Filter-enhanced MLP is All You Need for Sequential Recommendation

Kun Zhou[1,4†], Hui Yu[2,5†], Wayne Xin Zhao[3,4*] and Ji-Rong Wen[3,4]
[1]School of Information, Renmin University of China, Beijing, China
[2]University of Chinese Academy of Sciences, Beijing, China
[3]Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
[4]Beijing Key Laboratory of Big Data Management and Analysis Methods, Beijing, China
[5]Key Laboratory of Petroleum Resources Research, Institute of Geology and Geophysics, Chinese Academy of Sciences, Beijing, China
francis_kun_zhou@163.com,ishyu@outlook.com,batmanfly@gmail.com,jrwen@ruc.edu.cn

ABSTRACT

Recently, deep neural networks such as RNN, CNN and Transformer have been applied in the task of sequential recommendation, which aims to capture the dynamic preference characteristics from logged user behavior data for accurate recommendation. However, in online platforms, logged user behavior data is inevitable to contain noise, and deep recommendation models are easy to overfit on these logged data. To tackle this problem, we borrow the idea of filtering algorithms from signal processing that attenuates the noise in the frequency domain. In our empirical experiments, we find that filtering algorithms can substantially improve representative sequential recommendation models, and integrating simple filtering algorithms (*e.g.,* Band-Stop Filter) with an all-MLP architecture can even outperform competitive Transformer-based models. Motivated by it, we propose **FMLP-Rec**, an all-MLP model with learnable filters for sequential recommendation task. The all-MLP architecture endows our model with lower time complexity, and the learnable filters can adaptively attenuate the noise information in the frequency domain. Extensive experiments conducted on eight real-world datasets demonstrate the superiority of our proposed method over competitive RNN, CNN, GNN and Transformer-based methods. Our code and data are publicly available at the link: https://github.com/RUCAIBox/FMLP-Rec.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Sequential Recommendation, All-MLP Model, Filtering Algorithm

†Equal contribution.
*Corresponding author.

WWW '22, April 25–29, 2022, Virtual Event, Lyon, France.

ACM ISBN 978-1-4503-9096-5/22/04…$15.00
https://doi.org/10.1145/3485447.3512111

ACM Reference Format:
Kun Zhou[1,4†], Hui Yu[2,5†], Wayne Xin Zhao[3,4*] and Ji-Rong Wen[3,4]. 2022. Filter-enhanced MLP is All You Need for Sequential Recommendation. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3485447.3512111

1 INTRODUCTION

Recommender systems [32, 67, 69, 71] have been widely deployed in online platforms (*e.g.,* Amazon and Taobao) for predicting the potential interests of users over a large item pool. In real-world applications, users' behaviors are dynamic and evolving over time. Thus, it is critical to capture the sequential characteristics of user behaviors for making appropriate recommendations, which is the core goal for sequential recommendation [17, 22, 42, 53].

To characterize the evolving patterns of users' historical behaviors, a number of sequential recommendation models have been developed in the literature based on deep neural networks [67]. Typical solutions are based on RNN [17, 31] and CNN [53, 62]. Furthermore, a surge of works adopt more advanced neural network architectures (*e.g.,* memory network [9, 20] and self-attention mechanism [26, 27]) to enhance the modeling capacity for effectively capturing dynamic user preference. More recently, Transformer-based approaches [22, 50, 70] have shown remarkable performance in this task by stacking multi-head self-attention layers.

However, stacked self-attention layers involve a large number of parameters, which might lead to the over-parameterized architecture of Transformer-based methods [13, 35]. Besides, these methods mainly fit the model parameters based on logged user behavior data, which are in essence noisy [1, 43] or even contains malicious fakes [8, 65]. It has been found that deep neural networks tend to overfit on noisy data [7, 24]. The case becomes more severe for self-attention based recommenders when the logged sequence data contains noise, since it attends to all items for sequence modeling.

Considering the above issues, we aim to simplify the Transformer-based sequential recommender as well as increase its robustness to resist the noise in logged data. Our key idea is borrowed from the digital signal processing field, where filtering algorithms are used to reduce the influence of noise [38] for sequence data. We suspect that when the sequence data was denoised, it would become easier to capture sequential user behaviors. If this was true, we might be able to simplify the heavy self-attention components from Transformer-based approaches. To examine this hypothesis, we conduct several

empirical experiments in Section 3 by simply denoising the item embeddings with three classical filtering algorithms (more details are referred to Table 1). We find that filtering algorithms can substantially improve these deep sequential recommendation models, including the Transformer-based SASRec [22].

Motivated by the empirical findings, we propose a novel **F**ilter-enhanced **MLP** approach for sequential **Rec**ommendation, named **FMLP-Rec**. By removing the self-attention components from Transformers, FMLP-Rec is solely based on MLP structures for stacking blocks. As the major technical contribution, we incorporate a filter component in each stacked block, where we perform Fast Fourier Transform (FFT) [45] to convert the input representations into the frequency domain and an inverse FFT procedure recovers the denoised representations. The filter component plays a key role in reducing the influence of the noise from item representations[1]. To implement it, we incorporate learnable filters to encode the input item sequences in the frequency domain, which can be optimized from the raw data without human priors. Our approach can effectively attenuate noise information and extract meaningful features from all the frequencies (*e.g.,* long/short-term item interactions). Theoretically speaking, according to convolution theorem [45], it can be proved that learnable filters are equivalent to the circular convolution in the time domain, which has a larger receptive field on the whole sequence, and can better capture periodic characteristics of user behaviors. Another merit of the filter component is that it requires less time cost without considering pairwise item correlations, which results in a lighter and faster network architecture.

To the best of our knowledge, it is the first time that a filter-enhanced all-MLP architecture has been applied to the sequential recommendation task, which is simple, effective and efficient. To validate the effectiveness of our model, we conduct extensive experiments on eight real-world datasets from different scenarios for sequential recommendations. Experimental results show that FMLP-Rec outperforms state-of-the-art RNN, CNN, GNN and Transformer-based baseline models.

2 PRELIMINARIES

2.1 Problem Statement

Assume that we have a set of users and items, denoted by $\mathcal{U}$ and $\mathcal{I}$, respectively, where $u \in \mathcal{U}$ denotes a user and $i \in \mathcal{I}$ denotes an item. The numbers of users and items are denoted as $|\mathcal{U}|$ and $|\mathcal{I}|$, respectively. For sequential recommendation with implicit feedback, a user u has a context c, a chronologically-ordered interaction sequence with items: $c = \{i_1, \cdots, i_n\}$, where n is the number of interactions and i_t is the t-th item that the user u has interacted with. For convenience, we use $i_{j:k}$ to denote the subsequence, *i.e.,* $i_{j:k} = \{i_j, \cdots, i_k\}$ where $1 \le j < k \le n$.

Based on the above notations, we now define the task of sequential recommendation. Formally, given the contextual item sequence of a user $c = \{i_1, \cdots, i_n\}$, the task of sequential recommendation is to predict the next item that the user is likely to interact with at the $(n+1)$-th step, denoted as $p(i_{n+1}|i_{1:n})$.

[1]Note that we do not learn to remove the noisy items, since it is a more difficult task in practice, usually without any ground-truth labels. Instead, we consider directly improving the item representations in order to reduce the potential influence of noise.

2.2 Fourier Transform

Discrete Fourier transform. Discrete Fourier transform (DFT) is essential in the digital signal processing area [38, 45] and is a crucial component in our approach. In this paper, we only consider the 1D DFT. Given a sequence of numbers $\{x_n\}$ with $n \in [0, N-1]$, the 1D DFT converts the sequence into the frequency domain by:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} nk}, \quad 0 \le k \le N-1. \tag{1}$$

where i is the imaginary unit. For each k, the DFT generates a new representation X_k as a sum of all the original input tokens x_n with so-called "twiddle factors". In this way, X_k represents the spectrum of the sequence $\{x_n\}$ at the frequency $\omega_k = 2\pi k/N$. Note that DFT is an one-to-one transformation. Given the DFT X_k, we can recover the original sequence $\{x_n\}$ by the inverse DFT (IDFT):

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{\frac{2\pi i}{N} nk}. \tag{2}$$

Fast Fourier Transform. To compute the DFT, the Fast Fourier Transform (FFT) is widely used in previous works [16, 56]. The standard FFT algorithm is the Cooley–Tukey algorithm [10, 14], which recursively re-expresses the DFT of a sequence of length N and reduces the time complexity to $O(N \log N)$. The inverse DFT in Eq. 2, which has a similar form to the DFT, can also be computed efficiently using the inverse fast Fourier transform (IFFT).

Since FFT can convert input signals into the frequency domain where the periodic characteristics are easier to capture, it is widely used in digital signal processing area to filter noise signals [2, 45, 48]. A commonly-used way is the Low-Pass Filter (LPF) that attenuates high-frequency noise signals after processed by FFT. In this paper, we consider using FFT and filtering algorithms to reduce the influence of noisy features within the user interacted item sequence.

3 EMPIRICAL ANALYSIS WITH FILTERING ALGORITHMS FOR RECOMMENDATION

In this section, we conduct an empirical study to test: (1) the effectiveness of filtering algorithms in sequential recommendation models, and (2) the effectiveness of integrating filtering algorithms with all-MLP architectures.

3.1 Analysis Setup

For empirical study, we select the Amazon [34] *Beauty* and *Sports* datasets for the evaluation of sequential recommendation methods.

Sequential recommendation algorithms. We conduct experiments on GRU4Rec [17] and SASRec [22], two representative sequential recommendation models. The two models largely follow the standard framework of deep sequential models [17, 41, 53], consisting of an embedding layer, a sequence encoder layer and a prediction layer, but adopt RNN and Transformer in the sequence encoder layer, respectively. Despite that the two models have shown promising results, they may not be robust to noise in the user behavior sequence [51, 63]. Thus, we directly add a non-parameter filter layer between the embedding layer and the sequence encoder layer of the two models, and do not change other components.

Table 1: Performance comparison of SASRec and GRU4Rec with different filtering algorithms.

Methods	Filter	Beauty		Sports	
		HR@10	NDCG@10	HR@10	NDCG@10
GRU4Rec		0.4106	0.2584	0.4299	0.2527
	+HPF	0.3828	0.2228	0.3654	0.2063
	+LPF	0.4351	**0.2689**	**0.4481**	**0.2578**
	+BSF	**0.4372**	0.2658	0.4432	0.2563
SASRec		0.4696	0.3156	0.4622	0.2869
	+HPF	0.4544	0.3037	0.4530	0.2785
	+LPF	0.4941	0.3320	0.5040	0.3138
	+BSF	**0.5011**	**0.3334**	**0.5115**	**0.3172**

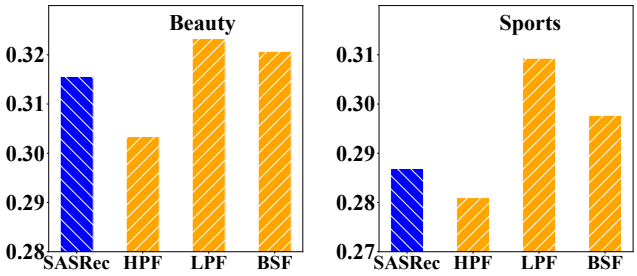

Figure 1: Performance (NDCG@10) comparison of all-MLP variants of SASRec with different filtering algorithms.

Filtering algorithms. In the filter layer, given the embedding matrix of the item sequence, we conduct the following operations for each dimension of features: *FFT* → *Filtering Algorithm* → *IFFT*. After filtering, we take the denoised embedding matrix as the input of the sequence encoder layer. For filtering algorithms, we select three classical methods [45] as follows:

• *High-Pass Filter (HPF)* passes signals with a higher frequency and attenuates ones with a lower frequency. After FFT, we set the values of the lower-frequency half of signals into zero.

• *Low-Pass Filter (LPF)* passes signals with a lower frequency and attenuates ones with a higher frequency. After FFT, we set the values of the higher-frequency half of signals into zero.

• *Band-Stop Filter (BSF)* attenuates signals with a medium frequency, and passes others. After FFT, we set the values of the medium-frequency half of signals into zero.

3.2 Results and Findings

In this part, we present the results and discuss the findings.

The effect on representative models. We report the results in Table 1. As can be seen, adding Low-Pass Filter (LPF) and Band-Stop Filter (BSF) lead to consistent improvements, especially on SASRec. For GRU4Rec, LPF achieves the best performance than other filtering algorithms, while for SASRec, BSF achieves the best. In contrast, High-Pass Filter (HPF) causes performance degradation in most cases. From these observations, we can conclude that:

• The item embedding matrix are likely to contains noise that affects the performance of sequential recommendation models.

• A proper filtering algorithm on the embedding layer is useful to alleviate the above problem. But for different models, the most suitable filtering algorithm may be also different.

• The low-frequency information within the embedding matrix seems more important for sequential recommendation. This phenomenon is similar to findings in hydrology [44], seismology [49] and praxiology [55] that low-frequency signals in nature and human behavior are usually meaningful periodic characteristics.

The effect on all-MLP models. The above study confirms that filtering algorithms can improve the performance of RNN and Transformer based sequential recommendation models. In this part, we continue to examine the performance of a simply variant that integrates these filtering algorithms with all-MLP architectures. Based on the architecture of SASRec [22], we remove the multi-head self-attention blocks within the Transformer-based sequence encoder layer, but add a filter layer after the embedding layer. We also select HPF, LPF and BSF algorithms as in section 3.1, and other components are not changed. In this way, the variant models only rely on MLPs to model the item sequence.

We report the performance of the all-MLP variant models with SASRec in Figure 1. As we can see, after removing multi-head self-attention blocks, most of the model variants still perform well. And the variant model with LPF even outperforms SASRec model with a large margin. It indicates that proper filtering algorithms can inspire the potential of simple all-MLP models to surpass complex Transformer-based models. By removing both the noise information and the self-attention blocks, the model more lightweight, which reduces the risk of overfitting. Based on the above analysis, it is promising to design an effective and efficient all-MLP model with a proper filtering algorithm for sequential recommendation.

4 METHOD

The empirical findings in Section 3 have demonstrated that an all-MLP architecture with proper filtering algorithms (*e.g.,* LPF) can yield very good recommendation performance. However, previous filtering algorithms [45] usually require domain knowledge or expert efforts in designing proper filters and setting hyperparameters (*e.g.,* filtering thresholds). It is still a challenge to effectively integrate the filtering algorithms into existing deep recommendation frameworks. In this section, we present an all-MLP architecture (named as **FMLP-Rec**) for sequential recommendation by stacking MLP blocks with learnable filters, which can automatically learn proper filters for various sequential recommendation scenarios.

4.1 FMLP-Rec: An All-MLP Sequential Recommender with Learnable Filters

Similar to the original Transformer architecture, our FMLP-Rec also stacks multiple neural blocks to produce the representation of sequential user preference for recommendation. The key difference of our approach is to replace the multi-head self-attention structure in Transformer with a novel filter structure. Besides the effect of noise attenuation by filters, such a structure is mathematically equivalent to the circular convolution (proved in Section 4.2), which can also capture sequence-level preference characteristics [45]. Next, we present the details of our approach.

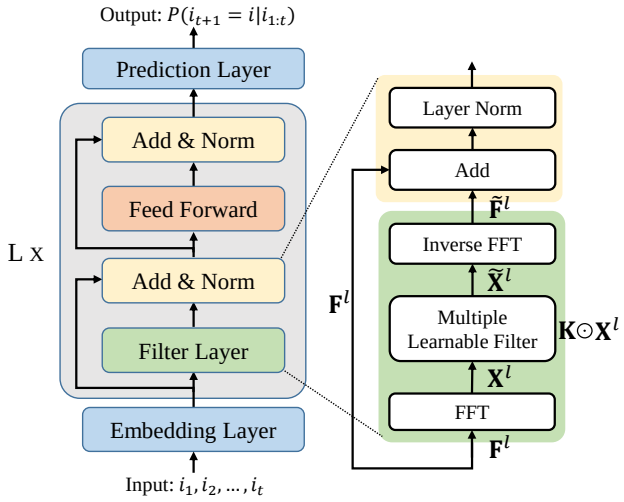

Figure 2: The overview of our FMLP-Rec, an all-MLP model that stacks multiple learnable filter-enhanced blocks.

4.1.1 Embedding Layer. In the embedding layer, we maintain an item embedding matrix $\mathbf{M}_I \in \mathbb{R}^{|\mathcal{I}| \times d}$ to project the high-dimensional one-hot representation of an item to low-dimensional dense representation. Given a n-length item sequence, we apply a look-up operation from $\mathbf{M}_I$ to form the input embedding matrix $\mathbf{E} \in \mathbb{R}^{n \times d}$. Besides, we incorporate a learnable position encoding matrix $\mathbf{P} \in \mathbb{R}^{n \times d}$ to enhance the input representation of the item sequence. By this means, the sequence representation $\mathbf{E}_I \in \mathbb{R}^{n \times d}$ can be obtained by summing the two embedding matrices. Since the item and position embedding matrices are randomly initialized, it may affect the filtering mechanism and cause the training process unstable. Inspired by recent works [12, 22], we perform dropout and the layer normalization operations to alleviate these problems. Thus, we generate the sequence representation $\mathbf{E}_I \in \mathbb{R}^{n \times d}$ by:

$$\mathbf{E}_I = Dropout(LayerNorm(\mathbf{E} + \mathbf{P})). \tag{3}$$

4.1.2 Learnable Filter-enhanced Blocks. Based on the embedding layer, we develop the item encoder by stacking multiple learnable filter blocks. A learnable filter block generally consists of two sublayers, *i.e.,* a filter layer and a point-wise feed-forward network.

Filter Layer. In the filter layer, we perform filtering operation for each dimension of features in the frequency domain, and then perform skip connection and layer normalization. Given the input item representation matrix $\mathbf{F}^l \in \mathbb{R}^{n \times d}$ of the l-th layer (when $l = 0$, we set $\mathbf{F}^0 = \mathbf{E}_I$), we first perform FFT along the item dimension to convert $\mathbf{F}^l$ to the frequency domain:

$$\mathbf{X}^l = \mathcal{F}(\mathbf{F}^l) \in \mathbb{C}^{n \times d} \tag{4}$$

where $\mathcal{F}(\cdot)$ denotes the one-dimensional FFT. Note that $\mathbf{X}^l$ is a complex tensor and represents the spectrum of $\mathbf{F}^l$. We can then modulate the spectrum by multiplying a learnable filter $\mathbf{W} \in \mathbb{C}^{n \times d}$:

$$\widetilde{\mathbf{X}}^l = \mathbf{W} \odot \mathbf{X}^l, \tag{5}$$

where $\odot$ is the element-wise multiplication. The filter $\mathbf{K}$ is called the *learnable filter* since it can be optimized by SGD to adaptively represent an arbitrary filter in the frequency domain. Finally, we adopt the inverse FFT to transform the modulated spectrum $\widetilde{\mathbf{X}}^l$ back to the time domain and update the sequence representations:

$$\widetilde{\mathbf{F}}^l \leftarrow \mathcal{F}^{-1}(\widetilde{\mathbf{X}}^l) \in \mathbb{R}^{n \times d}. \tag{6}$$

where $\mathcal{F}^{-1}(\cdot)$ denotes the inverse 1D FFT, which converts the complex tensor into a real number tensor. With the operations of FFT and inverse FFT, the noise from the logged data can be effectively reduced, and we can therefore obtain purer item embeddings. Following SASRec [22], we also incorporate the skip connection [15], layer normalization [3] and dropout [47] operations to alleviate the gradient vanishing and unstable training problems as:

$$\widetilde{\mathbf{F}}^l = \text{LayerNorm}(\mathbf{F}^l + \text{Dropout}(\widetilde{\mathbf{F}}^l)) \tag{7}$$

Feed-forward layers. In the point-wise feed-forward network, we incorporate MLP and ReLU activation functions to further capture the non-linearity characteristics. The computation is defined as:

$$\text{FFN}(\widetilde{\mathbf{F}}^l) = \big(\text{ReLU}(\widetilde{\mathbf{F}}^l \mathbf{W}_1 + \mathbf{b}_1)\big)\mathbf{W}_2 + \mathbf{b}_2, \tag{8}$$

where $\mathbf{W}_1$, $\mathbf{b}_1$, $\mathbf{W}_2$, $\mathbf{b}_2$ are trainable parameters. Then, we also perform skip connection and layer normalization operations as in Eq. 7 to generate the output of the l-layer.

4.1.3 Prediction Layer. In the final layer of FMLP-Rec, we calculate the user's preference score for the item i in step $(t+1)$ under the context from user history as:

$$P(i_{t+1} = i|i_{1:t}) = \mathbf{e}_i^\top \mathbf{F}_t^L, \tag{9}$$

where $\mathbf{e}_i$ is the representation of item i from item embedding matrix $\mathbf{M}_I$, $\mathbf{F}_t^L$ is the output of the L-layer learnable filter blocks at step t, and L is the number of learnable filter blocks. We adopt the pairwise rank loss to optimize the model parameters as:

$$L = -\sum_{u \in \mathcal{U}} \sum_{t=1}^{n} \log \sigma\Big(P(i_{t+1}|i_{1:t}) - P(i_{t+1}^-|i_{1:t})\Big), \tag{10}$$

where we pair each ground-truth item i_{t+1} with a negative item i_{t+1}^- that is randomly sampled.

4.2 Theoretical Analysis with Filter Layers

Besides noise attenuation, we now show that the proposed filter blocks can also capture sequential characteristics from logged data. We first theoretically prove that our proposed learnable filter is equivalent to circular convolution.

In the filter layer of FMLP-Rec, the input information is firstly converted to spectrum representations in the frequency domain via FFT, and then further multiplied the learnable filter $\mathbf{W}$. In this way, the formulation of the learnable filter $\mathbf{W}$ can be regarded as a set of learnable frequency filters $\{\mathbf{w}^{(1)}, \cdots, \mathbf{w}^{(d)}\}$ for different hidden dimensions in the item embedding matrix, where d denotes the hidden size. According to the convolution theorem [38, 45], the *multiplication* in the frequency domain is equivalent to the *circular convolution* in the time domain. It is a special case of periodic convolution between two periodic functions that have the same period. For the t-th dimension features $\{f_n^{(t)}\}_{n=0}^{N-1}$ from the item

representation matrix $\mathbf{F}^l$ and a filter $\{h_n^{(t)}\}_{n=0}^{N-1} = \mathcal{F}^{-1}(w^{(t)})$, their circular convolution is defined as $\{y_n^{(t)}\}_{n=0}^{N-1}$, where

$$y_n^{(t)} = \sum_{m=0}^{N-1} h_m^{(t)} \cdot f_{(n-m) \bmod N}^{(t)} \tag{11}$$

where mod denotes the integer modulo operation and n is the sequence length. Consider that the DFT of the sequence $\{y_n^{(t)}\}_{n=0}^{N-1}$ is $\{\tilde{x}_n^{(t)}\}_{n=0}^{N-1}$, we have the following derivations:

$$\begin{aligned}
\tilde{x}_k^{(t)} &= \sum_{n=0}^{N-1}\sum_{m=0}^{N-1} h_m^{(t)} f_{(n-m)\%N}^{(t)} e^{-\frac{2\pi i}{N}kn} \\
&= \sum_{m=0}^{N-1} h_m^{(t)} e^{-\frac{2\pi i}{N}km} \sum_{n=0}^{N-1} f_{(n-m)\%N}^{(t)} e^{-\frac{2\pi i}{N}k(n-m)} \\
&= w_k^{(t)} \left(\sum_{n=m}^{N-1} f_{n-m}^{(t)} e^{-\frac{2\pi i}{N}k(n-m)} + \sum_{n=0}^{m-1} f_{n-m+N}^{(t)} e^{-\frac{2\pi i}{N}k(n-m)} \right) \\
&= w_k^{(t)} \left(\sum_{n=0}^{N-m-1} f_n^{(t)} e^{-\frac{2\pi i}{N}kn} + \sum_{n=N-m}^{N-1} f_n^{(t)} e^{-\frac{2\pi i}{N}kn} \right) \\
&= w_k^{(t)} \sum_{n=0}^{N-1} f_n^{(t)} e^{-\frac{2\pi i}{N}kn} = w_k^{(t)} x_k^{(t)},
\end{aligned}$$

where in the right hand, $x_k^{(t)}$ is exactly the k-th number of the t-th dimension feature from $\mathbf{X}$ in the frequency domain, and $w_k^{(t)}$ is the k-th weight from the t-th filter $w^{(t)}$. In conclusion, we prove that

$$f^{(t)} * h^{(t)} = \mathcal{F}^{-1}(\mathbf{w}^{(t)} \odot \mathbf{x}^{(t)}) \tag{12}$$

where "$*$" and "$\odot$" denotes circular convolution and element-wise multiplication, respectively. Therefore, the multiplication of the t-th dimension feature from item representations $\mathbf{X}^l$ and the t-th filter $\mathbf{w}^{(t)}$ from $\mathbf{W}$ in Eq. 5, is equivalent to the circular convolution operation on the t-th feature of $\mathbf{F}^l$ using convolution kernel $\{h_n^{(t)}\}_{n=0}^{N-1}$. Similar to RNN, CNN and Transformer, the proposed filter block is also able to capture sequential characteristics, since it has in essence the same effect of circular convolution. Besides, compared with traditional linear convolution in previous works [53, 62], the circular convolution has a larger receptive field on the entire sequence and is able to better capture *periodic patterns* [45]. Such merit is particularly appealing for the recommendation task, where users' behaviors tend to show certain periodic trends [11, 19, 21].

4.3 Discussion

4.3.1 Comparison with Transformer-based Sequential Recommenders. Transformer-based models such as SASRec [22] and BERT4Rec [50] typically stack multi-head self-attention blocks for learning the sequential representations. It relies on a heavy self-attention structure to learn item-to-item correlations. As a comparison, our approach FMLP-Rec directly removes all the self-attention structures, and the entire approach is solely based on MLP-based structures by integrating additional filter layers. Such a design can largely reduce both the space and time complexity for modeling sequence data. More importantly, we have shown that the learnable filter layer is equivalent to the circular convolution operation using convolution kernels with the same size as the feature map [38, 45]. As a result, it can own the same receptive field as the self-attention mechanism but largely reduces the number of involved parameters. Besides, the circular convolution is able to capture periodic characteristics, which is also an important feature for sequential recommendation.

Table 2: Time complexity and receptive field of the proposed FMLP-Rec with Caser and SASRec, where n and k denote the sequence length and convolution kernel size, respectively. For simplicity, we omit other same terms (*e.g.,* hidden size) and highlight the difference at the sequence level.

	Time Complexity per Layer	Receptive Field
Caser	$O(kn)$	k
SASRec	$O(n^2)$	n
FMLP-Rec	$O(n \log n)$	n

Table 3: Statistics of the datasets after preprocessing.

Dataset	# Sequences	# Items	# Actions	# Sparsity
Beauty	22,363	12,101	198,502	99.93%
Sports	25,598	18,357	296,337	99.95%
Toys	19,412	11,924	167,597	99.93%
Yelp	30,431	20,033	316,354	99.95%
Nowplaying	145,612	59,593	1,085,410	99.99%
Retailrocket	321,032	51,428	871,637	99.99%
Tmall	66,909	37,367	427,797	99.98%
Yoochoose	470,477	19,690	1,434,349	99.98%

4.3.2 Time Complexity and Receptive Field Analysis. In this part, we compare our model with representative sequential recommendation models in terms of time complexity and receptive field. We select CNN-based Caser [53] and Transformer-based SASRec [22] as comparisons, since they represent two lines of different neural architectures. Since these models involve different algorithmic details (*e.g.,* dropout and normalization), it may not be fair to directly compare them. Instead, we consider a more general comparison by only on the sequence length n. The comparison results of the model complexity and receptive field are shown in Table 2.

First, Caser performs convolution operations in time domains, so its time complexity and receptive field are $O(k \cdot n)$ and k, respectively. However, since Caser relies on convolution kernels to capture sequential pattern characteristics, it usually requires a larger kernel size to extend the receptive field for better performance. Besides, in Transformer-based models, the self-attention layers require calculating the similarity for each pair of items. Its time complexity and receptive field are $O(n^2)$ and n [57, 64], respectively. In contrast, our FMLP-Rec consists of FFT and IFFT operations with the time cost of $O(n \log n)$ [16, 56], and point-wise feed-forward networks with the time cost of $O(n)$, which means the total time complexity is $O(n \log n)$. Besides, since our filter layer is equivalent to the circular convolution on the whole sequence, its receptive field is the same as Transformer. Therefore, our FMLP-Rec can achieve larger receptive field and meanwhile lower complexity.

5 EXPERIMENT

5.1 Experimental Setup

5.1.1 Dataset. We conduct experiments on eight datasets with varying domains. Their statistics are summarized in Table 3.

(1) **Beauty, Sports, and Toys**: these three datasets are obtained from Amazon review datasets in [34]. We select three subcategories: Beauty, Sports and Outdoors, and Toys and Games.

(2) **Yelp**[2] is a dataset for business recommendation. As it is very large, we only use the transaction records after *January 1st, 2019*.

(3) **Nowplaying**[3] contains music listening events collected from Twitter, where users posted tracks that were currently listening.

(4) **RetailRocket**[4] is collected from a personalized e-commerce website. It contains six months of user browsing activities.

(5) **Tmall**[5] comes from IJCAI-15 competition, which contains user's shopping logs on Tmall online shopping platform.

(6) **Yoochoose**[6] contains a collection of sessions from a retailer, where each session encapsulates the click events.

Note that the item sequences in Beauty, Sports, Toys and Yelp are user transaction records, while in Nowplaying, RetailRocket, Tmall and Yoochoose are click sessions. For all datasets, we group the interaction records by users or sessions, and sort them by the timestamps ascendingly. Following [18, 42], we filter unpopular items and inactive users with fewer than five interaction records.

5.1.2 Evaluation Metrics. Following [68], we employ top-k Hit Ratio (HR@k), top-k Normalized Discounted Cumulative Gain (NDCG@k), and Mean Reciprocal Rank (MRR) for evaluation, which are widely used in related works [22, 42, 70]. Since HR@1 is equal to NDCG@1, we report results on HR@{1, 5, 10}, NGCG@{5, 10}, and MRR. Following the common strategy [20, 22], we pair the ground-truth item with 99 randomly sampled negative items that the user has not interacted with. We calculate all metrics according to the ranking of the items and report the average score. Note that we also rank the ground-truth item with all candidate items and report the full-ranking results in supplementary materials.

5.1.3 Baseline Models. We compare our proposed approach with the following baseline methods: (1) **PopRec** ranks items according to the popularity measured by the number of interactions; (2) **FM** [41] characterizes the pairwise interactions between variables using factorized model; (3) **AutoInt** [46] utilizes the multi-head self-attentive neural network to learn the feature interactions; (4) **GRU4Rec** [17] applies GRU to model item sequences; (5) **Caser** [53] is a CNN-based method that applies horizontal and vertical convolutions for sequential recommendation; (6) **HGN** [33] adopts hierarchical gating networks to capture long-term and short-term user interests; (7) **RepeatNet** [40] adds a copy mechanism on RNN architecture that can choose items from a user's history; (8) **CLEA** [37] is recently proposed and performs item-level denoising via a contrastive learning model; (9) **SASRec** [22] is a unidirectional Transformer-based sequential recommendation model; (10) **BERT4Rec** [50] uses a Cloze objective loss for sequential recommendation by the bidirectional Transformer; (11) **SRGNN** [59] models session sequences as graph-structured data and uses an attention network; (12) **GCSAN** [60] utilizes both graph neural network and self-attention mechanism for session-based recommendation.

5.2 Experimental Results

The results of different methods on datasets containing user transaction record are shown in Table 4, and results on session-based datasets are shown in Table 5. Based on the results, we can find:

First, non-sequential recommendation methods (*i.e.,* PopRec, FM and AutoInt) perform worse than sequential recommendation methods. It indicates that the sequential pattern is important in this task. As for sequential recommendation methods, SASRec and BERT4Rec utilize Transformer-based architectures, and mostly achieve better performance than RNN-based models (*i.e.,* GRU4Rec and RepeatNet), CNN-based model (*i.e.,* Caser) and gate-based model (*i.e.,* HGN). A possible reason is that Transformer-based models have more parameters corresponding to stronger capacity to capture sequential characteristics. Besides, CLEA achieves comparable performance with SASRec and BERT4Rec in part of datasets. Since CLEA adopts the item-level denoising strategy, it indicates that alleviating the influence of noise is useful to improve the recommendation performance. We can also see that GNN-based models (*i.e.,* SRGNN and GCSAN) also achieve comparable performance as Transformer-based models. It suggests that GNNs are also promising to capture useful characteristics for accuracy recommendation.

Finally, by comparing our approach with all the baselines, it is clear to see that FMLP-Rec performs consistently better than them by a large margin on most of datasets. Different from these baselines, we adopt an all-MLP architecture with learnable filters to encode the item sequence. The learnable filters can alleviate the influence of noise information, and are equivalent to circular convolutions that can capture the periodic characteristics in the item sequences with a larger receptive field. The all-MLP architecture largely reduces the model scale, leading to a lower time complexity. As a result, our FMLP-Rec is effective and efficient. This result also shows that all-MLP architectures are effective for sequential recommendation.

6 FURTHER ANALYSIS

6.1 Ablation tudy

Our proposed FMLP-Rec contains filter layers, feed-forward network and Add & Norm operations. To verify the effectiveness of each component, we conduct the ablation study on Beauty and Sports datasets to analyze the contribution of each part. Besides, to validate if the learnable filters are more useful than classical filtering algorithms, we also conduct variation study by replacing the learnable filters with high-pass, low-pass and band-stop filters.

From the results in Table 6, we can observe that removing any components would lead to the performance degradation, especially the filter layer. It indicates all the components in FMLP-Rec are useful to improve the recommendation performance. Besides, we can see that our FMLP-Rec outperforms all the other variants with classical filtering algorithms. The reason is that our learnable filters can adaptively learn to filter the noise information via SGD, which is promising to better adapt to the data distribution.

[2]https://www.yelp.com/dataset
[3]https://dbis.uibk.ac.at/node/263#nowplaying
[4]https://www.kaggle.com/retailrocket/ecommerce-dataset
[5]https://tianchi.aliyun.com/dataset/dataDetail?dataId=42
[6]https://www.kaggle.com/chadgostopp/recsys-challenge-2015

Table 4: Performance comparison of different methods on four datasets containing user transaction records. The best performance and the second best performance methods are denoted in bold and underlined fonts respectively.

Datasets	Metric	PopRec	FM	AutoInt	GRU4Rec	Caser	HGN	RepeatNet	CLEA	SASRec	BERT4Rec	SRGNN	GCSAN	FMLP-Rec
Beauty	HR@1	0.0678	0.0405	0.0447	0.1337	0.1337	0.1683	0.1578	0.1325	0.1870	0.1531	0.1729	0.1973	**0.2011**
	HR@5	0.2105	0.1461	0.1705	0.3125	0.3032	0.3544	0.3268	0.3305	0.3741	0.3640	0.3518	0.3678	**0.4025**
	NDCG@5	0.1391	0.0934	0.1063	0.2268	0.2219	0.2656	0.2455	0.2353	0.2848	0.2622	0.2660	0.2864	**0.3070**
	HR@10	0.3386	0.2311	0.2872	0.4106	0.3942	0.4503	0.4205	0.4426	0.4696	0.4739	0.4484	0.4542	**0.4998**
	NDCG@10	0.1803	0.1207	0.1440	0.2584	0.2512	0.2965	0.2757	0.2715	0.3156	0.2975	0.2971	0.3143	**0.3385**
	MRR	0.1558	0.1096	0.1226	0.2308	0.2263	0.2669	0.2498	0.2376	0.2852	0.2614	0.2686	0.2882	**0.3051**
Sports	HR@1	0.0763	0.0489	0.0644	0.1160	0.1135	0.1428	0.1334	0.1114	0.1455	0.1255	0.1419	**0.1669**	0.1646
	HR@5	0.2293	0.1603	0.1982	0.3055	0.2866	0.3349	0.3162	0.3041	0.3466	0.3375	0.3367	0.3588	**0.3803**
	NDCG@5	0.1538	0.1048	0.1316	0.2126	0.2020	0.2420	0.2274	0.2096	0.2497	0.2341	0.2418	0.2658	**0.2760**
	HR@10	0.3423	0.2491	0.2967	0.4299	0.4014	0.4551	0.4324	0.4274	0.4622	0.4722	0.4545	0.4737	**0.5059**
	NDCG@10	0.1902	0.1334	0.1633	0.2527	0.2390	0.2806	0.2649	0.2493	0.2869	0.2775	0.2799	0.3029	**0.3165**
	MRR	0.1660	0.1202	0.1435	0.2191	0.2100	0.2469	0.2334	0.2156	0.2520	0.2378	0.2461	0.2691	**0.2763**
Toys	HR@1	0.0585	0.0257	0.0448	0.0997	0.1114	0.1504	0.1333	0.1104	0.1878	0.1262	0.1600	**0.1996**	0.1935
	HR@5	0.1977	0.0978	0.1471	0.2795	0.2614	0.3276	0.3001	0.3055	0.3682	0.3344	0.3389	0.3613	**0.4063**
	NDCG@5	0.1286	0.0614	0.0960	0.1919	0.1885	0.2423	0.2192	0.2102	0.2820	0.2327	0.2528	0.2836	**0.3046**
	HR@10	0.3008	0.1715	0.2369	0.3896	0.3540	0.4211	0.4015	0.4207	0.4663	0.4493	0.4413	0.4509	**0.5062**
	NDCG@10	0.1618	0.0850	0.1248	0.2274	0.2183	0.2724	0.2517	0.2473	0.3136	0.2698	0.2857	0.3125	**0.3368**
	MRR	0.1430	0.0819	0.1131	0.1973	0.1967	0.2454	0.2253	0.2138	0.2842	0.2338	0.2566	0.2871	**0.3012**
Yelp	HR@1	0.0801	0.0624	0.0731	0.2053	0.2188	0.2428	0.2341	0.2102	0.2375	0.2405	0.2176	0.2493	**0.2727**
	HR@5	0.2415	0.2036	0.2249	0.5437	0.5111	0.5768	0.5357	0.5707	0.5745	0.5976	0.5442	0.5725	**0.6191**
	NDCG@5	0.1622	0.1333	0.1501	0.3784	0.3696	0.4162	0.3894	0.3955	0.4113	0.4252	0.3860	0.4162	**0.4527**
	HR@10	0.3609	0.3153	0.3367	0.7265	0.6661	0.7411	0.6897	0.7473	0.7373	0.7597	0.7096	0.7371	**0.7720**
	NDCG@10	0.2007	0.1692	0.1860	0.4375	0.4198	0.4695	0.4393	0.4527	0.4642	0.4778	0.4395	0.4696	**0.5024**
	MRR	0.1740	0.1470	0.1616	0.3630	0.3595	0.3988	0.3769	0.3751	0.3927	0.4026	0.3711	0.4006	**0.4299**

Table 5: Performance comparison of different methods on four session-based datasets. Since these datasets do not have attribute information and the item sequences are usually shorter, we remove several improper baseline methods.

Datasets	Metric	PopRec	GRU4Rec	Caser	HGN	RepeatNet	CLEA	SASRec	SRGNN	GCSAN	FMLP-Rec
Nowplaying	HR@1	0.0757	0.4035	0.3435	0.3491	0.3350	0.2728	0.4396	0.3819	0.4447	**0.4731**
	HR@5	0.2197	0.6829	0.6267	0.6026	0.5257	0.5575	0.7042	0.6028	0.6728	**0.7262**
	NDCG@5	0.1480	0.5536	0.4942	0.4835	0.4355	0.4226	0.5812	0.4986	0.5658	**0.6094**
	HR@10	0.3318	0.7720	0.7318	0.6992	0.6110	0.6766	0.7968	0.7007	0.7616	**0.8081**
	NDCG@10	0.1841	0.5825	0.5283	0.5149	0.4631	0.4612	0.6113	0.5304	0.5946	**0.6360**
	MRR	0.1602	0.5314	0.4752	0.4677	0.4297	0.4070	0.5615	0.4892	0.5520	**0.5895**
Retailrocket	HR@1	0.0817	0.7202	0.6871	0.6432	0.6744	0.3410	0.7588	0.7408	**0.7773**	0.7736
	HR@5	0.2315	0.8597	0.8348	0.7650	0.7724	0.6139	0.8769	0.8373	0.8650	**0.8810**
	NDCG@5	0.1577	0.7982	0.7689	0.7098	0.7270	0.4853	0.8250	0.7937	0.8259	**0.8338**
	HR@10	0.3388	0.8925	0.8719	0.8036	0.8112	0.7296	0.9024	0.8684	0.8901	**0.9060**
	NDCG@10	0.1922	0.8089	0.7809	0.7223	0.7395	0.5227	0.8333	0.8037	0.8340	**0.8419**
	MRR	0.1694	0.7858	0.7563	0.7029	0.7234	0.4702	0.8145	0.7878	0.8199	**0.8247**
Tmall	HR@1	0.1025	0.4178	0.3288	0.4467	**0.5556**	0.2895	0.4045	0.4026	0.4650	0.5173
	HR@5	0.2264	0.5855	0.5066	0.5793	0.6090	0.4573	0.5478	0.5335	0.5940	**0.6565**
	NDCG@5	0.1647	0.5062	0.4230	0.5168	0.5826	0.3768	0.4792	0.4710	0.5329	**0.5911**
	HR@10	0.2967	0.6636	0.5943	0.6471	0.6494	0.5478	0.6275	0.6082	0.6591	**0.7206**
	NDCG@10	0.1874	0.5315	0.4513	0.5386	0.5956	0.4060	0.5049	0.4950	0.5538	**0.6118**
	MRR	0.1723	0.5021	0.4209	0.5168	**0.5894**	0.3775	0.4804	0.4736	0.5328	0.5879
Yoochoose	HR@1	0.1794	0.7208	0.7041	0.6278	0.7450	0.5106	0.7611	0.7575	**0.7855**	0.7749
	HR@5	0.4990	0.8950	0.8818	0.8425	0.8673	0.7628	0.8976	0.8840	0.9073	**0.9084**
	NDCG@5	0.3432	0.8189	0.8026	0.7460	0.8123	0.6465	0.8375	0.8280	**0.8535**	0.8499
	HR@10	0.6574	0.9273	0.9172	0.8927	0.9001	0.8436	0.9273	0.9137	0.9320	**0.9359**
	NDCG@10	0.3948	0.8294	0.8141	0.7623	0.8230	0.6727	0.8471	0.8376	**0.8616**	0.8588
	MRR	0.3276	0.8004	0.7838	0.7246	0.8019	0.6257	0.8240	0.8164	**0.8414**	0.8364

Table 6: Ablation study of our FMLP-Rec, we report NDCG@10 on Beauty and Sports datasets.

	Beauty		Sports	
	HR@10	NDCG@10	HR@10	NDCG@10
FMLP-Rec	**0.4998**	**0.3385**	**0.5059**	**0.3165**
w/o Filter Layer	0.4317	0.2914	0.4243	0.2604
w/o FFN	0.4607	0.3067	0.4594	0.2838
w/o Add & Norm	0.4654	0.2919	0.4831	0.2860
+HPF	0.4567	0.3034	0.4595	0.2810
+LPF	0.4847	0.3233	0.4949	0.3093
+BSF	0.4822	0.3207	0.4835	0.2977

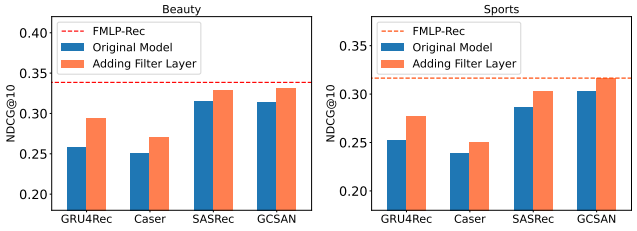

Figure 3: Performance (NDCG@10) comparison of different models enhanced by our learnable filters.

6.2 Applying Learnable Filters to Other Models

The key contribution of our FMLP-Rec is the learnable filters. It is a general module that can be applied to other sequential recommendation models. Thus, in this part, we examine whether our learnable filters can bring improvements to other models. Similar to the operations in Section 3, we add the learnable filters between the embedding layer and the sequence encoder layer, and select RNN-based GRU4Rec [17], CNN-based Caser [53], Transformer-based SASRec [22] and GNN-based GCSAN [60] as the base models.

The results are shown in Figure 3. We also report the performance of FMLP-Rec for comparison. First, after being integrated with our learnable filters, all the baselines achieve better performance. It shows that the learnable filters are generally useful to reduce the influence of noise information for other models, even for different architectures. Second, our FMLP-Rec still outperforms all the baselines and their variants. This is because our model only adopts MLP layers, which has less parameters and is more suitable for the learnable filters in sequential recommendation task.

7 RELATED WORK

Sequential Recommendation. Early works [42] on sequential recommendation are based on the Markov Chain assumption and focus on modeling item-item transition relationships to predict the next item given the last interaction of a user. A series of works follow this line and extend it to high-order MCs [18, 22, 53]. With the development of the neural networks, Hidasi et al. [17] introduced GRU to capture sequential patterns, and a surge of works leverage other neural network architectures for sequential recommendation, *e.g.,* CNN [53], GNN [52, 60] and Transformer [22, 50]. Based on these neural network architectures, various studies introduce other contextual information (*e.g.,* item attributes and reviews) by adding memory networks [20], hierarchical structures [25], data augmentation [58, 66] and pre-training technique [5, 6, 70], etc. Despite the success of these deep models in the sequential recommendation task, we find that these models are easy to be affected by noise information from the user historical behaviors. To solve this problem, we adopt learnable filters to reduce the influence of noise signals, and devise a lightweight all-MLP architecture to alleviate overfitting.

All-MLP Models. Multi-Layer Perceptron (MLP) [61] is a classical feed-forward neural network that consists of fully-connected neurons and non-linear activation functions. It is widely used as assistant modules to couple with CNN [4], RNN [36] and self-attention network [57] to construct deep models. Recently, several studies question the necessity of CNN [54] and self-attention network [28, 29], and propose to use MLP to replace the above architectures. These all-MLP models mostly aim to devise effective MLP-based mixing architectures to capture the interaction of input information, *e.g.,* mixer layer [54], axial shift block [29] and spatial shift block [30], and have performed well in various tasks, *e.g.,* image classification [54]and semantic segmentation [29]. More recently, GFNet [39] adopts 2D Fourier transform with the global filter layer to learn long-term spatial dependencies in the frequency domain, which shows competitive performance to Transformer-based models. However, all-MLP models are hard to capture sequential characteristics, which are essential to the sequential recommendation task. In our approach, we design an all-MLP model that includes the filter layers to encode the sequence in the frequency domain. It outperforms competitive RNN, CNN and Transformer-based baseline models in eight datasets, with a lightweight architecture.

8 CONCLUSION

In this paper, we found that the logged user behavior data usually contains noisy interactions, and performed empirical study analysis to show that filtering algorithms from the digital signal processing area are useful to alleviate the influence of the noise in deep sequential recommendation models. Inspired by it, we proposed FMLP-Rec, an all-MLP model with learnable filters for sequential recommendation task. The all-MLP architectures endowed our model with lower time complexity, and the learnable filters can be optimized by SGD to adaptively attenuate the noise information in the frequency domain. We also showed that the learnable filters are equivalent to the circular convolution in the time domain, which have a larger receptive field and can better capture periodic characteristics. Experimental results have shown that our approach outperforms several competitive RNN, CNN, GNN and Transformer-based baselines.

ACKNOWLEDGMENTS

This work was partially supported by the National Natural Science Foundation of China under Grant No. 61872369 and 61832017, Beijing Outstanding Young Scientist Program under Grant No. BJJWZYJH012019100020098, the Outstanding Innovative Talents Cultivation Funded Programs 2021 and Public Computing Cloud, Renmin University of China. This work is supported by Beijing Academy of Artificial Intelligence (BAAI). Xin Zhao is the corresponding author.

REFERENCES

[1] Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving web search ranking by incorporating user behavior information. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*. 19–26.

[2] John G Anderson and Susan E Hough. 1984. A model for the shape of the Fourier amplitude spectrum of acceleration at high frequencies. *Bulletin of the Seismological Society of America* 74, 5 (1984), 1969–1993.

[3] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. *CoRR* abs/1607.06450 (2016). arXiv:1607.06450 http://arxiv.org/abs/1607.06450

[4] Yoshua Bengio, Yann LeCun, and Donnie Henderson. 1993. Globally Trained Handwritten Word Recognizer Using Spatial Representation, Convolutional Neural Networks, and Hidden Markov Models. In *Advances in Neural Information Processing Systems 6, [7th NIPS Conference, Denver, Colorado, USA, 1993]*, Jack D. Cowan, Gerald Tesauro, and Joshua Alspector (Eds.). Morgan Kaufmann, 937–944.

[5] Shuqing Bian, Wayne Xin Zhao, Kun Zhou, Jing Cai, Yancheng He, Cunxiang Yin, and Ji-Rong Wen. 2021. Contrastive Curriculum Learning for Sequential User Behavior Modeling via Data Augmentation. In *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong (Eds.). ACM, 3737–3746. https://doi.org/10.1145/3459637.3481905

[6] Shuqing Bian, Wayne Xin Zhao, Kun Zhou, Xu Chen, Jing Cai, Yancheng He, Xingji Luo, and Ji-Rong Wen. 2021. A Novel Macro-Micro Fusion Network for User Representation Learning on Mobile Apps. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 3199–3209. https://doi.org/10.1145/3442381.3450109

[7] Rich Caruana, Steve Lawrence, and Lee Giles. 2001. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. *Advances in neural information processing systems* (2001), 402–408.

[8] Huiyuan Chen and Jing Li. 2019. Data Poisoning Attacks on Cross-domain Recommendation. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019*, Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu (Eds.). ACM, 2177–2180. https://doi.org/10.1145/3357384.3358116

[9] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential Recommendation with User Memory Networks. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, Yi Chang, Chengxiang Zhai, Yan Liu, and Yoelle Maarek (Eds.). ACM, 108–116. https://doi.org/10.1145/3159652.3159668

[10] James W Cooley and John W Tukey. 1965. An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation* 19, 90 (1965), 297–301.

[11] Angel Herrero Crespo and Ignacio Rodriguez Del Bosque. 2010. The influence of the commercial features of the Internet on the adoption of e-commerce by consumers. *Electronic Commerce Research and Applications* 9, 6 (2010), 562–575.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT 2019*. 4171–4186.

[13] Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556* (2019).

[14] Matteo Frigo and Steven G Johnson. 2005. The design and implementation of FFTW3. *Proc. IEEE* 93, 2 (2005), 216–231.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 770–778. https://doi.org/10.1109/CVPR.2016.90

[16] Michael T Heideman, Don H Johnson, and C Sidney Burrus. 1985. Gauss and the history of the fast Fourier transform. *Archive for history of exact sciences* 34, 3 (1985), 265–277.

[17] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR 2016*.

[18] B. Hidasi, M. Quadrana, A. Karatzoglou, and D. Tikk. 2016. Parallel Recurrent Neural Network Architectures for Feature-rich Session-based Recommendations. In *RecSys 2016*. 241–248.

[19] Chin-Lung Hsu and Hsi-Peng Lu. 2007. Consumer behavior in online game communities: A motivational factor perspective. *Computers in Human Behavior* 23, 3 (2007), 1642–1659.

[20] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *SIGIR 2018*. 505–514.

[21] Long Jin, Yang Chen, Tianyi Wang, Pan Hui, and Athanasios V Vasilakos. 2013. Understanding user behavior in online social networks: A survey. *IEEE Communications Magazine* 51, 9 (2013), 144–150.

[22] W.-C. Kang and J. J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *ICDM 2018*. 197–206.

[23] D. P. Kingma and J. Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR 2015*.

[24] Jake Lever, Martin Krzywinski, and Naomi Altman. 2016. Points of significance: model selection and overfitting. *Nature methods* 13, 9 (2016), 703–705.

[25] Chenliang Li, Xichuan Niu, Xiangyang Luo, Zhenzhong Chen, and Cong Quan. 2019. A Review-Driven Neural Model for Sequential Recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, Sarit Kraus (Ed.). ijcai.org, 2866–2872. https://doi.org/10.24963/ijcai.2019/397

[26] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma. 2017. Neural Attentive Session-based Recommendation. In *CIKM 2017*. 1419–1428.

[27] Jiacheng Li, Yujie Wang, and Julian J. McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, James Caverlee, Xia (Ben) Hu, Mounia Lalmas, and Wei Wang (Eds.). ACM, 322–330. https://doi.org/10.1145/3336191.3371786

[28] Yawei Li, Kai Zhang, Jiezhang Cao, Radu Timofte, and Luc Van Gool. 2021. LocalViT: Bringing Locality to Vision Transformers. *CoRR* abs/2104.05707 (2021). arXiv:2104.05707 https://arxiv.org/abs/2104.05707

[29] Dongze Lian, Zehao Yu, Xing Sun, and Shenghua Gao. 2021. AS-MLP: An Axial Shifted MLP Architecture for Vision. *CoRR* abs/2107.08391 (2021). arXiv:2107.08391 https://arxiv.org/abs/2107.08391

[30] Hanxiao Liu, Zihang Dai, David R. So, and Quoc V. Le. 2021. Pay Attention to MLPs. *CoRR* abs/2105.08050 (2021). arXiv:2105.08050 https://arxiv.org/abs/2105.08050

[31] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. 2016. Context-Aware Sequential Recommendation. In *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, 2016, Barcelona, Spain*, Francesco Bonchi, Josep Domingo-Ferrer, Ricardo Baeza-Yates, Zhi-Hua Zhou, and Xindong Wu (Eds.). IEEE Computer Society, 1053–1058. https://doi.org/10.1109/ICDM.2016.0135

[32] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. 2015. Recommender system application developments: a survey. *Decision Support Systems* 74 (2015), 12–32.

[33] C. Ma, P. Kang, and X. Liu. 2019. Hierarchical Gating Networks for Sequential Recommendation. In *KDD 2019*. 825–833.

[34] J. J. McAuley, C. Targett, Q. Shi, and A. van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *SIGIR 2015*. 43–52.

[35] Sachin Mehta, Marjan Ghazvininejad, Srinivasan Iyer, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2020. Delight: Deep and light-weight transformer. *arXiv preprint arXiv:2008.00623* (2020).

[36] Fernando J. Pineda. 1987. Generalization of Back propagation to Recurrent and Higher Order Neural Networks. In *Neural Information Processing Systems, Denver, Colorado, USA, 1987*, Dana Z. Anderson (Ed.). American Institue of Physics, 602–611.

[37] Yuqi Qin, Pengfei Wang, and Chenliang Li. 2021. The World is Binary: Contrastive Learning for Denoising Next Basket Recommendation. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 859–868. https://doi.org/10.1145/3404835.3462836

[38] Lawrence R Rabiner and Bernard Gold. 1975. Theory and application of digital signal processing. *Englewood Cliffs: Prentice-Hall* (1975).

[39] Yongming Rao, Wenliang Zhao, Zheng Zhu, Jiwen Lu, and Jie Zhou. 2021. Global Filter Networks for Image Classification. *CoRR* abs/2107.00645 (2021). arXiv:2107.00645 https://arxiv.org/abs/2107.00645

[40] Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten de Rijke. 2019. RepeatNet: A Repeat Aware Neural Recommendation Machine for Session-Based Recommendation. In *AAAI 2019*. 4806–4813.

[41] S. Rendle. 2010. Factorization Machines. In *ICDM 2010*. 995–1000.

[42] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *WWW 2010*. 811–820.

[43] Alan Said, Brijnesh J Jain, Sascha Narr, and Till Plumbaum. 2012. Users and noise: The magic barrier of recommender systems. In *International Conference on User Modeling, Adaptation, and Personalization*. Springer, 237–248.

[44] Yan-Fang Sang, Dong Wang, Ji-Chun Wu, Qing-Ping Zhu, and Ling Wang. 2009. The relation between periods' identification and noises in hydrologic series data. *Journal of Hydrology* 368, 1-4 (2009), 165–177.

[45] Samir S Soliman and Mandyam D Srinath. 1990. Continuous and discrete signals and systems. *Englewood Cliffs* (1990).

[46] W. Song, C. Shi, Z. Xiao, Z. Duan, Y. Xu, M. Zhang, and J. Tang. 2019. AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks. In *CIKM 2019*. 1161–1170.

[47] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[48] David H Staelin. 1969. Fast folding algorithm for detection of periodic pulse trains. *Proc. IEEE* 57, 4 (1969), 724–725.

[49] Klaus Stammler. 1993. SeismicHandler—programmable multichannel data handler for interactive and automatic processing of seismological analyses. *Computers & geosciences* 19, 2 (1993), 135–140.

[50] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *CIKM 2019*. 1441–1450.

[51] Yatong Sun, Bin Wang, Zhu Sun, and Xiaochun Yang. 2021. Does Every Data Instance Matter? Enhancing Sequential Recommendation by Eliminating Unreliable Data. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, Zhi-Hua Zhou (Ed.). ijcai.org, 1579–1585. https://doi.org/10.24963/ijcai.2021/218

[52] Qiaoyu Tan, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. 2021. Sparse-interest network for sequential recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 598–606.

[53] J. Tang and K. Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *WSDM 2018*. 565–573.

[54] Ilya O. Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. 2021. MLP-Mixer: An all-MLP Architecture for Vision. *CoRR* abs/2105.01601 (2021). arXiv:2105.01601 https://arxiv.org/abs/2105.01601

[55] Munetoshi Unuma, Ken Anjyo, and Ryozo Takeuchi. 1995. Fourier principles for emotion-based human figure animation. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. 91–96.

[56] Charles Van Loan. 1992. *Computational frameworks for the fast Fourier transform*. SIAM.

[57] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention is All you Need. In *NeurIPS 2017*. 5998–6008.

[58] Zhenlei Wang, Jingsen Zhang, Hongteng Xu, Xu Chen, Yongfeng Zhang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Counterfactual data-augmented sequential recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 347–356.

[59] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-Based Recommendation with Graph Neural Networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 346–353. https://doi.org/10.1609/aaai.v33i01.3301346

[60] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph Contextualized Self-Attention Network for Session-based Recommendation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, Sarit Kraus (Ed.). ijcai.org, 3940–3946. https://doi.org/10.24963/ijcai.2019/547

[61] Eyal Yair and Allen Gersho. 1988. The Boltzmann Perceptron Network: A Multi-Layered Feed-Forward Network Equivalent to the Boltzmann Machine. In *Advances in Neural Information Processing Systems 1, [NIPS Conference, Denver, Colorado, USA, 1988]*, David S. Touretzky (Ed.). Morgan Kaufmann, 116–123.

[62] F. Yuan, A. Karatzoglou, I. Arapakis, J. Jose, and X. He. 2019. A Simple Convolutional Generative Network for Next Item Recommendation. In *WSDM 2019*. 582–590.

[63] Zhenrui Yue, Zhankui He, Huimin Zeng, and Julian J. McAuley. 2021. Black-Box Attacks on Sequential Recommenders via Data-Free Model Extraction. In *RecSys '21: Fifteenth ACM Conference on Recommender Systems, Amsterdam, The Netherlands, 27 September 2021 - 1 October 2021*, Humberto Jesús Corona Pampín, Martha A. Larson, Martijn C. Willemsen, Joseph A. Konstan, Julian J. McAuley, Jean Garcia-Gathright, Bouke Huurnink, and Even Oldridge (Eds.). ACM, 44–54. https://doi.org/10.1145/3460231.3474275

[64] Shuangfei Zhai, Walter Talbott, Nitish Srivastava, Chen Huang, Hanlin Goh, Ruixiang Zhang, and Josh M. Susskind. 2021. An Attention Free Transformer. *CoRR* abs/2105.14103 (2021). arXiv:2105.14103 https://arxiv.org/abs/2105.14103

[65] Hengtong Zhang, Yaliang Li, Bolin Ding, and Jing Gao. 2020. Practical Data Poisoning Attack against Next-Item Recommendation. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen (Eds.). ACM / IW3C2, 2458–2464. https://doi.org/10.1145/3366423.3379992

[66] Shengyu Zhang, Dong Yao, Zhou Zhao, Tat-Seng Chua, and Fei Wu. 2021. Causerec: Counterfactual user sequence synthesis for sequential recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 367–377.

[67] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)* 52, 1 (2019), 1–38.

[68] Wayne Xin Zhao, Junhua Chen, Pengfei Wang, Qi Gu, and Ji-Rong Wen. 2020. Revisiting Alternative Experimental Settings for Evaluating Top-N Item Recommendation Algorithms. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2329–2332.

[69] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, et al. 2021. Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4653–4664.

[70] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux (Eds.). ACM, 1893–1902. https://doi.org/10.1145/3340531.3411954

[71] Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. 2020. Improving conversational recommender systems via knowledge graph based semantic fusion. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1006–1014.

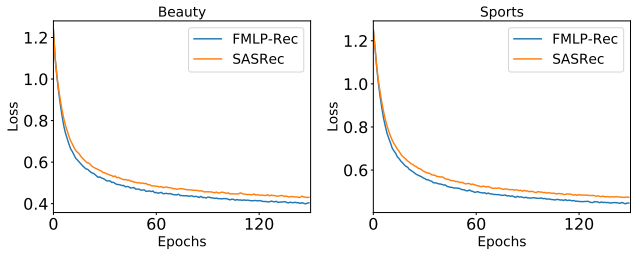

Figure 4: Training loss of our approach and SASRec on Beauty and Sports datasets with the increasing epochs.

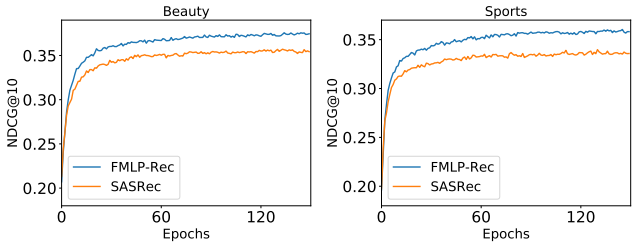

Figure 5: Testing recall (NDCG@10) of our approach and SASRec on Beauty and Sports datasets with the increasing epochs.

A IMPLEMENTATION DETAILS

For Caser, HGN and BERT4Rec, we use the source code provided by their authors. For CLEA, we implement it by PyTorch [7]. For other methods, we implement them based on RecBole [69]. All hyper-parameters are set following the suggestions from the original papers. For our proposed FMLP-Rec, we develop it based on the codes and data released by S^3-Rec [70], the dimension of the embedding is 64, and the maximum sequence length is 50. We set the number of the learnable filter blocks as 2, the batch size is set as 256. We use the Adam optimizer [23] with a learning rate of 0.001, and adopt early-stopped training if the MRR performance on the validation set decreases for 10 continuous epochs.

B MORE ANALYSIS

B.1 Training Curves Analysis

To reveal that our proposed FMLP-Rec is able to converge better than baseline models, we analyze the training process of SASRec and FMLP-Rec. We conduct the experiments on Beauty and Sports datasets and plot the curves of training loss and testing recall of the two models in Figure 4 and Figure 5, respectively. Note that in the figures we show the training processes under the optimal hyper-parameter setting for both methods.

In these figures, we can find that along the training process, FMLP-Rec consistently obtains lower training loss, which indicates that FMLP-Rec is able to better fit the training data than SASRec. Moreover, the lower training loss successfully transfers to better testing accuracy, which indicates the strong generalization capacity of our FMLP-Rec. The reason is that FMLP-Rec adopts learnable filters to attenuate noise information within the item representation matrices, which can better capture the useful characteristics and alleviate the overfitting problem on noise information. In contrast, the higher training loss and lower testing accuracy of SASRec reflect the practical difficulty to train an over-parameterized Transformer-based model well.

[7] https://pytorch.org/

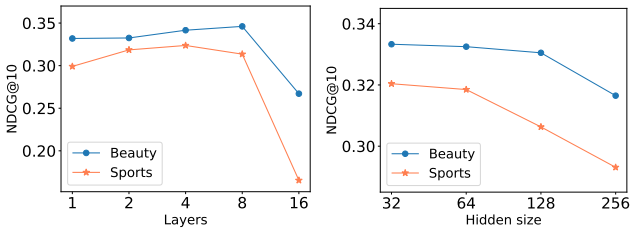

Figure 6: Performance (NDCG@10) comparison w.r.t. different numbers of pre-training epochs on Beauty and Toys datasets.

B.2 Parameter Tuning

When applying our FMLP-Rec to a new dataset, besides the standard hyper-parameters *e.g.,* learning rate and training epochs, the most important hyper-parameters to tune are the layer number and the hidden size. Here we investigate the performance change of FMLP-Rec with the respect to the layer number and the hidden size on Beauty and Sports datasets.

As shown in Figure 6, with the increasing of the layer number, the performance of FMLP-Rec can be further improved. However, when the layer number achieves 16, the gain starts to become trivial. It indicates that it is promising to deepen our FMLP-Rec for achieving more exciting performance, but the most proper architecture requires to be further investigated. We leave it into future works. For hidden size, we can see that the best value is about 64. The reason may be that too large hidden size can involve more parameters that are easy to overfit into the noise information, while too small size is not enough to capture the user preference.

B.3 Visualization of Learned Filters on Other Datasets

The core operation in FMLP-Rec is the the element-wise multiplication between frequency domain features $\mathbf{F}^l$ and the learnable filter $\mathbf{X}$ in Eq. 5. In this part, we visualize and interpret the learnable filters in the frequency domain. We average the values of learnable filters in the first layer for all dimensions of features, and depict it in Figure 7.

We can see that the learnable filters have clear patterns in the frequency domain, where the low-frequency signals are assigned larger positive weights than high-frequency signals. Therefore, the learnable filters can be seen as special low-pass filters to attenuate the high-frequency noise. This finding is similar to the empirical results in Section 3 that high-frequency information in item embedding matrix is usually noise.

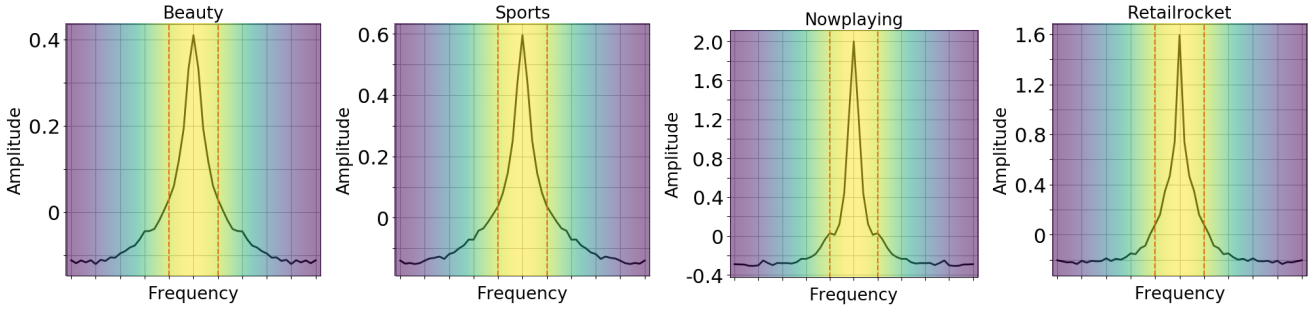

Figure 7: Visualization of our learned filters on four datasets, the amplitude denotes the weight from X to multiply the corresponding frequency in $\mathbf{F}^l$.

Table 7: Performance comparison of different methods on four datasets under full-sort setting. The best performance and the second best performance methods are denoted in bold and underlined fonts respectively.

Datasets	Metric	GRU4Rec	Caser	SASRec	FMLP-Rec
Beauty	HR@5	0.0164	0.0205	0.0387	**0.0398**
	NDCG@5	0.0099	0.0131	0.0249	**0.0258**
	HR@10	0.0283	0.0347	0.0605	**0.0632**
	NDCG@10	0.0137	0.0176	0.0318	**0.0333**
	HR@20	0.0479	0.0556	0.0902	**0.0958**
	NDCG@20	0.0187	0.0229	0.0394	**0.0415**
Sports	HR@5	0.0129	0.0116	**0.0233**	0.0218
	NDCG@5	0.0086	0.0072	**0.0154**	0.0144
	HR@10	0.0204	0.0194	**0.0350**	0.0344
	NDCG@10	0.0110	0.0097	**0.0192**	0.0185
	HR@20	0.0333	0.0314	0.0507	**0.0537**
	NDCG@20	0.0142	0.0126	0.0231	**0.0233**
Toys	HR@5	0.0097	0.0166	**0.0463**	0.0456
	NDCG@5	0.0059	0.0107	0.0306	**0.0317**
	HR@10	0.0176	0.0270	0.0675	**0.0683**
	NDCG@10	0.0084	0.0141	0.0374	**0.0391**
	HR@20	0.0301	0.0420	0.0941	**0.0991**
	NDCG@20	0.0116	0.0179	0.0441	**0.0468**
Yelp	HR@5	0.0152	0.0151	0.0162	**0.0179**
	NDCG@5	0.0099	0.0096	0.0100	**0.0113**
	HR@10	0.0263	0.0253	0.0274	**0.0304**
	NDCG@10	0.0134	0.0129	0.0136	**0.0153**
	HR@20	0.0439	0.0422	0.0457	**0.0511**
	NDCG@20	0.0178	0.0171	0.0182	**0.0205**

B.4 Performance under Full-Ranking Setting

To further verify the effectiveness of our FMLP-Rec, we also conduct experiments under the full-ranking setting. We select RNN-based GRU4Rec, CNN-based Caser, and Transformer-based SASRec as representative baseline models. The results on Beauty, Sports, Toys and Yelp datasets are shown in Table 7.

From the table, we can see similar tendency as in Table 4. It can be found that SASRec and FMLP-Rec perform much better than the other three methods. In most of cases, FMLP-Rec outperforms SASRec a lot. It indicates that the all-MLP architecture and learnable filters are exactly effective for the sequential recommendation task.