1 Music Streaming Sessions Dataset

The Music Streaming Sessions Dataset (MSSD) was released in late 2018 by Spotify, with around 160 million listening sessions and associated user actions (Brost, Mehrotra, and Jehan 2019). In addition, metadata and audio features for approximately 3.7 million unique songs are included. It is one of the biggest datasets of song metadata and streaming sessions, and was intended to aid tasks like music information retrieval and session-based recommendations.

Spotify acknowledged the fact that while the need for improvements in music streaming session recommendation continuously increased, there weren't any publicly available user log data to help design recommendation systems and learn information between sequence of interactions. One of the main ambitions listed by the authors was sequentially recommending items for users since they deemed it was important to music streaming services as well as other services like movie and retail. This thesis conducted research to fulfill this ambition by developing a novel recommendation system for the MSSD.

Ideally, it would be best to have a dataset that contains sessions of songs that users approved they liked. In other words, this would be a dataset with explicit feedbacks. However this is highly unlikely, which makes creating RecSys from large interaction datasets hard (Oard and Kim 1998). MSSD helps solve

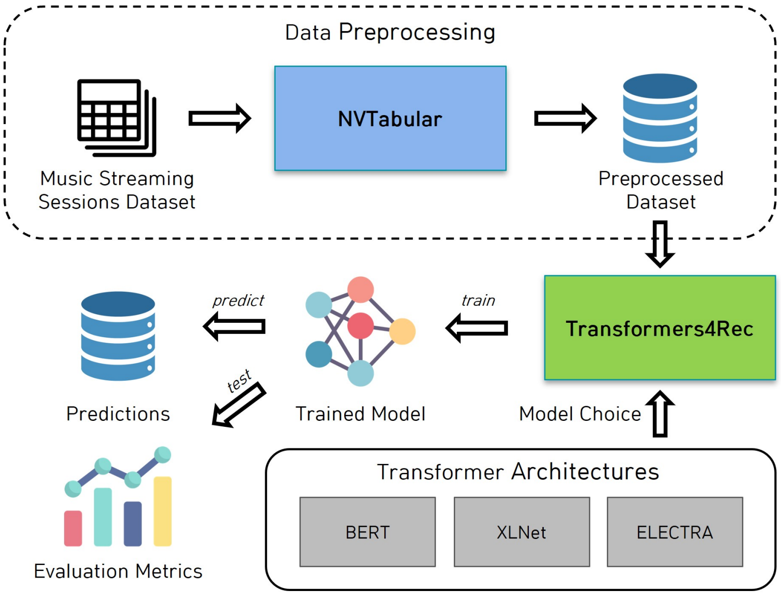

Figure 3.1 Model Pipeline.

with a large number of implicit feedbacks. Sadly, just the fact a user listened to a song does not equal to positive feedback. Therefore later on, adequate preprocessing with information about the streaming sessions will be mentioned to develop a dataset with stronger feedbacks.

3.2 Music Recommendation Model

The end-to-end recommendation system framework from NVIDIA, Transformers4Rec, will be used as the music streaming session-based recommendation model. Transformers4Rec was intended to be seamlessly integrated with the NVTabular [1] library, in order to allow fast preprocessing for large size datasets. Therefore, this thesis will utilize the NVTabular library for the preprocessing of MSSD. The entire model pipeline is shown in Figure 3.1.

[1]`https://github.com/NVIDIA-Merlin/NVTabular`

3.2.1 NVTabular

NVTabular supports both common and advanced feature engineering with GPU acceleration. It also has specialized operations for session-based recommendation, like sorting interactions by time then grouping them together by user or item. Truncating sequences are also possible. Due to these convenient functions and compatibility with Transformers4Rec, NVTabular will be used in this research for data preprocessing, like shown in the top part of Figure 3.1.

When preprocessing is done, NVTabular saves the data into Parquet format. NVTabular can load these files directly into GPU memory, allowing faster training and evaluation.

3.2.2 Transformers4Rec

Transformer4Rec works in the following way. With the given preprocessed dataset and chosen Transformer architecture in Figure 3.1, a model is trained. All songs in every session are transformed into embedded representations. In this section, the user can decide to which features to use. We can only use item id like past architectures like GRU4Rec and BERT4Rec (Hidasi et al. 2016; Sun et al. 2019a), or all the audio features of every track. Then the session sequences are masked based on pre-training procedures for the model chosen, and sent into the main training blocks of the model. Lastly, the model is trained by comparing the prediction to the ground truth.

The Transformer architecture can be any model in the HuggingFace (HF)'s Transformers library, implemented in either TensorFlow or PyTorch. Transformers4Rec inherits HF's optimized training and evaluation pipeline for NLP tasks, controlled by the *Trainer* class. Then the *Trainer* class is modified to fit session-based recommendation. Transformers4Rec also provides numerous traditional evaluation metrics for use, such as NDCG@N, Recall@N and Precision@N.

In this thesis, only BERT, XLNet and ELECTRA will be used. BERT is chosen due to its previous success with movie and retail data in BERT4Rec (Sun et al. 2019a). XLNet is chosen because it showed great results with not only retail, but also with news data which is relatively under explored (Souza Pereira Moreira et al. 2021). ELECTRA is chosen because its size is relatively smaller than most Transformer architectures, making it seem suitable for fast computation that is needed with real-time next song recommendation on streaming platforms (Clark et al. 2020).

The final trained models will output evaluation metrics for performance measurement.

3.3 Feature Embeddings

track id	duration	release year
popularity	acousticness	beat strength
bounciness	danceability	dyn range mean
energy	flatness	instrumentalness
key	liveness	loudness
mechanism	mode	instrumentalness
speechiness	tempo	time signature
valence	acoustic vector	

Table 3.1 Track metadata features.

Table 3.1 shows all the track metadata provided to us from MSSD (Brost, Mehrotra, and Jehan 2019). There are 23 features in total. Descriptions and distributions of the audio feature values are on the MSSD website. Using these audio features, a few features that seem to distinguish tracks best are added as extra information in the initial feature embedding stage of Transformers4Rec described in Section 3.2.2. Simple embedding measures like only using item id or simple soft one-hot encoding will be experimented as well. Then, performance differences resulting from different embedding methods will be investigated to provide insight on whether to use feature information or not, and if so, which features to use.

The some possible questions addressed at this stage are: Do similar features with high correlations both need to be included? Do some features disrupt actual representations of tracks? For example, 'energy' and 'liveness' seems redundant at face value. In addition, maybe not all songs with fast tempo should be considered similar, since there might be genre differences, and difference in methods measuring beats per minute.

Positional embedding, another important part of Transformers embeddings that learns from position of items in a sequence, is reviewed in the works by ATRank (Zhou et al. 2017). The authors of ATRank found out that it is difficult to learn good embedding on a continuous time feature, and decided on using log scale to discretize the elapse time between session sequences, and then represent it as categorical feature embeddings. However in our case, tracks occur right after each other with no time gaps, therefore simple sine cosine positional embeddings used for sentences are deemed satisfactory.

3.4 Session Information

Session information of MSSD is shown in Table 3.2 (Brost, Mehrotra, and Jehan 2019). Using this information, this research will try to experiment with

Column name	Column description	Example value
session id	unique session identifier	57_55129e3f-29bf-4ef6-aa72-d140333eac9c
session position	position of track within session	18
session length	length of session	20
track id	unique track identifier	t_aae12819-de17-4dd3-97b0-cad4dd7b9a56
skip 1	whether the track was only played very briefly	false
skip 2	whether the track was only played briefly	false
skip 3	whether most of the track was played	true
not skipped	whether the track was played in its entirety	false
context switch	whether the user changed context between the previous row and the current row	true
no pause	whether there was no pause between playback of the previous track and current track	false
short pause	whether there was a short pause between playback of the previous track and current track	true
long pause	whether there was a long pause between playback of the previous track and current track	true
num seekfwd	the number of times the user scrubbed forward during playback	0
num seekbk	the number of times the user scrubbed backward during playback	3
shuffle	whether the track was played with shuffle mode activated	false
hour of day	hour of day (integers between 0 and 23)	18
date	date in YYYY-MM-DD format	2018-09-10
premium	whether the user was on premium or not	true
context type	what type of context the playback occurred within	catalog
reason start	cause of this track play starting	forward button
reason end	cause of this track play ending	track done
uniform random	whether shuffle would be uniformly random for this session	false

Table 3.2 Session information.

various criteria when selecting interactions inside sessions to be deemed as feedback. In other words, some listening actions by uses may be discarded, since they should not be deemed as actual positive feedback. For example, if a track was only played very briefly then skipped, the model might probably have to learn that the user does not prefer this track. If the track was not skipped at all, which means 'not skipped' column would be 'true' in session metadata, it may have to concerned as stronger positive feedback than other interactions. Another interesting example to point out is that the track might have been played during 'shuffle' mode, where songs are played at random. This might mean that the user has no preference over this song.

Therefore, this research aims to figure out how to weight track interactions inside sessions based on their session metadata shown in Table 3.2, by experimenting based on several hypotheses like the ones mentioned above.

3.5 Transformer Architectures

The following Transformer architectures will be implemented to compare performances on MSSD.

- **GRU4Rec** (Hidasi et al. 2016): Not exactly a Transformer based model but a RNN based RecSys that had superior performances before Transformer based systems came out. Included to serve as a comparison with non Transformer models.
- **BERT** (Devlin et al. 2019): A deep Transformer based model which utilizes stacked Transformer encoders and masked language modeling task for pre-training.
- **XLNet** (Yang et al. 2020): A deep Transformer based model which utilizes permutation language modeling task for pre-training.
- **ELECTRA** (Clark et al. 2020): A deep Transformer based model which utilizes replacement token detection task for pre-training.

GRU4Rec is a RNN based method is included to compare with Transformer based methods in our research. Although simpler baseline models like POP (choosing the most popular item) and BPR-MF (Matrix factorization method with implicit feedback and pairwise ranking loss) Rendle et al. 2009 do exist, they are omitted since they already showed worse performances than GRU4Rec. Along with these GRU4Rec, the three models (BERT, XLNet, ELECTRA) will be experimented, making a total of four models compared.

3.6 Metrics

Related works mentioned in Section 2 utilized several evaluation metrics, including *Hit Ratio* (HR), *Normalized Discounted Cumulative Gain* (NDCG) and *Mean Reciprocal Rank* (MRR). In this thesis, for comparison with other works, only HR and NDCG will be used. The two are selected due to their popularity among related works.

HR is reported with $k = 10, 20$. HR@1 means it is a hit if the desired ground truth is in the top 1 predicted items, and HR@5 means it is a hit if the desired ground truth is in the top 5 predicted items, and the same logic applies for HR@10. HR is the same as recall if there is only one relevant item in the recommendation list. The formula for calculating HR is as below:

$$Hit\ Ratio = \frac{number\ of\ hits}{number\ of\ hits + number\ of\ misses}$$

NDCG is a measure of ranking quality, and asserts that very relevant results are more useful than somewhat relevant results which are more useful than irrelevant results (cumulative gain). It is a more complex metric than HR, and is expected to help measure performances more finely.

4 Experiments

4.1 Data Preprocessing

Since the dataset size is bigger than 300 gigabytes, it is hard to only use NVTabular. Therefore, initial data preprocessing was done by PySpark[1], which is an interface for Apache Spark in Python. PySpark makes big data preprocessing possible even with limited GPU memory.

MSSD in its entirety contains session csv data from 2018 July 15th to 2018 September 18th. Each row contains an interaction, meaning an instance where the user listened to a song. For each row, there are columns containing information about the interactions, mentioned in Section 3.4. Multiple rows from the same user session would then form a single streaming session. Due to the limit of available computing resources, only sessions from 2018 August 1st to 2018 August 31st are used, which is about half of the entire dataset. After loading the data with PySpark, the following preprocessing is done.

(a) Non-premimum interactions are eliminated, and sleeping sessions are discarded.

- When a user doesn't use Spotify premium, they are bound to advertisements and other hassles that may affect their listening behavior. Therefore, interactions by non-premium users are discarded.

[1]`https://spark.apache.org/docs/latest/api/python/`

- Sleeping sessions do not inform us about users' preferences, since they are mindlessly listening. Sleeping sessions are distinguished by filtering out sequences that do not have any user actions throughout the whole session.

(b) Create datetime for each interaction.

- For each interaction, information about the date and time of its occurrence is converted into a datetime object for future grouping.

(c) Merge different session features in to one column

- In section 3.4, we can see that columns like 'skip_1', 'skip_2' can be merged into one categorical column, since collectively they form a one hot vector showing how long a song was listened before being skipped.
- Therefore, columns 'long_pause_before_play', 'short_pause_before_play', 'no_pause_before_play' are combined into one column named 'pause_before_play'.

(d) Only keep songs listened to fully.

- To gain more confidence for each interaction being a positive feedback, only songs listened to the end without skipping are kept.

(e) Using NVTabular, all rows interactions are then grouped to a single session by their session ID, with information like track IDs and categorical features grouped as lists ordered by time.

(f) Resulting sessions with length 1 will be ignored. All session lengths will be capped at length 20, which already seems to be the case with MSSD.

Statistics of the final preprocessed version of our dataset are shown in Table 4.1. The dataset is saved as Parquet files ready for training with Transformers4Rec.

Dataset	days	users	items	interactions	Avg. length	Density
MSSD	31	39,582,988	2,377,710	271,053,259	19.2	< 0.01%

Table 4.1 MSSD statistics only.

4.2 Embedding

4.2.1 No features

For most of the experiments with the chosen Transformer architectures, a simple embedding just by using item IDs is used. This means that the initial embedding vector to be fed into the model will be equal to the list of track IDs for each session that we obtained at the end of preprocessing in Section 4.1. Of course, the list of track IDs will be transformed into the embedding vector dimension set during training parameter settings, and be added with traditional positional embedding information in the form of sine and cosine functions. When the embedding vector is finally ready, the four models (GRU4Rec, BERT, XLNet and ELECTRA) will be trained and evaluated to find the best performing model.

4.2.2 Session features

Our experiments will then extend from a simple embedding with no features to incorporating session information. This experiment will show if certain session information can prove its correlation with a user's preference on a track, by enhancing performance. In addition, models will have different capacities in learning rich feature information, so this experiment will show if our Transformer based models are capable of learning music related features.

The four models to be tested, GRU4Rec, BERT, XLNet and ELECTRA, will all be added session features from Section 3.4 to test the effect of additional features. The session features selected are as follows:

(a) Reason for track start

(b) Reason for track end

- The above two features are chosen due to the possible connections with user preference of the track. If the reason for track start is forward button, it may imply that the user skipped tracks and stopped at this one, since it was the track the user wanted. If implications like these are meaningful, it would be a good idea to make the model pay attention to these features as well.

(c) Pause before play

- If there was no pause between the previous track and the current track, it may infer that the current track was simply played due to being the next track on the playlist. If there was a long pause, on the other hand, it might mean that the user searched for this track after the previous track ended. If these possible implications are true, they may be meaningful to seeing which track the user prefers more.

The features selected above will test if the hypotheses on the relationship between session information and user preference is indeed meaningful. Although there are a lot of features in Table 3.2, only the above features are selected for simplicity and possible relevance to user preference. After feature information is added, their statistics like minimum and maximum values are provided in a schema file to be read by Transformers4Rec framework before training. With the schema file provided, similar to the experiments with no features, all models with experiments will be evaluated to find the best performing model. This will show which model out of the four is able to take maximum advantage from additional feature information.

4.2.3 Song features

In addition to the session information, track metadata, or song features, will be utilized to produce an even more rich embedding vector. Track metadata is shown in Table 3.1. When we use track metadata for embeddings, we are basically implementing a content-based method into our RecSys. This is because now songs will be represented by features that are directly correlated with their content. For example, if a song's tempo is 160 beats per minute and energy value is 90 and we decide to only use tempo and energy for embeddings, the song's embedding vector will be $[160, 90]$. This type of embedding method places songs with similar content closer. A song with 170 beats per minute and 95 energy, ($[170, 95]$ embedding vector) will be deemed much similar to the $[160, 90]$ song than a 100 beats per minute, 50 energy ($[100, 50]$ embedding vector) song.

This way, songs with similar track metadata to the previous songs played may have a higher change of getting recommended, if our model learns that this algorithm is beneficial. For example, if the train data shows that when users listen to generally high tempo songs, they do indeed listen to high tempo songs for the next sequence, the model will learn this pattern and infer based on information.

There are many track metadata available, but only four were finally chosen to use in training. The reason is because some features were too skewed or

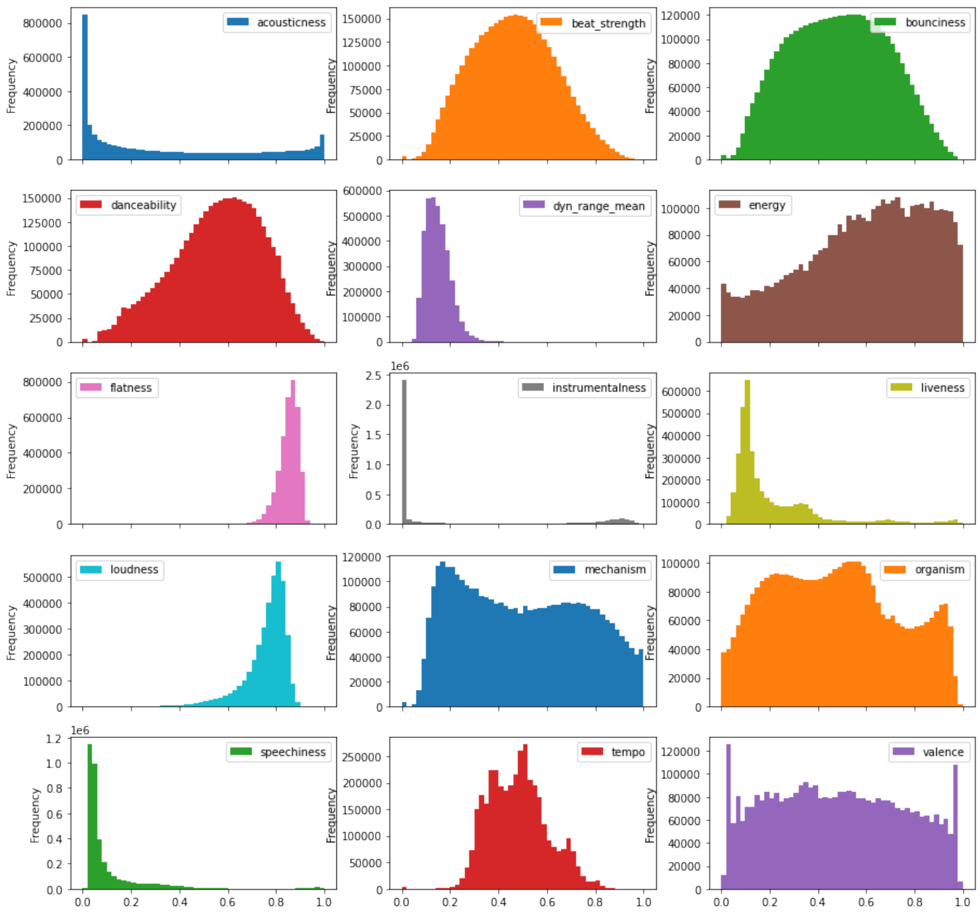

Figure 4.1 Distributions of feature values across songs for audio features.

had almost no variance, and didn't prove useful in distinguishing tracks. This analysis was done by first normalizing all feature values to fit in 0 1 range, and then drawing a plot to see how they are distributed. The plots are shown in Figure 4.1 (Brost, Mehrotra, and Jehan 2019). You can see that some of them actually have very similar distributions, and including all of them would be redundant in distinguishing songs. For example, *beat strength*, *bounciness* and *danceability* shows similar plot shape, and it seems like only one of them need to be included.

After this analysis, *energy*, *danceability*, *organism* and *valence* were chosen. The four values will go through the process that session information went through in the above section to have statistics on a schema file and be added in the embedding vectors for training and evaluation.

4.3 Hyperparameters

Since four models had to be experimented over a huge dataset, it was hard to perform extensive hyperparameter tuning on all the models. Therefore, based on experiments on the Transformers4Rec paper, initial hyperparameter settings were done. Out of the four datasets in that paper, our dataset best resembled *REES46 eCommerce* [2], which contains user sessions from a multi-category online store. The resemblance came from the size of the dataset, with the most number of items and interactions, and the fact that it was also spanned for a month. The exact hyperparameter settings are mentioned in this appendix [3] provided by the authors. From the starting points for each model, minor

[2] `https://www.kaggle.com/mkechinov/ecommerce-behavior-data-from-multi-category-store`

[3] `https://github.com/NVIDIA-Merlin/publications/blob/main/2021_acm_recsys_transformers4rec/Appendices/Appendix_C-Hyperparameters.md`

changes are made to improve performances. The models are all trained from scratch on a single NVIDIA TITAN RTX GPU.

4.4 Training

4.4.1 Problem Statement

The problem statement is equal to that of previous session based RecSys. $U = \{u_1, u_2, ..., u_{|U|}\}$ will denote the set of users, and $V = \{v_1, v_2, ..., v_{|V|}\}$ will denote a set of items, and $S_u = [v_1^u, ...v_t^u, ..., v_{n_u}^u]$ will denote the session sequence for user u, where v_t^u is the item that user u interacted with at time step t. n_u is the length of given session S_u. Our model's goal is to predict what item user u will interact at the next time step, $n_u + 1$, based on S_u. This can be seen as probability:

$$p(v_{n_u+1}^u = v|S_u)$$

4.4.2 Pipeline

Training will be done with the Transformers4Rec library which utilizes the HuggingFace Transformers library, as mentioned in Section 3. The pipeline follows the one shown in Figure 3.1, where we use the preprocessed dataset in Parquet file form, load to Transformers4Rec framework and train the model. Inside the framework, we choose the Transformer model of our choice (BERT, XLNet, ELECTRA), and set its hidden dimension, number of heads, embedding dimension, etc. Once our chosen model is ready solve the problem of next item prediction, data is loaded from the Parquet file in batches and trained.

4.4.3 Incremental Training, Evaluation

An interesting difference between session-based recommendation and other forms of recommendations, is that session-based recommendations receive a larger dataset every day due to the increasing number of user-item interactions every day (Zhang et al. 2020). Therefore, it is important for the RecSys to adapt to this large increasing dataset with a training method named incremental training, which is a norm seen in past works (Moreira, Jannach, and Cunha 2019; Souza Pereira Moreira, Ferreira, and Cunha 2018; Sun et al. 2019b). Incremental training works by setting a sliding window with a single time unit, and training and evaluating the model at every time unit. All sessions are split into time windows T based on the unit size, and evaluation is performed for each next time window T_{i+1}, where $i \subseteq [1, n-1]$, using model trained until the current time window T_i.

For our experiment, the time unit is a single day. Since the dataset spans 31 days, the sessions are split into 31 parts, and trained incrementally. To provide an example, the model will be trained with only the training set of the first day, and then evaluated with the evaluation set of the second day. Then the model will be trained on the training set of the second day, and then evaluated with the evaluation set of the third day. Evaluation is done with metrics described in Section 3, and the final performances of models are calculated as an average of all time windows, to see if the model performs well over time.

Dataset	Metric	GRU	BERT	XLNet	ELECTRA	Density
MSSD	HR10	0.2706	**0.2767**	0.2642	0.2680	$< 0.01\%$
	HR20	0.3344	**0.3734**	0.3215	0.3291	
	NDCG10	**0.1833**	0.1803	0.1764	0.1709	
	NDCG20	0.1994	**0.2003**	0.1901	0.1897	
Beauty	HR10	-	0.3025	-	-	0.02%
	NDCG10	-	0.1862	-	-	

Table 4.2 Performance with simple item embeddings.

4.5 Results

4.5.1 Simple item IDs

Table 4.2 shows the final results of the experiments. We can first see the surprising competence of GRU4Rec, with performances of no big difference with Transformer-based architectures. BERT was the best performing Transformer-based model. This can be explained by XLNet and ELECTRA needing more hyperparameter tuning for better performance, since they are more complex architectures. Another explanation that seems more probable is that BERT and masked language modelling might just be of better fit for MSSD, since even after extensive hyperparameter tuning, BERT still performed best.

BERT showed the best performance with NDCG20 at 0.2003 and HR20 at 0.3734. The small gap showed some superiority of a Transformer-based method. Comparing to some past experiments in BERT4Rec (Sun et al. 2019a) and Transformers4Rec (Souza Pereira Moreira et al. 2021), this result shows significance. The Beauty [4] dataset shown in BERT4Rec is similar with our dataset in terms of density, which calculates the fraction of interactions seen

[4]`http://jmcauley.ucsd.edu/data/amazon/`

and interactions possible from our users and items. In fact, our dataset is much lower than Beauty in density, which has a density of 0.02%. It is known from BERT4Rec and other works that datasets with lower densities are harder to show good performances, and our dataset even has more items and users, which makes it harder. Considering this, out model performed extremely well, with a HR10 and NDCG10 similar with Beauty's, shown in Table 4.2.

A more important fact is that our dataset was not comprised of explicit feedback or even strong implicit feedback. Streaming might be done mindlessly by turning on a playlist or while driving a car, which hinders the aspect of positive feedback. The results show that the extensive preprocessing to make this dataset work with a session-based RecSys actually did a good job in keeping the sessions with stronger positive signs. Overall, even with just simple item ID embeddings, our model performed reasonably well despite the sparsity and coarseness of the music streaming session dataset.

4.5.2 Item IDs + Session Information

In addition to simple item IDs, session information was taken into account by incorporating them into embeddings before training. From this point on, only Transformer models were experimented. The results in Table 4.3 showed a slight increase in overall performance, but not for all metrics for all models. This means that the hypotheses mentioned in Section 4.2 had a bit of significance, although not incredibly strong. In terms of increase due to adding session information, XLNet did show the highest value, but was still lower in performance than BERT. However, since the gap between gains for all models are very small, so it would be hard to firmly conclude that XLNet has a better capacity for incorporating rich feature information. It may have been that since XLNet was lower in performance, it had more capacity to improve.

Dataset / Features	Metric	BERT	XLNet	ELECTRA	Density
MSSD	HR10	**0.2799**	0.2745	0.2723	< 0.01%
+	HR20	**0.3680**	0.3271	0.3322	
Session Information	NDCG10	**0.1888**	0.1858	0.1737	
	NDCG20	**0.2005**	0.1905	0.1919	

Table 4.3 Performance with added session information.

4.5.3 Item IDs + Session Information + Track Metadata

To utilize all the feature information given for our dataset, track metadata was incorporated after session information. The results are shown below in Table 4.4. There was no evident increase from the results in Table 4.3 with only session information. We can interpret this result in several ways. One is that song preferences of users do not depend a lot on strictly similar numeric audio feature values. This can make sense since we usually listen to songs that match our taste, and a human's taste is hard to quantify. Another explanation is that after incorporating session information, our models' capacity in learning more patterns have saturated. This seems plausible since Transformer based models are already learning extremely complex patterns from sequences without any additional feature embeddings. The second explanation seems more likely, since even with session information added, there wasn't much increase in performance. It seems that in the future, there needs to be research on how to incorporate rich feature information in a way other than adding them to embedding vectors.

Overall, all these results seem to have significance in that this is a pioneer in this field of music streaming session based recommendation with Transformer architectures. Future research can now base their comparison on this research, and gain insights on aspects like data preprocessing and feature engineering specifically for music streaming session datasets.

Dataset / Features	Metric	BERT	XLNet	ELECTRA	Density
MSSD	HR10	**0.2849**	0.2701	0.2756	< 0.01%
+	HR20	**0.3615**	0.3489	0.3293	
Session Information	NDCG10	**0.1812**	0.1795	0.1756	
Track Metadata	NDCG20	**0.2010**	0.1870	0.1889	

Table 4.4 Performance with added session information and track metadata.

5 Conclusion and Future Works

Transformer architectures have shown success in session-based recommendation for various domains. In this thesis, we dove into a yet unexplored domain of music streaming sessions. Various preprocessing stages were performed to turn an unrefined data into a dataset with somewhat strong implicit feedbacks, and be ready in a format for session-based RecSys. Feature information were also refined to categorical columns, and added to help train the model. For training, the Transformers4Rec framework was used with incremental training procedures. The final experimental results on popular metrics show that despite the extreme sparsity and coarseness of our dataset, our best model with features are in competence with other datasets with more density.

Overall, this thesis contributed a novel model in the field of music streaming session-based recommendation with Transformer architectures. It investigated MSSD's unique characteristics to show differences with other datasets, and performed a comparison of different Transformer architectures to show compatibility with music data. Future researchers can refer to our methodology regarding data preprocessing, feature engineering, etc. as a starting point or a method to compare with.

There is still room for improvement in the future. Extensive hyperparameter tuning with increased computing resources is expected to boost the performance, since the experiment settings for this thesis did not allow much room for increased epochs, dimensions, etc. In addition, the features chosen in this thesis were based on hypotheses that seemed plausible. It might be beneficial in the future to test all the features empirically to see which feature contains the most information on user preference for songs.

Bibliography

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio, 2016: *Neural machine translation by jointly learning to align and translate.* arXiv: 1409.0473 [cs.CL].

Brost, Brian, Rishabh Mehrotra, and Tristan Jehan, 2019: The music streaming sessions dataset. *The World Wide Web Conference.* WWW '19. San Francisco, CA, USA: Association for Computing Machinery, 2594–2600. ISBN: 9781450366748. DOI: 10.1145/3308558.3313641. URL: https://doi.org/10.1145/3308558.3313641.

Chen, Xusong, Dong Liu, Chenyi Lei, Rui Li, Zheng-Jun Zha, and Zhiwei Xiong, 2019: Bert4sessrec: content-based video relevance prediction with bidirectional encoder representations from transformer. *Proceedings of the 27th ACM International Conference on Multimedia.* MM '19. Nice, France: Association for Computing Machinery, 2597–2601. ISBN: 9781450368896. DOI: 10.1145/3343031.3356051. URL: https://doi.org/10.1145/3343031.3356051.

Chung, Junyoung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio, 2014: *Empirical evaluation of gated recurrent neural networks on sequence modeling.* arXiv: 1412.3555 [cs.NE].

Clark, Kevin, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning, 2020: *Electra: pre-training text encoders as discriminators rather than generators.* arXiv: 2003.10555 [cs.CL].

Cremonesi, Paolo, Yehuda Koren, and Roberto Turrin, 2010: Performance of recommender algorithms on top-n recommendation tasks. *Proceedings of the Fourth ACM Conference on Recommender Systems.* RecSys '10. Barcelona, Spain: Association for Computing Machinery, 39–46. ISBN: 9781605589060. DOI: 10.1145/1864708.1864721. URL: https://doi.org/10.1145/1864708.1864721.

Datta, Hannes, George Knox, and Bart Bronnenberg, Apr. 2017: *Changing their tune: how consumers' adoption of online streaming affects music consumption and discovery.* DOI: 10.2139/ssrn.2782911.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, 2019: *Bert: pre-training of deep bidirectional transformers for language understanding.* arXiv: 1810.04805 [cs.CL].

Fang, Jun, 2021: *Session-based recommendation with self-attention networks.* arXiv: 2102.01922 [cs.IR].

Gemulla, Rainer, Erik Nijkamp, Peter J. Haas, and Yannis Sismanis, 2011: Large-scale matrix factorization with distributed stochastic gradient descent. *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* KDD '11. San Diego, California, USA: Association for Computing Machinery, 69–77. ISBN: 9781450308137. DOI: 10.1145/2020408.2020426. URL: https://doi.org/10.1145/2020408.2020426.

Grbovic, Mihajlo, Vladan Radosavljevic, Nemanja, Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp, 2015: *E-commerce in your inbox: product recommendations at scale.* arXiv: 1606.07154 [cs.CL].

Hidasi, Balázs, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk, May 2016: Session-based recommendations with recurrent neural networks.

Jarzabek, Przemysław, 2018: *Are new movies longer than they were 10, 20, 50 year ago?* URL: https://towardsdatascience.com/are-new-movies-longer-than-they-were-10hh20-50-year-ago-a35356b2ca5b#:~:text=The\%20most\%20popular\%20runtime\%20is,is\%2080\%E2\%80\%93120\%20minutes\%20long. (visited on 12/26/2018).

Jin, Rong, Joyce Y. Chai, and Luo Si, 2004: An automatic weighting scheme for collaborative filtering. *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* SIGIR '04. Sheffield, United Kingdom: Association for Computing Machinery, 337–344. ISBN: 1581138814. DOI: 10.1145/1008992.1009051. URL: https://doi.org/10.1145/1008992.1009051.

Kang, Wang-Cheng, and Julian McAuley, Nov. 2018: Self-attentive sequential recommendation. ICDM '18, 197–206. DOI: 10.1109/ICDM.2018.00035.

Karatzoglou, Alexandros, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver, 2010: Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. *Proceedings of the Fourth ACM Conference on Recommender Systems*. RecSys '10. Barcelona, Spain: Association for Computing Machinery, 79–86. ISBN: 9781605589060. DOI: 10.1145/1864708.1864727. URL: https://doi.org/10.1145/1864708.1864727.

Koren, Yehuda, 2008: Factorization meets the neighborhood: a multifaceted collaborative filtering model. *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '08. Las Vegas, Nevada, USA: Association for Computing Machinery, 426–434. ISBN: 9781605581934. DOI: 10.1145/1401890.1401944. URL: https://doi.org/10.1145/1401890.1401944.

Koren, Yehuda, 2009: Collaborative filtering with temporal dynamics. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '09. Paris, France: Association for Computing Machinery, 447–456. ISBN: 9781605584959. DOI: 10.1145/1557019.1557072. URL: https://doi.org/10.1145/1557019.1557072.

Le, Quoc V., and Tomas Mikolov, 2014: *Distributed representations of sentences and documents*. arXiv: 1405.4053 [cs.CL].

Ludewig, Malte, Noemi Mauro, Sara Latifi, and Dietmar Jannach, 2020: Empirical analysis of session-based recommendation algorithms. *User Modeling and User-Adapted Interaction*, **31**, 149–181. ISSN: 1573-1391. DOI: 10.1007/s11257-020-09277-1. URL: http://dx.doi.org/10.1007/s11257-020-09277-1.

Luo, Anjing, Pengpeng Zhao, Yanchi Liu, Fuzhen Zhuang, Deqing Wang, Jiajie Xu, Junhua Fang, and Victor S. Sheng, 2020: Collaborative self-attention network for session-based recommendation. English. *Proceedings of the 29th International Joint Conference on Artificial Intelligence, IJCAI 2020*. Edited by Christian Bessiere. IJCAI International Joint Conference on Artificial Intelligence. International Joint Conferences on Artificial Intelligence, 2591–2597.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean, 2013: *Distributed representations of words and phrases and their compositionality*. arXiv: 1310.4546 [cs.CL].

Moreira, Gabriel De Souza P., Dietmar Jannach, and Adilson Marques Da Cunha, 2019: Contextual hybrid session-based news recommendation with recurrent neural networks. *IEEE Access*, **7**, 169185–169203. ISSN: 2169-3536. DOI: 10.1109/access.2019.2954957. URL: http://dx.doi.org/10.1109/ACCESS.2019.2954957.

Oard, Douglas W, and Jinmook Kim, 1998: Implicit feedback for recommender systems.

Radford, Alec, and Karthik Narasimhan, 2018: Improving language understanding by generative pre-training.

Rendle, Steffen, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme, 2009: Bpr: bayesian personalized ranking from implicit feedback. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. UAI '09. Montreal, Quebec, Canada: AUAI Press, 452–461. ISBN: 9780974903958.

Sanchez, Daniel, 2019: *What's the average length of a song? it's shorter than you think.* URL: https://www.digitalmusicnews.com/2019/01/18/streaming-music-shorter-songs-study/ (visited on 01/18/2019).

Souza Pereira Moreira, Gabriel de, Felipe Ferreira, and Adilson Marques da Cunha, 2018: News session-based recommendations using deep neural networks. *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems*. DOI: 10.1145/3270323.3270328. URL: http://dx.doi.org/10.1145/3270323.3270328.

Souza Pereira Moreira, Gabriel de, Sara Rabhi, Jeong Min Lee, Ronay Ak, and Even Oldridge, 2021: Transformers4rec: bridging the gap between nlp and sequential / session-based recommendation. *Fifteenth ACM Conference on Recommender Systems*. RecSys '21. Amsterdam, Netherlands: Association for Computing Machinery, 143–153. ISBN: 9781450384582. DOI: 10.1145/3460231.3474255. URL: https://doi.org/10.1145/3460231.3474255.

Sun, Fei, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang, 2019a: Bert4rec: sequential recommendation with bidirectional encoder representations from transformer. *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. CIKM '19. Beijing, China: Association for Computing Machinery, 1441–1450. ISBN: 9781450369763. DOI: 10.1145/3357384.3357895. URL: https://doi.org/10.1145/3357384.3357895.

Sun, Shiming, Yuanhe Tang, Zemei Dai, and Fu Zhou, 2019b: Self-attention network for session-based recommendation with streaming data input. *IEEE Access*, **7**, 110499–110509. DOI: `10.1109/ACCESS.2019.2931945`.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, 2017: *Attention is all you need.* arXiv: `1706.03762 [cs.CL]`.

Wang, Shoujin, Longbing Cao, Yan Wang, Quan Z. Sheng, Mehmet Orgun, and Defu Lian, 2021: *A survey on session-based recommender systems.* arXiv: `1902.04864 [cs.IR]`.

Wang, Shoujin, Liang Hu, Longbing Cao, Xiaoshui Huang, Defu Lian, and Wei Liu, 2018: Attention-based transactional context embedding for next-item recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence*, **32**. URL: `https://ojs.aaai.org/index.php/AAAI/article/view/11851`.

Wolf, Thomas et al., 2020: *Huggingface's transformers: state-of-the-art natural language processing.* arXiv: `1910.03771 [cs.CL]`.

Yang, Zhilin, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le, 2020: *Xlnet: generalized autoregressive pretraining for language understanding.* arXiv: `1906.08237 [cs.CL]`.

Zhang, Yang, Fuli Feng, Chenxu Wang, Xiangnan He, Meng Wang, Yan Li, and Yongdong Zhang, 2020: How to retrain recommender system? *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* DOI: `10.1145/3397271.3401167`. URL: `http://dx.doi.org/10.1145/3397271.3401167`.

Zhou, Chang, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao, 2017: *Atrank: an attention-based user behavior modeling framework for recommendation.* arXiv: `1711.06632 [cs.AI]`.

초 록

최근 트랜스포머 기반 추천시스템들이 다양한 분야에서 높은 성능을 보여왔다. 하지만 음악 스트리밍 분야에는 적용되지 않았었고, 이 논문을 통해 음악 스트리밍 분야에 트랜스포머 기반 세션 추천시스템이 어떤 성능을 보여주는지 탐색해 보았다. 데이터 전처리를 통해 유저들이 음악을 실제로 좋아해서 들었을 법한 세션들만 남기려 노력했고, 세션 기반 추천시스템에 맞게 데이터를 정제했다. 음악과 관련된 다양한 정보들도 모델 훈련에 반영하기 위해 카테고리 형태로 바꿔주었고, 훈련 자체는 세션 기반 추천시스템에서 자주 쓰이는 점진적 훈련법을 활용했다. 최종 실험 결과에서는 데이터의 비정제성과 비밀집성을 극복하고 비슷한 데이터셋과 경쟁력을 갖추는 성과를 보여주었다. 이 연구를 통해 음악 스트리밍 세션 추천시스템에 트랜스포머 기반 모델이라는 새로운 가능성을 보여주었고, 추후 연구자들이 참고할 수 있는 시작점을 제공하였다.

주요어: 서울대, 석사, 자연어처리 연구실
트랜스포머, 추천시스템, 음악 스트리밍 세션
학 번: 2020-25184