Adaptive Collaborative Filtering with Personalized Time Decay Functions for Financial Product Recommendation

Ashraf Ghiye, Baptiste Barreau, Laurent Carlier, Michalis Vazirgiannis

Adaptive Collaborative Filtering with Personalized Time Decay Functions for Financial Product Recommendation

ASHRAF GHIYE, BNP Paribas Corporate and Institutional Banking, France and École Polytechnique, France
BAPTISTE BARREAU, BNP Paribas Corporate and Institutional Banking, France
LAURENT CARLIER, BNP Paribas Corporate and Institutional Banking, France
MICHALIS VAZIRGIANNIS, École Polytechnique, France

Classical recommender systems often assume that historical data are stationary and fail to account for the dynamic nature of user preferences, limiting their ability to provide reliable recommendations in time-sensitive settings. This assumption is particularly problematic in finance, where financial products exhibit continuous changes in valuations, leading to frequent shifts in client interests. These evolving interests, summarized in the past client-product interactions, see their utility fade over time with a degree that might differ from one client to another. To address this challenge, we propose a time-dependent collaborative filtering algorithm that can adaptively discount distant client-product interactions using personalized decay functions. Our approach is designed to handle the non-stationarity of financial data and produce reliable recommendations by modeling the dynamic collaborative signals between clients and products. We evaluate our method using a proprietary dataset from BNP Paribas and demonstrate significant improvements over state-of-the-art benchmarks from relevant literature. Our findings emphasize the importance of incorporating time explicitly in the model to enhance the accuracy of financial product recommendation.

CCS Concepts: • **Recommender Systems**; • **Collaborative Filtering**;

Additional Key Words and Phrases: Dynamic Collaborative Signals, Adaptive Filtering, Time-Dependent, Context-Aware, Finance

ACM Reference Format:
Ashraf Ghiye, Baptiste Barreau, Laurent Carlier, and Michalis Vazirgiannis. 2023. Adaptive Collaborative Filtering with Personalized Time Decay Functions for Financial Product Recommendation. In *Seventeenth ACM Conference on Recommender Systems (RecSys '23), September 18–22, 2023, Singapore, Singapore.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3604915.3608832

1 INTRODUCTION

In many e-commerce applications, such as music streaming services, the primary goal of recommender systems is to maximize user engagement. Products in these applications typically have low maintenance costs and negligible inventory risk. On the contrary, corporate and institutional banks operating as market makers need to ensure liquidity in financial markets by constantly offering bid (buy) and ask (sell) prices which involve taking frequent positions in both directions. The client can send an electronic notification asking for detailed information, which shows her interest in buying or selling a specific product, through a process known as a request for quotation (RFQ). Conversely, salespeople may contact clients to offer competitive quotes on products held by the bank. Having a reliable recommendation algorithm for corporate banks has two main consequences: (1) increasing customer satisfaction by providing them with relevant investment opportunities and (2) reducing inventory risk by enabling risk managers and salespeople to be more proactive in anticipating the needs of their clients.

Manuscript submitted to ACM

Modeling the clients' behaviour in terms of their historical interests is of paramount importance to understand their profile and anticipate their needs. Some clients adopt a more conservative approach, displaying enduring interest in specific products; others adopt a rather diverse strategy, showing limited recurring interest in products. To provide personalized recommendations that resonate with all clients (e.g., to match their risk appetite), it is crucial to consider the context of their previous trades. For instance, products similar to those a client had inquired a few days ago may be more salient in determining her current interest compared to products she consulted months ago.

To that end, we present a time-dependent recommender system tailored to the unique characteristics of the financial industry. We start by defining the clients' and products' context in terms of their first-hop neighbourhoods in the user-item interaction graph. In this work, we assume that interest persists for a certain period after the interactions have occurred, after which the utility of the interactions progressively fades over time. Therefore, we propose to personalize the rate at which this decay occurs. We conduct an extensive study on a proprietary database of G10 Bonds RFQs to test the effectiveness of different aggregation functions in encoding the dynamic collaborative signals. In addition, we compare our proposed method to relevant benchmarks from prior studies (HCF [1], LightGCN [10]). Our findings highlight the importance of incorporating time explicitly in the modeling process. In summary, our contributions are:

- We propose personalized time decay functions to aggregate the past, assuming that the utility of past interactions decreases monotonically with time.
- We demonstrate that using time explicitly to down-weight user-item interactions enhances the performance of graph collaborative filtering algorithms.
- Our approach shows substantial improvements compared with different aggregation strategies, from a time-agnostic average to more advanced methods like Gated Recurrent Unit [4] and Attention [24].

In the following sections, the terms "user-item" or "client-product" will be used interchangeably to align with the vocabulary commonly used in the recommender systems literature.

2 BACKGROUND AND RELATED WORK

The main objective of a recommender system is to predict the level of interest a user u exhibits towards a specific item i by scoring any user-item pair from the set of all users U and all items I. Typically, historical interest are stored in the form of an interaction matrix, denoted by $\mathbf{A}$, where each entry a_{ui} corresponds to the feedback (e.g., ratings for explicit or clicks for implicit) provided by a user u for an item i. In our study, we deal with an implicit feedback problem, where interactions correspond to a binary signal, i.e., the requests for quotation (RFQs).

Collaborative Filtering (CF) is a widely used technique in recommender systems [13], with the core task of predicting the missing entries in the interaction matrix $\mathbf{A} \in \{0,1\}^{|U|\times|I|}$. For instance, Matrix Factorization (MF) [17, 22] approximates $\mathbf{A}$ by the product of two lower-rank matrices: $\mathbf{P} \in \mathbb{R}^{|U|\times E}$ and $\mathbf{Q} \in \mathbb{R}^{|I|\times E}$, such that $\hat{\mathbf{A}} = \mathbf{P} \times \mathbf{Q}^\top$ recovers the original matrix with minimal loss of information. Each row $\mathbf{p}_u$ in $\mathbf{P}$ and each row $\mathbf{q}_i$ in $\mathbf{Q}$ are commonly referred to as user and item embeddings, respectively, with E representing the embedding size. Recent neural recommender models [12] use similar embedding components but improve the interaction modeling by using a stack of non-linear transformations to learn better representations and incorporating side information to improve the factorization [31].

Related Work

Conventional CF algorithms fail to account for crucial contextual factors, such as the time or order of user interactions. Prior research has attempted to incorporate time into these methods through various approaches [3, 6], such as adding

time as a third dimension in the rating matrix [16, 20, 30], using time in the pre-processing step to down-weight item ratings based on recency [7, 9], or only considering items rated within a fixed time window [21]. However, most of these approaches have focused on explicit feedback and received less attention in the context of implicit feedback.

Another line of research has used historical interactions as pre-existing features to introduce dynamics in neural network recommender systems [5, 11, 15]. In particular, Sequential Recommender Systems (SRS) model the patterns in user history to provide dynamic recommendations [18, 25, 26]. One such method is Caser [23], which uses convolutional filters to model short- and long-term behaviors in a user's history. However, SRS models often focus on modeling only one side of the interactions, namely the user history.

More recently, Graph Neural Networks (GNNs) have emerged as the state-of-the-art approach for recommender systems [8, 32]. By representing historical user-item interactions as a bipartite graph, GNNs capture collaborative signals by learning representations of users and items based on their connectivity. Specifically, GNNs refine a user's (or an item's) embedding by aggregating the embeddings of their local neighbourhoods [10, 27]. However, most existing methods represent interactions as a static graph and fail to record the order of the user-item pairs, limiting their ability to capture short-term preferences. Few studies have begun to explore this challenge [19, 29]. Notably, Wu et al. [28] demonstrate how to construct dynamic graphs to frame the next-item prediction task as a link prediction problem.

Our research targets financial recommender systems and focuses on the link prediction approach. Building on the previous work of Barreau et al. [1], we introduce time in the propagation step to discount the utility of past interactions.

3 METHODOLOGY

We propose a new temporal aggregation function that aims at producing better time-aware recommendations by adaptively down-weighting the utility of past interactions - we call it Adaptive Collaborative Filtering (ACF).

Our goal is to learn a mapping function to generate temporal embedding for each user u (and item i), at any given time t, based on their sequential behaviour in the past. The model prediction at time t, measured in days, is defined as the inner product of user and item final embeddings at that time: $\hat{a}^t_{ui} = \sigma(\langle \mathbf{p}^t_u,\ \mathbf{q}^t_i \rangle)$, where σ is the sigmoid function. The model's outputs are used as the ranking scores to generate daily recommendations.

We decompose the temporal embedding of user u at time t into two components: (1) a static embedding, denoted by $\mathbf{e}^t_u = f_1(\mathbf{x}^t_u)$, which captures the user's features (i.e., IDs), and (2) a history embedding, denoted by $\mathbf{h}^t_u = \text{AGG}(\{\mathbf{e}^{t_j}_j : t_j < t\})$, which aggregates the embeddings of the historical items, where $\mathbf{e}^{t_j}_j = f_2(\mathbf{x}^{t_j}_j)$ represents the static embedding of the j^{th} item inquired by the user before t.

The final user embedding is computed using a 1×1 convolutional network ϕ_1. The filters operate along the embedding axis, and considers both static and history embeddings as separate channels (Barreau et al. [1]):

$$\mathbf{p}^t_u = \phi_1(\mathbf{e}^t_u,\ \mathbf{h}^t_u) \tag{1}$$

Similarly, we apply the same decomposition on the item side to obtain the final item embedding using ϕ_2:

$$\mathbf{p}^t_i = \phi_2(\mathbf{e}^t_i,\ \mathbf{h}^t_i) \tag{2}$$

The sub-scripted functions (f_1, ϕ_1) and (f_2, ϕ_2) denote separate weights for the user and item sides, respectively.

Our research focuses on the dynamic component. We define the temporal neighbourhood of user u at query time t as the set $\mathcal{N}_t(u) = [(i_1, t_1), (i_2, t_2), \ldots, (i_n, t_n)]$, which consists of the n most recent **time-stamped** interactions of

user u. t_j describes the timestamp when the interaction occurred ($t_j < t$). The temporal neighbourhood of an item i is defined similarly. The history embedding is obtained by mapping the temporal set into a vector in $\mathbb{R}^E$.

Barreau et al. [1] propose taking a simple average of the embeddings over the set of history, which can be viewed as linearly propagating the static embeddings on the user-item interaction graph (LightGCN[10]):

$$\mathbf{h}_u^t = \frac{1}{\mid \mathcal{N}_t(u) \mid} \sum_{(j,t_j) \in \mathcal{N}_t(u)} \mathbf{e}_j^{t_j}; \qquad \mathbf{h}_i^t = \frac{1}{\mid \mathcal{N}_t(i) \mid} \sum_{(j,t_j) \in \mathcal{N}_t(i)} \mathbf{e}_j^{t_j} \tag{3}$$

where $|\mathcal{N}_t(u)|$ and $|\mathcal{N}_t(i)|$ denote the neighbourhood size. The current formulation assigns equal importance to all past interactions, regardless of their temporal order. However, we suppose that recent RFQs provide a stronger signal of interest than older ones. Consequently, it is crucial to incorporate temporal information in the aggregation to capture the evolving nature of user preferences and account for the shifting dynamics of user-item interactions.

Moreover, not all clients share the same level of trading activity. Some may have multiple inquiries per day, while others may have long periods between interactions. Considering a fixed number of past interactions can lead to sets of inquiries with varying time intervals and possibly starting from different points in time. As a result, relying solely on time to discount older interactions may be sub-optimal in representing the diverse range of time horizons.

To address the above issues, we propose to learn different time factors to reflect the relative importance of an item i to a user u given the time the interaction has occurred t_j. More generally, we want to learn personalized weighting coefficients $w_{ui}^{t_j}$ to model the varying utility of the interaction based on time and the user-item pair:

$$\mathbf{h}_u^t = \sum_{(j,t_j) \in \mathcal{N}_t(u)} w_{uj}^{t_j} \, \mathbf{e}_j^{t_j} \tag{4}$$

To that end, we propose to parameterize the aggregation function with power law kernels and personalized decay rates, i.e., we model the weights w_{ui}^t to be inversely proportional to the time elapsed since the interaction ($\Delta t_j = t - t_j$):

$$w_{uj}^{t_j} = \frac{1}{\Delta t_j^{\alpha_{ui}}} \tag{5}$$

where α_{ui} are learnable decay rates given by a feed-forward neural network. To compute the decay rate, we pass the concatenated user and item's embeddings through a linear layer and apply a sigmoid activation to constrain α_{ui} in the range $[0, 1]$ - as it has shown better empirical results compared to other activation functions (e.g., ReLU):

$$\alpha_{uj} = \sigma(g_1(\mathbf{e}_u \mid\mid \mathbf{e}_j^{t_j})) = \sigma(\mathbf{W}_1(\mathbf{e}_u \mid\mid \mathbf{e}_j^{t_j})) \tag{6}$$

Figure 1 illustrates the global architecture of the model.

4 EXPERIMENTS

In this study, we conducted a series of experiments to evaluate the performance of our proposed method (**ACF**) against two competing benchmarks: **HCF** [1] and **LightGCN** [10]. In addition, we compared against two other standard benchmarks: a vanilla matrix factorization (**MF**) and a most popular rule (**POP**). Finally, we implemented variants of **ACF** to study the impact of different aggregation functions on the model performance. All models are described below:

- **POP:** We recommend the items (or users) based on their popularity in the training set.
- **MF:** We maintain the same learning procedure while excluding the dynamic component from the model, i.e., the history blocks. The only learnable parameters are the two lookup tables f_1 and f_2 of combined size $(\mid U \mid + \mid I \mid) \times E$.

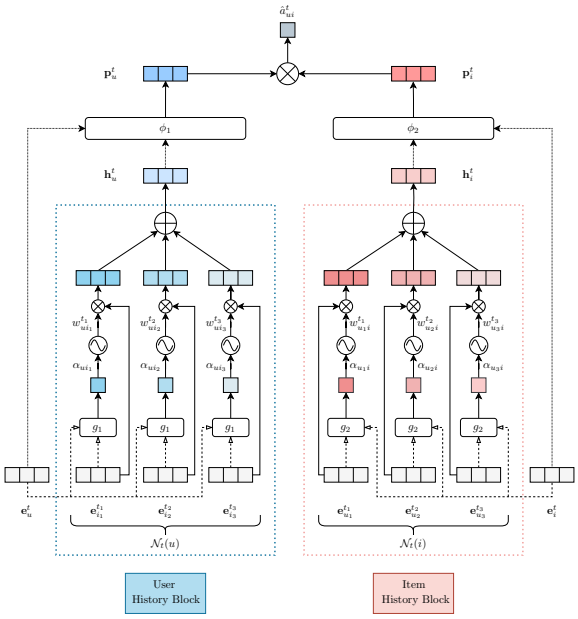

Fig. 1. Summary of the model architecture for a history of size $n = 3$. The static embedding of the k^{th} item (user) in the user's (item's) history is denoted by $\mathbf{e}_{i_k}^{t_k}$ ($\mathbf{e}_{u_k}^{t_k}$). Dashed and dotted arrows correspond to the concatenate and stack operations, respectively. ϕ_1 and ϕ_2 combine the ID embedding (static embedding) and first-order signal (history embedding). The history embedding is obtained by taking a weighted sum of the static embeddings in the temporal neighbourhood. The weights w_{ui}^t are optimized to capture the varying utility of the interaction given the user-item pair, and the time elapsed since the interaction occurred.

- **HCF:** We use an average to aggregate the history (Eq. 3). ϕ_1 and ϕ_2 are each a 2-layers 1×1 convolutional network with 64 and 32 filters respectively. The total number of trainable parameters is $(\mid U \mid + \mid I \mid) \times E + 2 \times (2 \times 64 \times 32)$.
- **LightGCN:** We use a one-layer LightGCN, which is similar to HCF with the only difference of using a linear combination for the final embedding, i.e., $\mathbf{p}_u^t = \mathbf{e}_u^t + \frac{1}{2}\mathbf{h}_u^t$. This algorithm has the same number of parameters as MF.
- **ACF-LP, ACF-LE:** We keep the same architecture as HCF, with extra $2 \times (2 \times E)$ trainable parameters to produce the learnable decay rates (Eq. 6). **P** and **E** indicate respectively a power law or an exponential kernel for the decay.
- **ACF-P, ACF-E:** Same as previous models, but with fixed decay rates ($\alpha_{ui} = 1$).
- **ACF-ATT:** The weights of Equation 4 are defined as attention coefficients directly: $w_{ui}^{t_j} = \sigma(\mathbf{W}(\mathbf{e}_u^t \mid\mid \mathbf{e}_j^{t_j} + \mathbf{e}_{\Delta t_j}))$, where $\mathbf{e}_{\Delta t_j}$ is a time embedding given by a look-up table (f_3) of size $100 \times E$.
- **ACF-GRU:** We use a Gated Recurrent Unit (GRU) with one hidden layer of size E. We take the last hidden state of the sequence to represent the history embedding.

4.1 Data

We use a proprietary dataset from BNP Paribas, which consists of 492,702 daily requests for quotation (RFQs) between 2118 institutional clients and 2246 governmental bonds from the G10 countries recorded over 453 days. In addition to their IDs, each client (resp. bond) at time t is represented by the list of their $n = 20$ previous bonds (resp. clients) within a window of the past 100 days. If this list is empty, we replace the history embedding with a zero vector embedding.

To obtain three contiguous sets, D_{train}, D_{val} and D_{test}, we apply a straightforward temporal split to avoid data leakage. The training set encompasses one year, from 01/08/2018 to 31/07/2019; the validation set spans one month, from 01/08/2019 to 31/08/2019; the test set also spans one month, from 01/09/2019 to 30/09/2019.

4.2 Training

We train all models using a symmetrized version of the *Bayesian Personalized Ranking* (BPR) loss [22], defined as follows:

$$\begin{aligned}\mathcal{L}_{BPR} = -(1-p) \sum_{(t,u,i,j)\in D_u} \log(\sigma(\hat{a}^t_{ui} - \hat{a}^t_{uj})) \\ -p \sum_{(t,u,v,i)\in D_i} \log(\sigma(\hat{a}^t_{ui} - \hat{a}^t_{vi}))\end{aligned} \tag{7}$$

where p, the probability of sampling a negative client, is originally set to zero. We set it to 0.5 to give equal importance to both tasks, i.e., recommending clients and products. Let $D = \{(t,u,i) \mid a^t_{ui} = 1\}$ be the set of all observable interactions. Then, $D_u = \{(t,u,i,j) \mid a^t_{ui} = 1, a^t_{uj} \neq 1\}$ is obtained by considering all possible quadruplets (t,u,i,j), such that the triplet $(t,u,i) \in D$, but $(t,u,j) \notin D$. The objective is to learn a personalized ranking, such that each user defines a ranking relation between items. In this case, the user u is more interested in item i than item j at time t. We also define $D_i = \{(t,u,v,i) \mid a^t_{ui} = 1, a^t_{vi} \neq 1\}$ similarly. Finally, we approximate the loss in equation 7 via negative sampling.

Our method employs non-positive uniform sampling with time constraints. Specifically, for every positive sample (t,u,i), we uniformly sample a valid negative item j for D_u if it satisfies two conditions: it is not in the temporal neighbourhood of user u at time t ($j \notin \mathcal{N}_t(u)$), and if the bond has not expired yet. Similarly, a negative user v is considered valid for D_i if $v \notin \mathcal{N}_t(i)$, and if the client is still registered in the catalogue.

To ensure a fair comparison, we train multiple instances of each model with different hyper-parameters. We select the best model based on the lowest validation loss. For training, we use Adam [14] with early stopping to prevent over-fitting. The final evaluation in the subsequent section is performed on the test set.

4.3 Evaluation

After training, the model is used to generate daily recommendations. For each day t, the model produces embeddings for all users and items. The evaluation is carried out on the scoring perimeter, defined as the Cartesian product of all valid users and items on that given day, which encompasses all catalogued clients and unexpired bonds.

Consider a query q that can be either a user u or an item i, for which we aim to provide a list of recommended items or users, respectively. Let $R_q = [r_q^{(1)}, \ldots, r_q^{(l)}, \ldots, r_q^{(k)}]$ denote the recommendation list in descending order of output scores, where l denotes the *rank* position, and k represents the total number of items or users. To measure the quality of recommendations, we use two standard metrics [2]:

- Mean Reciprocal Rank (MRR) defined as:

$$\text{MRR}_t = \frac{1}{\mid Q_t \mid} \sum_{q\in Q_t} \frac{1}{rank(r_q^{(f)})} \tag{8}$$

- Mean Average Precision (mAP) defined as:

$$\text{mAP}_t = \frac{1}{\mid Q_t \mid} \sum_{q \in Q_t} \text{AP}(q) \tag{9}$$

where $r_q^{(f)}$ refers to the first relevant recommendation in the list, and $\text{AP}(p)$ is the average precision. Q_t represents the set of queries on day t, which consists of either the clients who made inquiries or the products that were inquired on that day. The two query sets lead to user-side and item-side scores for each metric. To evaluate the model's effectiveness in providing relevant recommendations for both clients and risk managers, we use the symmetrized MRR (s-MRR) and mAP (s-mAP) scores [1], defined as the harmonic mean of user- and item- MRR and mAP, respectively.

5 RESULTS

Table 1 reports the average performance of all models during the test period. The results demonstrate that our proposed method (ACF) outperforms all other baselines: The best model, ACF-LE, shows substantial improvements in both metrics - with gains of up to 16.86% in s-mAP and 12.65% in s-MRR compared to HCF. ACF-LP and ACF-ATT follow with second and third-best scores, with only ACF-GRU underperforming the benchmark. One possible interpretation is that recurrent-based models are not well-suited to handle sequences with variable time intervals between actions [33].

Moreover, The performance gap between HCF and LightGCN can be attributed to the choice of an appropriate combination layer for the final embeddings. Indeed, HCF combines the static and history embeddings using two stacks of convolutional layers, which proves to be a more effective strategy than LightGCN's approach. In turn, both models outperform MF, which disregards the history blocks. It suggests that the collaborative signal carried in the first-order connectivity of the graph is valuable for the recommendation task.

The daily evaluation shown in Figure 2 reveals that most graph-based models maintain a relatively stable performance throughout the test period, except for ACF-GRU and LightGCN. These models exhibit a decrease in performance as we move away from the training period, showing their inability to cope with evolving interests, possibly due to the inappropriate aggregation module (GRU) or the ineffective combination layer (LightGCN). As a result, their performance follows the same trend as MF, which only uses static embeddings as input. The history blocks allow the model to update the latent representations by incorporating new incoming interactions at inference time. As a result, models equipped with history blocks can maintain a high level of performance over prolonged periods, requiring less frequent retraining. Notably, our proposed time-encoding strategy consistently outscoring all other baselines, demonstrating its superiority in providing reliable recommendations that capture evolving interests in dynamic settings.

Table 1. Symmetrized MRR (s-MRR) and mAP (s-mAP) are reported in percentage. The Gain row shows the percentage improvement of the best model (**highlighted in bold**) compared to the benchmark (underlined).

Model / Metric	s-MRR (%)	s-mAP (%)
POP	1.08	0.64
MF	17.78	13.18
LightGCN	22.13	16.01
HCF	27.28	20.35
ACF-GRU	18.80	13.72
ACF-ATT	28.28	21.54
ACF-LP	30.54	23.59
ACF-LE	**30.73**	**23.78**
Gain (%)	12.65	16.86

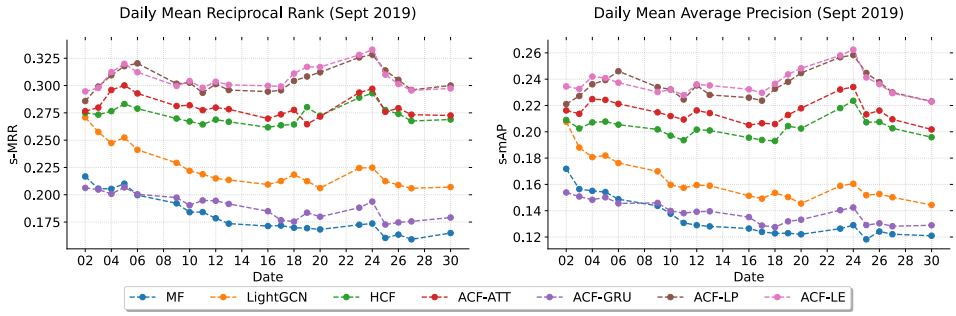

Fig. 2. Daily evolution of the evaluation metrics during the test period, shown for (left) s-MRR and (right) s-mAP.

To analyze the main components of our model, we compare the impact of different types of aggregation and neighbourhood sizes on the model's performance in Table. 2. We summarize the following observations:

- Larger neighbourhood sizes consistently correlate with better performance, as observed by the general trend of increasing s-MRR and s-mAP values as n increases. Interestingly, even when using a small history size (e.g. $n = 5$), any model with a history block surpasses MF, suggesting that using the first-order interactions to enrich the embeddings boosts the performance of collaborative filtering.
- The choice of a time kernel is crucial. Our experimental results show that an exponential kernel is highly sensitive to decay rates, as seen in the stark difference between ACF-LE and ACF-E. Conversely, a power law with fixed decay performs nearly as good as personalized decays, as shown by the slight difference between ACF-LP and ACF-P. One possible explanation could be that interest decay might be relatively uniform across users and items in our dataset and can be adequately modeled with a power law.
- Our experiments show that time-decayed approaches outperform attention mechanisms with time embeddings, although both perform better than a simple average (HCF). The performance gap between the two can be attributed to the following reasons: (1) the history sequences exhibit significant variability with many gaps and missing values, which can hinder the effectiveness of attention mechanisms; (2) our dataset is limited in size, whereas these models require large datasets to capture the full range of patterns and dependencies. Moreover, attention is typically employed to highlight the most relevant interactions in a sequence, whereas our approach assumes all history to be important but with decreasing utility over time.

The results of this section are twofold. First, incorporating a history block in recommendation models to leverage the first-order interactions can enhance collaborative filtering. Second, selecting suitable aggregation functions, such as time-decayed functions, is crucial for capturing the varying utility of historical data and achieving optimal performance.

6 CONCLUSION

In this paper, we introduce ACF, a novel time-dependent collaborative filtering algorithm for capturing the dynamic signal of user-item interactions. We argue that personalizing the varying utility of the interactions using time is crucial for recommendation tasks. Furthermore, experimental results show the effectiveness of time-decayed approaches compared to more complicated aggregations, such as attention and recurrent-based models. This work represents an initial attempt to integrate time explicitly into the propagation step of dynamic graphs. In future work, we plan to

Table 2. Evaluation Metrics for different aggregation functions (vertically) and history sizes n (horizontally). For $n = 0$, all models collapse to an MF and hence share the same performance. The best model is **highlighted in bold**

	s-MRR (%)					s-mAP (%)				
Model / History size	$n = 0$	$n = 5$	$n = 10$	$n = 15$	$n = 20$	$n = 0$	$n = 5$	$n = 10$	$n = 15$	$n = 20$
HCF	17.78	19.98	23.14	24	27.28	13.18	14.52	17.22	18.05	20.35
ACF-GRU	17.78	17.39	18.12	18.24	18.80	13.18	12.69	13.28	13.47	13.72
ACF-ATT	17.78	22.72	24.65	25.97	28.28	13.18	17.32	18.11	19.68	21.54
ACF-P	17.78	24.29	26.01	27.55	30.24	13.18	18.5	19.97	21.37	23.49
ACF-E	17.78	18.32	20.40	21.83	28.04	13.18	13.76	15.11	16.40	21.54
ACF-LP	17.78	**24.85**	**26.39**	**28.43**	30.54	13.18	**19**	**20.37**	**21.93**	23.59
ACF-LE	17.78	23.78	25.56	27.18	**30.73**	13.18	18.01	19.59	20.58	**23.78**

investigate more design choices for time-aware propagation with higher-order connectivity and extend our approach by incorporating dynamic attributed graphs, which involve nodes whose features change over time.

REFERENCES

[1] Baptiste Barreau and Laurent Carlier. 2020. History-Augmented Collaborative Filtering for Financial Recommendations. In *Fourteenth ACM Conference on Recommender Systems*. ACM, Virtual Event Brazil, 492–497. https://doi.org/10.1145/3383313.3412206

[2] Stefan Büttcher, Charles Clarke, and Gordon V. Cormack. 2010. *Information Retrieval: Implementing and Evaluating Search Engines*. The MIT Press.

[3] Pedro G. Campos, Fernando Díez, and Iván Cantador. 2014. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction* 24, 1 (Feb. 2014), 67–119. https://doi.org/10.1007/s11257-012-9136-x

[4] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1724–1734. https://doi.org/10.3115/v1/D14-1179

[5] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems* (Boston, Massachusetts, USA) *(RecSys '16)*. Association for Computing Machinery, New York, NY, USA, 191–198. https://doi.org/10.1145/2959100.2959190

[6] Eduardo José de Borba, Isabela Gasparini, and Daniel Lichtnow. 2017. Time-Aware Recommender Systems: A Systematic Mapping. In *Human-Computer Interaction. Interaction Contexts*, Masaaki Kurosu (Ed.). Springer International Publishing, Cham, 464–479.

[7] Yi Ding and Xue Li. 2005. Time weight collaborative filtering. ACM, Bremen Germany, 485–492. https://doi.org/10.1145/1099554.1099689

[8] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 1025–1035.

[9] Ayat Yehia Hassan, Etimad Fadel, and Nadine Akkari. 2022. Exponential Decay Function-Based Time-Aware Recommender System for e-Commerce Applications. *International Journal of Advanced Computer Science and Applications* 13, 10 (2022). https://doi.org/10.14569/IJACSA.2022.0131071

[10] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, YongDong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 639–648. https://doi.org/10.1145/3397271.3401063

[11] Xiangnan He, Zhankui He, Jingkuan Song, Zhenguang Liu, Yu-Gang Jiang, and Tat-Seng Chua. 2018. NAIS: Neural Attentive Item Similarity Model for Recommendation. *IEEE Transactions on Knowledge and Data Engineering* 30, 12 (dec 2018), 2354–2366. https://doi.org/10.1109/tkde.2018.2831682

[12] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web* (Perth, Australia) *(WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 173–182. https://doi.org/10.1145/3038912.3052569

[13] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *2008 Eighth IEEE International Conference on Data Mining*. 263–272. https://doi.org/10.1109/ICDM.2008.22

[14] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[15] Yehuda Koren. 2008. Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Las Vegas, Nevada, USA) *(KDD '08)*. Association for Computing Machinery, New York, NY, USA, 426–434. https://doi.org/10.1145/1401890.1401944

[16] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '09)*. Association for Computing Machinery, New York, NY, USA, 447–456. https://doi.org/10.1145/1557019.1557072

[17] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37. https://doi.org/10.1109/MC.2009.263

[18] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. ACM, Houston TX USA, 322–330. https://doi.org/10.1145/3336191.3371786

[19] X. Li, M. Zhang, S. Wu, Z. Liu, L. Wang, and P. S. Yu. 2020. Dynamic Graph Collaborative Filtering. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE Computer Society, Los Alamitos, CA, USA, 322–331. https://doi.org/10.1109/ICDM50108.2020.00041

[20] Tongtong Liu, Wenming Ma, and Yulong Song. 2020. Deep Time-Aware Matrix Factorization. In *2020 13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. 952–956. https://doi.org/10.1109/CISP-BMEI51763.2020.9263503

[21] O. Nasraoui, Jeff Cerwinske, Carlos Rojas, and Fabio A. González. 2007. Performance of Recommendation Systems in Dynamic Streaming Environments. In *SDM*.

[22] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (Montreal, Quebec, Canada) *(UAI '09)*. AUAI Press, Arlington, Virginia, USA, 452–461.

[23] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (Marina Del Rey, CA, USA) *(WSDM '18)*. Association for Computing Machinery, New York, NY, USA, 565–573. https://doi.org/10.1145/3159652.3159656

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[25] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z. Sheng, Mehmet A. Orgun, and Defu Lian. 2021. A Survey on Session-Based Recommender Systems. *ACM Comput. Surv.* 54, 7, Article 154 (jul 2021), 38 pages. https://doi.org/10.1145/3465401

[26] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet Orgun. 2019. Sequential Recommender Systems: Challenges, Progress and Prospects. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Macao, China, 6332–6338. https://doi.org/10.24963/ijcai.2019/883

[27] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*. Association for Computing Machinery, New York, NY, USA, 165–174. https://doi.org/10.1145/3331184.3331267

[28] Lingfei Wu, Peng Cui, Jian Pei, and Liang Zhao. 2022. *Graph Neural Networks: Foundations, Frontiers, and Applications*. Springer Singapore, Singapore. 725 pages.

[29] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based Recommendation with Graph Neural Networks. In *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence* (Honolulu, HI, USA), Pascal Van Hentenryck and Zhi-Hua Zhou (Eds.), Vol. 33. AAAI Press, 346–353. https://doi.org/10.1609/aaai.v33i01.3301346

[30] Liang Xiong, X. Chen, Tzu-Kuo Huang, Jeff G. Schneider, and Jaime G. Carbonell. 2010. Temporal Collaborative Filtering with Bayesian Probabilistic Tensor Factorization. In *SDM*.

[31] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen. 2017. Deep Matrix Factorization Models for Recommender Systems. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Melbourne, Australia, 3203–3209. https://doi.org/10.24963/ijcai.2017/447

[32] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, Yike Guo and Faisal Farooq (Eds.). ACM, 974–983. https://doi.org/10.1145/3219819.3219890

[33] Yu Zhu, Hao Li, Yikang Liao, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. 2017. What to Do next: Modeling User Behaviors by Time-LSTM. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (Melbourne, Australia) *(IJCAI'17)*. AAAI Press, 3602–3608.