Modelling Time Delta

in User-Item Interactions Using Deep Recommender Systems

by

Norman Knyazev

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday April 23, 2020 at 14:00.

Student number:	4627083	
Project duration:	March 12, 2018 – April 23, 2020	
Thesis committee:	Prof. dr. Alan Hanjalic,	TU Delft, supervisor
	Dr. Julián Urbano Merino,	TU Delft
	Dr. Jan van Gemert,	TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Abstract

Many widely used Recommender System algorithms estimate user tastes without accounting for their evolving nature. In recent years there has been a gradual increase in methods incorporating such temporal dynamics through sequential processing of user consumption histories. Some works have also included additional temporal features such as time stamps and intervals between a given user's interactions with the platform. The latter, in particular, may be a strong signal providing additional context with respect to the current user preferences. However, in previous works this source of information has only been used passingly, without any significant analysis of its impact on recommendation. In this thesis we examine the effects of such intervals, termed *time gaps*, on recommendation accuracy. In order to do so, we propose a family of novel *Deep-TimeDelta* models, extending a state-of-the-art sequential Recurrent Neural Network based recommender. Through the comparison of our time-dependent models to the sequential baseline we demonstrate that the use of time gaps leads to improvements in recommendation performance, in particular for cases following longer user inactivity. Furthermore, we examine the mechanisms regulating the model recommendation behaviour. Our results suggest that the above performance improvements may be achieved through increased reliance on user long term preferences as well as strong regulation of the importance of the recently consumed items. Finally, we examine the performance differences for users groups with distinct consumption behaviours, demonstrating some improvement for groups featuring less active users as well as users consuming more popular content.

Acknowledgements

I would like to thank Professor Alan Hanjalic for his unrelenting support, invaluable feedback and gentle guiding towards the finish line. I would also like to express my sincerest gratitude to my *de facto* daily supervisor Jaehun Kim, with whom we've spent way too many hours on too many occasions discussing and debating various aspects of my research. I've enjoyed our conversations on everything from recommendation and deep learning to people getting twitchy at craft coffee festivals. Moreover, I am grateful to all other members of the Multimedia Computing staff for their ideas, encouragement and accepting me as one of their own. Furthermore, I want to thank all members of the Data Intelligence team at RTL Netherlands, and especially Dr Daan Odijk and Mateo Gutierrez Granada, from whom I've learned so much about the practical side of research. Finally, I would like to thank my friends, family and my other half, Francesca, for being by my side during this long sail and supporting me at every moment.

Norman Knyazev
Delft, April 2020

Contents

1 Introduction

In modern age, users are faced with more choice than ever. Many online services, such as on-demand entertainment platforms, news portals and e-commerce websites have vast catalogues of content at the users' disposal. With the amount of content added every day it has become progressively more difficult for an individual to be familiar with all the options available to them [47, 59]. This overwhelming choice has been previously shown to have a negative impact on the user's ability to make a good choice as well as the overall user experience [16, 47, 59]. As a result, Recommender Systems (RS) have been proposed as a potential solution to this problem, providing each user with a tailored selection of items deemed to be the most relevant for them. Nowadays, RS are an important component of many popular services such as Pandora[1], Netflix[2] and Spotify[3].

A large number of widely used RS methods rely on a given user's consumption history to estimate what other items may be relevant to them [6, 33, 80, 89, 108, 142]. Items that are predicted to be the most preferred by that specific user can then be presented to them in the form an ordered list of recommendations. Importantly, throughout the recent history of RS, most of the commonly used algorithms have approached the task of recommendation as a static problem [40, 89, 97]. For those models, separate instances of a customer consuming pieces of content tend to be processed independently from one another and without any consideration for the order in which the items were consumed. However, while these methods yield a satisfying performance and often boast good computational efficiency, one may argue that the problem of recommendation is inherently dynamic - users do not consume all items at once but instead sequentially, with their tastes changing ever so slightly as the time passes. Intuitively, when predicting items that may be relevant to the user today, the item the user consumed yesterday may be more predictive of their current needs than an item consumed a long time ago. As such, it may be important for the model to be able to assign varying levels of importance to different items in the user's history based on the recency of the interaction.

In the view of the above, an increasing number of authors have started incorporating different forms of temporal information into their models [32, 33, 50, 108]. For instance, a number of authors have chosen to cast recommendation as a sequence prediction problem [48, 53, 114]. Given a list of items consumed by a single user ordered by the interaction time, their aim is to predict the subsequent item at each point in the list. Popular approaches include Hidden Markov Models [48, 114, 123] and, more recently, Recurrent Neural Networks [33, 52, 53, 108]. Other approaches, alternatively, include the interaction time in their otherwise static models [32, 50, 68, 79]. Different forms of including time as an additional feature are thought to allow models to account for the gradual changes in popularity of items, continuous shifts in user tastes over time as well as the effects of periodicity [79]. These algorithms have been shown to perform particularly well under custom evaluation schemes, such as the prediction of masked ratings during the Netflix prize competition [80]. However, alternative evaluation designs have also been proposed, allowing only the user's past rating information to be used for the prediction of the future ratings [20]. Some authors suggest that under such more realistic scenarios these algorithms may struggle to accurately predict the future user behaviour, potentially rendering them less valuable in practice [86].

[1] pandora.com
[2] netflix.com
[3] spotify.com

Notably, there exists another potential source of temporal information not leveraged by the above works. Intervals between consumption events, *time gaps*, may be an important source of information allowing to further improve the performance of sequential models. For instance, when a user listened to two tracks directly one after another, in many cases the first may be a relatively good predictor of the second. On the other hand, in case the interval between two songs is longer (e.g. a month), intuitively, the expected level of dependency between the two tracks may be lower. Moreover, as the user's tastes may have changed during this period, the second song may be much more more representative of what is relevant to the user now. Conversely, if the user instead listened to the first track only for a few seconds before switching to the next one, the duration of the interval may indicate that the first song was actually not relevant to the user.

Simple sequential models may fail to capture the above dynamics as they do not have access to temporal information that would allow them to do so. Including time gaps in an otherwise sequential model may, on the other hand, allow the model to learn the effect different time gaps may have on estimating shifting user preferences. Interestingly, the few works in literature that do incorporate time gaps either address their effects on recommendation passingly or do not discuss them at all [26, 63].

1.1. Research Questions and Scientific Contributions

We believe that time gaps may lead to improvements in prediction accuracy of recommender systems over existing models. Moreover, we are interested in understanding how the models would leverage this temporal information in order to do so. Motivated by the current gaps in research, we aim to answer our overarching research question:

Research Question: Are time gaps a useful source of information for improving recommendation accuracy in sequential recommendation?

We propose to answer this question by focusing on three key aspects. Firstly, time gaps may be included in a recommender model in various ways and thus serve different roles in recommendation. We are interested in determining which of those uses may improve the estimation of user preferences. As such, our first Research Sub Question is as follows:

Research Sub Question 1: Which modes of incorporating time gaps into a sequential model lead to an improvement of recommendation accuracy?

Secondly, we hypothesise that the implicit meaning of time gaps and how they affect recommendations may vary depending on their value. For instance, it may be possible that certain values of time gaps may be treated as a positive signal while some may be seen as negative. Thus, we are interested in determining the answer to the following question:

Research Sub Question 2: How are different values of time gaps interpreted by the model and what effects do they have on recommendation accuracy?

Moreover, it is hypothesised that users consuming content with different frequency may benefit from inclusion of temporal information to a different extent. For instance, increased preference uncertainty for the less active users may be potentially captured and accounted for through the leveraging of time gaps. Furthermore, users with distinct tastes may also interact with the platform in a different fashion from one another. Overall, it is of interest to examine the benefit of including temporal information for different users, raising the third and final Research Sub Question:

Research Sub Question 3: Are time gaps beneficial in predicting the behaviour of specific groups of users?

In order to address the above questions we propose a family of novel sequential time gap models. We then contrast their performance in an offline evaluation study against a state-of-the-art sequential neural network model, on which our model is based. Following that, we explore the model performance as well as differences in model behaviour over smaller subsets of interactions associated with non-overlapping ranges of time gap sizes. Finally, model performance is compared on subsets of users with distinct temporal behaviour and content preferences.

Overall, the scientific contributions of this work are as follows:

1. We introduce a novel time gap based recommendation algorithm along with its implementation.[4]
2. We perform an analysis of the benefits to the recommendation accuracy provided by the time gaps. Moreover, we examine and interpret how distinct values of time gaps affect the process of recommendation.

This thesis is structured as follows. The Background section of Chapter 2 first provides context on Recommender Systems and Collaborative Filtering. This is followed by additional information on static, sequential and time-dependent recommender models. Finally, we provide relevant information on neural networks, which form the basis of our model as well as of our baseline. Following that, sequential and temporal neural network based models are described in the Related Work section of Chapter 2. From then onwards, Chapter 3 describes our proposed set of models as well as the motivation behind them. The chapter also introduces the rationale behind the experiments performed to address our overarching Research Question. Chapter 4 describes the specifics of the experimental setup, including the datasets used, data preprocessing, model parameter selection and evaluation of the results. Subsequently, experimental results, along with our analysis, are presented to the reader in Chapter 5. We then provide our conclusions, discuss the limitations of our approach and explore potential future work in Chapter 6. Finally, as the offline study may not fully reflect the recommender's effect on the end user's satisfaction, an online study on the user data from a video-on-demand platform Videoland[5] was originally envisaged. However, due to challenges stemming from the implementation of the experiment no significant conclusions can be reported. Ideas underlying the online experiment, along with information on its setup, the observed results as well as a more detailed description of the aforementioned difficulties and their impact on the progression of the thesis are included in the Appendix.

[4] www.github.com/NKNY/DeepTimeDelta
[5] www.videoland.com

2

Background and Related Work

2.1. Aim of recommender systems

The rapid growth of the internet, e-commerce and entertainment industries has led to a surge in the amount of content available for consumption. While this growth may initially seem to be a positive phenomenon for the consumer, it also comes with its challenges. Work by Iyengar and Lepper [59] suggests that having to select an item from a large number options may actually decrease the customer's motivation to make that choice. This effect has been termed *choice overload*. On the other hand, a study by Häubl and Trifts [47] demonstrated that having an interactive system provide the user with a tailored subset of items led to the user being able to make better choices while expending less mental effort. Taken together, the above results suggest that in situations featuring an excessive number of options it is important a user be provided with a pre-filtered bespoke list of items. Methods employed for this task are collectively known as Recommender Systems (RS). Nowadays RS are integral components of many businesses, including Pandora, Netflix and Spotify. For a comprehensive list of benefits RS provide both to the user as well as the service platform we urge the reader to consult Ricci et al. [115].

2.2. Basic problem setting

A recommender system, recommendation system or simply a recommender, is an algorithm or a technique that assists a user in satisfying their informational need [92]. This may, for instance, come in the form of suggesting a product, an article or a song that is the most appropriate to the user at the time they interact with the system. In the classical definition, a recommender model aims to identify item or items i from the set of all items I which would lead to the highest satisfaction $R_{v,i}$ of user v. Note that the above term *satisfaction* is subjective and thus difficult to measure in practice. In the early RS works authors often used ratings assigned to items as a measure of their relevance to a given user v [29, 54, 77, 78, 125]. The aim of such works is to predict the rating $\hat{r}_{v,i}$ that v would assign to a previously unseen item i. Learning the user's preferences is done by leveraging the available ratings $r_{v,i}$ that the user had previously entered on the platform. The quality of rating predictions produced by such models is often evaluated using Root Mean Squared Error (RMSE) on a withheld set of existing ratings. At recommendation (prediction) time the model is queried for items previously not consumed by v for which $\hat{r}_{v,i}$ is the highest.

On the other hand, recommendations are generally presented to the user in the form of an ordered list reflecting the estimated relative preference of items, without the direct need for the calculation of the exact ratings. As such, more recently, authors have found success in skipping the rating prediction altogether and instead directly focusing on ranking the items [4, 61, 70]. This approach is particularly useful when the rating information is not available and is discussed in Subsection 2.4.2.

2.3. Types of recommender systems

There has been a large body of work approaching the problem of recommendation from different angles. Burke [18] lists four main types of recommender systems: collaborative, content-based, demographic and knowledge-based. More recent types of RS also include social RS [42] and hybrid models [19]. While there

have been developments in all the directions of recommendation, the largest amount of research effort has gone into content-based approaches and collaborative filtering [30, 81].

Content-based approaches attempt to identify certain underlying characteristics of items and then match them to users who exhibit affinity for those characteristics. In practice, this means that a content-based model is often attempting to find items most similar to the ones that have previously been relevant to the user (e.g. assigned a high rating). For example, if user positively rated a few romantic comedies, a content-based model may identify the user's preference for this type of movies. It may then try recommend other items whose metadata (e.g. genre, actors, language, keywords) are similar to the previously consumed items. However, the main limitation of such approaches is the difficulty of extracting meaningful features from the content. It has been previously shown that only using content metadata may insufficient for producing high quality recommendations [73]. Moreover, this dependence often makes content-based approaches rely on custom crafted features [90], making them non-generalisable across different recommendation domains e.g. books and music.

2.4. Collaborative Filtering

The other widespread type of models instead leverages the aggregated feedback from multiple users. For this reason, these methods are collectively termed Collaborative Filtering (CF). They are often the go-to models as they can be used without the need to extract and process any content-related features.

In the previously described recommendation setting, all user-item interactions can be summarised using a user-item matrix. An example of such matrix is presented in the left part of Figure 2.1. CF models aim to leverage the information captured by such matrix in order to provide relevant recommendations. The classical paradigm is that CF models can be split into *neighbourhood-based* (also known as *memory-based*) and *model-based* methods [30]. The former can be further split into *user-based* and *item-based* neighbourhood methods [30]. Briefly, given a user u, user-based neighbourhood methods attempt to identify other users whose consumption patterns (e.g. ratings) are similar to those of u. They then aggregate the information from k (hyperparameter) such neighbours in order to identify what other items may be relevant to u. Item-based neighbourhood methods, on the other hand, employ a variation on the above approach. Given a set of items I_u consumed by u, for each item $i \in I_u$ these methods calculate the similarity of i to all other items in the dataset. An example of such similarity metric is cosine similarity of two vectors where each pair of entries is ratings from a user who has consumed both items [119]. These similarities are then employed to recommend other items similar to those already previously consumed by u. For a more comprehensive overview of neighbourhood-based methods we suggest the work of Desrosiers and Karypis [30].

Neighbourhood models are known for their ability to identify localised relations but struggle with identifying global underlying trends [78]. As an example, for a given user who previously watched Brave, Moana and Kill Bill, a likely set of recommendations under a neighbourhood model would be (Disney) animations and Tarantino-esque movies. On the other hand, one may argue that a successful model must also learn to infer the underlying concepts defining the user preferences (e.g. strong lead female characters) and recommend accordingly, in effect looking past direct neighbours for recommendation.

2.4.1. Latent factor models

The recent years, following increases in available computational power, have seen a rise in the use of the so-called *model-based* or *latent factor* approaches [10, 43, 79, 108]. Instead of selectively focusing on parts of the user-item matrix as with neighbourhood methods, all the information in the matrix is used in full. The success of these models is illustrated by the work of Koren [80] winning the Netflix prize, providing 10% lower RMSE compared to Netflix's own algorithm at the time. Paired with the availability of computationally efficient implementations, the *model-based* algorithms are widely adopted in industry [25, 58, 97, 113].

Collectively, latent factor methods attempt to model users and items as k-dimensional feature vectors. In this formulation, each value in the item vector symbolises the extent to which the item possesses some unknown (latent) characteristic. Similarly, each respective value in the user vector denotes the user's preference (or a lack thereof) for the above characteristic. This notion is shown in Figure 2.1. The reader may note that this is conceptually similar to the idea presented in section 2.3 in the context of Content-based RS. Whilst latent features are difficult to interpret, they are learned on the basis of aggregated user consumption patterns and as such do not rely on content processing.

A number of seminal works in the area of latent factor methods draw inspiration from Single Value Decomposition (SVD) [10, 78, 107, 134]. This technique has been pivotal in the fields of Information Retrieval

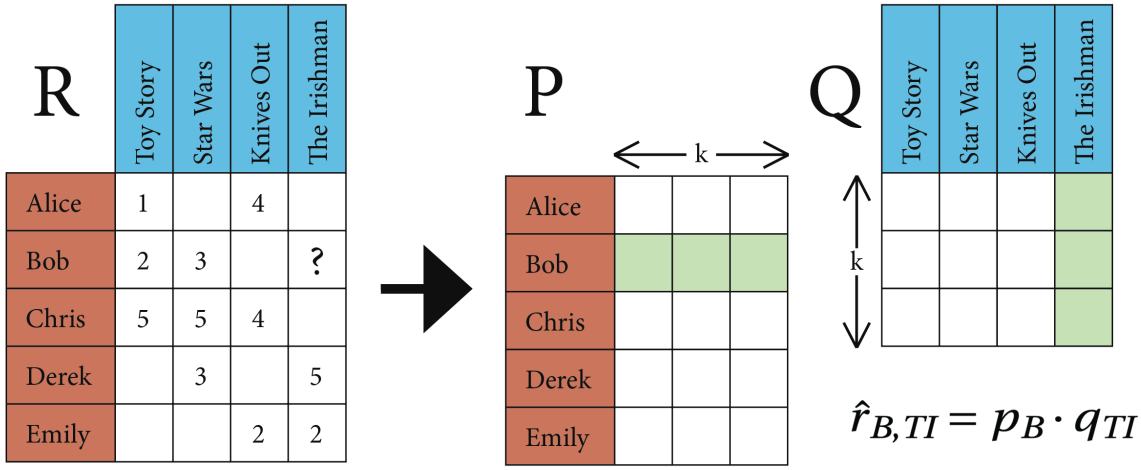


Figure 2.1: Singular Value Decomposition (SVD) of the user-item matrix. An individual entry in the rating matrix R denotes the rating assigned by the row's user to the column's item. Missing values denote that a given user has not rated the chosen item. An unknown user-item rating combination (?) can be estimated by decomposing R into a user matrix P and an item matrix Q. The inner dimension of the two resulting matrices, denoting the dimensionality of the resulting latent user and item (embedding) vectors, is a hyperparameter k. The missing rating is calculated as the dot-product of the latent vectors belonging to the user and the item in question.

[14] and Machine Learning [128]. User-item matrices, however, tend to be very sparsely populated as users tend to interact with only a small proportion of available items, making the data not amenable to SVD [84]. While it is possible to impute the missing values as described in [27, 132], the imputed values are unlikely to be reliable [147]. As such, many authors have instead focused on modelling the user behaviour solely on the basis of existing values [10, 40, 78, 107].

One of the most studied types of latent factor models is Matrix Factorisation (MF). Its most common implementation, popularised by Funk [40] and employing gradient descent, is still often employed in industry due to its scalability and ease of implementation. Under this framework, an $m \times n$ user-item interaction matrix R is decomposed into an $m \times k$ user latent vector matrix P and a $k \times n$ item latent vector matrix Q. Given a user v and an item i one would predict v's preference for i as the dot product between the user embedding vector p_v and the item embedding q_i. The length of these embedding vectors k is a hyperparameter and can be tuned on the validation set or via cross-validation. Notably, higher values of k, both for MF and other models, may lead to improved performance at the expense of longer computation time and memory requirements. However, too high k values may also increase the risk of overfitting [84, 97].

The main difference of this technique from the previously mentioned SVD is that in MF it is possible to learn the user and items embeddings exclusively on the basis of the available ratings. This is done by calculating the prediction loss on those ratings and updating the embeddings based on the extent to which they contributed to the error. This process is then repeated until a sufficiently low disparity between predicted and true ratings is reached. This iterative approach is known as gradient descent - for an overview of this technique we suggest Ruder [117]. Funk's model placed third in the Netflix Challenge and inspired a large body of related work [78, 97, 118, 122].

2.4.2. Implicit ratings and additional features

Whilst relying on user ratings is a convenient proxy for estimating user preferences, this also comes with a number of drawbacks. The main disadvantage of relying on the rating data is the relative scarcity of such feedback. In fact, a number of authors, including those from YouTube, suggest that the overwhelming majority of user interactions often comes from other forms of feedback [25, 51, 66, 113]. These may include click data, user hovering over a given item's thumbnail, textual queries, user watching a trailer for a given movie, among many others. Consumption data (clicks), simply indicating that the user interacted with a given item, is particularly prevalent in the majority of domains [21, 25, 58, 100]. This type of feedback is known as *implicit*. A clear limitation of such input is that the user satisfaction, previously available in the form of ratings, is no longer measured. However, it is likely that the user made a conscious decision to interact with the given item, indicating potential interest in it. Moreover, they chose that item above all other available items, suggesting some form of preference. As such, implicit feedback is crucial for providing an insight into the user's

tastes as well as partially ameliorating the data sparseness encountered when solely relying on ratings.

One of the models able to incorporate such feedback is SVD++ [78]. This approach builds on MF, extending it with additional weights that model different additional types of interactions that the user may have with the item. In their work, Koren demonstrates the superior performance of SVD++ over standard MF on the Netflix dataset. Other works successfully leveraging implicit feedback alongside explicit ratings include [6, 106]. Altogether, the above works underline the importance of utilising implicit feedback for prediction.

Notably, due to the drastic differences in the availability of the data, a number of authors have chosen to focus on modelling user preferences on the basis of only implicit feedback, omitting the dependence on ratings [58, 101, 113, 114]. Note that this task is inherently more complex compared to the rating prediction as there is no clear positive or negative feedback and thus standard regression models that dominated the Netflix challenge cannot be applied [114]. Instead, the problem is often recast as one of predicting the preference directly (bypassing ratings), determining the preference with respect to other items (ranking) or estimating the probability of a given item being consumed next. For example, in their work Hu et al. [58] substitute the rating of the item with the binary preference (or non-preference) for each item and an associated vector displaying their confidence in the estimated preference. Confidence is high if a user consumed the item in full and on multiple occasions, whereas low confidence values tend to be used if the item was only consumed partially. Similar to rating prediction, the authors still aim to learn the user and item embeddings. However, in contrast to MF, the embedding dot-product instead models the binary user preference, with the confidence vector weighting the interaction's contribution to the total loss, reflecting the model's certainty in the correctness of the estimated preference. Rendle et al. [113], on the other hand, propose a model that aims to learn the ranking of items by contrasting each consumed item with a random item not consumed by the user. The authors operate under the assumption that a consumed item is preferred by the user over any unconsumed item. While this assumption is often violated in practice (e.g. user is unaware of an item they would have otherwise liked), the authors, regardless, demonstrate performance comparable to or better than the state-of-the-art at the time. In fact, both of the above approaches have found widespread adoption in industry due to their fast training times and good performance.

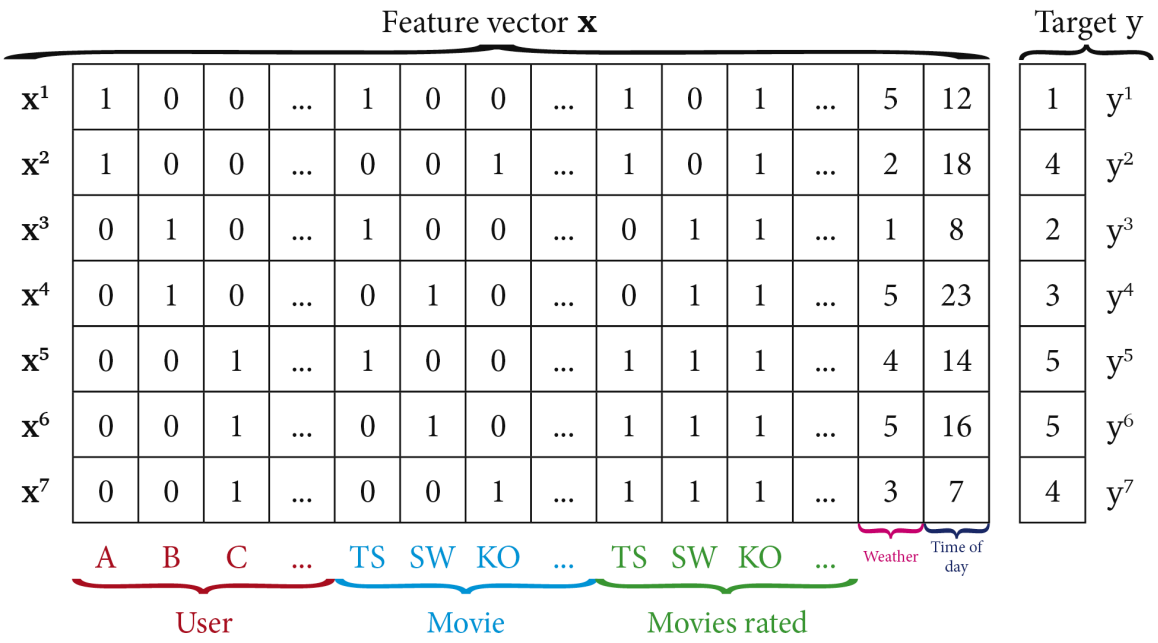
Note that in the formulation of SVD++ all the additional terms involve only various types of implicit feedback associated with the items. As previously alluded, recommendation platforms often have access to additional information related to interactions outside of explicit and implicit feedback. This data may include but is not limited to spatial, temporal and social information. Various works have demonstrated the importance of this additional information in increasing the recommendation quality [2, 6, 63, 91, 112, 144]. For instance, intuitively, it is not unusual for users to watch Christmas movies in the period leading up to the holiday season. However, it is useful for the model to incorporate the understanding of the context in which those movies were watched e.g. as not to attempt to later recommend Christmas movies in summer. Rendle has shown that their framework termed *Factorisation Machines* (FM) is able to successfully leverage many types of such additional information for the task of the rating prediction [112]. FM is conceptually related to MF, albeit now the rating is determined via the strength of interactions not only between the item and user but also the circumstances of interaction. Moreover, similar to MF, Factorisation Machines rely on gradient descent. In the case of FM it is used to find the most effective way of combining the multimodal information of the interaction as to predict the rating assigned by the user to the item. An example of the input and output of a FM model is presented in Figure 2.2.

They key aspect, in addition to allowing for any type of features, leading to the success enjoyed by the FM, is the introduction of interactions between features. Second order FM's (two features interacting at a time) are frequently used due to their simplicity of implementation and good performance [88]. Blondel et al. [15] and Xiao et al. [143] also recently demonstrated that FM's of higher order (3+) may provide even better results on some tasks compared to the lower-order FM's ($1-2$). More recent works by Guo et al. [45] and Loni et al. [89] have also addressed the fact that FM's basic implementation may perform suboptimally when used directly on implicit feedback. With the above modification FM's can thus be seen as a generalisation of both MF and SVD++. Overall, the seminal work by Rendle et al. highlights two key points:

1. Auxiliary information can be included in the model to increase the quality of recommendation.

2. Exploiting more complex relationships between input variables is highly beneficial in improving the model's expressiveness.

However, one limitation of FM, visible in Figure 2.2, is that FM's do not capture certain types of temporal information, for example the order in which items were consumed. That is, rows $\mathbf{x^1}$ and $\mathbf{x^2}$ may be switched



$\mathbf{x}^1$	1	0	0	...	1	0	0	...	1	0	1	...	5	12	1	y^1
$\mathbf{x}^2$	1	0	0	...	0	0	1	...	1	0	1	...	2	18	4	y^2
$\mathbf{x}^3$	0	1	0	...	1	0	0	...	0	1	1	...	1	8	2	y^3
$\mathbf{x}^4$	0	1	0	...	0	1	0	...	0	1	1	...	5	23	3	y^4
$\mathbf{x}^5$	0	0	1	...	1	0	0	...	1	1	1	...	4	14	5	y^5
$\mathbf{x}^6$	0	0	1	...	0	1	0	...	1	1	1	...	5	16	5	y^6
$\mathbf{x}^7$	0	0	1	...	0	0	1	...	1	1	1	...	3	7	4	y^7

Figure 2.2: Factorization machines can be used to incorporate a wide range of inputs. Data is presented to the model as set of tuples ($\mathbf{x}^{\mathbf{i}}$, y^i). Every aspect of the interaction is encoded as $\mathbf{x}^{\mathbf{i}} = (x_1, x_2, ..., x_n)$ using numeric or indicator variables, whilst y^i is the rating assigned by the user for that particular interaction. For each interaction the user and the movie from Figure 2.1 encoded using indicator variables alongside the list of all items rated by the given user. The weather on a scale from 1 to 5 as well as the hour of interaction denoted using integers. Adapted from Rendle [112].

and FM's global optimum would be unchanged. As such, while FM's can process a wide array of features, some dependencies between inputs may still be handled suboptimally.

2.5. Temporal dynamics

The approach employed by MF and other standard model-based methods can be viewed as modelling the user's overall *long-term* preferences [48]. The main disadvantage of such approaches is that even the most drastic recent changes in user behaviour may be ignored by the model, especially if the recent interactions only make up a small percentage of the user's history. Intuitively, the user's most recent interactions may reflect the user's current needs more than their interactions a long time ago. As such, some authors have chosen to view the items consumed by the users not as a randomly ordered sets but instead as a sequences, with the aim of modelling the users' *short-term* preferences [33, 48, 114, 129].

There appears to be no consensus in the literature in terms of the naming convention for different temporal dynamics. According to Shi et al. [126], *time-dependent* models exploit time as a continuously changing feature (e.g. Unix epochs, days since a particular date etc.), whereas *time-aware* models would instead use only the cyclic aspect of the temporal information (e.g. day of the week). On the other hand, Campos et al. [20] use the term *time-aware* to refer to the first of the above definitions, while Du et al. [34] refer to the same concept as *time-sensitive*. Finally, Donkers et al. [33] refer as *time-dependent* to the approaches that use the ordering of the interactions, disregarding the actual time stamps and timespans between interactions, while Campos et al. [20] classify such approaches as *time-adaptive*.

In this work, the following nomenclature is used:

- *Time-aware* models use temporal information as a temporal context for when a certain interaction happened or what context a prediction must be made for (Saturday, summer, Christmas). No distinction is made between the same phase of two successive cycles (Christmas 2014 and Christmas 2015).
- *Sequential* models may use raw temporal information (e.g. time stamps) to establish the order of interactions. However, no information conveying the time of the interaction or the time elapsed between any two interactions is retained. As such, interactions are treated purely as an ordered sequence.
- *Time-dependent* models use the raw temporal information or its derivatives (binned time stamps, time

gaps), allowing the model to determine the time of an interaction, the relative order of two interactions or the time elapsed between them. Time is not viewed as cyclical and thus two successive Friday nights are viewed as distinct inputs.

This work will focus on contrasting *sequential* and *time-dependent* approaches. *Time-aware* models, on the other hand, are often regarded as a subtype of context-aware recommender systems [2]. As many works have previously demonstrated the importance of leveraging (temporal) context in recommendation, the reader is invited to familiarise themselves with [2, 6, 7].

2.5.1. Sequential models

Sequential models expand on the non-temporal models by leveraging the order in which the user consumed the items. Interestingly, many authors do not attempt to predict the item ratings and instead focus on predicting the next item consumed by a given user given their previous items [26, 33, 34, 48, 63].

A natural angle to approach this problem is to model the last chosen item as the current state and the next item chosen as the next state. Markov Chains have been widely used to model such interstate transitions in Natural Language Processing and Information Retrieval [17, 85]. In RS, early works only provided non-personalised time-aware recommendations [123]. These models were later combined with latent factor models to provide personalised recommendations [48, 114].

One limitation of such models is that multiple interactions in a quick succession are treated the same as an equal number of interactions separated by different periods of time. More broadly, while sequential models are able to utilise the implicit ordering in the consumption events, all other temporal information is discarded.

2.5.2. Time-dependent models

Time-dependent models are able to leverage auxiliary temporal information in addition to the implicit ordering. The most common type of such information are the time stamps of interactions. Hermann proposes to use the time between consumption of two items by one user (time gap) as a measure of their similarity [50]. Ding and Li [32], instead, extend an item-based neighbourhood model with a temporal decay factor. In their formulation, items that were consumed further in the past contribute less to the prediction compared to the more recent interactions. Moling et al. [98], Pampalk et al. [103] demonstrated that accounting for premature track switching (defined as a hard coded percentage of the track's duration in [98], any listening time below the track's length in [103]) may lead to improved recommendation performance. Other authors have also attempted to model the user's return to the platform and their recurrent consumption of items via temporal point processes and other statistical models [34, 68, 69].

Koren [79] proposed *timeSVD++*, decomposing the user and item embeddings into several static and temporal components. Firstly, for dynamic components, the authors propose to include factors that model changes in each item's ratings over time. Secondly, the algorithm contains factors that account for changes in each user's ratings over time with one factor per user per day. Finally, Koren and Bell also include factors to model each user's change of preferences over time, similarly maintaining one factor per user per day. The authors suggest that accounting for the dynamic nature of users and items is what allowed them to achieve the lowest RMSE in the Netflix Prize [80].

One may notice that the above model is very intricate and tailored to the Netflix Prize problem. A common criticism of *timeSVD++* is that it does not generalise to other domains [142]. Moreover, as the authors point out themselves [81], the model is not able to actually make future predictions directly. Instead, the authors suggest that their model is able to decompose the user signal into separate long-term and short-term components, allowing to interpolate between the past interactions. However, Lathia et al. [86] suggest that whilst a model's ability to predict missing ratings may be highly successful for the Netflix Prize, this performance does not necessarily transfer to the problem of future rating prediction. Nevertheless, *timeSVD++* demonstrates that using temporal information may be key to leveraging the dynamically changing user behaviour.

Overall, one of the main limitations of the above approaches is that they assume specific distributions for temporal information's contribution to the prediction (exponential in [32] and *timeSVD++*, Hawkes process in [34]). Moreover, work on Support Vector Machines [3] and Factorisation Machines, as discussed in subsection 2.4.2, suggests that models containing complex (e.g. nonlinear) interactions may be more expressive and thus in effect able to capture the fine-grained user preferences. In recent years, a type of models leveraging the above concepts, termed neural networks, has seen a resurgence in various fields, including Recommendation Systems [23, 26, 33, 53, 121].

2.6. Neural Network based models

Artificial Neural Networks (or simply *Neural Networks*) is a class of Machine Learning models characterised by multiple stages of computation and the intertwined use of linear and nonlinear transformations. In a basic neural network, the computation can be visualised as layers, where each layer applies linear and nonlinear transformations on the output of the previous layer. A schematic of the basic implementation of a neural network, *Feed-Forward Neural Network* (*FFNN*) is shown in Figure 2.3.

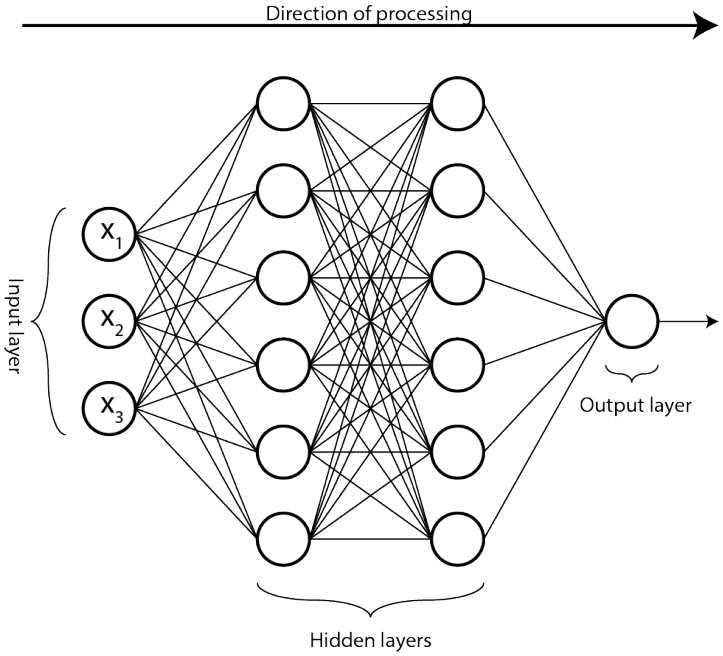


Figure 2.3: Three layer Feed-Forward Neural Network (FFNN). For any hidden layer each of its units is a linear weighted sum of the outputs of the previous layer followed by a non-linear transformation. The output of the final layer is a linear combination of the outputs of the last hidden layer.

In FFNN, an input vector $x \in \mathbb{R}^N$ is provided to the classifier. Each of the N input values $x_1, x_2, ..., x_N$ is then combined as a weighted sum $h_k(x) = \sum_{i=1}^{N} w_{i,k} \cdot x_i + b$, where b_k is known as the bias term. A nonlinear transformation $g(\cdot)$ such as sigmoid, hyperbolic tangent (tanh) or Rectified Linear Unit (ReLU) [99] is then applied to the sum. The combined computation $g(h_k(x))$ is referred to as a unit or a neuron. Each layer in a neural network usually consists of multiple such neurons. In the above example the first *hidden layer*, processing the input, consists of M neurons. Note that two neurons in the same layer usually do not share the weights w and biases b, thus leading to $M \cdot N$ weights and M biases for a particular layer, where N denotes the number of inputs to the layer. The main idea of neural networks is that the output of the first hidden layer can be treated as an input to the second hidden layer, which in turn may be used as input for the third layer etc. These stacked layers may be seen as receiving an input, extracting some underlying features, successively combining them into progressively more sophisticated features. Finally, the features of the last hidden layer's may be processed as a linear combination in the *output* layer in order to make the prediction (e.g. classification, regression). The successive use of linearities followed by nonlinearities is what is thought to allow neural networks to approximate any distribution [57] and perform well in various scenarios, such as object detection [111], speech recognition [28] and sequence prediction [11]. In recent years, there has also been a growing interest in neural networks by the RS community. Various works have approached both the static and dynamic views of recommendation problem [23, 25, 33, 121, 139, 142].

2.6.1. Static Neural Network models

Static neural network based models can be seen as an extension to standard CF-based approaches, such as neighbourhood methods and FM's. For instance, [121] uses an Autoencoder to learn a user-specific mapping for each item based on which users have consumed the item. Autoencoders are a special type of FFNN, where the input vector (e.g. ratings for one item i from each of the $|U|$ users) is passed into a model with one hidden layer and the task of the model is to reconstruct the original input [5]. The hidden layer is thus forced to learn

a mapping to a latent representation for each item, which can then be used to find unavailable ratings at test time, analogous to item-item neighbourhood models, discussed in Subsection 2.4.

Authors from YouTube propose a neural network as a generalisation of MF, highlighting the ease with which any features can be incorporated [25]. In their work the authors jointly learn item embeddings, aggregated static embeddings of user watch and search histories as well as a geographic embedding. Those are then concatenated with additional categorical and continuous features. The combined vector is then passed through a series of forward layers, finally outputting the final user state vector. This vector is then used to extract item probabilities indicating the likelihood of any given item being chosen by the user next. The authors also propose using sampled softmax to alleviate the computational constraint of having to score each item at every training step, as first described in [62]. Employing this approach, the authors demonstrate improved offline metrics and a significant increase in watch time on the platform. The latter is considered to be indicative of the increased user satisfaction, which is the ultimate objective of RS. However, as previously described in Section 2.5, while the above neural network based models may provide an improvement over the classical MF and FM, they still fail to leverage the temporal dynamics present in the data.

2.6.2. Neural Networks for sequential data

Recurrent Neural Networks (RNN's) are a type of neural networks proposed to deal with dynamic input. Unlike the models described in the previous section, RNN's process the input sequentially. These models condition their output for the input at time step T not only on $\mathbf{x}^{(T)}$ but also their inputs at time steps $T-1$, $T-2$, ..., 1. For example, when using day T's measurements ($\mathbf{x}^{(T)}$) to predict weather for day $T+1$ (output at day T), an RNN also takes into account the measurements from all the previous days. In the simple formulation, this historical information is summarised in a *hidden state* vector $\mathbf{h}^{(T-1)}$. As shown in Figure 2.4, the output of the model (weather prediction) $\mathbf{o}^{(T)}$ at time step T depends on our latent (hidden) representation of our knowledge $\mathbf{h}^{(T)}$. This latent vector in turn depends on the input (measurements) $\mathbf{x}^{(T)}$ at time step T as well as our prior knowledge vector $\mathbf{h}^{(T-1)}$, which in turn depends on $\mathbf{x}^{(T-1)}$ and $\mathbf{h}^{(T-2)}$ etc. At each time step the model is able to update its hidden state and output a prediction based on this hidden state. This idea has been widely used in stock price prediction [22], Natural Language Processing [24] and many other fields [8, 104, 109].

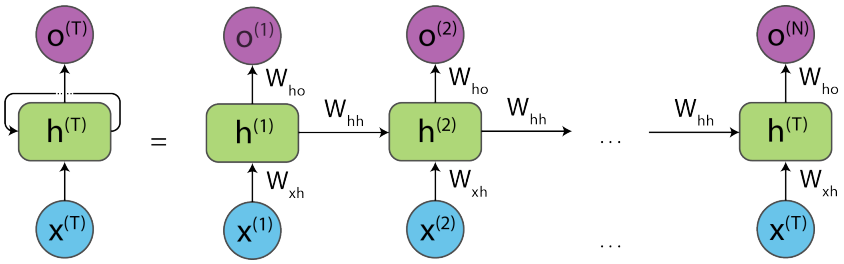


Figure 2.4: Recurrent Neural Network (RNN). The hidden state at time step T depends on the hidden state at time step $T-1$, which in turn depends on the hidden state of $T-2$ and so on. By explicitly denoting each step's dependency on all the previous steps an RNN can be unrolled into a Feed-Forward Neural Network with shared weights across all time steps. Adapted from Olah [102].

The main drawback of the simple version of RNN is the problem known as *vanishing gradient*. Due to the gradient flow dynamics during training such models are often unable to learn long-distance dependencies and can only end up correctly learning the relationship between time step T and a small number of preceding time steps. For example, when training a model to generate a sentence, at the 20$^{\text{th}}$ word the model might output a word consistent with the last five words but that would not make sense when taking into consideration the first six words. An example of such situation is given in Figure 2.5.

I was born in the Netherlands. <additional text> I speak _.

Figure 2.5: Example of vanishing gradient problem. When tasked with predicting the missing word a simple RNN may be able to make a link to the first sentence, given that the gap between the sentences is small enough. But as the gap increases the model is progressively less likely to make that connection.

To ameliorate this issue, Cho et al. [24] proposed a modified RNN architecture termed Gated Recurrent Unit (GRU). An example of one GRU neuron is shown in Figure 2.6. In their implementation, the input $\mathbf{x}^{(T)}$

and the hidden state $\mathbf{h}^{(T)}$ are used to calculate $\mathbf{r}$ (the *reset gate*) and $\mathbf{u}$ (the *update gate*). These gates allow the model to regulate the extent to which the old hidden state $\mathbf{h}^{(T-1)}$ contributes to the new hidden state $\mathbf{h}^{(T)}$. Other Recurrent models proposed to learn the longer-term dependencies include Long Short Term Memory (LSTM) [56] with its various variants [41, 44] as well as Clockwork RNN's [82]. Karpathy [71] suggests that the ability of the above models to learn long-distance dependencies stems from the fact that the gradient can be propagated through many more time steps before dying off. This behaviour, he argues, is due to the summation operation during the update of the hidden state, contrasted with the multiplicative update in RNN. Interestingly, work by Greff et al. [44] suggests that there is little difference in terms of performance between the above models and their variations - for example one may learn only the update gate $\mathbf{u}$ in GRU while setting the reset gate $\mathbf{r}$ to be $1-\mathbf{u}$. Nevertheless, most works opt to use either GRU or LSTM units. This is also true for the constantly increasing number of works, addressing the problem of temporal recommendation using Recurrent Networks [53, 63, 129, 142].

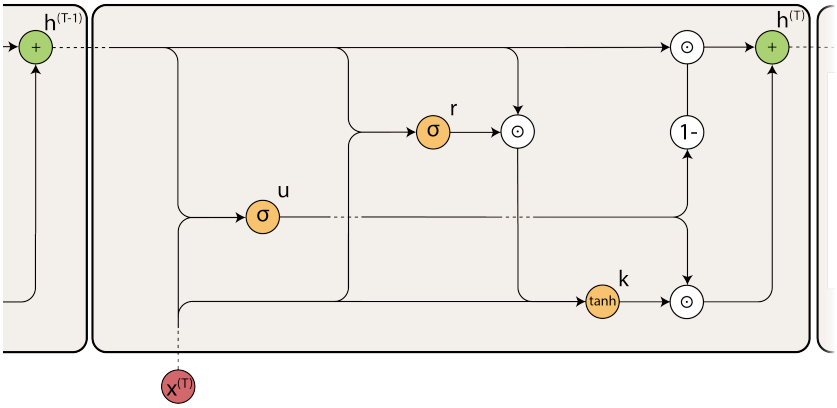


Figure 2.6: Gated Recurrent Unit (GRU) cell. At each point in the sequence the hidden state of the last step is leakily combined with the new input. Joining of arrows denotes concatenation of multiple vectors. Non-linear transformations denoted in orange. $\odot$ denotes element-wise multiplication. Multiplication with weight matrices omitted from the schematic for clarity. Adapted from Olah [102].

2.7. Related Work

2.7.1. Recurrent Neural Networks for sequential recommendation

Recurrent networks are a natural way of tackling the problem of predicting the item to be consumed by the user given the sequence of their past consumption events. The first work employing RNN-based recommendations tackled the problem of session-based recommendation [53]. In their work, the authors attempted to predict the next item in the sequence, given only the interactions of the current browsing session. However, whilst they demonstrate an improvement over their baselines, their approach does not rely on item embeddings and does not provide personalised recommendations outside the scope of the given session. The authors later extended their work with multimodal inputs (images, textual descriptions), highlighting the flexibility of neural networks compared to their predecessors [52].

Pei et al. [108] propose a sophisticated approach combining sequential recommendation with the attention architecture [138], demonstrating improved rating accuracy over their baselines, including MF and timeSVD++. The authors also argue that the attention module allows them to have a more interpretable model, which has been shown to increase user satisfaction [49, 136]. On the other hand, one of the drawbacks of the above model is its general and computational complexity due to having two separate networks to model user and item evolution.

Conversely, the model recently proposed by Donkers et al. [33] aims to learn user and item embeddings jointly in a more straightforward fashion. The authors regulate the influence of both on the recommendations using a recommendation-specific version of GRU. In their formulation, illustrated in Figure 2.7, a gating vector $\boldsymbol{\xi}$ is calculated on the basis of the current interacting user and item as well as the previous hidden state, symbolising the current user preferences. The user and item embedding vectors contributing to the hidden state update are then gated by $\boldsymbol{\xi}$ and $1-\boldsymbol{\xi}$ respectively, allowing the model to selectively balance the influence of the two. For instance, the authors suggest that when the previous item consumed by user v was a first part of the film series (e.g. Toy Story), the prediction may find it beneficial to give more weight to the

item vector. This would be done as it may be likely the user would want to watch the sequel (Toy Story 2) over other available items. Conversely, the model may determine that in another situation an item may be an outlier, shutting down its influence on the recommendation. The function to learn the balance between the item information and the user characteristics is learned during training. Similar to the GRU, the gated user and item vectors are then used to leakily calculate the new hidden state $\mathbf{h}^{(T)}$, which is in turn used to obtain the probability scores for all items. Notably, the gating mechanisms used in the model may also be potentially seen as providing a limited degree of interpretability through the extent of their activations. The authors demonstrate a noticeable improvement over simpler non-temporal and time-dependent models as well as neural network based sequential recommendation models, illustrating the strength of their approach.

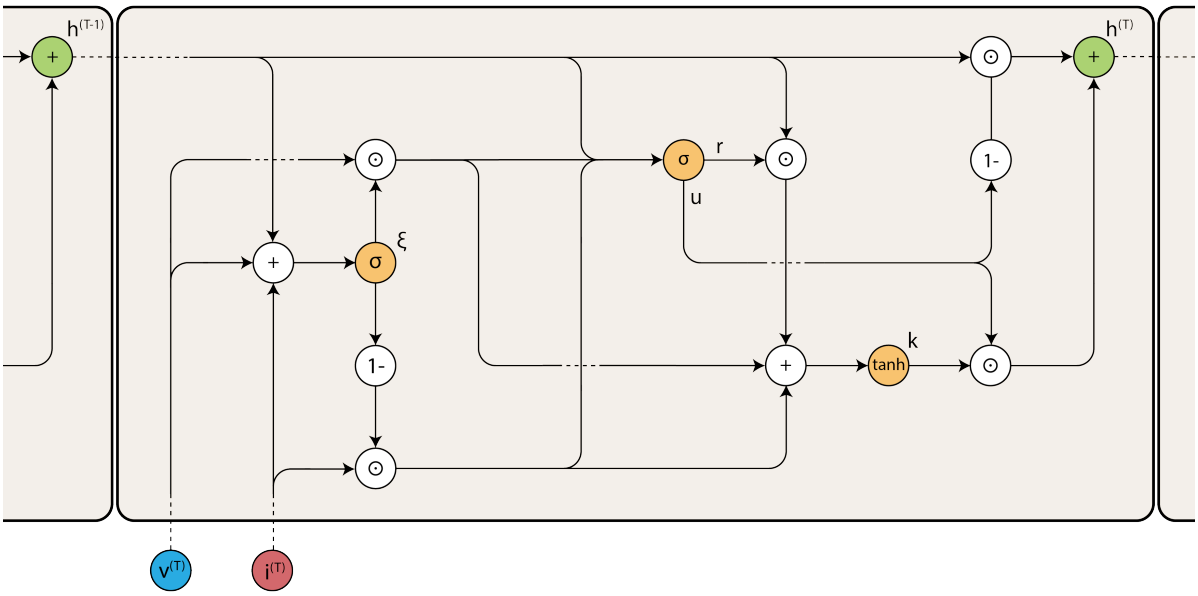


Figure 2.7: Attentional User-based GRU of Donkers et al. [33]. Interacting user and item vectors are combined with the last hidden state into the gating vector $\boldsymbol{\xi}$ used to balance the influence of the user and the item on hidden state update. Joining of arrows denotes concatenation of multiple vectors. Non-linear transformations denoted in orange. $\odot$ denotes element-wise multiplication. Multiplication with weight matrices omitted from the schematic for clarity. Note that the (weighted) summation in this model and the matrix multiplication of concatenated inputs in GRU are conceptually equivalent linear transformations.

However, whilst improving over the traditional baselines, the above models still discards a substantial amount of information that may be captured by time stamps and their derivatives. RNN-based time-dependent models have thus been proposed to solve the above limitation while still providing improvements in performance enjoyed by RNN-based sequential models.

2.7.2. Recurrent Neural Networks for time-dependent recommendation

There is currently a limited number of works in the domain of time-dependent recommendation that employ RNN-based methods. It is expected that this will change in the coming years as neural networks find wider adoption by the Recommendation community. Nevertheless, some authors have already employed RNN's for time-dependent recommendation. Wu et al. [142] propose to learn an architecture consisting of two jointly trained RNN's. Given a sequence of interactions, at each time step, one RNN outputs the current user state and the other outputs the current item state. These are then also combined with the static user and item embeddings to collectively predict the rating $\hat{r}_{v,i}$. The authors also use the current binned time stamp to allow the model to have a sense of where in the timeline it is. The authors argue that this allows their model to adapt to effects such as the change to the rating scale and a movie being nominated for an award. They argue that such events may lead to drastic changes in rating distribution for interactions following a certain point in time and should be accounted for.

A different approach was employed by Dai et al. [26]. The authors propose their version of a two-RNN scheme attempting to predict user's return time for each user-item pair. In their formulation, the user (item) state at each time step depends on four components: the state's previous value, the state value of the item (user) interacting, the rating and the time since the last interaction involving the user (item) - the time gap.

The reasoning for using the time gap information or its effects on recommendation, however, were not discussed. Moreover, the authors train the hidden states to predict the parameter of a Rayleigh distribution - restricting the expressiveness generally provided by neural networks.

Finally, Jing and Smola [63] propose the use of time gaps to estimate the return time of the user to the platform and the items they will consume during that session. The authors calculate time gaps between the sessions and discretise them to learn an embedding for each of the resulting bins as well as for the contextual vectors for each hour of the week. This temporal information together with the user and item information is then fed into an LSTM network, with the network aiming to jointly model the return time and the items to be consumed during the next session. The authors report a noticeable improvement in performance over their baselines. They also find that sessions following longer user absences are associated with a lower predictive performance of the model. We aim to further expand on their analysis and investigate the additional effects and the importance of including time gaps in the model.

2.7.3. Summary and open research directions

As presented in this chapter, Recommender Systems is an actively developing field of research. Combined with a rise of neural networks, the development of models leveraging the dynamic nature of consumption has led to improved capturing of evolving user tastes, paired with strong gains in performance.

A large number of successful works simply leverage the sequential nature of interactions, demonstrating substantial improvements over static models [33, 52, 53, 108]. A smaller number of works also rely on more explicit temporal information for recommendation [142], with a few authors incorporating time gaps into their models [26, 63]. Whilst the latter demonstrate improved performance over the chosen baselines, the benefits of time gap information, as opposed to general improvements stemming from other changes in model architectures, are not clear. Based on the work of Pampalk et al. [103] and Moling et al. [98] detecting early track switching may be beneficial in improving the recommendation accuracy. Furthermore, the work of Ding and Li [32], Jing and Smola [63] collectively suggests that items consumed a long time ago may only be weakly associated with the current user tastes. Both of the above use cases may be potentially represented using time gap information. However, little research has been done on the impact of accounting for the duration of user inactivity on the model performance and recommendation behaviour. Furthermore, to our knowledge, the value of time gaps for subsets of users with distinct consumption behaviours has not been previously studied. The aim of this thesis is thus to examine the effects and benefits of using temporal information. In the next chapter we describe the main ideas underlying our analysis.

3

Methodology

Temporal recommendation is an emerging and rapidly advancing area of the Recommender Systems. In recent years, in large part due to resurgence of Neural Networks, there has been a substantial increase in the number of works addressing both sequential as well as time-dependent recommendation [33, 52, 53, 63, 108, 129]. A number of authors have demonstrated improvements over the previous state-of-the-art using their LSTM and GRU based models, particularly in the domain of sequential recommendation [33, 52, 53, 108]. A smaller number of works have also chosen to include additional temporal features, aiming to further increase the models' performance [26, 63]. Yet in most cases these features are added in tandem with other non-temporal modifications and as such the effects of temporal information are not properly explored.

We focus on one such feature, the time gaps. In this chapter, we provide our formal definition of time gaps. We then provide our rationale for focusing on that particular type of temporal information. Next, we introduce our models, collectively termed *DeepTimeDelta*, incorporating time gaps into recommendation. Finally, we propose a set of experiments with the goal of examining the benefits and the effects of time gaps on recommendation.

3.1. Time gaps

Given a history of k ordered interactions from one user $I_u = [i^{(T=1)}, i^{(T=2)}, i^{(T=3)}, \ldots, i^{(T=k)}]$ we define $\Delta t^{(T)}$ as the time interval between the starts of two subsequent interactions $\Delta t^T = t^{(T)} - t^{(T-1)}$. Note that we measure this interval between the starts of two interactions (e.g. song streams) and not between the end of the first and the start of the second. This is done as in many publicly available datasets only the start time stamps are available. In cases where the interaction is instantaneous and is not associated with a pair of start and end time stamps (e.g. user rating a movie, confirming a purchase), the time gap is measured between the time stamps recorded for such events. In this thesis, when referring to time gaps in general, the simplified notation Δt is also used.

3.2. Motivation

As discussed in the previous chapter, sequential and time-dependent models appear to achieve state-of-the art performance in the context of temporal recommendation. Whilst time gaps have been less explored overall, they have been featured in architectures proposed in [26, 63]. Intuitively, time gaps may provide the model with signals otherwise unavailable to the pure sequence-based models. It is hypothesised that time gap information may be particularly useful in contexts featuring exceptionally long or exceptionally short times between subsequent item consumptions. For instance, in case of a movie streaming platform, users consume content with different frequency. We propose a hypothetical user, who on average consumes one movie per day. When such a user returns to the platform and chooses an item of a different genre to what they normally consume, a sequential model may adjust its recommendation to some degree to indicate this potential shift in user preferences. This adjustment would be identical irrespective of the duration of the user's last absence. However, as described by Jing and Smola [63], as the user spends more time away from the platform, our confidence in the estimated preferences of that user may be decreased. Conversely, when the user does come back to the platform after a long period of inactivity, the item they consume upon their return may

be strongly indicative of their current tastes. If this item is substantially different from the items consumed prior to the user's departure, this potential shift in user preferences should be immediately reflected in the new recommendations. Moreover, users operate at different scales. An absence of one month may be considered long for a user consuming the content daily but normal for a different user consuming content every few weeks. As such, the interpretation of the time gap should not be simply hard coded into the model and instead vary based on additional factors [32].

Furthermore, time gaps of ~4 minutes may be standard for a user of a music streaming service as those may indicate a continuous streaming session. Yet exceptionally short time gaps between starts of two consecutive tracks (e.g. 5 seconds) may indicate that the user chose to abandon the first track in favour of the second. This negative signal towards the first item may be indicative of the user's current preferences and should be accounted for when generating recommendations, as suggested by work of Moling et al. [98], Pampalk et al. [103]. This information, however, would be ignored by a sequential model, that would simply see that the second item was consumed at some point after the first. More generally, time gaps of various sizes may provide a recommender system with improved feedback relating to the user's current preferences.

A potential alternative to using the time gaps as the source of temporal feedback is using the time stamps of interactions directly. For instance, the raw time stamps were used in the RNN-based model of Wu et al. [142]. However, the authors suggested that the main benefit of that feature was that it allowed the model to account for the seasonal effects and trends taking place at that point in time. It is unclear whether the models would be able to leverage this form of feedback to identify the patterns described above. As such, we propose incorporating the time gap information in a more explicit fashion. Moreover, the extent to which time stamps are amenable to various embedding procedures such as those used in [26, 142] is unclear. Namely, time stamps used at prediction time would fall outside of the range for which the embedding function was learned. Such discrepancy between the training and testing input distributions is an example of covariate shift and is likely to lead to suboptimal performance on at prediction time [127, 133]. For time gaps, however, most of the input space is likely to be represented in the training set. Whilst there may be Δt values for which there are no exact corresponding training examples, the model may still generalise by relying on surrounding Δt values. Nevertheless, we hypothesise that incorporating time stamps together with the time gap data may lead to the model receiving an even greater amount of information. In the future work we would like to address the extent to which either of the both features would benefit the model when used in tandem.

3.3. Model

In order to analyse the impact the time gaps may have on recommendation, it is essential to include a baseline sequential model to which the time gap model is contrasted. Importantly, the architecture of both models should be similar as to facilitate the direct comparison and interpretation of the results. Moreover, the baseline architecture should ideally also be a state-of-the-art model that is simple in concept and implementation as to allow for a more straightforward interpretation of the effects of Δt. For the above reasons we choose to extend the model proposed by Donkers et al. [33] and use their model as our baseline. As previously described in Subsection 2.7.1, in contrast to other models such as [108, 129], which consist of a user network and item network, the model of Donkers et al. is a single RNN combining both user and item components. Using their simpler architecture, the authors still report strong results, improving over their baselines. The model of Donkers et al. is defined as follows:

$$\boldsymbol{\xi} = \sigma(W_{n,n}\mathbf{h}^{(T-1)} + W_{n,n}\mathbf{E}_v\mathbf{v}^{(T)} + W_{n,n}\mathbf{E}_i\mathbf{i}^{(T)}) \tag{3.1}$$

$$\begin{bmatrix}\mathbf{u}\\ \mathbf{r}\end{bmatrix} = \begin{bmatrix}\sigma\\ \sigma\end{bmatrix} W_{3n,2n} \begin{bmatrix}\mathbf{h}^{(T-1)}\\ \boldsymbol{\xi}\odot\mathbf{E}_v\mathbf{v}^{(T)}\\ (1-\boldsymbol{\xi})\odot\mathbf{E}_i\mathbf{i}^{(T)}\end{bmatrix} \tag{3.2}$$

$$\mathbf{k} = \tanh(W_{n,n}\mathbf{r}\odot\mathbf{h}^{(T-1)} + W_{n,n}\boldsymbol{\xi}\odot\mathbf{E}_v\mathbf{v}^{(T)} + W_{n,n}(1-\boldsymbol{\xi})\odot\mathbf{E}_i\mathbf{i}^{(T)}) \tag{3.3}$$

$$\mathbf{h}^{(T)} = (1-\mathbf{u})\odot\mathbf{h}^{(T-1)} + \mathbf{u}\odot\mathbf{k} \tag{3.4}$$

In the above formulation $\mathbf{v}^{(T)}$ and $\mathbf{i}^{(T)}$denote one-hot vectors indicating the user and item interacting at time step T. In turn, $\mathbf{E}_v$ and $\mathbf{E}_i$ represent the user and item embedding matrices. Thirdly, $\mathbf{h}^{(T-1)}$ indicates the hidden state of the previous time step. For simplicity, embeddings, hidden state and intermediary vectors are of the same dimensionality n, also referred to as embedding width. The embeddings are initialised randomly and are learned during training. Element-wise multiplication is denoted by $\odot$. Linear transformations $\mathbb{R}^m \rightarrow$

$\mathbb{R}^n$ are denoted as $W_{m,n}$ with bias omitted for clarity. Note that all the above weight matrices are different and are learned independently from one another. Finally, item scores used for prediction can be extracted by applying another linear transformation $W_{n,|I|}$ to the current hidden state $\mathbf{h}^{(T)}$, where $|I|$ is the total number of items. Similar to the user and item embeddings, values of the hidden state may be interpreted as a position in a (different) latent space describing user's recent preferences. Importantly, the aim of the model at any point in time is to predict the next item consumed by the user. Whilst not fully equivalent to recommending the most relevant items, the estimated probability of interaction with an item is used as a proxy for relevance.

Notably, the model of Donkers et al. does not leverage time gaps nor any other temporal information besides implicit ordering. It can, however, be extended to do so with relatively little effort.

3.3.1. Equation 1

Equation 3.1 gives the model of Donkers et al. the ability to selectively switch focus between the user and item embeddings when updating the model state and predictions. This gating of user and item components is done via the vector $\boldsymbol{\xi}$, with user and item (embedding) vectors interacting during a given time step scaled element-wise by $\boldsymbol{\xi}$ and $1-\boldsymbol{\xi}$ respectively. When a particular element $\boldsymbol{\xi}_j$ is close to zero, the respective part of the user embedding $(\mathbf{E}_v\mathbf{v})_j$ is scaled to be low as well. Conversely, $(\mathbf{E}_i\mathbf{i})_j$ is scaled by $1-\boldsymbol{\xi}_j$ and thus stays close to its original value.

Conceptually, this gating mechanism can be interpreted as the model being able to decide whether to give more weight to the user or the item component in making the prediction for the next time step. The model takes this decision on the basis of the user and item interacting (via their embeddings) as well as the user's recently consumed items (captured by the hidden state). Through this mechanism the model can reduce the effect of the user embedding on the prediction if it is deemed to be less important for the decision. Conversely, the model may instead reduce the impact of the item on the recommendation and focus on the user component, if it is deemed more relevant.

We introduce temporal information into the model by allowing $\boldsymbol{\xi}$ to make use of the time gap between the current and the previous item. In order to achieve this, Equation 3.1 is modified to be as follows:

$$\boldsymbol{\xi} = \sigma\left(W_{n,n}\mathbf{h}^{(T-1)} + W_{n,n}\mathbf{E}_v\mathbf{v}^{(T)} + W_{n,n}\mathbf{E}_i\mathbf{i}^{(T)} + W_{n,n}\mathbf{E}_{\Delta t}(\Delta t^{(T)})\right) \tag{3.5}$$

As users and items contribute to the model in the form of n-dimensional embeddings, we propose the embedding function $\mathbf{E}_{\Delta t}(\Delta t, d, f)$, also used as $\mathbf{E}_{\Delta t}(\Delta t)$:

$$\mathbf{E}_{\Delta t}(\Delta t, d, f) = \begin{cases} W_{1,n}\Delta t & d = 0 \\ f(\mathbf{E}_{\Delta t}(\Delta t, 0, f)) & d = 1 \\ f(W_{n,n}\mathbf{E}_{\Delta t}(\Delta t, d-1, f)) & d > 1 \end{cases} \tag{3.6}$$

The temporal embedding function allows the model to capture the influence the time gap may have on the recommendation and incorporate it into the model along with the embeddings of the user and the item. In the above definition, d is referred to as the depth of the temporal embedding and is a hyperparameter. Setting d to 0 leads to the model using a simple linear embedding of the time gap, as done in [26]. However, we hypothesise that a simple linear transformation may not be sufficient to capture the effect of time on recommendation. Intuitively, when dealing time gaps on the order of months, there is little difference between the user behaviours for $\Delta t = x$ seconds $\Delta t = x + 30$ seconds. Conversely, such minor differences may be more important when the value of the time gap is on the order of seconds. Due to this nonlinear interpretation of time we believe that applying an additional nonlinearity may better represent the effects of time on recommendation. By setting $d = 1$ we thus also apply a nonlinear function f such as ReLU, tanh or sigmoid on top of the linear transformation. Values of d exceeding 1 imply further application of the linear and nonlinear transformations, with $W_{n,n}$ of different depth layers learned separately. For simplicity f is kept constant across different depth levels. Through application of the linear transformation and f the model is able to capture the chosen complexity with which the time gap affects the recommendation.

Through the formulation proposed in Equation 3.5 the model may now additionally selectively switch between the user and item components based on the time since the user's previous interaction. This may allow the model to learn the relationship between the time gap size and the extent to which both user and item components should be included in predicting the next item.

As hypothesised in Section 3.2, there are various contexts where the model may leverage this ability. In particular, in case of unusually long time gaps the model may find it beneficial to reduce the values of the components of $\boldsymbol{\xi}$ to be as low as possible. Shifting focus from the user's historic preferences to the recently

consumed item may serve as a mechanism for the model to rapidly adapt to the changing user preferences. On the other hand, we hypothesise that the model may interpret an extraordinarily short interval between two items as negative feedback with respect to the first one. We envisage two distinct ways how the model may react to such input. Firstly, the model may learn to strongly augment the item component by decreasing $\boldsymbol{\xi}$. As the hidden state update cannot include the previous item directly, it may instead choose to correct its previous hidden state update on the basis of the new item. Alternatively, the model may instead choose to overwrite itself by applying an update with a stronger user component, representing the user's more stable preferences. This may be done as to undo the changes to the hidden state introduced by the previous update. In the same vein, analogous interpretations may be learned for any given value of Δt.

As previously alluded, by modifying just the first equation the time gaps are only able to influence the the balance between the user and the item. The model's flexibility may be further increased by allowing Δt to regulate the influence of the previous hidden state $\mathbf{h}^{(T-1)}$. Theoretically the model may already have the capacity to do so indirectly via $\boldsymbol{\xi}$ influencing the values of $\mathbf{u}$ and $\mathbf{r}$. As these in turn regulate $\mathbf{h}^{(T-1)}$, changes in $\boldsymbol{\xi}$ may also have an accompanying impact on the influence of the $\mathbf{h}^{(T-1)}$ on recommendations. In practice, however, it may be beneficial to allow Δt to regulate $\mathbf{h}^{(T-1)}$ in a more explicit fashion.

3.3.2. Equation 2

Equation 3.2, describing the second equation of the baseline model, allows the model to use the attenuated embeddings together with $\mathbf{h}^{(T-1)}$ to infer vectors $\mathbf{r}$ and $\mathbf{u}$. The vector pair is then used to regulate the effect of $\mathbf{h}^{(T-1)}$ on $\mathbf{h}^{(T)}$ as described in Equations 3.3 and 3.4. Namely, vector $\mathbf{r}$ regulates the contribution of $\mathbf{h}^{(T-1)}$ to the update relative to the user and item embeddings. On the other hand, $\mathbf{u}$ balances the direct contribution of $\mathbf{h}^{(T-1)}$ and the update vector $\mathbf{k}$ to the new hidden state $\mathbf{h}^{(T)}$. On a conceptual level, both $\mathbf{u}$ and $\mathbf{r}$ control the importance of recently consumed items on the next item prediction.

By modifying the second equation of Donkers et al. the model is provided with two additional ways in which Δt may affect the prediction. We propose to modify Equation 3.2 as follows:

$$\begin{bmatrix}\mathbf{u}\\ \mathbf{r}\end{bmatrix} = \begin{bmatrix}\sigma\\ \sigma\end{bmatrix} W_{4n,2n} \begin{bmatrix}\mathbf{h}^{(T-1)}\\ \boldsymbol{\xi} \odot \mathbf{E}_v \mathbf{v}^{(T)}\\ (1-\boldsymbol{\xi}) \odot \mathbf{E}_i \mathbf{i}^{(T)}\\ \mathbf{E}_{\Delta t}(\Delta t^{(T)})\end{bmatrix} \tag{3.7}$$

In the altered formulation the update vector Δt may now directly affect the direction in the latent space in which the model's hidden state is updated. The time gap's embedding, via $\mathbf{u}$, may now dictate whether any of the hidden state components need to be updated as well as the extent of the update. $\mathbf{r}$, on the other hand, regulates the direction of the update, which previously only depended on the (attenuated) user and item embeddings. This effectively allows the model to regulate the extent to which the user's recent interaction history is taken into account relative to the user and item information.

In practice, the model is presented with a mechanism through which it can regulate the similarity of the new predictions to the old ones based on the value of the time gap. Similar to scenario presented for the first equation, this mechanism may be particularly beneficial in the context of long time gaps. We hypothesise that in such scenarios the model may learn to strongly augment the values of $\mathbf{u}$ as to base the prediction in most part on the current interaction, captured by the update vector $\mathbf{k}$. Similarly, we expect the model to diminish the contribution of the hidden state to $\mathbf{k}$ by regulating $\mathbf{r}$ to be low. As such, the increased dependency on the item for the prediction, as described for the first equation, would be inflated even further.

The mechanism may also act in a similar fashion for short time gaps, as to drive the model away from its current hidden state, representing recommendations related to the previous item. We envisage the model updating itself strongly either on the basis of the user or the item vector, in both ways overwriting the erroneous previous update of the hidden state.

Lastly, certain time gaps may instead serve as a positive feedback with respect to the previous item. For instance, if a user consumes a recommended sitcom episode in full and then tunes into another piece of content, the associated time gap may serve as an additional signal that the recommendation was relevant. In that context the model may want to increase the value of $\mathbf{r}$ used in the hidden state update as to reflect a higher confidence in the estimated user preferences, modelled by the hidden state.

Note that based on the findings of Greff et al. [44], discussed in Subsection 2.6.2, $\mathbf{u}$ and $\mathbf{r}$ have a related role and thus may in theory be replaced by $\mathbf{u}$ and $1-\mathbf{u}$ without any loss in performance. In our implementation we keep two separate variables to maintain the similarity to the baseline model. Moreover, in order to reduce the complexity of the model, the depth and the nonlinear function of $\mathbf{E}_{\Delta t}$ are shared across the equations.

3.3.3. Equation 3

Finally, the third equation of baseline model (Equation 3.3), reflects how the model combines the summary of the recent interactions via $\mathbf{h}^{(T-1)}$ with the descriptors of the current interaction - the (attenuated) user and item embeddings. We hypothesise that there may be a benefit in employing the time gap information as another such descriptor. This is reflected in the proposed formulation for the third equation:

$$\mathbf{k} = \tanh(W_{n,n}\mathbf{r} \odot \mathbf{h}^{(T-1)} + W_{n,n}\boldsymbol{\xi} \odot \mathbf{E}_v\mathbf{v}^{(T)} + W_{n,n}(1-\boldsymbol{\xi}) \odot \mathbf{E}_i\mathbf{i}^{(T)} + W_{n,n}\mathbf{E}_{\Delta t}(\Delta t^{(T)})) \tag{3.8}$$

The proposed change allows the time factor to directly contribute to the update. As the term $W_{n,n}\mathbf{E}_{\Delta t}(\Delta t^{(T)})$ is shared across all users it may be interpreted as a movement in the latent space towards a certain hidden state specific for each value of Δt. This can effectively be seen as a decay towards a baseline recommendation associated with that time gap. For instance, there may certain popular key items in the catalogue that users tend to return to after prolonged absences. In such cases the potential user interests may be best served by updating the hidden state mostly on the basis of the time gap itself.

3.3.4. Combined model

Combining all the proposed modifications, the full *DeepTimeDelta* model can be expressed as follows:

$$\boldsymbol{\xi} = \sigma(W_{n,n}\mathbf{h}^{(T-1)} + W_{n,n}\mathbf{E}_v\mathbf{v}^{(T)} + W_{n,n}\mathbf{E}_i\mathbf{i}^{(T)} + W_{n,n}\mathbf{E}_{\Delta t}(\Delta t^{(T)})) \tag{3.9}$$

$$\begin{bmatrix}\mathbf{u}\\ \mathbf{r}\end{bmatrix} = \begin{bmatrix}\sigma\\ \sigma\end{bmatrix} W_{4n,2n} \begin{bmatrix}\mathbf{h}^{(T-1)}\\ \boldsymbol{\xi} \odot \mathbf{E}_v\mathbf{v}^{(T)}\\ (1-\boldsymbol{\xi}) \odot \mathbf{E}_i\mathbf{i}^{(T)}\\ \mathbf{E}_{\Delta t}(\Delta t^{(T)})\end{bmatrix} \tag{3.10}$$

$$\mathbf{k} = \tanh(W_{n,n}\mathbf{r} \odot \mathbf{h}^{(T-1)} + W_{n,n}\boldsymbol{\xi} \odot \mathbf{E}_v\mathbf{v}^{(T)} + W_{n,n}(1-\boldsymbol{\xi}) \odot \mathbf{E}_i\mathbf{i}^{(T)} + W_{n,n}\mathbf{E}_{\Delta t}(\Delta t^{(T)})) \tag{3.11}$$

$$\mathbf{h}^{(T)} = (1-\mathbf{u}) \odot \mathbf{h}^{(T-1)} + \mathbf{u} \odot \mathbf{k} \tag{3.12}$$

The model may leverage the temporal information at every step of the decision making by selectively regulating the effects the user, the item and the user's previous history exert on recommendations. Moreover, temporal information is allowed to directly affect the change to the recommendations. Furthermore, should the model find any such use of the time gap not useful, it may learn to switch that use off via the temporal term's weights in the associated equation or the embedding function itself.

Importantly, modifications to the original equations need not all be applied simultaneously. It may be, for example, that the optimal performance is already achieved by modifying a smaller subset of equations. Such a model, due to its lower number of learnable parameters, may also be marginally faster to train. As a consequence, we propose the family of *DeepTimeDelta* models, also referred to as *time delta* models. For the remainder of the thesis the models are referred by their number, as described in Table 3.1.

Table 3.1: Addition of time gap information to equations in different models. Model 0 denotes the sequential baseline model.

Model	Modified equations
0	None
1	1
2	2
3	3
4	1, 2
5	1, 3
6	2, 3
7	1, 2, 3

3.4. Addressing research aims

As this work is exploratory in nature, we do not propose *DeepTimeDelta* models as the new state-of-the-art. Instead, our goal is to use the models as a tool for analysis of temporal information, with the purpose of determining the benefits of time gaps as a feature. More precisely, our overarching Research Question is *Are time gaps a useful source of information for improving recommendation accuracy in sequential recommendation?*. We aim to leverage these models to tackle its distinct aspects, captured by our three Research Sub Questions.

3.4.1. Research Sub Question 1

Our first Sub Question is *Which modes of incorporating time gaps into a sequential model lead to an improvement of recommendation accuracy?* We propose to address this question by contrasting the performance of our models with the baseline through offline evaluation across multiple datasets. Combinations of equations leading to increases in performance may be considered beneficial for recommendation accuracy.

3.4.2. Research Sub Question 2

Following that, we aim to address Research Sub Question 2: *How are different values of time gaps interpreted by the model and what effects do they have on recommendation accuracy?*. This is done by evaluating the behaviour and performance of the above models for different values of Δt. In addition to improvements across the spectrum of various time gap values, we focus on short and long time gaps in particular. Note that the definitions of short and long time gaps are both subjective as well as dataset and use specific - we describe our threshold choices and their motivation in the following chapter. Our hypotheses for those two types of time gaps, previously discussed in the context of individual equations in Subsections 3.3.1 and 3.3.2, are as follows:

- For datasets where time stamps denote the time of consumption, short time gaps may serve as source of negative feedback and can be exploited by time delta models to improve recommendation accuracy for such interactions.
- For all datasets, long time gaps, indicating prolonged user absence from the platform, may serve as a signal for a given time delta model to perform a larger update of its hidden state as to capture more recent user preferences. Such an adaptation may then lead to improved recommendation accuracy following those interactions.

In addition to the above, time gaps may also prove useful in the context of other types of data. Namely, Donkers et al. evaluate their model on two datasets, one of which is a dataset of users assigning ratings to movies. As such, the true item consumption times are unknown. Moreover, the dominant behaviour present in the dataset is marking multiple consumed items within a short window (minutes) and returning to the platform only after a longer absence (weeks). Whilst the true consumption order is unknown, short inter consumption intervals may provide *DeepTimeDelta* with additional information, allowing it to identify cases where the item consumption order in the data may be ambiguous. The existence of such an adaptation may highlight further data-dependent uses of time gaps and may be accompanied by performance gains for those interactions. Whilst such improvements may not necessarily lead to improved user satisfaction when used in a real system, they may still indicate that the model overall has a better understanding of the underlying user behaviour. As such, our third hypothesis is as follows:

- For datasets where time stamps denote the time when a rating was assigned, short time gaps may serve as an indication of uncertainty with respect to the order of the item consumption. As such, they can be leveraged by time delta models to improve models' recommendation accuracy following those interactions.

We first aim to address the above hypotheses through a more general analysis by splitting all interactions into a limited number of categories on the basis of their time gap. Performance of time delta models for those smaller interaction groups is to be contrasted to that of the baseline - groups of interactions for which a significant result improvement is observed can be regarded as benefiting from the use of time gaps.

However, whilst the above the above analysis provides a quantitative overview of the effect of time gaps on performance, it may do so at at excessively low resolution. Moreover, the above analysis does not establish a link between the improvements in performance and specific model behaviours outlined in the hypotheses. As such, in addition to the above, we propose to separate interactions into more fine grained groups on the basis of Δt, focusing on differences in the performance and recommendation behaviour as well as internal model responses. Discrepancies across the values of Δt for a single model as well as differences between the sequential and time delta models can then be used to link improvements in performance to specific mechanisms through which time delta models leverage the time gaps.

3.4.3. Research Sub Question 3

Lastly, we aim to answer the final Sub Question: *Are time gaps beneficial in predicting the behaviour of specific groups of users?*. Such knowledge may be crucial for cases where the platform's aim is to improve recommendations for a specific set of users underserved by the current recommendation model. We propose to identify

user groups benefiting from the use of time gaps by assessing the performance of proposed models for those users. User groups for whom time delta models demonstrate significant improvements over the baseline can be said to benefit from the use of temporal information.

Firstly, in our view, users with differing listening habits in terms of item consumption rate may benefit from time delta models to a different degree. More specifically, we believe that models leveraging time gaps may show improved performance for users consuming content infrequently, reflecting the potential improvements in adapting to users developing new tastes following long periods of inactivity, as hypothesised earlier. In a similar fashion, improvements may be observed for users consuming content with a high frequency, mirroring the expected improvements associated with enhanced recommendation following short periods of inactivity.

Finally, previous work also suggests that users consuming items of different popularity may benefit from recommender systems to a different degree [9, 38]. Bauer and Schedl [9] refer to the extent to which certain users are inclined to consume popular content as user *mainstreamness*. In their work, the authors suggest that it may be more difficult to predict the behaviour (and thus provide recommendations) for users with more niche preferences as opposed to the users consuming mostly popular content. We hypothesise that users with different content preferences may also have a distinct temporal consumption pattern. As such, we aim to identify the mainstreamness user groups for which time delta models exhibit improved performance compared to the sequential baseline. The exact method of partitioning users into groups along with the setup of the remaining parts of the experiment are presented to the reader in the following chapter.

4

Experimental setup

4.1. Datasets

The analysis of the effects of inclusion of temporal information in recommendation was performed on the basis of three datasets. The first dataset, *LastFM 1K user*[1] (*LastFM*), represents listening habits of 992 users for 1500661 tracks over the span of four years. An interaction between a user and an item denotes the user commencing to listen to the specified track. Time stamps provide information on the starts of the streams but no information is available for stream end times. Δt of *LastFM* approximately follows a log-normal distribution with a median of 239 seconds (Figure 4.1a).

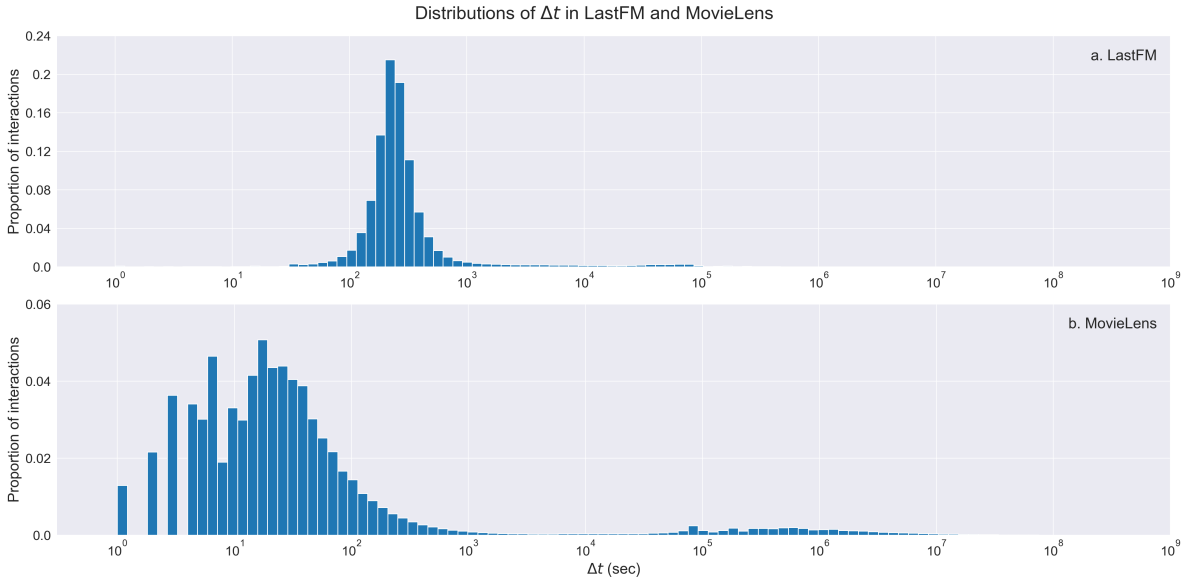


Figure 4.1: Δt distributions of *LastFM* and *MovieLens*. Note that the bins are exponentially increasing. Moreover, interactions with $\Delta t = 0$ are excluded. As such, the sum of bins for either dataset is lower than 1.

The second dataset, *MovieLens 10M* dataset[2] (*MovieLens*) [46] contains ratings from 71567 users assigned to 10681 movies over the span of 15 years. An interaction between a user and an item denotes the user assigning the rating to the item on a five point scale at the time denoted by the time stamp of interaction. The time when the item was in reality consumed is not known. Importantly, ~27% of the dataset's rating events take place with the same time stamp as another interaction for the same user. Without those interactions, Δt values in *MovieLens* form a log-normal Gaussian mixture with the modes of approximately 15 seconds and 6.75 days (Figure 4.1b). The first mode suggests that a common behaviour in this dataset is to rate multiple

[1] Last.fm & http://ocelma.net/MusicRecommendationDataset/lastfm-1K.html
[2] grouplens.org/datasets/movielens/10m/

movies over a short period of time. Additional analysis of the data (not shown) demonstrates that ~60% of the dataset's users log in only once, rate on average ~69 items and do not return later. Users who do return on average rate 17 items per session (with no time gap exceeding 1 hour).

The above datasets were chosen as they represent different domains in which time gaps may have different interpretations that may be learned by our models. Moreover, these datasets are widely used in literature, including the work of Donkers et al. [33, 51, 63, 67, 87, 108, 135, 148]. Choosing these datasets allowed for a facilitated comparison of the proposed models to the baseline by acting as a sanity check for the correctness of the implementation of the baseline model. As discussed in Methodology Section 3.4, interactions in *MovieLens* may not be indicative of the true consumption times, particularly when Δt between two interactions is short (e.g. seconds or minutes). As such, we choose to include the dataset to examine whether short time gaps may be leveraged as a signal for this uncertainty. Note that we assume that items whose consumptions are instead separated by longer periods (e.g. weeks) were in reality also consumed in that order.

The third and final dataset came from a Dutch Video on demand platform *Videoland*. The dataset consists of movie and series streaming events gathered over a period of one year. An interaction between a user and an item denotes the user starting the consumption of the item. The time stamp of interaction denotes the start of the video stream.

4.2. Dataset specific preprocessing

Data was first preprocessed separately with distinct steps performed for each dataset. In order to increase the computational efficiency and reduce the memory load, the training and evaluation were performed on a sampled version of *LastFM* dataset, as described by Donkers et al. [33]. Namely, for each user, a random slice of successive interactions totalling 10% of the length of their consumption history was obtained, retaining interactions with 407128 items. Applying subsampling is essential in order to store the embedding matrices on the GPU during training, particularly when using a high embedding width. While it is possible to store the embeddings on the CPU, this would lead to a drastic decrease in the training speed. As such, the sampling strategy was employed.

A different form of preprocessing was applied to *MovieLens 10M*. As the models evaluated in this thesis rely on implicit and not explicit feedback, the ratings have to be converted into a implicit representation. Under our evaluation procedure all the dataset's interactions are kept whilst removing the rating values given, identical to [33, 113]. While some works apply a different scheme whereby only the interactions with a rating exceeding a certain threshold are considered [51, 60], the former was chosen as to reflect the fact that not all implicit interactions are associated with a positive user experience. This is a important aspect of true implicit feedback data and as such crucial to retain.

Finally, the platform of *Videoland* provides users with recommendations for both movies and series. However, the platform only permits recommendations of series as a whole and not of individual episodes. As such, all episodes of a single show were instead set to be represented by the same item index. The above choice, though, led to the dataset comprising of a high number of cases of long sequences indicating the repeated consumption of the same item by one user. Our preliminary attempts to use such data indicated that simple sequential models (e.g. those with embedding size of 1) appeared to already achieve strong performance by recommending the same item the user had just consumed. Whilst, in theory, such a recommendation strategy is not inherently wrong, *Videoland* uses personalised recommendations to help users discover new content. As such, any items previously seen by the user are in practice filtered out from the recommendation strip. Employing the data without any additional modifications would thus lead to a discrepancy between the model's objective during training and its use at prediction time as the model's top prediction(s) would never be actually shown to the user in practice.

With the above considerations in mind, we propose an alternative *Videoland*-only training and evaluation scheme that would encourage the model to look further than the previously consumed item when choosing which items to recommend. Given a sequence of k interactions from one user $\mathrm{I}_v = [i^{(T=1)}, i^{(T=2)}, \ldots, i^{(T=k)}]$, we define the associated weights for the training set as well as for validation and test sets as follows:

$$w_{\text{train}}(T, x) = \begin{cases} x & i^{(T)} = i^{(T+1)} \\ 1 & \text{otherwise} \end{cases}, \quad 0 \leqslant x \leqslant 1 \tag{4.1}$$

$$w_{\text{val,test}}(T) = \begin{cases} 0 & i^{(T)} = i^{(T+1)} \\ 1 & \text{otherwise} \end{cases} \tag{4.2}$$

Under the proposed scheme, the loss and the contribution to the evaluation metrics for a given interaction (discussed in Sections 4.3 and 4.4 respectively) are scaled by the associated weight. During training, an interaction with an item that is followed by further interactions with the same item is weighted by a hyperparameter x, whose value may be between 0 and 1. Contribution of individual interactions to the batch loss is set to be proportional to this weight. As such, lower values of x force the model to assign more importance to other interactions where the next item is different. This in turn, encourages the model to recommend items different from the one consumed. Moreover, repeated interactions in validation and test sets are assigned a weight of 0, masking their contribution to the the loss and evaluation metrics. This is done to reflect the fact that in practice the model would not be allowed to make the recommendation of the repeated item and should not be penalised for failing to do so.

We hypothesised that setting $0 < x \ll 1$ (e.g. $x = 0.01$) may still allow the model to leverage repeated interactions with the same item, indicating higher user interest in that item, analogous to the use of confidence in [58]. However, our experiments using a holdout validation set on $x \in [1, 0.8, 0.5, 0.2, 10^{-1}, 10^{-2}, 10^{-3}, 0]$ indicated consistently higher performance for $x = 0$ across all models.

Other evaluation schemes, such as removing the middle interactions when the same item was repeated 3 or more times or removing all but 1 interaction were also considered. In general, it was found that explicit removal of items from the user history led to ambiguity with respect to the choice of Δt for remaining interactions. Moreover, the latter alternative, corresponding to removing $i^{(T=3)}$ in the above example, is in part captured by the our chosen evaluation scheme at $x = 0$. Finally, we would like to stress that our proposed scheme is only applied to the *Videoland* dataset as the remaining two datasets do not suffer from repeated interactions to the same extent (3% for *LastFM*, 0% for *MovieLens*).

4.3. Further preprocessing and model training

The dataset specific preprocessing was then followed up with additional data processing steps applied to all datasets. These steps are adapted from the preprocessing applied by Donkers et al. [33] as to maintain the consistency of implementation and evaluation, only expanding it with steps necessary to include time gaps.

First, the interactions of each user are grouped together and ordered by the ascending time stamp of the interaction. In order to incorporate temporal information to be used by our models, for each interaction the time stamp of the preceding interaction is also subtracted from the current time stamp, yielding the associated time gap.[3] Next, for each interaction in the user's sequence (a triplet of v, i, Δt) [4] an associated target label indicating the item the user consumed next is obtained. The user's sequence of these input-target pairs is then split into the training, validation and test sets. The test encompasses the most recent five percent of the user's feature-label pairs, rounded down. Similarly, validation set encompasses the same proportion of interactions preceding those included in the test set. Finally, the remaining input-output pairs are assigned to the training set. For instance, for a user's history featuring 80 interactions, 78 feature-label pairs would be retained after calculating Δt and the target labels. From those, first 72 pairs would be assigned to the training set, 3 subsequent pairs to the validation set and the final 3 to the test set. Moreover, interactions with items not seen in the train set are also purged from the validation and test sets as embeddings belonging to those items would not have been learned during training. Finally, the remaining interactions for each set are then split into non-overlapping sequences of length 20, applying padding at the ends of the sequences where necessary.[5]. At each unpadded point in the sequence the model's task is to predict the next item consumed by the user, as captured by the target label.

During a training step, the model receives a shuffled batch of such sequences, generating predictions at each time step. At the end of the sequence, softmax transformation is applied to item scores for all unpadded time steps. These scores are then compared to the target labels in order to calculate the cross entropy loss. As described previously in Section 4.2, losses for individual interactions of *Videoland* are further scaled by w_{train}. The resulting losses are then averaged in order to produce the batch loss, which is in turn used to update the parameters utilised for the processing of the next batch using ADAM [75]. For validation and test sets, no parameter update is performed. Instead, item scores are used to return a list of the highest ranked items to be recommended to the user.

[3] In our implementation, the the same dataset files are used for the baseline model and our time-dependent models. However, during the data feeding the temporal information is discarded for the sequential model.

[4] v and i refer to an index used to denote a single user or item. $\mathbf{v}$ and $\mathbf{i}$ refer to one hot encoded user and item vectors. Conceptually, both representations refer to the same input and as such are used interchangeably.

[5] We apply zero padding to ensure the same length of each sequence, as generally required for batch processing by computational frameworks such as TensorFlow.

We choose the above data partitioning scheme as to maintain similarity to Donkers et al. [33]. Moreover, the scheme of partitioning user interactions into subsequences of length 20 is able to account for substantial differences in user history lengths. Providing the model with full user histories instead of subsequences may lead to a substantial computational time overhead. Namely, due to the disparity in the lengths of user histories a recurrent model may quickly reach the end of the batch's shorter user histories. However, it would be unable to start the next batch until the few remaining long users are processed. The above situation is likely to take place with *LastFM* where user history lengths vary between 1 and 16480 interactions, with 25% of users having fewer than 350 interactions and another quarter of users having consumed more than 2325 items.

We hypothesise that the sequence length value may have an effect on the training and evaluation dynamics. Whilst for this work its value is set at 20, we would like to investigate the effects of varying this value as a part of our future work. Moreover, as described in [20], it may be preferable to partition interactions between training, validation and test sets such that all interactions that took place between two time stamps belong to the same set. Explicitly, we only use a given user's earlier interactions to predict their later ones. However, some information from interactions of other users whose train set interactions happened after the first user's validation or test set may be reflected in the learned parameters. These could in turn provide the model with additional information for the first user's validation and test set interactions, which it would not have access to during real use. As such, we would also like to extend our evaluation in our future work to be based on the above scheme.

4.4. Metrics

In most recommendation contexts users are only recommended a certain number of items at a time. This limit on the number of predictions must be reflected by the metrics used to evaluate the model in offline experiments. Moreover, in our evaluation framework (and that of Donkers et al.) the model's task is to predict the single next item. As such, traditional metrics relying on ratings (e.g. RMSE, Mean Absolute Error) or reflecting the presence of multiple relevant items (Mean Average Precision) are not directly applicable.

For each instance of recommendation, the model is to capture the user's next (and the only relevant) item among the limited number of guesses available to it. We reflect this via *Recall at 20* (Recall@20), defined as the percentage of cases where the next item consumed by the user was among the top 20 recommendations. Moreover, in many cases not all items are presented to the user at once. For instance, recommendations may be presented to the user in the form of a scrollable recommendation strip, where only a subset of recommendations is visible at a time. As such, items that are more likely to be relevant to the user should be ranked higher, placing them at the start of the recommendation list. This is reflected by the choice of our second metric *Mean Reciprocal Rank at 20* (MRR@20), defined as the mean of the *Reciprocal Ranks at 20* (RR@20) over all interactions. RR@20 itself is a measure of the correctness of a single ranking, defined as 0 if the true item was not among the 20 recommended ones and the inverse of its rank otherwise. The above metrics reflect the two important aspects of recommendation and as such are widely used in sequential recommendation literature [33, 52, 53]. We choose a steep ranking penalty of Reciprocal Rank in place of a more forgiving one (e.g. Normalised Discounted Cumulative Gain) as the model's ability to include the target item at lower position of recommendation is already captured by Recall@20. Moreover, note that the number of recommendations 20 is chosen in order to maintain the similarity with Donkers et al.

However, a criticism can be made that the above metrics are skewed towards more active users who have more interactions contributing to the metrics. As such we introduce two additional metrics, UserRecall@20 and UserMRR@20. These are analogous to Recall@20 and MRR@20 but are instead calculated for each user individually and then averaged over all users. As such, users with short histories are considered to the same extent as users with longer histories. These user based metrics, along with Recall@20 and MRR@20, are as defined below.

$$\text{RR@20} = \begin{cases} \frac{1}{\text{rank}(i^{(T+1)})} & i^{(T+1)} \in \text{pred}^{(T)} \\ 0 & \text{otherwise} \end{cases} \tag{4.3}$$

$$\text{Recall@20} = \frac{1}{\sum_{v=1}^{U} |I_v|} \sum_{v=1}^{U} \sum_{T=1}^{|I_v|} i^{(T+1)} \in \text{pred}^{(T)} \tag{4.4}$$

$$\text{MRR@20} = \frac{1}{\sum_{v=1}^{U} |I_v|} \sum_{v=1}^{U} \sum_{T=1}^{|I_v|} \text{RR@20} \tag{4.5}$$

$$\text{UserRecall@20} = \frac{1}{|U|} \sum_{v=1}^{U} \left(\frac{1}{|I_v|} \sum_{T=1}^{|I_v|} i^{(T+1)} \in \text{pred}^{(T)} \right) \tag{4.6}$$

$$\text{UserMRR@20} = \frac{1}{|U|} \sum_{v=1}^{U} \left(\frac{1}{|I_v|} \sum_{T=1}^{|I_v|} \text{RR@20} \right) \tag{4.7}$$

In the above formulation, U refers to the set of all users, whilst I_v denotes the ordered list of items consumed by a particular user v. At each point $T \in \mathbb{Z} : 1 \leq T \leq |I_v|$ the aim of the model is to predict $i^{(T+1)}$ - the next item user v consumed after $i^{(T)}$. As such, $\text{pred}^{(T)}$ refers to the set of 20 highest scoring predicted items returned by the model following the consumption of $i^{(T)}$. Furthermore, $\text{rank}(i^{(T+1)})$ denotes the rank of the target item $i^{(T+1)}$ among the 20 predictions made at step T. Note that the last item in the user history is not utilised as input during evaluation due to the lack of a corresponding target item. Finally, as described in the Section 4.2, scores obtained for individual interactions belonging to *Videoland* dataset are further scaled by $w_{\text{val,test}}$.

4.5. Regularisation

Neural Network based models are known for their susceptibility to overfitting [130, 146]. We thus propose three aspects to regularising our model. Firstly, we applied Dropout to user, item and time embedding vectors [130], with separate probabilities of keeping an individual neuron for each one of the three. Secondly, we applied L_2 regularisation on the user side, as described by Donkers et al. The authors suggest that this may serve a twofold goal of reducing overfitting as well as decreasing the effects of supplying the model with the same user embedding at every time step. Finally, model performance on validation set was repeatedly measured during training at a predetermined frequency. Training was then stopped prematurely when a chosen metric would fail to improve over a given number of steps. Model parameters used during the evaluation where the model achieved the best result would be then exported as the final parameters. UserMRR@20 was chosen as the early stopping metric due to sensitivity to both correct item being present in the recommendations as well as the ordering within the ranking. While this behaviour is also captured by MRR@20, we ultimately believe that the goal of the recommender system should be to maximise the satisfaction of all users, not the select few.

4.6. Implementation and hardware

Our models along with the baseline were implemented using TensorFlow 1.13 [1]. The implementation with support for single GPU and multi GPU training is released alongside this publication.[6] We also include our data preprocessing, training and evaluation code alongside our implementation. Training and evaluation for *MovieLens* and *LastFM* were performed using NVIDIA GeForce GTX 1080 Ti GPU for approximately 3 hours and 20 hours per run respectively. Training and evaluation of *Videoland* were instead performed in the SageMaker environment of Amazon using Tesla V100 GPU.

[6] www.github.com/NKNY/DeepTimeDelta

4.7. Hyperparameter selection

We performed an extensive grid search using the validation sets, analogous to Donkers et al. [33].[7] Evaluated parameter values are presented in Appendix Table B.1. Out of the evaluated values, the following set of parameters was found to be optimal for every setting:

Table 4.1: Dataset and model agnostic hyperparameters.

Parameter	Value
Optimiser	ADAM
Learning rate	0.001
Initialisation distribution	Uniform
Initialisation range	0.1
Sampling softmax	False

We chose ADAM [75] (with its TensorFlow default parameters) due to its use of momentum and learning rate decay as well as its widespread use in literature [31, 33, 67]. It was found that the learning rate of 0.001 was optimal as decreases to its value led to no improvement in performance whereas increases of the value led to the training loss frequently diverging to infinity. Finally, sampling softmax [62] was originally considered as a solution for dealing with the large number of items and associated memory constraints of *LastFM*. We found that this improvement came at a cost of a substantial drop of performance, unless when applying low amounts of downsampling. As such, sampled softmax was not used.

Furthermore, the choice of certain hyperparameters was influenced by computational constraints imposed by specific datasets. These parameters were assigned to be different for each dataset and are presented in Table 4.2. Notably, our internal experiments generally indicated a positive correlation between the size of the embeddings used and the performance of both the baseline as well our models. However, a high number of users or items in a dataset was associated with high memory requirements arising from storing embedding matrices on the GPU, as previously alluded in Section 4.2. As such, a dataset specific upper limit was set for the embedding sizes considered.

The duration of the training and the parameter indicating the number of epochs without metric improvement needed to trigger early stopping were chosen based on the shape of the loss and metric curves of the validation set. These hyperparameters were chosen as to ensure that the training would not be stopped prematurely while any metrics were still improving. Moreover, the frequency of evaluation was chosen to be sufficiently high as to decrease the risk of skipping over parameters leading to a higher validation set performance.

Table 4.2: Hyperparameters related to execution or chosen on the basis of computational constraints. Max num units refers to the size of embeddings and hidden states.

Parameter	LastFM	MovieLens	Videoland
Max num units	500	1000	200
Batch size	100	1000	2000
Max num epochs	50	50	20
Evaluation frequency (epochs)	0.25	0.25	0.5
Early stopping (epochs)	10	10	5

Finally, the best combinations of the remaining hyperparameters were chosen for each model. As there generally was no one set of parameters optimising all metrics at once, the hyperparameter choice was made by combining the rankings produced on the basis of each of the metrics using Borda count [37]. Unlike the case for the stopping metric, no one metric had to be prioritised over the others. Borda count was chosen due to its low score decay, allowing to find a consensus between four metrics. The hyperparameters chosen for each of the datasets are shown below in Table 4.3.

[7] While some authors e.g. [13] suggest that using random search is preferred over grid search, we chose grid search as to perform the parameter selection in a structured and non stochastic way.

Table 4.3: Dataset-specific hyperparameters of different models.

Dataset	Model	Initial state	Dropout keep probability			Time embedding	
			Δt	user	item	d	f
LastFM	0	0	-	0.5	0.8	-	-
	1	0	0.5	0.5	0.5	1	sigmoid
	2	0	0.5	0.5	0.5	1	sigmoid
	3	0	0.5	0.5	0.5	1	sigmoid
	4	0	0.5	0.5	0.5	0	sigmoid
	5	0	0.5	0.5	0.5	1	sigmoid
	6	0	0.5	0.5	0.5	1	ReLU
	7	0	0.5	0.5	0.5	1	ReLU
MovieLens	0	0	-	0.5	0.8	-	-
	1	0	0.5	0.5	0.5	1	ReLU
	2	0	0.5	0.5	0.8	1	sigmoid
	3	0	1	0.5	0.8	1	sigmoid
	4	0	1	0.5	0.8	1	sigmoid
	5	0	1	0.5	0.8	1	sigmoid
	6	0	0.5	0.5	0.8	1	sigmoid
	7	0	1	0.5	0.8	1	sigmoid
Videoland	0	Trainable	-	0.5	0.8	-	-
	1	Trainable	0.8	0.5	0.8	1	sigmoid
	2	Trainable	0.8	0.5	0.8	1	ReLU
	3	Trainable	0.8	0.5	0.8	1	sigmoid
	4	Trainable	0.8	0.5	0.8	1	sigmoid
	5	Trainable	0.8	0.5	0.8	1	ReLU
	6	Trainable	0.8	0.5	0.8	1	ReLU
	7	Trainable	0.5	0.5	0.8	1	ReLU

In the above table Dropout keep probabilities refer to separate probabilities of keeping each individual neuron in the user, item and time gap embedding vectors, as discussed in Section 4.5. Trainable Initial state refers to the model being able to learn the hidden state used as input for the first time step in each input sequence. Alternatively, the value of 0 denotes the alternative approach, whereby the model receives the vector of zeroes throughout the training. Allowing the model to learn the initial state may allow it to be set to a hidden state associated with generic recommendations. In contrast, when relying on all zero initial hidden state the model has to adapt its parameters to accept divergent inputs of the zero hidden state at the start of the sequence and non zero values thereafter.

4.8. Statistical evaluation of Research Sub Question 1

Our first aim was to identify the best performing models, used as an indication of the beneficial modes of incorporating time delta information. We pooled results from distinct datasets as to represent the overall effects of temporal information on performance. Note that the below analysis was performed separately for each metric.

First, test set results of time delta and baseline models were obtained for each of the datasets. The results were first to be evaluated via a one-way ANOVA [39] in order to identify whether there appeared to be any significant differences across means of performances of different models. It was noted, however, that both the means and the variances of results appeared to be highly dependent on the dataset in question and strongly different across datasets, violating ANOVA's assumption of homoscedasticity [94]. In order to allow for a direct comparison, the results of different models were standardised per dataset, analogous to Kim et al. [74], allowing a subsequent application of a one-way ANOVA to the transformed data [39]. The omnibus test was followed by Dunnett's test (H_1 : time delta > baseline) in order to identify specific models demonstrating statistically significant improvement over the baseline [36]. Dunnett's test was chosen due to its design allowing for many-to-one comparisons whilst limiting the Family Wise Error Rate (FWER) associated with multiple

comparisons [55]. Note that the post-hoc test was only carried out for metrics for which the model factor was associated with significant differences in means.

The standardisation procedure was chosen due to our belief that small yet constant improvements over the baseline still indicate an overall improvement. This may be particularly relevant in cases where improvements take place only on a specific range of Δt values, with the extent of the improvement masked by the remaining interactions. On the other hand, the standardisation operation may also distort the obtained results to a certain degree. Namely, if the assumption that even minor improvements are significant as long as they are constant does not hold, standardisation may inflate the differences and make insignificant differences appear substantial.

Conversely, the standardisation procedure may have a masking effect leading to an inability to detect the presence of significant differences. For instance, all models may achieve a substantial improvement over the baseline, with one model scoring particularly high. In the above case standardisation may actually diminish the extent to which the remaining models are considered to be improving over the baseline. The presence of such outliers may in some cases lead to no models being considered significantly better than the baseline, particularly if the best results for different datasets are achieved by distinct models.

A nonparametric alternative to the above standardisation procedure is Kruskall-Wallis one way analysis of variance [83]. This omnibus test may be followed up with post hoc analysis, such as individual Mann-Whitney U test [93] with adjustments for FWER or related False Discovery Rate (FDR) [12]. Kruskall-Wallis test functions by applying the rank transform to the data, removing the previously described sensitivity towards outliers. However, the rank based approach also discards the scale of inter model differences. Whilst the approach is less susceptible to outliers, it also loses information on the scale of relative improvement over the baseline, which may be important to capture. With this consideration in mind we chose the procedure relying on standardisation, as described above.

Finally, note that whilst Dunnett's test limits the FWER within the analysis associated with one metric, the probability of type I error may be further inflated due to presence of multiple evaluation metrics. Whilst it may be possible to apply further adjustments, such as Bonferroni correction [35] or Benjamin-Hochberg procedure [12], these methods may lead to a low overall power or there may exist a dependency structure between different metrics, complicating the analysis. Moreover, a dependency structure may be also present between different models. As such, we report p-values without any further adjustments. However, the reader may also wish to draw their own additional conclusions on the basis of the reported results.

4.9. Time gap ranges

Following the evaluation of general performance, the focus was then shifted to identifying the hypothesised association between particular types of time gap values and improvements in recommendation performance. In order to address Research Sub Question 2 we first split all interactions on the basis of the time between the given interaction and the preceding interaction. Each interaction's Δt was classified as either *low*, *medium* or *high*. As described previously, a time gap of the same length may have different interpretations based on the data from which it arises. As such, the time gap group ranges were defined separately for each of the datasets. The assigned values are presented in Table 4.4.

Table 4.4: Time gap duration groups and their dataset-specific values.

Dataset	Duration group		
	low	*medium*	*high*
LastFM	0sec - 2min	2min - 18h	18h+
MovieLens	0sec - 37min	37min - 30d	30d+
Videoland	0sec - 5min	5min - 14d	14d+

The choice of thresholds for *LastFM* was based on two aspects. Namely, we aimed to balance the ability to maintain an intuitive interpretation of time gaps with the need to maintain a reasonably large sample size. For instance, for short (*low*) time gaps to be consistent with our interpretation outlined in Methodology Section 3.2, their values have to be sufficiently small in order to be interpretable as negative signal. However, using too low of a threshold between *low* and *medium* values (e.g. on the scale of a few seconds) may lead to too few samples being assigned to the *low* group. As such, the 120 second cutoff point, corresponding to the 6^{th} percentile of all time gaps, was chosen as the upper bound for time gaps belonging to the *low* group. On

the other hand, the lower bound of 17.9 h (99$^{\text{th}}$ percentile) was set for interactions to be classified as *high*. As described in Section 3.2, we hypothesise that user preferences may drift following longer periods of inactivity. While user tastes may be less likely to drastically change following an absence of 18 hours, increasing the threshold would further decrease the low number of interactions classified as *high*.

In contrast, as previously described in Section 4.1, non-zero time gaps in *MovieLens* are distributed with two modes. We fit a Gaussian Mixture Model with two components on logarithmically transformed non-zero data as to distinguish between the densities. The value of 37 minutes, at which both Gaussians had equal densities, was chosen as the cutoff point between the two mixture components. The data from IMDb [8] suggests that only 1.9% of the movies do not exceed 37 minutes by duration. As such, based on our interpretation of short *MovieLens* time gaps described in Subsection 3.4), we propose that any interaction with a time gap equal to or below this value (including $\Delta t = 0$) be assigned to the *low* group. On the other hand, the minimum cutoff value of 30 days was chosen to represent a long user absence. We hypothesise that this period of inactivity may be sufficient for the user preferences to drift. The high threshold was chosen due to more frequent prolonged user absences, in contrast to the high activity of *LastFM* users. The total number of such *high* length interactions also amounts to one percent of dataset's interactions.

Finally, for *Videoland*, the *low* Δt threshold was set at 5 minutes while the *high* Δt threshold was chosen to be 14 days. These first limit was chosen as interactions with movies or shows of under 5 minutes may represent a user indicating their lack of preference for the content. Values below this threshold may encompass both the user instantly switching to a different item as well as consuming the item in part and then switching to a different one. The former may be an example of an accidental wrong click, whereas the latter may indicate that the user did not like the first item. Both cases suggest potential preference of the second item over the first. On the other hand, we chose the threshold of *high* duration time gaps for on the basis of the observed user inactivity behaviour and our belief that 14 days may be sufficient for a user preference shift. Finally, note that contrary to our publicly available datasets, threshold choices for *Videoland* were less influenced by the number of interactions due to the high availability of interactions with a wide range of Δt values.

4.9.1. Statistical evaluation of Research Sub Question 2

Similar to the procedure used for Research Sub Question 1 (Section 4.8), the aim was to identify models providing significant changes to performance for distinct types of time gaps. First, test set results were obtained for each metric for each dataset's three time gap categories. A three-way ANOVA [9] would not have a sufficient number of degrees of freedom and thus could not be directly applied to the data. Analogously to Research Sub Question 1, a two-way ANOVA [10] could still not be applied to the data due to the heteroscedastic nature of model performances across distinct time delta groups of one dataset. As such, we chose to standardise the results within a specific combination of dataset, time delta group and metric, subsequently applying a separate two-way ANOVA [11] for each of the metrics. Individual post hoc Dunnett's tests (H_1 : time delta > baseline) were then applied to the data in order to identify models exhibiting a significant improvement over the baseline for each of the Δt groups and without additional adjustments for FWER or FDR, analogous to Section 4.8.

4.9.2. Fine grained analysis of influence of time gaps

In order to gain further insight into the functioning of the baseline and time-dependent models, a qualitative analysis focusing on the lower level relationship between time gaps and the associated model behaviour was set up. This was done by focusing on a higher number of more compact ranges of Δt. Namely, we partitioned interactions on the basis of Δt into 100 logarithmically spaced bins. Following that, for each of the statistics, its mean value for each bin was calculated along with the associated 95% confidence interval and plotted as a function of each bin's time gaps. The statistics were then used as a basis of visual exploratory analysis with the aim of identifying qualitative underlying trends between the model function and associated time gaps. We identified three distinct groups of statistics able to provide a more detailed understanding of the model behaviour:

[8] Information courtesy of IMDb, (http://www.imdb.com). Used with permission.

[9] Three-way ANOVA with the dataset, time delta group and model as independent variables and a given metric as a response variable.

[10] Two-way ANOVA with time delta group and model as independent variables and a given metric as a response variable applied to data standardised for each dataset.

[11] Two-way ANOVA with time delta group and model as independent variables and a given metric as a response variable applied to data standardised for each dataset for each time delta group.

- Model performance.
- Similarity between recommendations and previously consumed items.
- Gating vector activations.

Firstly, the focus was again turned to model performance. However, in this case the performance was examined over a wider range of more concentrated time delta values. We decided to exclusively focus on MRR@20 as it reflects both the ranking order of recommendations as well as the binary presence of the target item in the recommendations. UserRecall@20 and UserMRR@20 were not considered for this role as users could not be guaranteed to have a sufficient number of interactions in each of the bins.

Following that, we set out to analyse the relationship between the time gap preceding a consumption event and the items recommended following that interaction. This was done by examining the similarity of recommendations to the item consumed prior to generating the recommendations as well as their similarity to the preceding item. Moreover, we also include the similarity between the user consumed items themselves as to provide further context to the item-recommendation similarity information. The aim of the above analysis was to provide a more interpretable item focused understanding of the model behaviour as a function of specific time gaps. We measure inter item similarity using Euclidean distance of embedding vectors. Distance between a consumed item and the 20 recommendations generated following the consumption of that item is calculated as the mean of individual item-recommendation distances. In theory, the item embeddings may lie on a more complex data manifold, making them less amenable to our analysis. However, we observe that items sharing similarities with other items also appear more likely to be close to those items in the Euclidean space, similar to the word embeddings learned in NLP [96]. Furthermore, note that only cases where the item before and after time gap are different are considered by the above analysis. Finally, as embeddings learned by distinct model may differ, we use embeddings of the sequential baseline model for the analysis of inter item distances for all models.

Thirdly, our aim was to interpret the fashion in which the size of the time gap may affect the decisions taken by the model. *DeepTimeDelta* models, together with the baseline model, employ a number of gating vectors limiting the effects of other components, guiding the hidden state update process: $\boldsymbol{\xi}$, $\mathbf{u}$, $\mathbf{r}$. As all components of those gating vectors lie between 0 and 1, it is possible to summarise their values using summary statistics, describing the amount of input allowed to pass through the gate. Vector mean was considered for this role along with the following percentiles: $\{5, 10, 25, 50, 75, 95\}$. Vector mean was ultimately adopted as it was found to provide the best insight into the observed results, with the notation $\bar{\boldsymbol{\xi}}$, $\bar{\mathbf{u}}$, $\bar{\mathbf{r}}$ used to denote the respective means of $\boldsymbol{\xi}$, $\mathbf{u}$ and $\mathbf{r}$. The update vector $\mathbf{k}$ was not included in the analysis, as unlike the other vectors, tanh cannot be simply interpreted as only letting a part of the input through, complicating the interpretation of its effects on recommendation.

The main limitation of the above set of exploratory analyses is that they may be unable to guarantee an exhaustive relationship between the time gap and the model behaviour. Namely, gating vector activations may lead to complex interactions, reducing the interpretability of their effects on the recommendation. For instance, whilst a high value of $\mathbf{u}$ may lead to a stronger divergence from the current hidden state and a high value of $\mathbf{r}$ may lead to a strong dependency on the hidden state, the combined effect of the two is unclear. Such interactions may limit the interpretability of effects of changes to the activations on the final produced recommendations. Furthermore, certain components of the embedding vectors may be more crucial in updating the hidden state than others. Our approach discards such semantics and treats all components equally. Karpathy et al. [72] demonstrated that it is possible to identify individual neurons responsible for specific tasks in NLP. However, their approach relies on manual discovery of such patterns. Whilst applicable to NLP due to the generally unambiguous roles of individual words in a sentence, the use of this approach on recommendation data is further complicated by the challenging interpretation of latent semantics of users and items. Thirdly, in case of large differences between item embeddings originating from different models, employing the baseline model's item embeddings may complicate the interpretation of a given model's activations' effects on recommendation. However, differences between the same model's recommendations and mean activations for distinct time gaps may still be interpretable and may allow to examine general temporal trends within the same model.

Finally, note that for *Videoland*, as described in Section 4.2, models were trained such that only cases where the item to predict was not the item just consumed contributed to parameter updates. Analogously, only cases where the item after the time gap and the target item were different were also included in the analysis described above for *Videoland*.

4.10. User grouping

In order to address Research Sub Question 3, two distinct user partitionings were obtained. Firstly, for a given dataset its users were categorised on the basis of their mean time gap of all their interactions. Mean user time gap duration ($\overline{\Delta} t_u$) was chosen above other statistics due to its ability to capture both inter as well as intra session behaviour. For instance, *LastFM* users on average consume more than 2 songs per listening session. In this case, a user's mean time gap duration would partially reflect both the duration of their listened songs as well as the duration of their average inactivity. On the other hand, median user Δt, another statistic considered, would only effectively describe the song duration within the session. Note that in case of large time gaps the mean may be skewed towards the large time gaps on the logarithmic scale. This is intended as whilst we are interested in capturing the user intra session behaviour, we place higher emphasis on capturing the user's inter session behaviour. The limitation of using the mean is that multiple distinct behaviours may result in the same mean Δt value. Other statistical measures such as the first or third quartiles of the user's Δt distribution may be used instead. However, as the appropriate statistic may be strongly dependent on the user behaviour trends in each dataset, we choose $\overline{\Delta} t_u$ due to its expected general applicability.

The second split was performed on the basis of the calculated mainstreamness of each user. Inspired by the work of Bauer and Schedl [9] dealing with song mainstreamness, we adapt their mainstreamness measure $M_{R,APC}^{global}(v)$. This measure is based on artist play counts (*APC*), which is a vector of length corresponding to the number of artists. The values of *APC* indicate the number of times each artist's tracks were listened to by all users in the dataset. Moreover, the authors define a separate measure $APC(v)$, also of dimensionality $|I|$, to reflect the artist's play counts by a specific user.

As there is no defined artist available for *MovieLens* and *Videoland*, we instead define our own related measure, replacing artist play counts with item consumption counts (*ICC*). As such, each user can be represented with an $|I|$-dimensional vector $ICC(v)$, where an individual value $ICC(v)_j$ corresponds to the number of times that user consumed item I_j. In a similar fashion, *ICC* is the measure of each item's consumption counts over the whole dataset, analogous to *APC*. With the above in mind, we propose our measure of user mainstreamness for a given user v:

$$M_{R,IIC}(v) = \tau\Big(\text{ranks}(IIC(v)), \text{ranks}(IIC)\Big) \tag{4.8}$$

In the above definition *ranks(IIC(v))* denotes a rank transformation of a user v's consumption counts. Our measure, analogous to Bauer and Schedl [9], relies on Kendall's Tau (τ) to compare the such item rankings to the global ranking. For a given user, τ close to 1 indicates strong similarity to overall play counts (mainstream) while values close to -1 indicate disagreement with the dataset's total play counts (niche).

For each of the above measures every user was classified as belonging to a *low*, *medium* or *high* group for that measure, similar to Research Sub Question 2. Note that in this instance the classification was performed on the user level, as opposed to individual interactions. The thresholds between user groups were chosen such that each group would contain one third of the user base. The thresholds for the above measures are shown in Tables 4.5 and 4.6.

Table 4.5: Consumption frequency user groups and their dataset-specific $\overline{\Delta} t_u$ values. *Videoland* values not included due to the sensitivity of the data.

Dataset	Consumption frequency group		
	low	*medium*	*high*
LastFM	0sec - 30min	30min - 77min	77min+
MovieLens	0sec - 20sec	20sec - 18min	18min+

Note that, in theory, more dataset specific definitions of *low*, *medium* and *high* may instead be used for either of the groupings. These may be chosen manually based on interpretations of the meanings of the thresholds or via modelling techniques such as Gaussian Mixture Models, discussed in Section 4.1. However, in our view, a mean user time gap may have a different interpretation from that of the time gap of a single interaction. For instance, a time gap of 15 minutes before a song in *LastFM* likely indicates that the user consumed a song fully or partially and then left the platform for a short period of time. However, $\overline{\Delta} t_u = 15$ min may, in addition to the repeated behaviour described above, may also indicate a user consuming a number of songs in succession, followed by a prolonged absence. Due to the plurality of associated scenarios $\overline{\Delta} t_u$ is

Table 4.6: Mainstreamness user groups and their dataset-specific τ values. *Videoland* values not included due to the sensitivity of the data.

Dataset	Mainstreamness group		
	low	*medium*	*high*
LastFM	(-0.02) - 0.02	0.02 - 0.04	0.04 - 0.13
MovieLens	0 - 0.08	0.08 - 0.13	0.13 - 0.50

less interpretable for the more manual threshold selection, as done in Section 4.9. As such, we choose not to select thresholds in such a way and instead use equal user numbers for each group.

Finally, model test set performance was obtained for each group of each grouping measure. Following that, a statistical analysis, analogous to the one described in Subsection 4.9.1 was carried out. Standardisation was applied to the results such that for a given measure, model performances for one individual group (*low*, *medium* or *high*) were transformed in conjunction, separately for each dataset. This was then ultimately followed by separate two-way ANOVA tests [12] and Dunnett's post hoc tests (H_1 : time delta > baseline) for each metric for each user grouping measure.

[12] Two-way ANOVA with the given measure's user group and model as independent variables and a given metric as a response variable applied to results for the given grouping measure standardised for each dataset for each user group

5

Results and Discussion

5.1. Effect of model choice on overall performance

In order to determine beneficial modes of incorporating time gap information associated with increased performance, models were evaluated on the previously withheld test sets, as described in Section 4.8. The results of this evaluation are presented in Table 5.1. Following the above analysis, statistical evaluation consisting of one-way ANOVA and subsequent Dunnett's tests was performed as described in Section 4.8. The results of the statistical analysis are presented to the reader in Appendix Table B.2 and Table 5.2.

Table 5.1: Full dataset evaluation metrics of baseline and *DeepTimeDelta* models.
Best performance for the given dataset in terms of the chosen metric denoted in bold.

Dataset	model	Recall@20	MRR@20	UserRecall@20	UserMRR@20
LastFM	0	0.3313	0.2396	0.3117	0.2281
	1	**0.3446**	0.2409	**0.3349**	0.2335
	2	0.3396	**0.2463**	0.3233	**0.2364**
	3	0.3150	0.2185	0.2946	0.2005
	4	0.3345	0.2440	0.3177	0.2338
	5	0.3423	0.2386	0.3286	0.2305
	6	0.3186	0.2134	0.3032	0.2003
	7	0.3342	0.2318	0.3192	0.2237
MovieLens	0	0.2146	0.0680	0.2664	0.0899
	1	**0.2287**	**0.0734**	**0.2776**	**0.0961**
	2	0.2206	0.0700	0.2706	0.0913
	3	0.2070	0.0642	0.2603	0.0869
	4	0.2259	0.0728	0.2736	0.0949
	5	0.2156	0.0685	0.2669	0.0920
	6	0.2185	0.0677	0.2721	0.0914
	7	0.2185	0.0693	0.2708	0.0922
Videoland	0	0.6227	0.3551	0.5672	0.3192
	1	0.6247	0.3632	**0.5680**	0.3234
	2	**0.6250**	**0.3640**	0.5671	0.3236
	3	0.6236	0.3628	0.5655	0.3229
	4	0.6239	0.3635	0.5674	0.3235
	5	0.6243	0.3637	0.5676	**0.3238**
	6	0.6249	0.3633	**0.5680**	0.3231
	7	0.6247	0.3633	0.5679	0.3230

Table 5.2: Full dataset offline evaluation Dunnett's test p-values.

H_1	Recall@20	MRR@20	UserRecall@20	UserMRR@20
1 > 0	0.0081	0.0671	0.0042	0.0275
2 > 0	0.0355	0.0986	0.3805	0.1008
3 > 0	0.9849	0.9220	1.0000	0.9260
4 > 0	0.1094	0.0581	0.2369	0.0389
5 > 0	0.1583	0.2618	0.3122	0.1048
6 > 0	0.3517	0.7965	0.4410	0.6340
7 > 0	0.1341	0.3450	0.1864	0.2224

Firstly, we note substantial differences in the distributions of the results between different datasets, presented in Table 5.1. Namely, all models consistently demonstrate the highest mean performance on *Videoland* across all metrics (e.g. mean UserMRR@20 of 0.323). A lower set of scores is attained for interactions of *LastFM* (0.223), with *MovieLens* proving to be the most challenging (0.092). On the other hand, the obtained results are also spread out the most for *LastFM* (UserMRR@20 $\sigma = 0.015$), with inter model scores of *MovieLens* being dispersed to a lesser extent ($\sigma = 0.015$) and even less so for *Videoland* ($\sigma = 0.001$). We hypothesise that prediction may be easier following certain values of time gaps over the others, as reported by Jing and Smola [63]. As time gap distributions vary between different datasets, certain datasets may contain a higher percentage of interactions with time gaps for which models achieve a higher recommendation accuracy. Furthermore, there also appears to be a minor bias towards users with long histories for *LastFM* and *Videoland*, as all models exhibit a lower performance on user based metrics compared to their non user equivalents, with the opposite being true for *MovieLens*. This observation is likely linked to the training set distribution of interaction counts for each user, as consumption histories of individual *MovieLens* users are shorter compared to those of the remaining datasets.

In terms of individual models, we observe that no one model or equation achieves the best result across all datasets and metrics. A combination of models 1, 2 and 4 dominate the performance on *LastFM* and *MovieLens*, with less consistent improvements also observed for models 5 and 7. Model 1, in particular, achieves the best performance on *MovieLens* over all metrics, with an average increase of 6.4% over the baseline model. The model also achieves the best results on Recall@20 and UserRecall@20 of *LastFM* (average improvement of 5.7%), while MRR@20 and UserMRR@20 are best served by model 2 (average improvement of 3.2%). [1] On the other hand, for *Videoland* there was no clear best model, with model 2 narrowly achieving the highest combined performance (as measured by Borda count on the basis of all metrics), closely followed by models 5, 6, 1 and 7. Notably, with the exception of UserRecall@20, all time delta models demonstrated at least some improvement over the baseline model. However, the extent of the improvement was also generally lower than that observed for other datasets.

At the equation level, models modifying equations 1 and 2 appear to achieve the highest overall performance. Changes to those equations allow the model to leverage the temporal information in order to regulate the influence of other inputs on the hidden state update. The importance of the above mechanism is also reflected by the results of the statistical evaluation. Namely, model 1 was found to achieve a statistically significant improvement over the baseline in terms of Recall@20, UserRecall@20 and UserMRR@20 at 5% confidence level. Moreover, statistical analysis highlighted model 2 as significantly improving on Recall@20 and model 4 achieving the same feat on UserMRR@20. On the other hand, modifying equation 3 exhibited generally deteriorated results on both *LastFM* and *MovieLens* datasets compared to the baseline. Additionally, whilst providing some improvement over the baseline model on *Videoland*, the extent of the improvement observed for model 3 was lower than for other time delta models. Nevertheless, models 5 and 6 respectively demonstrate the highest *Videoland* UserMRR@20 and UserRecall@20, with the former also achieving the second highest Recall@20 and UserRecall@20 on *LastFM*. As such, effects of including time gap information in

[1] Note that our baseline model results for *MovieLens* are also approximately consistent with those reported by Donkers et al. [33], with our implementation of their model demonstrating a 4.5% improvement of Recall@20 and a 8.7% increase in MRR@20 over their reported results. However, the authors' stated results on *LastFM* appear to be substantially different from those reported by us. Namely, Donkers et al. indicate 30% lower Recall@20 and 28% lower MRR@20 compared to the results for their model presented in Table 5.1. We hypothesise that the source of the difference lies in the random user history subsampling, as described in the Experimental Setup Section 4.1. Our internal results demonstrated a level of result variability between independent random samples, consistent with the above difference. To our regret, the authors did not provide the sampled data nor their sampling code along with their publication nor during private correspondence.

equation 3 are more varied. Our interpretation of extending the above equation, as described in Subsection 3.3.3, is allowing the model to decay its recommendations towards a list associated with the given time gap value. Based on our intuition, such an effect may be particularly beneficial for larger time gap values, whilst being less relevant for items consumed in a quick succession. As such, we hypothesise that the differences discussed above arise from the prevalence of sufficiently large time gaps, with sufficiency being determined by the underlying user behaviour of the dataset.

Finally, the extent to which combining modifications to multiple equations improves recommendation remains unclear due to its high apparent dataset specificity. For instance, model 5 combining changes to equations 1 and 3 achieved higher *Videoland* MRR@20 and UserMRR@20 when contrasted with models modifying those equations in isolation. On the other hand, whilst model 4 achieves the second highest result on *MovieLens*, model 1 achieves a higher performance by extending only the first equation. Analogously, whilst further extending model 4 with a modification to equation 3 allows model 7 to achieve higher Recall@20 and UserRecall@20 on *Videoland*, such an extension generally provides a negative effect on the remaining two datasets.

5.2. Effect of time gap value on model performance

Following that, we turned to establishing which types of time gaps benefit from the use of time-dependent models. In order to address the above, performance of all models was analysed on the basis of predictions made following interactions with either *low*, *medium* or *high* duration time gaps, as defined in Section 4.9. The results of such evaluation are presented below in Table 5.3. This was followed by ANOVA on standardised data (as described in Subsection 4.9.1). The associated p-values and the percentage of explained variation are presented in Appendix Table B.3. The omnibus test was then followed up with Dunnett's post hoc tests (Table 5.4).

Analogous to the observed discrepancies in overall mean scores achieved for individual datasets in Section 5.1, there also exists a difference between the relative difficulties of distinct time gap groups across datasets. For instance, models generally achieve their best *LastFM* results on *low* time gaps, with marginally lower scores on *medium* interactions and a more significant loss of performance for the *high* group. For *Videoland*, the relationship is comparable to the above, with more similar levels of performance achieved on *low* and *medium*. The above observations are consistent with those of Jing and Smola [63] who report an inverse relationship of the interval between two sessions and their model's prediction accuracy. On the other hand, whilst the best *MovieLens* results are still consistently observed for the *low* group, models instead demonstrate their lowest scores on *medium* interactions. This in turn highlights the potential impact of the choice of the item domain as well as the data collection on the observed results.

On the model level, models 1 and 2 appear to achieve the most consistent improvement across all datasets, outperforming the baseline across all but one combination of datasets, metrics and groups. Models 4, 5 and 7 also demonstrate a strong performance on *MovieLens* and *Videoland*, with fewer cases of improvement on *LastFM* ($6-8$ out of 12 metric-group tuples). Finally, models 3 and 6, whilst generally outperforming the baseline on the other two datasets, fail to demonstrate any improvement on *LastFM*.

5.2.1. *Low*

In terms of individual time gap groups, inclusion of temporal information leads to no performance improvement or a minor improvement over the sole use of sequential information for predictions made following *low* duration inactivity. Importantly, whilst some dataset specific differences are present, models 1 and 2 are the only ones that achieve an improvement across most metrics. Model 1 demonstrates the largest UserMRR@20 increase over the baseline of 6.9% for *MovieLens*, whilst model 2 achieved the largest improvements of 4.4% and 1.6% for *LastFM* and *Videoland* respectively. These results suggest that limiting individual inputs may provide some minor benefit in adapting to the underlying temporal behaviour associated with items consumed in a quick succession. In contrast, as hypothesised earlier, the user agnostic decay term of equation 3 does not appear to provide sufficient expressiveness and as such only contributes towards smaller gains of MRR@20 and UserMRR@20 of *Videoland* observed for model 3.

However, regardless, the extent of the improvement of time delta models is generally the least substantial and the least consistent when contrasted to that of the remaining time gap groups. This is also reflected in the results of the statistical evaluation, whereby no model was found to show significant result increase over the baseline at 5% confidence level for any of the metrics on *low* time gaps. As such, no strong benefit is overall observed in utilising Δt information for improving prediction accuracy following short periods of inactivity.

Table 5.3: Offline evaluation metrics of baseline and *DeepTimeDelta* models on distinct time gap duration groups.
Best performance for the given dataset's time gap group in terms of the chosen metric denoted in bold.

Dataset	model	Recall@20			MRR@20			UserRecall@20			UserMRR@20		
		low	medium	high	low	medium	high	low	medium	high	low	medium	high
LastFM	0	0.3662	0.3293	0.2956	0.2807	0.2374	0.1841	0.3284	0.3111	0.2927	0.2501	0.2273	0.1836
	1	**0.3728**	**0.3429**	**0.3283**	0.2725	0.2393	0.1916	**0.3447**	**0.3352**	**0.3344**	0.2537	0.2333	0.1937
	2	0.3692	0.3378	0.3107	**0.2850**	**0.2442**	0.1890	0.3331	0.3227	0.3120	**0.2611**	**0.2355**	0.1925
	3	0.3291	0.3147	0.2503	0.2374	0.2179	0.1506	0.2916	0.2954	0.2501	0.2080	0.2008	0.1550
	4	0.3590	0.3332	0.2994	0.2785	0.2424	0.1698	0.3245	0.3183	0.2955	0.2531	0.2333	0.1761
	5	0.3488	0.3423	0.3082	0.2450	0.2386	**0.1987**	0.3261	0.3292	0.3099	0.2244	0.2312	**0.2026**
	6	0.3387	0.3178	0.2704	0.2297	0.2129	0.1563	0.3034	0.3034	0.2628	0.2108	0.2004	0.1583
	7	0.3556	0.3329	0.3132	0.2557	0.2306	0.1891	0.3200	0.3191	0.3061	0.2376	0.2230	0.1924
MovieLens	0	0.2258	0.1105	0.1706	0.0719	0.0326	0.0511	0.2673	0.1311	0.1706	0.0902	0.0400	0.0531
	1	**0.2405**	0.1190	0.1800	**0.0776**	0.0351	0.0549	**0.2787**	0.1403	0.1813	**0.0964**	0.0444	0.0578
	2	0.2306	0.1219	0.2035	0.0734	0.0370	0.0623	0.2710	**0.1517**	0.2051	0.0915	**0.0478**	0.0650
	3	0.2167	0.1132	0.1834	0.0675	0.0324	0.0541	0.2609	0.1363	0.1844	0.0872	0.0418	0.0561
	4	0.2360	**0.1251**	**0.2109**	0.0764	**0.0377**	**0.0651**	0.2740	0.1515	**0.2121**	0.0951	0.0473	**0.0681**
	5	0.2255	0.1184	0.1961	0.0721	0.0342	0.0567	0.2672	0.1424	0.1966	0.0923	0.0434	0.0586
	6	0.2280	0.1233	0.2028	0.0711	0.0346	0.0603	0.2726	0.1492	0.2017	0.0917	0.0449	0.0617
	7	0.2284	0.1203	0.2023	0.0728	0.0352	0.0616	0.2712	0.1472	0.2058	0.0924	0.0453	0.0655
Videoland	0	0.5659	0.6325	0.3219	0.2967	0.3644	0.1150	0.5733	0.5727	0.3305	0.3123	0.3238	0.1195
	1	0.5661	0.6345	0.3428	0.3013	0.3729	0.1261	0.5742	0.5733	0.3498	0.3171	0.3283	0.1304
	2	0.5660	**0.6349**	0.3422	**0.3017**	**0.3737**	0.1245	0.5735	0.5727	0.3488	**0.3172**	0.3287	0.1287
	3	0.5652	0.6333	0.3448	0.2999	0.3725	0.1264	0.5720	0.5709	0.3515	0.3155	0.3280	0.1310
	4	0.5653	0.6338	0.3396	0.3007	0.3732	0.1246	0.5733	0.5727	0.3467	0.3166	0.3286	0.1288
	5	0.5651	0.6340	**0.3479**	0.3007	0.3735	**0.1276**	0.5731	0.5729	**0.3547**	0.3164	**0.3288**	**0.1320**
	6	**0.5666**	0.6346	0.3437	0.3011	0.3730	0.1262	**0.5743**	0.5733	0.3506	0.3163	0.3281	0.1305
	7	0.5655	0.6345	0.3422	0.3005	0.3730	0.1249	0.5732	**0.5735**	0.3497	0.3161	0.3281	0.1290

Table 5.4: Dunnett's test p-values for offline evaluation on distinct time gap duration groups.

H_1	Recall@20			MRR@20			UserRecall@20			UserMRR@20		
	low	medium	high	low	medium	high	low	medium	high	low	medium	high
1 > 0	0.5510	0.0343	0.1139	0.2215	0.2520	0.3007	0.1102	0.2152	0.0556	0.0563	0.1157	0.2068
2 > 0	0.9647	0.0145	0.0356	0.3386	0.0406	0.1176	0.9384	0.2576	0.0198	0.2389	0.0130	0.0795
3 > 0	1.0000	0.9968	0.8161	1.0000	0.9830	0.9195	1.0000	1.0000	0.7078	1.0000	0.9377	0.8666
4 > 0	0.9990	0.0741	0.0501	0.3349	0.0377	0.1906	0.9624	0.3618	0.0367	0.1443	0.0229	0.1152
5 > 0	1.0000	0.0989	0.0313	0.9768	0.3359	0.0981	0.9996	0.4056	0.0162	0.8138	0.1473	0.0627
6 > 0	0.9988	0.1948	0.2110	0.9967	0.9036	0.5409	0.9715	0.6038	0.2158	0.9597	0.6765	0.5543
7 > 0	1.0000	0.0864	0.0350	0.9272	0.4233	0.1225	0.9973	0.2297	0.0214	0.6827	0.1943	0.0649

5.2.2. *Medium*

On the other hand, in the case of *Medium* time gaps, a wider range of time delta models achieved a more consistent improvement over the baseline. In addition to the gains demonstrated by models 1 and 2 (up to 3.6%, 19.5% and 1.5% UserMRR@20 improvement on *LastFM*, *MovieLens* and *Videoland* respectively), models 4 and 5 provide comparable increases in the majority of cases. Likewise, models 6 and 7 achieve an improvement on all metrics for *MovieLens* and *Videoland*. Notably, model 3 also demonstrates an improvement on 3 out of 4 metrics for each of the above two datasets, albeit the improvement is lower than that of the other time delta models.

With comparable or larger improvements compared to the *low* group, as well as a larger number of models achieving some level of improvement, models 1 and 2 were found to achieve a statistically significant improvement over the baseline in terms of Recall@20 at 5% significance level. Moreover, model 2, along with model 4 were both found to have demonstrated a significant *medium* time gap improvement over the baseline in terms of MRR@20 and UserMRR@20. The above thus indicates the positive effect the gating equations 1 and 2 are able to have on recommendation, with equation 3 providing a lesser or negative benefit.

5.2.3. *High*

Lastly, time-dependent models generally demonstrate the largest improvement following interactions with *high* time gaps. All time delta models outperform the baseline across all metrics for *MovieLens* and *Videoland* datasets, with models 1, 2, 5 and 7 achieving the same feat for *LastFM*. The highest overall improvements were observed for *MovieLens*, with model 4 showing a UserMRR@20 improvement of 28.2% over the baseline, followed by those of models 2 and 7. Model 5, on the other hand, in addition to demonstrating the largest *LastFM* improvements on MRR@20 and UserMRR@20 (7.9% and 10.3% respectively), also achieved the highest *Videoland* results across all metrics, exemplified by a 10.5% UserMRR@20 increase over the baseline. Notably, and in contrast to the remaining groups, the second highest *Videoland* UserMRR@20 improvement of 9.6% was also achieved by model 3. However, the model still failed to improve on *LastFM* and achieved some of the lowest improvements on *MovieLens*.

Out of the above models, 2, 5 and 7 achieved statistically significant mean result improvement for Recall@20 at 5% confidence level. The same models, along with model 4, were also found to significantly improve in terms of UserRecall@20. As such, we highlight the wide array of successful time delta models and modes of incorporating time gap information following long time gaps, including the use of non user specific decay of equation 3.

5.2.4. Limitations

Importantly, there also exist limitations on applicability of the above conclusions, stemming from the following sources. Firstly, as previously suggested in Section 4.8, a model may achieve an improvement over a given metric across all datasets. However, if there exists a model that achieves a further increase over the first model's performance, the standardisation procedure may decrease the overall impact of the improvement of the first model over the baseline. If there exists a better time delta model for multiple datasets, statistical evaluation may be less likely to identify the first model as providing a significant improvement. We hypothesise that, for instance, this may be the case for model 1 and its UserMRR@20 performance on *high* time gaps.

Likewise, the extent of the improvement over the baseline demonstrated by a certain model may be overshadowed by even more pronounced performance losses suffered by other models. This may be the case for model 1 and its UserRecall@20 results on *low* time gap interactions. Namely, whilst the model demonstrates the largest improvements on *LastFM* and *MovieLens*, while achieving a close second largest improvement on *Videoland*, the extent of the improvements is less than the difference between the baseline and the poorer result of model 3.

Thirdly, we hypothesise that the apparent lack of statistically significant results for low time gaps may also partially stem from the divergent implicit meanings of low time gaps across the datasets. Namely, whilst the shortest time gaps in *Videoland* may constitute an abrupt switch from one item to another, the same time gaps in *MovieLens* instead signify the quick rating assignment, which may be less interpretable as any form of positive or negative feedback. More generally, our results demonstrate the difficulty of combining datasets from different domains and user input types.

Finally, in our view, the above evaluation may have been influenced by the subjective choice of time gap thresholds. For instance, it may be possible that *Videoland* users generally take less than 14 days to exhibit a behaviour attributed to *high* duration absence in Section 3.2. In that case, the *medium* interaction group may also end up containing a number of such interactions, potentially skewing the results.

5.3. Fine grained analysis of effects of time gap value on model behaviour

To further improve our understanding of the underlying model behaviour as a function of the time gap, we proceeded to analyse the changes in models and their behaviour as a response to Δt partitioned into 100 exponentially increasing bins. Note that due to space and clarity constraints only a subset of models are presented in this section, chosen on the basis of performance as well as the diversity of recommendation behaviours and activations. Figures featuring all models are included in the Appendix.

5.3.1. *LastFM*

5.3.1.1 MRR@20

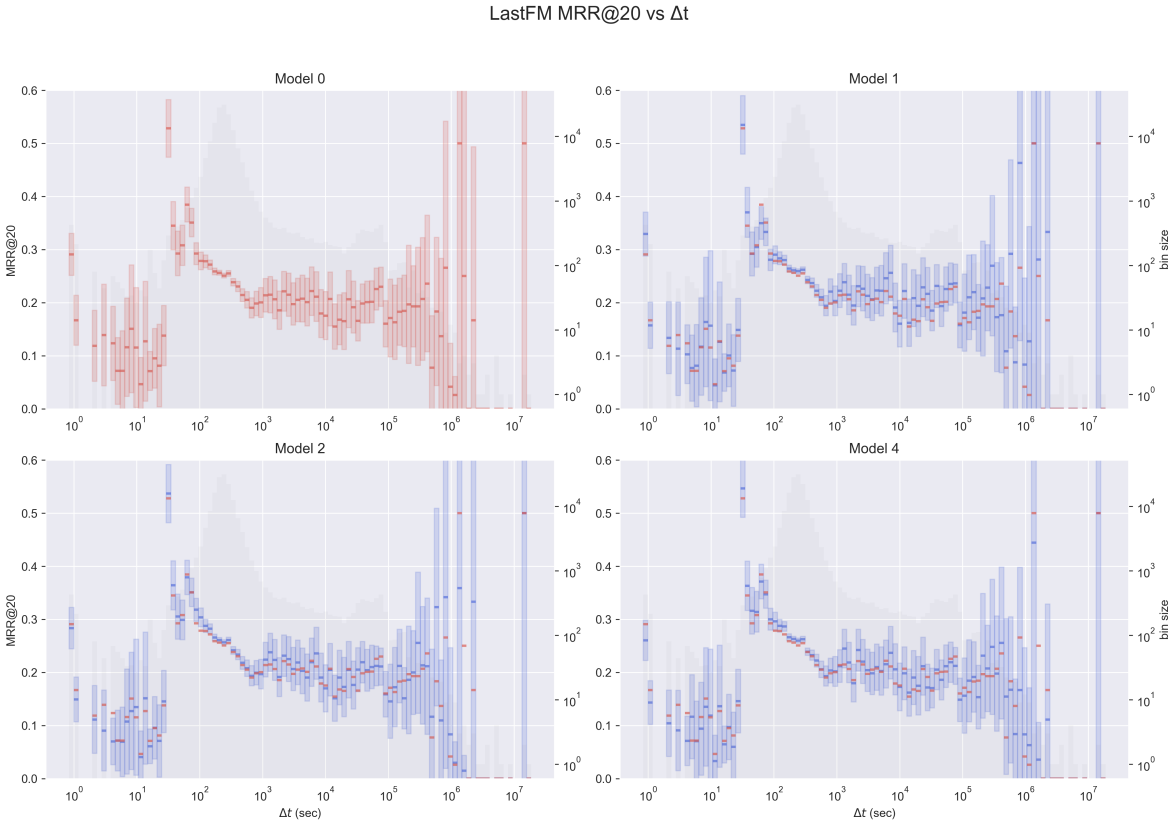


Figure 5.1: MRR@20 and the associated 95% confidence intervals of models 0, 1, 2, 4 for varying time gap duration time on *LastFM* dataset. Performance averaged over individual interactions falling into exponentially expanding bins. Model 0 presented in red, *DeepTimeDelta* models presented in blue. Model 0 MRR@20 included alongside that of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

As seen in Figure 5.1, *LastFM* interactions of both sequential as well as time delta models roughly fall into three categories on the basis of MRR@20. The overall lowest performance is observed for interactions with $\Delta t < 10^{1.5}$ sec (30 sec), followed by a rapid performance increase and a subsequent decline at $\Delta t = 10^{2.4}$ sec (4 min). Finally, from $\Delta t = 10^{2.8}$ sec (10 min) onwards, only minor fluctuations in performance are observed. Whilst not leveraged in this work, the above breakpoints between the bin clusters may be employed as an alternative choice of thresholds outlined in Section 4.9.

Importantly, the apparent plateau of performance for $\Delta t > 10^{2.8}$ sec suggests that users may overall exhibit comparable listening patterns following any medium-to-long period of inactivity. This raises the possibility that *LastFM* users may actually not interact with novel tracks even after prolonged absence, as hypothesised earlier. Moreover, the overall performance trends exhibited by both sequential as well as time delta models appear to be concordant. The latter suggests that user behaviour may be roughly captured by the sequential information, with time gaps playing a role in additional fine tuning. Whilst outside the scope of the current project, future work may wish to extend the above visual analysis to include static models as to identify the

precise value of sequential ordering.

Nevertheless, time delta models achieve consistent increases in MRR@20 over the baseline (up to 9.1% for model 2) for bins surrounding the mode of the interaction distribution. Likewise, time delta models, with model 1 in particular, may also provide a level of improvement on interactions with Δt on the order of hours or single days, consistent with the findings for *medium* and *high* groups in Section 5.2. However, additional work (e.g. on the full *LastFM* dataset) is required to clarify the exact ranges of such larger time gaps associated with improvements in performance,as well as for bins with $\Delta t < 10^{1.5}$ sec due to the wide confidence intervals arising from small bin counts.

5.3.1.2 Embedding distances

Following that, the focus was shifted to examining potential differences in recommendation. More precisely, embedding vectors of items directly before and after the time gap were compared to the embeddings of recommendations produced upon consumption of the second item ($\bar{p}$) in terms of Euclidean distance, as described in Subsection 4.9.2. We refer to the mean distance between the item before the time gap and each of the recommendations as $|i^{(T-1)} - \bar{p}|_2$. Analogously, $|i^{(T)} - \bar{p}|_2$ refers to the mean distance between the item following the time gap and the recommendations. Finally, we also include $|i^{(T)} - i^{(T-1)}|_2$, denoting the distance between the two items, for additional context. Interactions were binned on the basis of associated time gaps, as done for MRR@20.

As seen in Figure 5.2, items consumed with an interval of $3 < \Delta t < 10^{1.5}$ sec are found to be substantially further apart compared to any time gap range (high $|i^{(T)} - i^{(T-1)}|_2$). In contrast, items consumed in rapid succession from one another ($\Delta t \leq 3$ sec) are observed to be more similar. We interpret both of the above cases as the first item not being relevant to the user. Moreover, we hypothesise that the latter may be an example of the user immediately switching to the next track in the playlist or album, whilst the former may also be related to the user making an active choice of the next track. However, contrary to our expectations, no strong local divergence in terms of item-to-recommendation distances was observed between the sequential and time-dependent models for the above interaction types. Whilst both $|i^{(T-1)} - \bar{p}|_2$ as well as $|i^{(T)} - \bar{p}|_2$ of model 1 are lower compared to their baseline counterparts, the same shifts are also observed across all other Δt values, suggesting an overall absence of short time gap specific adaptation for time delta models.

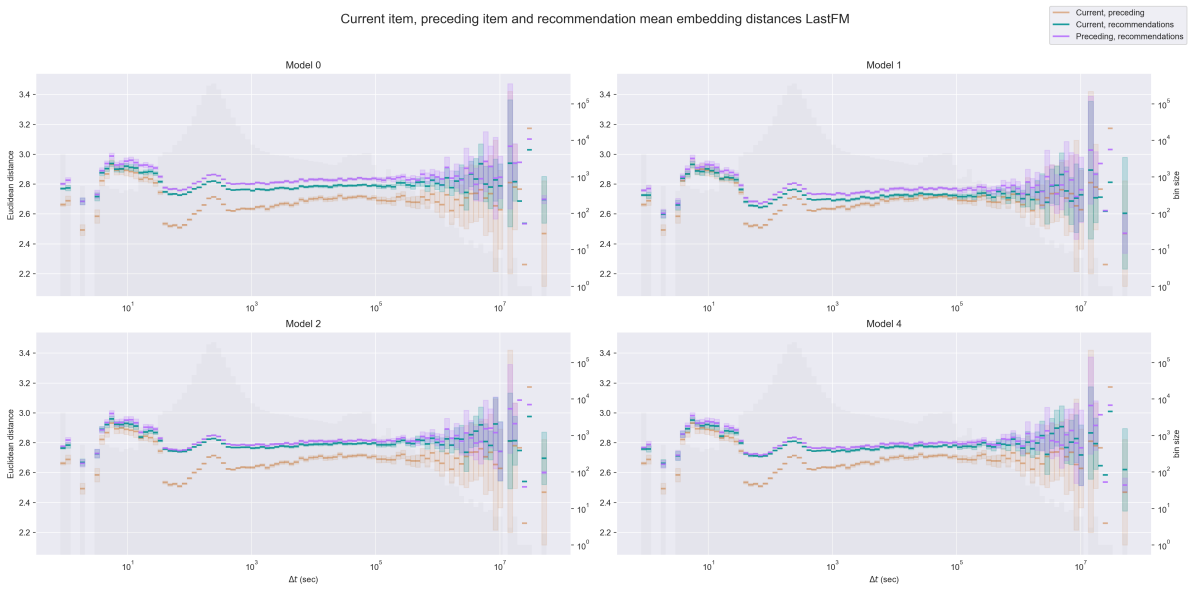


Figure 5.2: $|i^{(T-1)} - \bar{p}|_2$, $|i^{(T)} - \bar{p}|_2$ of models 0, 1, 2, 4, as well as $|i^{(T)} - i^{(T-1)}|_2$ with their associated 95% confidence intervals for varying time gap duration on *LastFM* dataset. Distances averaged over individual interactions falling into exponentially expanding bins. Number of interactions assigned to each bin shown in grey.

For absences of longer durations, we also observe a minor and decelerating increase in the inter item as well as item-to-prediction distances for interactions separated by more than $10^{2.8}$ sec. As such, as hypothesised earlier, in place of novel items, we find that users generally choose to engage with the same kind of

content they had consumed previously. Whilst users may discover new tracks during their listening session, their inter session behaviour is reminiscent of the *Filter Bubble*, explored more closely in Pariser [105].

5.3.1.3 Activations

Finally, mean gating vector activations $\bar{\boldsymbol{\xi}}$, $\bar{\mathbf{u}}$ and $\bar{\mathbf{r}}$ were examined as to improve our understanding of the impact temporal information may have on lower level decision making of time delta models. These binned activations are presented to the reader in Figure 5.3.

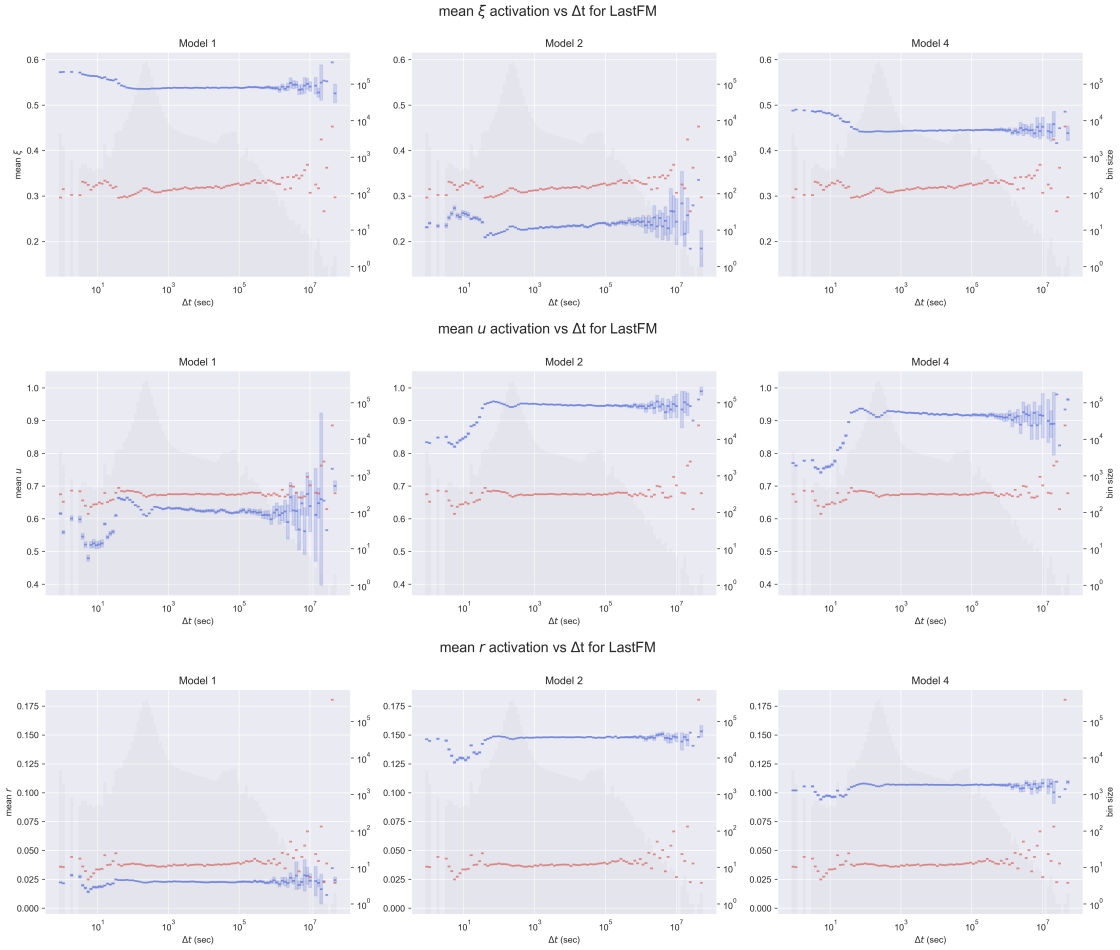


Figure 5.3: Mean $\boldsymbol{\xi}$, $\mathbf{u}$ and $\mathbf{r}$ with their associated 95% confidence intervals of models 0, 1, 2, 4 for varying time gap duration on *LastFM* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

Consistent with the observations for inter item distances, most gating vectors appear to exhibit visually distinct patterns related to interaction types discussed in Section 5.2. However, importantly, and distinct from prediction distances, there exist pronounced differences in the values as well as general trends of activations, both between time-dependent and sequential models, as well as between different time delta models. Interestingly, introduction of a temporal term in equation 1, used to determine the value of $\boldsymbol{\xi}$, appears to have a knock on effect on the trend exhibited by $\mathbf{u}$ and $\mathbf{r}$ wherein they depart from a baseline-like behaviour. On the other hand, changes to equation 2 do not appear to strongly affect the distribution of responses of $\bar{\boldsymbol{\xi}}$, suggesting that whilst theoretically those may also be altered indirectly via backpropagation through the learned user and item embeddings, the extent of those changes is not substantial.

Notably, whilst for $\Delta t \leq 3$s the baseline model appears to perform item-focused updates (low $\bar{\boldsymbol{\xi}}$, high $\bar{\mathbf{u}}$, medium-high $\bar{\mathbf{r}}$ relative to other bins), time delta models employ a different strategy for the above interactions. Namely, time-dependent models exhibit relatively strong levels of user dependency (high $\bar{\boldsymbol{\xi}}$ - model 1), lower extent of hidden state update (low $\bar{\mathbf{u}}$ - model 2) or both (model 4). It is not clear whether such strategy

lends itself to improving the performance for the associated interactions due to the wide confidence intervals described for MRR@20 as well as an apparent lack of short time gap specific differences from the baseline in terms of recommendations. However, the above observation would suggest that, instead of reducing the impact of $i^{(T-1)}$ as initially hypothesised, for short Δt time delta models either reduce the impact of $i^{(T)}$ or increase the importance of the user's long term preferences on recommendation. The above strategies are also used by both time delta and sequential models for $3 < \Delta t < 10^{1.5}$ sec as well as for more prolonged absences ($\Delta t > 10^{2.8}$ sec). The latter, however, suggests that for *LastFM* time delta models do not adapt to long periods of inactivity by strongly leveraging the item consumed upon the user's return to the platform as hypothesised originally. Instead, models progressively shift focus to static user preferences or the user's recent history directly preceding their departure from the platform, consistent with our previous observation that users do not appear to consume novel content after a prolonged absence.

Note, however, that it is not fully clear whether the observed differences in activations are solely responsible for the differences in performance and recommended items. For instance, a unique feature of models 1 and 4, achieving the largest MRR@20 improvements around $\Delta t = 10^{2.4}$ sec, is the lack of increase in $\bar{\boldsymbol{\xi}}$ seen for a lower performing models 2 and the baseline. However, we observe no strong local differences in recommendation distance trends between any models, suggesting an alternative mechanism leading to performance improvements, for example better parameter learning. Our analysis is also complicated by the fact that absolute values of activations cannot be directly compared between models. This is the case due to the observed differences in learned user and item embeddings between models. For instance, two models with distinct activations of $\boldsymbol{\xi}$ may lead to the same scaled user embedding used in equation 2. In a similar vain, values of $\bar{\boldsymbol{\xi}}$ exceeding 0.5 do not guarantee the overall model attention being focused on the user vector as user and item vectors distributions may be different. As such, further work, addressing the above limitations, may gain further insight into the effects of time gaps.

5.3.2. *MovieLens*

5.3.2.1 MRR@20

For *MovieLens*, the performance of both the baseline as well as time delta models, presented in Figure 5.4 exhibits an overall parabolic trend. The highest global accuracy is observed for the shortest time gaps, with performance decreasing to its local minimum by approximately $\Delta t = 10^4$ sec (3 h) and finally increasing for interactions with $\Delta t > 10^{5.4}$ sec (3 days). Note, however, that local trends for $10^3 <$ sec $\Delta t < 10^{4.7}$ sec as well as $\Delta t > 10^{6.8}$ sec are to some extent obfuscated by the wide confidence intervals arising from low numbers of interactions in each of the associated bins. Moreover, it is not clear why there exist substantial differences between different bins with Δt on the order or seconds or minutes as we do not believe users to be able to consume new content or change their tastes over such short periods of time.

Overall, we observe that time delta models are able to provide improved performance over the baseline across most values of time gaps. However, certain models appear to excel on different ranges of interactions. In particular, in addition to improvements across the majority of bins, models 1 and 4, in contrast to other time delta models, demonstrate improvements on shorter time gaps ($\Delta t < 10^{1.9}$) sec. On the other hand, models 2 and 4, as well as 6 and 7 (Appendix Figure B.6), appear to improve the most for larger time gaps, with the highest improvements observed for $\Delta t > 10^{5.7}$ sec (6 days). The latter is consistent with the results in Table 5.3, whereby models featuring a modified equation 2 were found to demonstrate consistent improvements on *MovieLens high* duration time gaps.

5.3.2.2 Embedding distances

As seen in Figure 5.5, the distance between items $i^{(T-1)}$ and $i^{(T)}$ undergoes a slow decrease, with a fluctuation for Δt on the order of seconds. This is followed by a more rapid descent starting at $\Delta t = 10^{3.7}$ sec (1.4 h) and lasting until $\Delta t = 10^{5.7}$ sec, at which point additional Δt increases are associated with growth of $|i^{(T-1)}\text{-}i^{(T)}|_2$.

Analogous to *LastFM*, item-to-prediction distances overall follow the trends of $|i^{(T-1)}\text{-}i^{(T)}|_2$, with recommendations being generally closer to the more recently consumed item. However, compared to *LastFM*, we observe a higher amount of deviation of $|i^{(T-1)} - \bar{p}|_2$ and $|i^{(T)} - \bar{p}|_2$ from $|i^{(T-1)}\text{-}i^{(T)}|_2$. Unexpectedly, no pronounced differences are observed for models 1 and 4 distinguishing them from other models. As such, the mechanism for improvement on short time gaps remains unclear. On the other hand, models 2 and 4, as well as 6 and 7 (Appendix Figure B.7), exhibit a rapid shift away from the item before the time gap (higher $|i^{(T-1)} - \bar{p}|_2$) and closer to the item after the time gap (lower $|i^{(T)} - \bar{p}|_2$) for $\Delta t > 10^{1.4}$ sec, with the behaviour

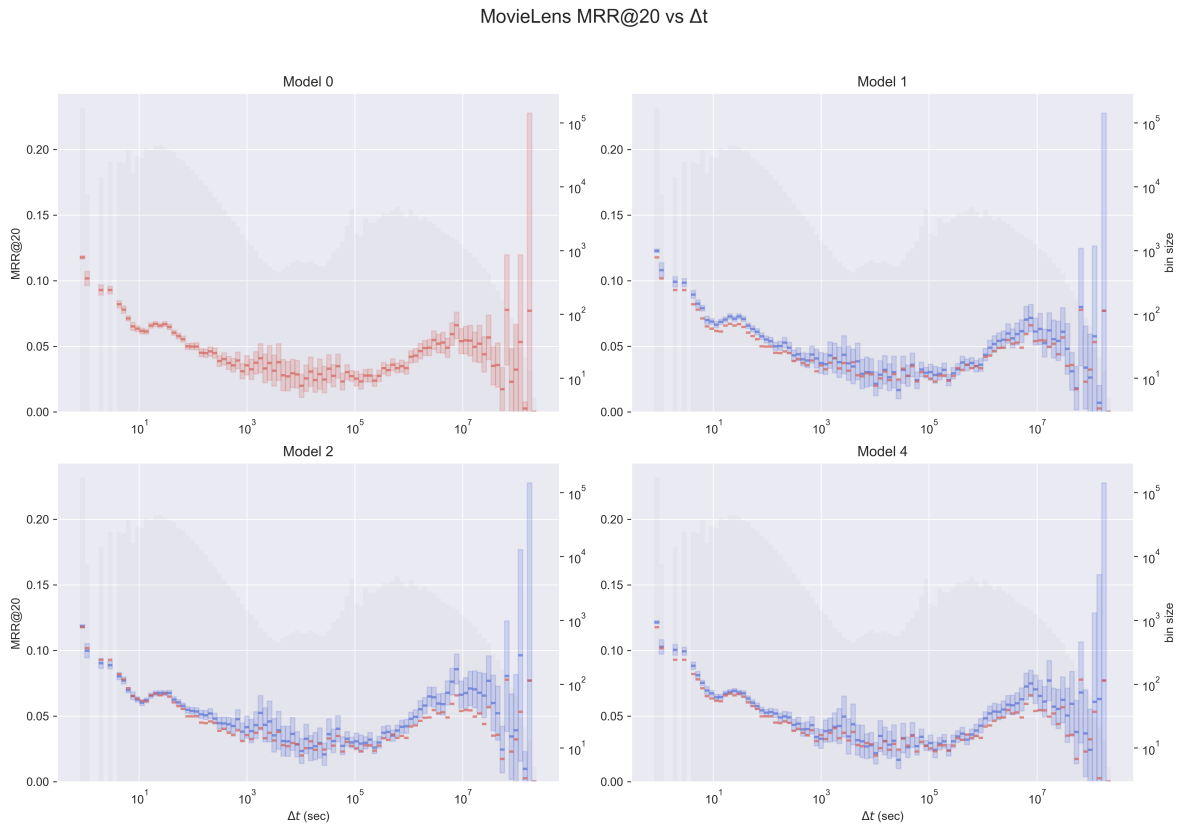


Figure 5.4: MRR@20 and the associated 95% confidence intervals of models 0, 1, 2, 4 for varying time gap duration on *MovieLens* dataset. Performance averaged over individual interactions falling into exponentially expanding bins. Model 0 presented in red, *DeepTimeDelta* models presented in blue. Model 0 MRR@20 included alongside that of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

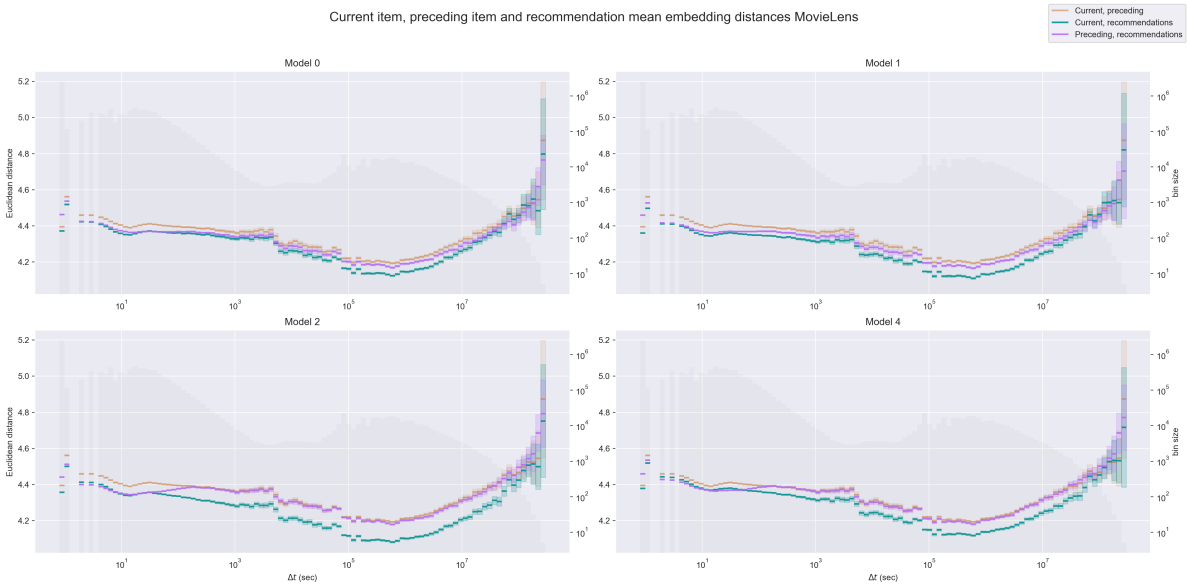


Figure 5.5: $|A-P|_2$, $|B-P|_2$ of models 0, 1, 2, 4, as well as $|B-A|_2$ with their associated 95% confidence intervals for varying time gap duration on *MovieLens* dataset. Distances averaged over individual interactions falling into exponentially expanding bins. Number of interactions assigned to each bin shown in grey.

persisting for all higher Δt values. Such swift changes are, however, not observed for models that do not feature a time gap term in their second equation, suggesting that direct regulation of the hidden state may be an important mechanism for improving the performance following medium or long inactivity.

5.3.2.3 Activations

Finally, in terms of activations, both sequential as well as time delta models exhibit a wide array of activation trends. These are presented in Figure 5.6.

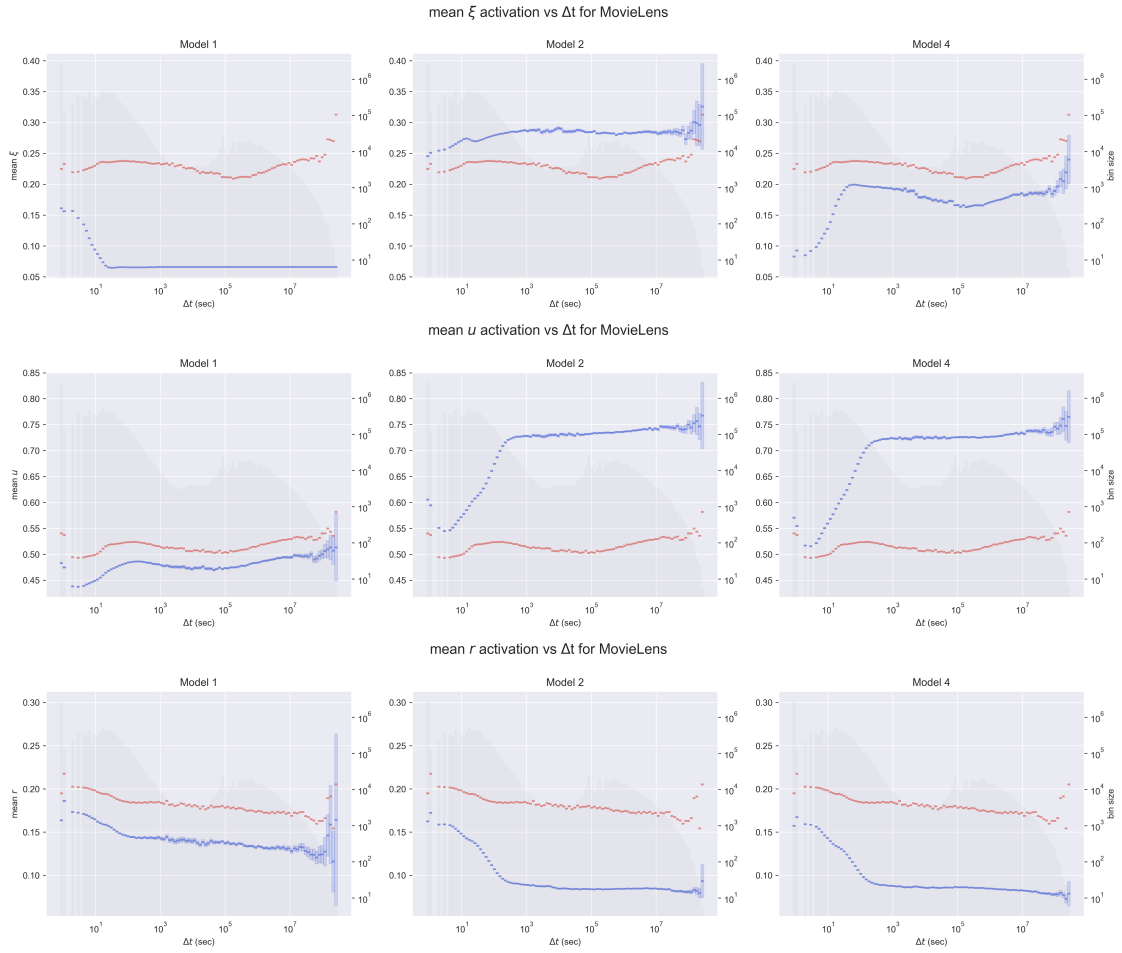


Figure 5.6: Mean $\boldsymbol{\xi}$, $\mathbf{u}$ and $\mathbf{r}$ with their associated 95% confidence intervals of models 0, 1, 2, 4 for varying time gap duration on *MovieLens* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

For large time gaps, both the baseline as well as models 2 and 4, as well as 6 and 7 (Appendix Figures B.8, B.9, B.10), increase the importance of the stable user preferences and the extent of the update whilst decreasing the importance of the recently consumed items. Notably, the above time delta models exhibit a substantially more rapid change in $\bar{\mathbf{u}}$ and $\bar{\mathbf{r}}$ for $\Delta t < 10^{2.4}$ sec, quickly moving away from $i^{(T-1)}$, as described in the previous subsubsection. This behaviour is not observed to the same extent in models achieving lower performance for long time gaps and thus is likely the primary source of the observed improvements. However, note that in spite of the above time delta models simultaneously further increasing the importance of the static user preferences, their recommendations strongly move towards $i^{(T)}$ from $\Delta t = 10^{1.4}$ sec. As such, we conclude that models utilise and balance both of the above inputs, leveraging them for recommendations following any longer absences, especially those on the order of hours, days or months.

On the other hand, whilst we observe substantial differences of time delta models to the baseline in terms of their activations for $\Delta t < 10^{1.9}$ sec, we are unable to determine the mechanism through which models 1 and 4 achieve the largest improvements for the above range of Δt values. Whilst both models rapidly alter

their ξ in response to minor increases in Δt, this change is done in opposing directions. Moreover, it is unknown whether the activation changes truly represent time delta models interpreting short interactions as not having the knowledge of the exact consumption order, as hypothesised in Section 3.4, or models instead acting through a different mechanism. To that end, in Chapter 6, we propose additional methodology to further disambiguate between the possible underlying mechanisms leading to improvements observed for short time gaps.

5.3.3. *Videoland*

Finally, the above analysis was also carried out for *Videoland*. Note that due to the sensitivity of the data the counts of interactions belonging to each bin are not presented to the reader as was done for the publicly available datasets. As the dataset was gathered from interactions of over 600,000 users [2] the number of interactions across every range of time delta values exceeds that of the remaining datasets.

5.3.3.1 MRR@20

As presented in Figure 5.7, all models exhibit a high level of global as well as local variation in terms of their performance across various values of Δt. Overall, models achieve peaks of performance at $10^{3.1}$ sec (25 min), $10^{4.6}$ sec (11 h) and $10^{5.7}$ sec (6 days), followed by a rapid decline of results, consistent with the previously described observation of Jing and Smola [63].

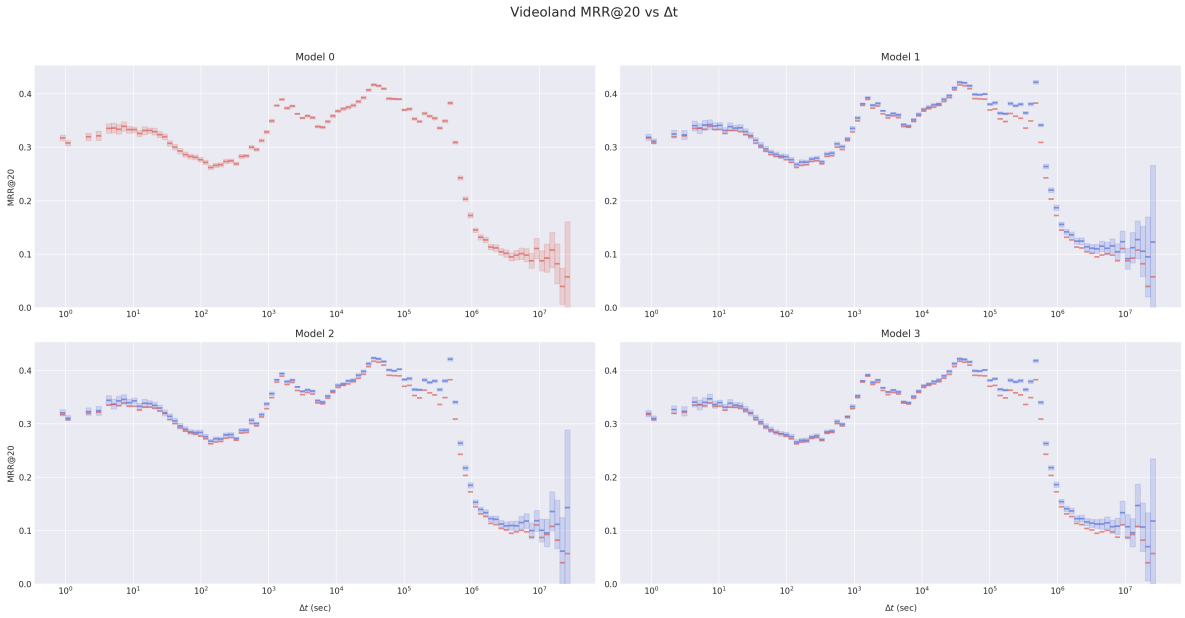


Figure 5.7: MRR@20 and the associated 95% confidence intervals of models 0 – 3 for varying time gap duration on *Videoland* dataset. Performance averaged over individual interactions falling into exponentially expanding bins. Model 0 presented in red, *DeepTimeDelta* models presented in blue. Model 0 MRR@20 included alongside that of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity.

Contrasted to the baseline, time delta models achieve an equal or improved performance for the vast majority of time gap values, with no model achieving performance lower than the baseline for more than 4 bins. Whilst some minor performance increases are observed for time gaps on the order of seconds or minutes, the largest absolute extent of the improvement was observed for interactions with $10^{4.9}$ sec $< \Delta t <$ $10^{5.9}$ sec (1 and 10 days respectively). The highest increase, totalling up to 10.5% of the baseline performance, is shown by model 4 for $\Delta t = 10^{5.7}$ sec (Appendix Figure B.11). As *Videoland* features a number of shows with weekly releases, the above improvement may be related to prediction of such weekly behaviour. This idea is explored further in relation to item and prediction distances. Further increases to 10^6 sec $\leq \Delta t \leq 10^{6.9}$ sec (3 months) are similarly associated with improvements over the baseline, with the largest improvements of approximately 10% observed for models 5 and 3, consistent with the ranking of models on *high* time gaps

[2] https://www.broadbandtvnews.com/2019/09/01/rtls-dutch-svod-service-videoland-to-add-commercials/

in Table 5.3. However, the extent of the improvement is less clear for $\Delta t > 10^{6.9}$ sec due to the widening confidence intervals. Nevertheless, the above improvements confirm that time gaps are a valuable source of information for absences of any duration and in particular for those on the order of days, weeks or months.

5.3.3.2 Embedding distances

For item and recommendation distances of *Videoland*, presented in Figure 5.8, we similarly observe a substantially higher variation of both item-to-item as well item-to-prediction distances compared to the previous datasets. Notably, recommendations of sequential but also time delta models appear to be closer to $i^{(T-1)}$ than to $i^{(T)}$ for $\Delta t < 10^{1.7}$ sec. This observation is at odds with our original hypothesis of short time gaps being used as a negative signal towards the first item. Whilst for a number models $|i^{(T)} - \bar{p}|_2 < |i^{(T-1)} - \bar{p}|_2$ for $10^{1.7}$ sec $< \Delta t < 10^{2.4}$ sec, the only difference between the sequential and time delta models is the marginally higher values of $|i^{(T-1)} - \bar{p}|_2$ of the latter - a trend not exclusive to the above Δt values. As such, we are unable confirm the viability of short time gaps as a source of negative feedback for $i^{(T-1)}$ for *Videoland*.

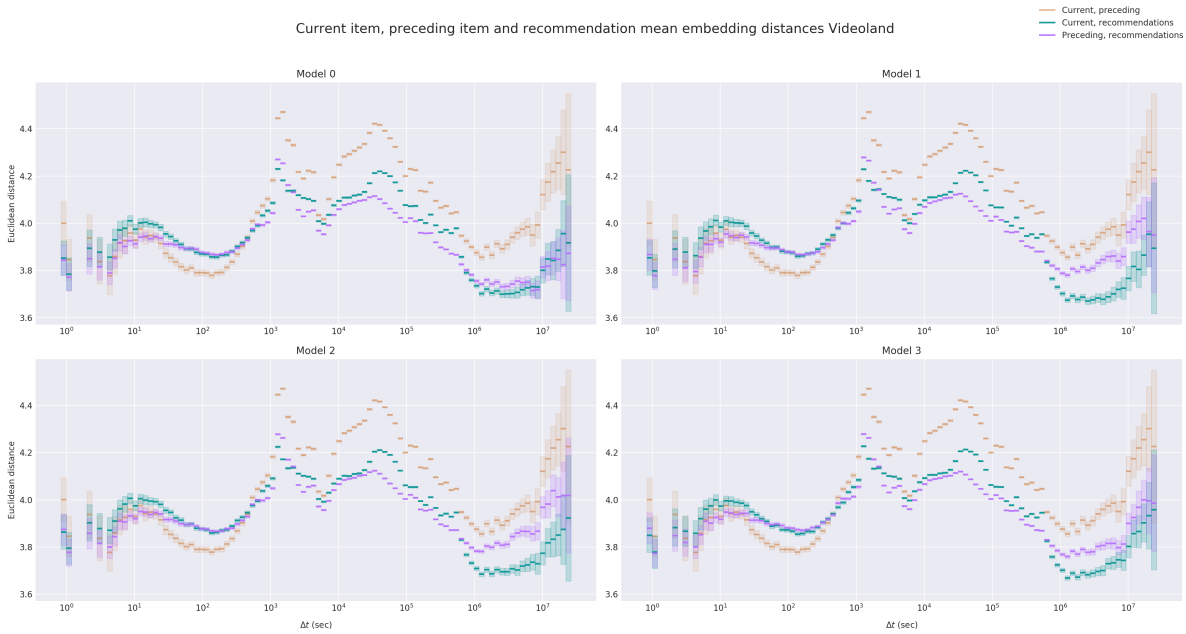


Figure 5.8: $|A-P|_2$, $|B-P|_2$ of models $0-3$, as well as $|B-A|_2$ with their associated 95% confidence intervals for varying time gap duration on *Videoland* dataset. Distances averaged over individual interactions falling into exponentially expanding bins.

On the other hand, compared to the baseline model, $|i^{(T)} - \bar{p}|_2$ of time delta models undergoes a more pronounced decline between $\Delta t = 10^{4.9}$ sec and $\Delta t = 10^{6.1}$ sec (2 weeks). Analogously, $|i^{(T-1)} - \bar{p}|_2$ of time delta models first declines less rapidly and later begins to increase. As such, time delta models appear to strongly reduce the importance of the recent history (hidden state) for medium and large time gaps. Interestingly, further increases of Δt to the order of months ($10^{6.5}$ sec) are also associated with increased movement away from $i^{(T)}$, suggesting that either hidden state still maintains some effect on recommendation or that predictions are strongly influenced by the static user preferences.

Finally, the above period of $10^{4.9}$ sec $< \Delta t < 10^{6.1}$ sec, coincides with improvements hypothesised to be related to consumption of weekly content. Users with inactivity whose duration falls into the above range appear to consume progressively more similar content to what they consumed directly before their departure from the platform, as evidenced by a rapid decline of $|i^{(T)} - i^{(T-1)}|_2$. Moreover, recommendations of time delta models also progressively become very close to both of the above items, indicating a high level of similarity between all three. Whilst there may exist simple rule-based dependencies between two successive items (e.g. user always watches current week's Ex on the Beach after Temptation Island), those are likely to already be captured by the sequential model. Whilst not shown directly, we propose that the performance improvements for the above time gaps may instead arise from time delta models leveraging time gaps as to identify the availability of the next episode of the related show. In the above case, however, improvements in metrics may not necessarily translate into improved user satisfaction, as in the end the user will not have discovered any

novel content.

5.3.3.3 Activations

Finally, mean gating vector activations, presented in Figures 5.9, 5.10 and 5.11, exhibit a higher level of inter model similarity compared to those of the remaining datasets, particularly when Δt is on the order of seconds or minutes. As such, the mechanism through which time delta models achieve performance improvements for lower time gaps reported in Section 5.2 and Subsubsection 5.3.3.1 is unclear. Importantly, model 3, excluding any direct effect of temporal information on the gating vectors, similarly outperforms the baseline for the above range of Δt values. Based on our intuition, temporal decay of equation 3 is unlikely to be beneficial in cases with short intervals. As such, we hypothesise that explicit temporal information may also lead to improved parameter learning through backpropagation, as described in Subsection 5.3.1. However, additional work is required to confirm this interpretation.

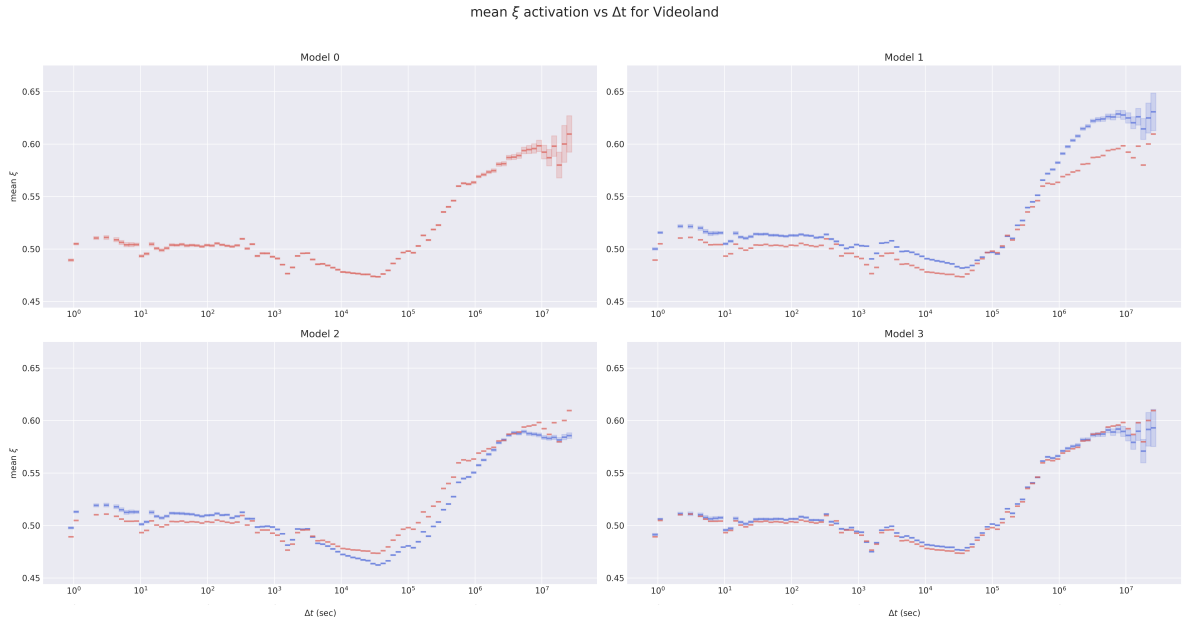


Figure 5.9: Mean $\boldsymbol{\xi}$ with its associated 95% confidence interval of models 0 – 3 for varying time gap duration on *Videoland* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity.

Starting from $\Delta t = 10^{3.5}$ sec (53 min) most time delta models exhibit minor decreases to $\bar{\boldsymbol{\xi}}$ and $\bar{\mathbf{r}}$ as well as increases to $\bar{\mathbf{u}}$. However, the most substantial changes commence between $\Delta t = 10^{4.5}$ sec and $\Delta t = 10^{5.5}$ sec (9h to 3 days respectively), whereby existing baseline model trends are strongly amplified. Models 1, 4 and 7 (Appendix Figure B.13) substantially increase the user vector contribution, whilst all time delta models except model 3 also increase the strength of the update. The latter is consistent with our observation whereby time delta models were believed to reduce their dependency on the hidden state for large time gaps. Moreover, the rapid increase in $\bar{\boldsymbol{\xi}}$ indicates models finding that following long periods of inactivity users also benefit from the use of their long term preferences, consistent with our observations for other datasets.

However, the exact means through which models 5 and 3 are able to achieve the highest performance among all time delta models for *high* time gaps is unclear. Importantly, models modifying the third equation also directly incorporate Δt into the calculation of the update vector k, providing another mechanism through which recommendations may be further improved. The reliance on this mechanism is consistent with our observations for model 3, which only exhibits minor differences from the baseline in terms of its activation pattern. However, generally, we believe that the mixture of reducing the hidden state contribution as well as increasing reliance on static user preferences is what allows the remaining time delta models to achieve particularly consistent improvements on interactions with long time gaps.

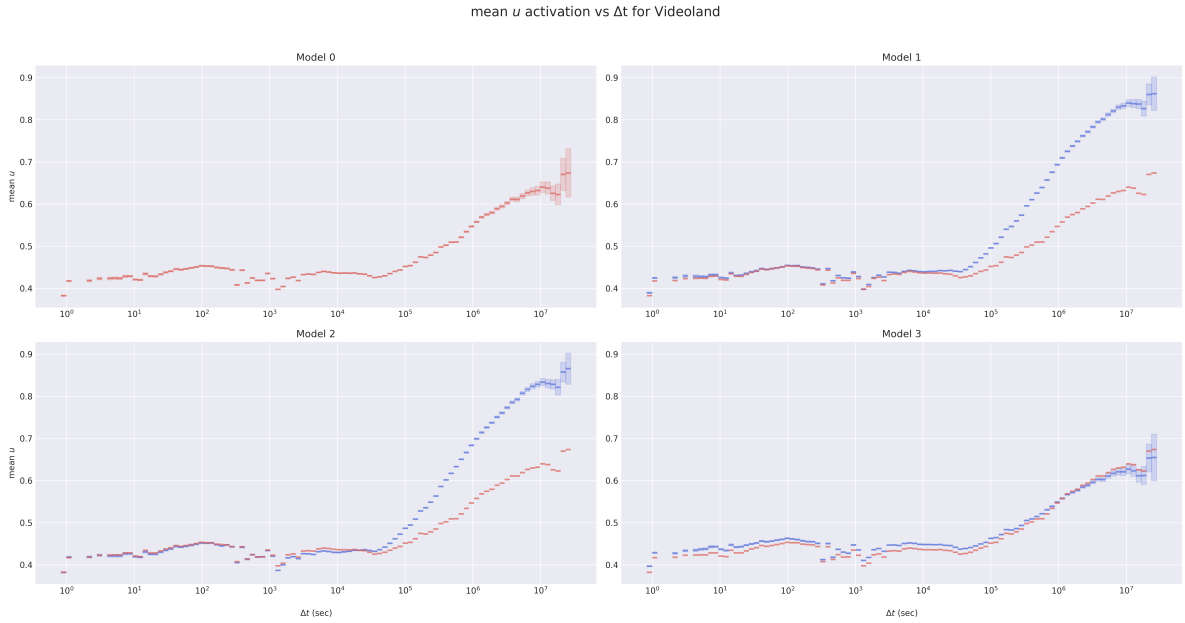


Figure 5.10: Mean **u** with its associated 95% confidence interval of models 0 – 3 for varying time gap duration on *Videoland* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity.

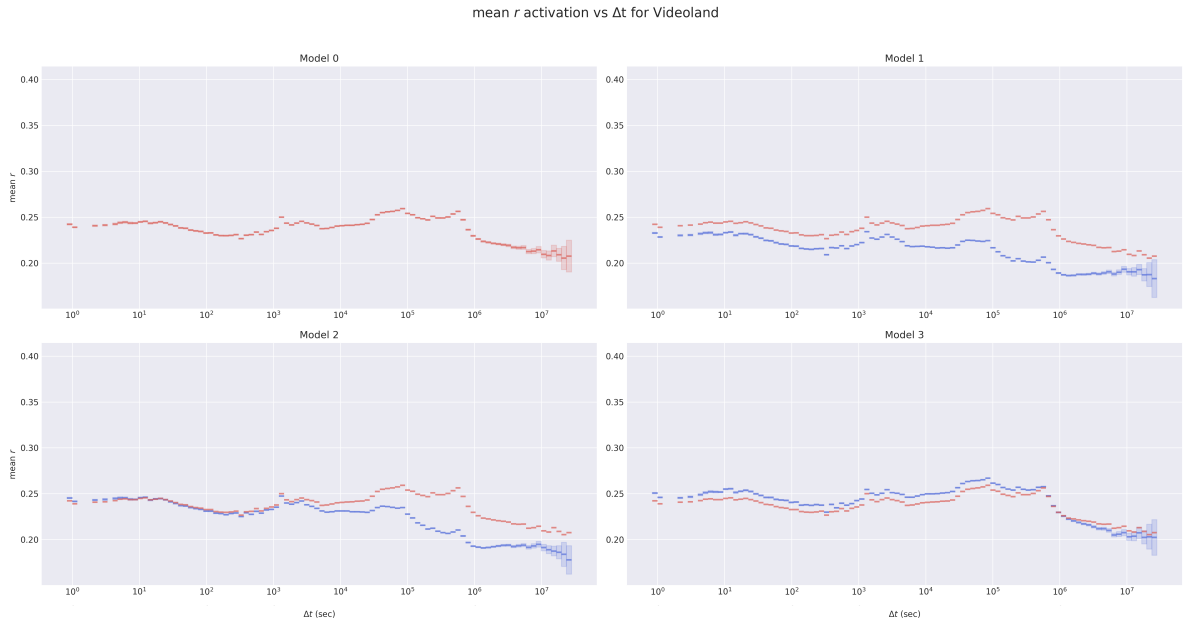


Figure 5.11: Mean **r** with its associated 95% confidence interval of models 0 – 3 for varying time gap duration on *Videoland* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity.

5.3.4. Conclusions

Overall, we observe strong dataset specific trends, with substantial differences in performance, embeddings and activations across different time gap values. However, overall we detect only minor improvements for short time gaps, consistent with the results described in Section 5.2. Notably, we discern no short time gap specific differences between the baseline and time delta models in terms of recommended items. Whilst

internally some differences in model behaviour may be observed, separating short time gaps from the rest, it is not clear whether these differences relate to interpreting short time gaps as signals of negative feedback, unknown consumption order or something else.

On the other hand, the use of time gaps overall leads to improved recommendation when user inactivity is on the order of days or longer. As seen for all datasets, time delta models exhibited increased reliance on static user preferences, highlighting their importance in recommendation following prolonged absences. For two of the datasets models also strongly reduced the importance of recently consumed items, consistent with our expectations. However, unexpectedly, for *LastFM* growth of Δt was instead associated with increases in the influence of the recently consumed items, suggesting a link between the last listening session and the current one.

Finally, some improvement for a wide range of time gaps may also arise from alternative mechanisms, not considered by the above analysis, most notably decay term of modified equation 3 as well as improved learning of user and item embeddings. Future work may choose to focus on these as to further improve our understanding of time gaps.

5.4. Effects on user groups

5.4.1. Temporal consumption

Lastly, an investigation into which subsets of users benefit from inclusion of time delta information was carried out. We first turned our attention to users with different levels of mean user time gap ($\overline{\Delta} t_u$). Note that *low* values of $\overline{\Delta} t_u$ signify a high interaction frequency, whereas users belonging to the *high* $\overline{\Delta} t_u$ category have longer average waiting periods between their interactions, indicating a lower item consumption frequency. The results of model evaluation are presented in Table 5.5. This was followed by statistical analysis, analogous Section 5.2. ANOVA results are presented in Appendix Table B.4, while Dunnett's test p-values are presented in Table 5.6.

Similar to the analysis of individual interactions, on the user level, the relative difficulty of distinct groups strongly depends on the choice of the dataset. Models exhibit the highest mean *MovieLens* and *Videoland* results across all metrics as well as on UserMRR@20 of *LastFM* for the *low* $\overline{\Delta} t_u$ group, with a lower performance on *medium* users. On the other hand, the above order is inverted for the remaining *LastFM* metrics. Nevertheless, users belonging to the *high* group pose the highest challenge to all models across all datasets. We propose that the latter may be to some extent related to the activation patterns observed in Section 5.3. Namely, prolonged user inactivity was previously found to be associated with increased leveraging of the static user preferences, which are learned during training. Such embeddings are likely to more closely correspond to the true underlying user tastes for users with higher number of interactions. Notably, such users are also more likely to have a shorter average inactivity duration ($\rho = -0.31$), consistent with the general trend of relative group difficulty ordering.

Similarly, we observe no uniting trend in terms of the relative size of the improvement over different user groups across all datasets. For instance, whilst the improvements for *MovieLens* and *Videoland* are consistently the smallest for the *high* user group, this group also enjoys the largest improvements for the *LastFM* dataset. Note, however, that the extent of *LastFM* improvement for the *high* $\overline{\Delta} t_u$ user group is lower than that observed for *high* interactions in Table 5.3. This observation may be related to the idea that a user's history is likely to feature absences of various durations. As such, the total extent of the gains for that user is likely to lie between the smallest and the largest improvements observed for distinct interaction groups, effectively dampening observed improvement for the *high* user group.

On individual model level, model 1 consistently improves over the baseline across all datasets, only failing to improve in terms of UserRecall@20 for *high* $\overline{\Delta} t_u$ users. Analogously, model 2 improves on 34 out of 36 cases, model 4 improves on 32, whilst models 3 and 6 outperformed the baseline only for a small proportion of dataset and user group combinations. However, only model 1 was found to achieve any statistically significant improvements at 5% confidence level, those being observed for *medium* $\overline{\Delta} t_u$ users in terms of Recall based metrics as well as the *high* user group in terms of Recall@20.

The low extent of statistically significant differences as well as lack thereof for *low* $\overline{\Delta} t_u$ users is likely related to the overall small size of observed improvements. In our view, whilst some improvement is observed for certain user groups, the source of the improvement may to some extent lie in the improvement on individual interactions, discussed in Sections 5.2 and 5.3. Furthermore, this contrast is likely in part observed due to the plurality of possible interpretations of a single given $\overline{\Delta} t_u$ value, described in Section 4.10. Moreover, Figures in Section 5.3 highlight the pronounced differences in the way users interact with content originating

Table 5.5: Offline evaluation metrics of baseline and *DeepTimeDelta* models on distinct $\overline{\Delta}t_u$ user groups. Best performance for the given dataset's $\overline{\Delta}t_u$ user group in terms of the chosen metric denoted in bold.

Dataset	model	Recall@20			MRR@20			UserRecall@20			UserMRR@20		
		low	medium	high	low	medium	high	low	medium	high	low	medium	high
LastFM	0	0.3207	0.3564	0.3090	0.2312	0.2572	0.2276	0.3284	0.3263	0.2799	0.2427	0.2377	0.2037
	1	**0.3292**	**0.3731**	**0.3321**	0.2320	0.2589	0.2299	**0.3398**	**0.3502**	**0.3138**	0.2445	0.2418	0.2139
	2	0.3267	0.3655	0.3235	0.2370	**0.2640**	**0.2370**	0.3352	0.3392	0.2948	0.2454	**0.2471**	**0.2164**
	3	0.3069	0.3356	0.2943	0.2143	0.2303	0.2048	0.3132	0.3042	0.2662	0.2207	0.2063	0.1745
	4	0.3247	0.3558	0.3183	**0.2375**	0.2586	0.2324	0.3314	0.3298	0.2915	**0.2472**	0.2405	0.2135
	5	0.3265	0.3721	0.3280	0.2290	0.2584	0.2257	0.3335	0.3426	0.3089	0.2402	0.2403	0.2107
	6	0.3053	0.3455	0.3021	0.2074	0.2277	0.2005	0.3112	0.3179	0.2796	0.2118	0.2059	0.1832
	7	0.3205	0.3615	0.3178	0.2219	0.2499	0.2243	0.3304	0.3319	0.2947	0.2377	0.2318	0.2014
MovieLens	0	0.3163	0.2399	0.1785	0.1074	0.0758	0.0547	0.3527	0.2505	0.2008	0.1275	0.0818	0.0624
	1	**0.3313**	**0.2564**	**0.1915**	**0.1141**	**0.0828**	**0.0592**	**0.3635**	**0.2603**	**0.2137**	**0.1341**	**0.0880**	**0.0682**
	2	0.3222	0.2448	0.1849	0.1087	0.0780	0.0569	0.3570	0.2539	0.2056	0.1278	0.0836	0.0645
	3	0.3066	0.2310	0.1720	0.1024	0.0720	0.0512	0.3481	0.2434	0.1944	0.1238	0.0794	0.0596
	4	0.3255	0.2512	0.1905	0.1127	0.0814	**0.0592**	0.3582	0.2544	0.2127	0.1329	0.0864	0.0674
	5	0.3146	0.2404	0.1804	0.1080	0.0775	0.0547	0.3519	0.2486	0.2048	0.1293	0.0846	0.0642
	6	0.3232	0.2439	0.1815	0.1088	0.0763	0.0537	0.3627	0.2541	0.2046	0.1310	0.0828	0.0626
	7	0.3220	0.2433	0.1821	0.1097	0.0779	0.0555	0.3576	0.2542	0.2055	0.1302	0.0847	0.0639
Videoland	0	0.6391	0.6269	0.5239	0.3662	0.3575	0.2901	0.5896	0.5930	**0.5038**	0.3321	0.3394	0.2750
	1	0.6404	0.6302	0.5250	0.3732	0.3682	0.2951	0.5898	0.5955	0.5027	0.3358	0.3459	0.2764
	2	**0.6410**	0.6307	0.5238	**0.3738**	0.3694	0.2953	0.5897	0.5951	0.5004	0.3361	0.3465	0.2760
	3	0.6388	0.6300	0.5238	0.3718	0.3689	**0.2957**	0.5864	0.5934	0.5009	0.3343	0.3459	0.2764
	4	0.6390	0.6304	0.5247	0.3728	**0.3695**	0.2954	0.5892	0.5948	0.5023	0.3357	0.3464	0.2762
	5	0.6395	0.6302	**0.5255**	0.3736	0.3689	0.2955	0.5890	0.5951	0.5028	0.3360	**0.3466**	**0.2767**
	6	0.6409	0.6303	0.5242	**0.3738**	0.3680	0.2938	**0.5904**	0.5955	0.5021	**0.3362**	0.3456	0.2754
	7	0.6397	**0.6312**	0.5254	0.3733	0.3685	0.2944	0.5895	**0.5958**	0.5023	0.3360	0.3456	0.2752

Table 5.6: Dunnett's test p-values for offline evaluation on distinct $\overline{\Delta}t_u$ user groups.

H_1	Recall@20			MRR@20			UserRecall@20			UserMRR@20		
	low	med	high	low	med	high	low	med	high	low	med	high
$1 > 0$	0.1001	0.0100	0.0279	0.1463	0.1200	0.1749	0.2615	0.0042	0.2983	0.1070	0.0572	0.0865
$2 > 0$	0.2569	0.1309	0.7920	0.3178	0.2622	0.2150	0.8601	0.2064	1.0000	0.4953	0.1941	0.4491
$3 > 0$	1.0000	0.9995	1.0000	0.9999	0.9991	0.9979	1.0000	1.0000	1.0000	1.0000	0.9988	0.9998
$4 > 0$	0.8999	0.1443	0.2407	0.1548	0.1409	0.1217	0.9548	0.4561	0.9165	0.1365	0.1033	0.1642
$5 > 0$	0.9715	0.2059	0.1585	0.6125	0.4483	0.6502	0.9989	0.4694	0.8635	0.5068	0.1969	0.2907
$6 > 0$	0.9328	0.6949	0.9930	0.9625	0.9885	0.9995	0.9713	0.5256	1.0000	0.9107	0.9713	0.9998
$7 > 0$	0.9194	0.1442	0.3489	0.7131	0.6308	0.7581	0.9492	0.1378	0.9938	0.4690	0.4304	0.9828

from different domains. As such, users from different datasets belonging to the same $\overline{\Delta} t_u$ group may exhibit vastly differing behaviours and therefore may benefit from temporal information to a different extent. We hypothesise that a more dataset-specific approach of partitioning users into groups may provide a clearer separation between users in order to identify any underlying performance trends.

5.4.2. Mainstreamness

Finally, users were instead split on their mainstreamness levels. The results of model performance for users with *low*, *medium* and *high* degrees of mainstreamness, as captured by Kendall's τ, are presented in Table 5.7, whilst the results of statistical evaluation are presented in Appendix Table B.5 and Table 5.8.

The choice of dataset is once again observed to be a key factor affecting the relative difficulty of user groups. For instance, models demonstrate their highest mean *MovieLens* score on *low* mainstreamness users (UserMRR@20 of 0.11), with a minor decrease in performance for the *medium* user group (0.10) and the lowest performance for the most mainstream users (0.07). Conversely, the above ordering is reversed for *Videoland*, where models achieve the highest mean result (0.35) on *high* users, followed by *medium* and *low* users (0.30 and 0.23 respectively). Finally, the relative difficulty of users groups for *LastFM* varies between evaluation metrics. The relative ease of recommendation for mainstream and difficulty of recommendation for niche users of *Videoland* is consistent with prior work of Bauer and Schedl [9], Farrahi et al. [38]. Moreover, as by definition there are many more consumption instances of popular items over the niche items, embeddings learned for the former are more likely to correctly reflect their underlying characteristics and, by proxy, the tastes of users consuming those items. However, as noted earlier, the above relative difficulty of mainstreamness user groups is not observed across all datasets, suggesting that the perceived difficulty of serving niche users described by [9, 38, 120] is strongly dependent on the choice of the dataset, as well as the evaluation metric.

In terms of the improvement over the baseline, there was similarly no group spanning all datasets for which the improvements would exceed those of other user groups. On the other hand, models 1, 2 and 4 demonstrated general improvements across all user groups, datasets and metrics, generally only failing to improve on *low* and *medium* mainstreamness users of *Videoland*. Models 3 and 6, conversely, achieved the least consistent extent of improvement, consistent with our observations for $\overline{\Delta} t_u$ groups.

As evidenced by statistical evaluation, model 1 was the only one to demonstrate any statistically significant improvements over the baseline at 5% confidence level. Such improvements were observed for Recall based metrics of *high* τ users as well as for *medium* users in terms of Recall@20. Notably, models 1, 2 and 4 also outperform the baseline across all dataset and metric combinations for *low* mainstreamness users. However, these improvements were not deemed to be statistically significant, likely reflecting the overall small effect size of the improvement. Moreover, it is unclear whether the observed improvements for *high* mainstreamness users are beneficial in practice, as such users are likely to be served well by simple popular item recommenders [9]. Finally, similar to our previous analysis, we hypothesise that users belonging to the same mainstreamness group of distinct datasets may to some extent also exhibit different sets of behaviours. As such, whilst we observe some benefit in the use of time gaps, more dataset centric analysis may provide further insight into their effects on specific mainstreamness user groups.

Table 5.7: Offline evaluation metrics of baseline and *DeepTimeDelta* models on distinct mainstreamness user groups.
Best performance for the given dataset's mainstreamness user group in terms of the chosen metric denoted in bold.

Dataset	model	Recall@20			MRR@20			UserRecall@20			UserMRR@20		
		low	medium	high	low	medium	high	low	medium	high	low	medium	high
LastFM	0	0.3516	0.3603	0.3183	0.2765	0.2652	0.2259	0.2853	0.3271	0.3185	0.2126	0.2421	0.2272
	1	**0.3719**	**0.3849**	0.3267	0.2799	0.2650	0.2275	**0.3171**	**0.3552**	0.3296	**0.2267**	0.2442	0.2285
	2	0.3603	0.3740	0.3245	**0.2868**	**0.2726**	**0.2318**	0.2986	0.3397	0.3277	0.2210	**0.2520**	**0.2339**
	3	0.3313	0.3457	0.3018	0.2389	0.2426	0.2072	0.2647	0.3126	0.3018	0.1707	0.2199	0.2061
	4	0.3575	0.3666	0.3200	0.2811	0.2681	0.2308	0.2967	0.3324	0.3208	0.2223	0.2461	0.2311
	5	0.3646	0.3736	**0.3283**	0.2756	0.2635	0.2251	0.3108	0.3423	**0.3298**	0.2213	0.2433	0.2255
	6	0.3342	0.3486	0.3058	0.2318	0.2343	0.2036	0.2787	0.3200	0.3069	0.1771	0.2171	0.2031
	7	0.3601	0.3675	0.3189	0.2652	0.2582	0.2182	0.2967	0.3351	0.3222	0.2083	0.2399	0.2206
MovieLens	0	0.2840	0.2702	0.1991	0.1018	0.0913	0.0612	0.2945	0.2840	0.2246	0.1059	0.0964	0.0697
	1	0.2925	**0.2820**	**0.2140**	**0.1079**	**0.0971**	**0.0665**	0.3020	**0.2928**	**0.2415**	**0.1122**	**0.1019**	**0.0764**
	2	0.2907	0.2738	0.2055	0.1034	0.0930	0.0633	0.3003	0.2855	0.2301	0.1072	0.0975	0.0715
	3	0.2844	0.2634	0.1908	0.1005	0.0878	0.0571	0.2946	0.2764	0.2148	0.1049	0.0931	0.0654
	4	0.2884	0.2776	0.2117	0.1073	0.0964	0.0659	0.2975	0.2891	0.2375	0.1114	0.1006	0.0750
	5	0.2839	0.2702	0.2004	0.1041	0.0930	0.0613	0.2949	0.2825	0.2272	0.1088	0.0986	0.0710
	6	**0.2942**	0.2739	0.2026	0.1045	0.0916	0.0606	**0.3046**	0.2875	0.2289	0.1094	0.0974	0.0699
	7	0.2916	0.2733	0.2029	0.1055	0.0934	0.0622	0.3017	0.2858	0.2294	0.1098	0.0984	0.0710
Videoland	0	0.4521	0.5425	0.6382	0.2326	0.3169	0.3638	**0.4160**	**0.5357**	0.6060	**0.2319**	0.3015	0.3412
	1	0.4532	**0.5434**	0.6404	0.2339	0.3204	0.3727	0.4149	0.5342	0.6084	0.2291	0.3018	0.3487
	2	0.4530	0.5424	**0.6409**	0.2341	0.3208	**0.3735**	0.4131	0.5329	0.6079	0.2290	0.3020	0.3490
	3	0.4493	0.5399	0.6397	0.2311	0.3197	0.3724	0.4116	0.5298	0.6071	0.2266	0.3007	0.3488
	4	0.4534	0.5419	0.6397	**0.2347**	0.3204	0.3730	0.4154	0.5336	0.6076	0.2298	0.3019	0.3487
	5	**0.4535**	0.5432	0.6399	0.2341	**0.3209**	0.3732	0.4140	0.5338	0.6080	0.2288	**0.3023**	**0.3491**
	6	0.4525	0.5425	0.6407	0.2331	0.3196	0.3729	0.4155	0.5342	0.6083	0.2300	0.3019	0.3480
	7	0.4520	0.5431	0.6404	0.2332	0.3203	0.3728	0.4138	0.5337	**0.6086**	0.2285	0.3015	0.3483

Table 5.8: Dunnett's test p-values for offline evaluation on distinct mainstreamness user groups.

H_1	Recall@20			MRR@20			UserRecall@20			UserMRR@20		
	low	med	high	low	med	high	low	med	high	low	med	high
$1 > 0$	0.0971	0.0470	0.0166	0.2362	0.0750	0.0809	0.4478	0.4320	0.0106	0.9401	0.5453	0.0669
$2 > 0$	0.4631	0.8579	0.0567	0.6516	0.1456	0.1307	0.9951	0.9995	0.2166	1.0000	0.7579	0.1885
$3 > 0$	1.0000	1.0000	1.0000	1.0000	0.9999	0.9996	1.0000	1.0000	1.0000	1.0000	1.0000	0.9993
$4 > 0$	0.6299	0.9259	0.2151	0.1322	0.0802	0.0596	0.9593	0.9957	0.2286	0.9487	0.5627	0.0848
$5 > 0$	0.7980	0.8572	0.3405	0.7432	0.2796	0.5012	0.9953	0.9993	0.2276	0.9997	0.6005	0.4158
$6 > 0$	0.8881	0.9992	0.7125	0.9999	0.9986	0.9895	0.8305	0.9999	0.7349	1.0000	0.9999	0.9904
$7 > 0$	0.6454	0.8526	0.3477	0.9078	0.5079	0.6668	0.9633	0.9990	0.1553	0.9999	0.9911	0.6636

6

Conclusions, Limitations and Future Work

6.1. Conclusions

In this thesis we examine the effects and benefits of extending a sequential state-of-the-art model with temporal information. We demonstrate that such addition, in the form of time gap information, is associated with models achieving higher predictive performance compared to the sequential baseline model across three datasets originating from different recommendation domains.

Moreover, we show that time-dependent models demonstrate improvements across a wide range of interactions, with the largest improvements observed for cases following medium and, in particular, long user inactivity. The largest improvements are generally observed for models incorporating temporal information in the gating process of the input vectors, whereas the use of temporal information as its own feature in a user agnostic fashion appears to have more dataset-specific benefits, mostly for cases of more prolonged user absence.

Additionally, we provide a closer look at the internal model mechanisms, the resulting recommendations and associated performance for a wide range of time gaps preceding the recommendations. Notably, different time delta models are observed to adopt distinct strategies for various extents of user inactivity. However, two uniting features of time delta models, observed for recommendation following longer absences across all datasets, are the increased reliance on user long term preferences as well as strong regulation of recent user history. Surprisingly, a large amount of the time delta model behaviour in response to varying values of the time gap is similarly observed for the baseline model, indicating a substantial amount of information already being encoded in the sequential ordering.

On the user level we also demonstrate some improvement of time-dependent models over the baseline for users of low and medium activity. However, we hypothesise that those may to some extent simply reflect the improvements observed on individual interactions. Finally, limited improvements are also observed for users consuming medium and high popularity content, with no statistically significant improvement for more niche users. Overall, our work demonstrates that time gaps are indeed a useful source of information for improving recommendation accuracy.

6.2. Limitations

6.2.1. Limitations of statistical evaluation

In general, there always exist time delta models providing an improvement over the baseline for any type of interactions, as well as for most types of users. However, we observe strong variation of user behaviour and model performance between different datasets and across different values of the time gap within the same dataset. Moreover, little or no similarity is seen for comparable time gap values across datasets. As such, distinct groups of time-dependent models excel on different time gap ranges or user types of different datasets.

This divergence, in turn, may be exerting a negative effect on our framework of statistical analysis used to combine results from different datasets. Notably, as described in Section 4.8 and Subsection 5.2.4, differing models excelling on distinct datasets may ultimately lead to improvements of some models being rendered statistically insignificant. Moreover, due to nonlinear nature of combining multiple time gap modifications, it

is overall difficult to establish a clear causative relationship between individual modifications and their effect on performance, as well as the benefits of combining multiple modifications e.g. via multi-factor ANOVA. Likewise, a large limitation of our statistical analysis is its focus on identifying specific models that show statistically significant improvements in place of identifying whether improvement is shown for time gap information as a whole. However, as observed, the same method of incorporating the time gap information may actually lead to improvements on some datasets and losses of performance on others. As such, it may be difficult to demonstrate the direct benefit of time gap information.

Furthermore, as outlined in Section 4.8, whilst Dunnett's correction is applied to reduce the FWER, this is done separately for each metric (and user/interaction group) and as such may still carry a risk of increased Type I error. Finally, the lack of temporal cross validation [116] or multiple rounds of training and testing may have contributed towards destabilising the observed results and thus contributed towards inflating the probability of Type I error.

6.2.2. Dataset-specific limitations

The differences in the observed results and model behaviour are likely strongly linked to the way users interact with the platform. Overall, in our view, *Videoland* provides the most rich source of user behaviour, with distinct user consumption behaviours across a wide spectrum of distinct time gaps. However, whilst we propose a scheme for dealing with consumption information relating to both movies and series, detailed in Section 4.2, there may exist alternative approaches tackling the above problem.

LastFM, on the other hand, appears to feature mostly highly active users, with only 1% of interactions being separated by 18 or more hours. As such, it is unclear how both time-dependent as well as sequential models would fare on less active users and in a more realistic setting, featuring more prolonged absences. Moreover, while on platforms such as those for movies, news or clothes users may be required to make more active content choices, this may not necessarily be the case for music. As tracks are often consumed in the form of albums or playlists, the user may not always be actively making the choice of the next track. Additionally, we hypothesise that as listening to a particular track requires less commitment in terms of time compared to movies and shows, *LastFM* users both tend to listen to a high number of items per session as well as may be more willing to accept a song that does not fully reflect their needs. In the above case, models may actually be learning to predict the output of the track or playlist based recommender system of the platform, not the user's true preferences. Furthermore, as described in Subsubsection 5.3.1.2, users of *LastFM* rarely appear to branch out in terms of new tracks between listening sessions, consistent with the discussion of filter bubbles of Pariser [105]. Additionally, the analysis of model performance on *LastFM* may be affected by the sampling procedure described in Section 4.2, with sampling potentially leading to a biased estimate of the true model performance across the analysis. Furthermore, sampling also led to the increased width of confidence intervals, as described in Subsection 5.3.1.

Finally, *MovieLens* appears to be the most distinct dataset in terms of user behaviour. As briefly described in Section 4.1, users of *MovieLens* show a general tendency to rate movies in quick bursts of over 50 movies at a time, without any guarantee for the consumption order. Additionally, over a quarter of the dataset's rating events take place with the same time stamp as another interaction for the same user. Moreover, as described previously, in the majority of cases users do not return to the platform following their initial session, in contrast to the users of *LastFM* and *Videoland*. As such, whilst the interactions featured in the dataset may be a useful source of feedback for static models, we caution the reader against the use of *MovieLens* data for evaluation of sequential models, a practice currently widely prevalent in literature [33, 67, 87, 135, 148].

6.2.3. Additional limitations

Limited insight is also obtained on the improvements associated with short time gaps. No conclusions are made with respect to time gaps on the order of seconds serving as a source of negative feedback, in part due to wide confidence intervals arising from a low number of eligible interactions. Moreover, whilst improvements are observed on *MovieLens* for interactions separated by time gaps on the order of seconds or minutes, it remains unclear whether those are achieved due to time delta models treating time gaps as a signal of the vagueness of the consumption order.

Likewise, whilst the higher granularity analysis provides additional information on the inner functioning of the model, there exist limitations in the employed approach. Firstly the activation analysis is limited in its ability to contrast different models. As different models may learn distinct user and item embeddings as well as other weights, the same value of $\bar{\mathbf{u}}$, for instance, may reflect non identical updates to the hidden state and recommendations. Analogously, for a particular model, due to differences in distributions of user and item

embeddings (in addition to the nonlinear nature of the model), the relative impacts of the user and the item vector on recommendation are not guaranteed to be approximately equal at $\bar{\xi} = 0.5$. Furthermore, whilst we observe strong apparent influence of the most recently consumed items on the resulting recommendations for different Δt values (high correlation between item-to-item and item-to-prediction distances), we are unable to separate the effect of the time gap from the effects of the item choices. Moreover, our results suggest a high amount of user behaviour already being captured by sequential data without the use of time gaps. However, the extent of user behaviour present in the ordering of the user history over their overall underlying preferences remains unclear due to the lack of additional static baselines. Importantly, the reliance on high quality user and item embeddings of both time delta as well as the baseline models also suggests that both are likely to suffer from the *cold start* problem.

Finally, it also remains unclear whether the observed offline metric improvements necessarily lead to improvements in user satisfaction. For instance, *LastFM* users are observed to be likely to return to familiar items following longer periods of inactivity. Similarly, whilst time delta models demonstrate the most substantial improvement for *Videoland* cases of user absence of 1 week, the improved prediction may be related to time delta models predicting the consumption of shows already familiar to the user. As such, the increase in prediction accuracy may to a certain extent exaggerate the tangible benefit of the recommendation to the user. Moreover, as suggested above and in Section 3.4, users likely already make certain consumption choices on the basis of the existing recommender systems. Combined with overbearingly large amount of choice available to the user, there can be no guarantee that any item consumed by the user was the most relevant to them at the given time. Due to the above bias and noise in the data, increase in prediction accuracy may not necessarily translate into increase in the user satisfaction. Whilst it was envisaged that A/B test provide indication of the user true satisfaction with recommendations, as hinted in the Introduction. However, due to technical challenges associated with the implementation of the experiment no conclusions can be drawn from the observed results. The setup, results and challenges as well as the impact of the online experiment on the thesis progression are included in Appendix Chapter A.

6.3. Future Work

Based on the observed dataset specific trends, future studies may choose focus on a lower number of datasets or multiple datasets reflecting consumption of a single domain, such as multiple movie, song or news datasets.

For additional examination of effects of time gaps on recommendation, we believe that splitting user histories into longer sequences or using full histories as individual samples may exert a strong effect on the achieved performance. Additionally, initial internal testing indicated the potential benefit of training models on the task of only predicting the item at the end of the sequence, as to reflect a more lifelike use case. A more realistic evaluation framework, analogous to that recommended by Campos et al. [20], may also provide a better understanding of the real life performance of the model.

Based on our intuition, time gap information may be further supplemented by additional inputs in order to provide a further increase in performance. Firstly, future work may want to consider additional context, such as the hour of the week embeddings used by Jing and Smola [63]. Moreover, additional temporal information may also be presented to the model in the form of the explicit item durations, providing the model with clear information on prematurely terminated item consumption, or interaction time stamps directly. Furthermore, incorporating content-based features in place of or together with the item embedding vectors may provide a partial amelioration of item version of the cold start problem [151]. Finally, newer architectures, such as those relying on self-attention, may provide a further improvement over our GRU based architecture [67, 148, 150].

In terms of the analysis of the effects of the time gaps we first and foremost suggest a focus on extending our framework for comparison of activations of distinct models as to increase the interpretability of the learned behaviour. It would be of interest to also examine whether the activation and recommendation behaviours observed in Section 5.3 are still present when varying the size of the time gap for a fixed recent history and the interacting item. Moreover, additional work, featuring varying both Δt as well as the item may provide further insight into the individual and combined effects of the two on the resulting recommendations. Furthermore, it is of interest to examine whether time delta models are able to adapt to users with similar preferences operating at different frequencies (e.g. two otherwise identical users, with one consuming items hourly and the other weekly). Additionally, a repeat of the user study, described in Appendix Chapter A, may confirm whether the observations with respect to the performance reported for the offline evaluation are also observed in the online setting. Further work may also wish to extend the evaluation to include anal-

ysis on the interaction or user level, analogous to Sections 5.2 and 5.4. The analysis may also benefit from a more prolonged duration of the experiment, as to capture a higher number of more prolonged absences, for which the main benefits were observed during the offline evaluation. Finally, both for online as well as offline evaluation, an inclusion of further state-of-the-art baselines as well as their time delta extensions may further clarify the importance of time gaps.

As suggested in Subsection 5.3.2, our study was unable to identify whether improved understanding of the hidden nature of the true consumption was what allowed time-dependent models to achieve an improved performance over the baseline on *MovieLens*. As such, we suggest a methodology relying on next basket recommendation [63, 114, 140]. Sequential and time-dependent models may be trained on the objective of predicting all items rated by the user during their next session. Models are then to be evaluated on multiple shuffled permutations of the same input sequences. We hypothesise that time delta models may enjoy improved prediction accuracy, attributable to the reduced overfitting on the last few items during a rating session.

Finally, time delta information may instead be used as a measure of the time passed after the current interaction, not leading up to it. In the above context, temporal information may serve as an important source of uncertainty with respect to the user's preferences. Prolonged inactivity may be viewed as a cause to decay the user's preferences as a function of the duration of the absence, for instance analogous to our use of equation 3. Such behaviour may be particularly beneficial for users returning following a long leave from the platform, as to capture their immediate needs without the need to interact with an item first.

Appendices

Appendix A. **Online Study**

In the following chapter we describe the main ideas and the setup of the online experiment introduced in the first chapter. We then report and interpret the observed results. Following that, as alluded in the introduction, we present the reasons for why the experiment was not successful and therefore not reported in the main part of the thesis. Finally, we discuss the impact of the online experiment on the duration of the thesis.

A.1. Methodology of the online evaluation

Previous works have demonstrated that superior accuracy in offline experiments does not necessarily translate into higher usefulness of recommendations for the end user [95, 137]. For instance, when dealing with implicit feedback, offline data may be affected by positional bias, whereby users tend to click more on the more highly ranked results [64]. Such bias has been shown to strongly influence the learning outcomes of recommendation algorithms [65, 141, 149]. As described in the work of Zhao et al. [149], both the training as well as testing data may be biased by the ranking produced by the recommendation algorithm that was employed during the data collection. As such, good performance in the offline experiment may, to some extent, simply indicate the new model successfully learning to predict the output of the old one. As such, we propose an online user study, also referred to as an A/B test, on a Dutch video-on-demand platform Videoland in order to evaluate whether employing time delta information truly leads to higher user satisfaction. This is done by contrasting the performance of the time delta model to that of the sequential baseline as well as the platform's static production recommender, with improvements indicating the benefit of time delta information. Moreover, a useful side effect of the above evaluation is that improvements of the sequential model over the static production model would also corroborate the benefit of accounting for temporal dynamics via sequential recommendation. For a comprehensive review on user experiments in RS we refer the reader to Knijnenburg and Willemsen [76].

A.2. Experimental setup of the online evaluation

A percentage of Videoland's users received recommendations generated by our best performing model, as determined during offline evaluation. Performance of the time-dependent model was then contrasted to the performance of our baseline model as well as the platform's production algorithm. This algorithm does not employ neural networks and does not leverage temporal dynamics.

New recommendations were calculated and served to users daily for the period of 20 days. Each day, all models were retrained on the basis of viewing data spanning a one year period preceding that day. A single user would be served recommendations by the same model for the duration of the experiment. A user was randomly assigned to receive recommendations from *DeepTimeDelta* with probability of 0.2, our sequential baseline with 0.2 and the production baseline with probability of 0.6. The choice of *DeepTimeDelta* model was made based on the overall performance on *Videoland* offline dataset, as measured by Borda count on the basis of all metrics.

A.2.1. Changes to data preprocessing and evaluation

Additional consideration was given to the limitations imposed by the requirements of online recommendation and the platform used. Firstly, in place of training, validation and test split used during the offline evaluation, only train and validation sets were produced, with validation set totalling up to 10% of a user's interactions (rounded down). Additionally, in order to generate recommendations shown to the user at serving time, each user's 20 most recent interactions were obtained to be used as input for a trained model. The neural network's hidden state after the processing of the last item was then used to calculate item scores. If a user had fewer than 20 interactions, all their interactions were used as input at serving time and the recommendations were generated on the basis of their last available interaction.

Secondly, the personalised recommendation strip of Videoland contains 25 items shown to the user at any given time. As such, the number of predictions on the basis of which the model was evaluated during daily training was increased to 25. Similarly, the number of recommendations produced by the trained model at serving time was also set to 25.

Finally, items the user had previously interacted with were also excluded from the recommendations as the purpose of the recommendation strip was to help users discover new items. Moreover, not all items that had previously existed on the platform were available for consumption in the item catalogue during the experiment. As such, while the unavailable items were used during training, they were excluded from the ranking when producing the final 25 recommendations.

A.2.2. Online evaluation metrics

Note that the recommendation procedure described above was performed once per day. This in contrast to the offline evaluation, where the aim was to predict the next item every time a given user consumed an item. As users may consume multiple items in one day, metrics described in Section 4.4 are no longer directly applicable.

Whilst it may be possible to adapt the above metrics to the case of multiple target items, click based metrics also fail to capture the distinct level of commitment for different types of content in terms of time required by the user. As such, we instead choose to focus on metrics measuring the duration of the interaction. More precisely, we measure the user engagement via the time users spend on platform (watch time), as well as the proportion of time spent watching recommended items (rec time %), defined below. For a given model, the metrics are calculated daily on the basis of interactions made by the model's users on that day.

$$\text{watch time} = \frac{1}{|U|}\sum_{v=1}^{|U|}\sum_{i\in I_v}\text{time}_v(i) \tag{A.1}$$

$$\text{rec time \%} = \frac{\sum_{v=1}^{|U|}\sum_{i\in I_v}\text{time}_v(i)\cdot i\in\text{pred}_v}{\sum_{v=1}^{|U|}\sum_{i\in I_v}\text{time}_v(i)} \tag{A.2}$$

Extending the definitions used in Section 4.4, U refers to the set of users assigned to a given model active on the given day. Furthermore, $\text{time}_v(i)$ refers to the time user v spent watching item i on that day. Additionally, $i \in \text{pred}_v$ is a step function indicating presence of item i in the set of recommendations for user v. Notably, due to additional constraints, only cases where the user consumed 80% or more of the item on a given day were included in the user history I_v.

A.2.3. Online evaluation statistical analysis

The above metrics were recorded daily for the duration of the experiment, with the aim of determining pairs of models where there exists a significant improvement of the mean of one model over the other. Due to the likelihood of existence of periodical effects (e.g. higher user engagement on certain days of the week), results from pairs of models were grouped together. Results of one model were then subtracted from the other, with the resulting differences being subjected to Shapiro-Wilk normality test [124]. This test was chosen due to its reported highest power among all normality tests [110]. This was then finally followed by three paired one-sample t-tests on the differenced data [131], with the following alternate hypotheses: time delta > production, sequential > production and time delta > sequential. Bonferroni correction with $m = 3$ was applied to the resulting p-values in order to offset the increased chance of Type I error.

A.3. Results and Discussion of the online evaluation

Based on the results of the offline evaluation model 2 was chosen to evaluate its effect on *Videoland* user behaviour and to be contrasted to model 0 and the platform's production algorithm. The experiment lasted 20 days, with the daily mean watch time and rec time % for three model types presented in Figure A.1. Note that, for visualisation purposes, the scores of all models for any given day are scaled by the performance of the production baseline as to eliminate the weekly seasonality trends.

As observed, users receiving recommendations from sequential and time delta models exhibit an activity ranging from 98.9 to 100.2 percent of that of the users receiving recommendations from the production algorithm in terms of watch time. The exact extent varies per day and per model. However, neither of the temporal models achieves an increase in the user watch time over the production baseline on more than 3 out of the 20 days. Lastly, there exists a high level of inter day variation between the sequential and the time delta models, with no model clearly outperforming the other.

On the other hand, rec time % of temporal models varies between 86 and 163 percent of the values obtained for the production baseline. Whilst both temporal models achieve a worse result compared to the production model on the first day of the experiment, no performance below 107% of the production level is

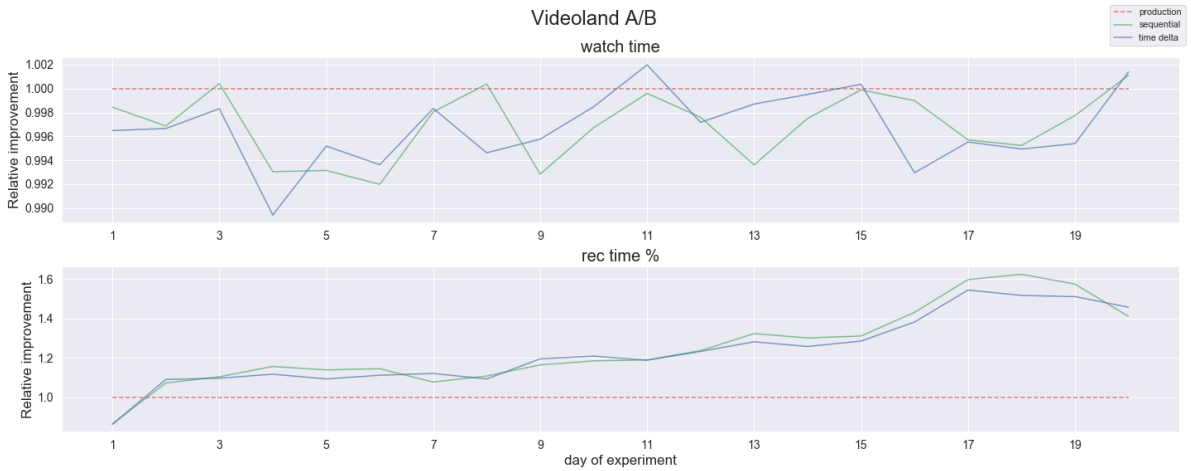


Figure A.1: Online experiment daily metrics for different models. Presented values scaled for a any given day scaled by the performance of the production model.

achieved by these models on any of the subsequent days. Furthermore, between days 8 and 9, both temporal models appear to undergo an ascent from levels of ~1.1 of the production's rec time % to ~1.2. This is followed by another increase to ~1.3 of the production's performance between days 11 and 13 as well as a more rapid increase to > 1.5 of the production's rec time % between days 15 and 17. Notably, whilst for days 1, 2, 7, 9, 10 and 20 time delta model also demonstrates a minor improvement over the sequential model, the latter outperforms the former on the remaining days, showing a particularly consistent trend of improvement between days 12 and 19.

Subsequent Shapiro-Wilk normality tests for daily differences of pairs of models were unable to reject the null hypothesis of the resulting distributions being normal at 5% confidence level. This was the followed by subsequent t-tests of the above pairs. The p-values for both types of tests are presented in Table A.1. In terms of time spent on the platform, the null hypotheses of the true mean difference between either of the temporal models and the production model being 0 or lower could not be rejected at 5% confidence level. Analogously, we could not reject the null hypothesis that the true mean difference between the time delta and the sequential did not exceed 0. On the other hand, the mean rec time % difference between either of the temporal models and the static model was found to be significantly higher than 0 at 5% confidence interval. Finally, the null hypothesis for mean rec time % difference between the time delta and the sequential model being 0 or lower could not be rejected at 5% confidence level.

Table A.1: Online experiment Shapiro-Wilk and t-test p-values for paired daily performances of different recommendation algorithms. t-test $H_0 : \mu \leq 0$, where μ denotes the mean difference of a given pair of daily results.

Test	Distribution	watch time	rec time %
Shapiro-Wilk	sequential - production	0.1941	0.3796
	time delta - production	0.9133	0.3018
	time delta - sequential	0.4742	0.9222
t-test	sequential - production	0.9999	< 0.0001
	time delta - production	1.0000	< 0.0001
	time delta - sequential	0.6007	0.9767

As such, overall, we do not observe a significant increase in terms of watch time of temporal models over the static model. Analogously, no improvement is observed for the time delta model over the sequential model. Conversely, users receiving recommendations from either sequential or time delta models appear to actually consume a marginally lower amount of content. However, the differences between the models are fairly minor, and as such additional work is required to corroborate the above observation.

On the other hand, both the sequential as well as time delta models appear to lead to a significant increase of the proportion of the watch time users spent watching recommendations compared to the static baseline.

Notably, however, the use of time delta information was not shown to provide a significant benefit over the sole use of sequential information. Moreover, we instead observe a trend of increasing improvement of sequential model over the baseline in the second half of the online experiment, with the exception of the final day. It may be that users require a certain amount of adaptation time before starting to consume recommendations of a new recommender more actively, suggesting an progressively increasing validity of the results of the latter days as a measure of true user satisfaction. This notion is also consistent with the observed relative increase in the recommendation consumption of both temporal models after ~1 week, as users may have been previously accustomed to the recommendations of the production algorithm.

Importantly, the observed results may be affected by additional seasonal user consumption trends not related to the weekly changes in user activity. We are unaware of any such trends having taken place over the duration of the experiment. However, paired with the seemingly unstabilised rec time %, a longer duration experiment may have allowed us to get a better understanding of long term effects of different recommenders. Notwithstanding the above limitation, the results are likely further distorted by additional limitations associated with model training. Firstly, the exact set of hyperparameters of the time delta model used in the online experiment was different from those reported in Table 4.3 and used in the offline evaluation. This is due to the fact that following making the choice of the hyperparameters used in the online experiment, a number of adjustments were made to the data preprocessing pipeline. Holdout method, performed following the completion of the A/B test, indicated suboptimal performance of the original parameters on the adjusted data. However, a subsequent rerun of the online experiment with the parameters used in the offline evaluation was not completed successfully due to platform instability issues and thus cannot reported. Secondly, a later analysis of the production model also revealed a separate error in the data preprocessing pipeline of the production algorithm. This is suspected to have potentially had a negative impact on the performance of the static baseline during the original run of the online experiment. As such, altogether, no conclusions with respect to the benefit of the use of time delta information over the solely sequential or static data can be drawn from these results.

A.4. Impact of online experiment on thesis duration

Finally, the inclusion of the A/B test may have also had a negative impact on the length of the thesis. Originally, prior to the decision to incorporate the analysis of *Videoland* offline and online data alongside the public datasets, taken 7 months after the start of this project, a Tensorflow implementation of time delta models was already available. However, due to the requirements of Amazon SageMaker at the time, the model had to be reimplemented using Tensorflow Estimator API. Moreover, due to the recency of the availability of SageMaker, only limited documentation on the use of the platform and its interactions with TensorFlow was available.

Additionally, we encountered difficulties in obtaining historical offline as well as online data. The former was made available 6 months into the duration of the internship, whilst a process allowing access to the daily batches of data from the previous day was functional with an additional delay of $3-4$ months.

Furthermore, due to the limited and varying amount of time between the availability of the fresh data and the deadline for producing the recommendations, a substantial effort was made into optimising the daily data preprocessing, model training and recommendation serving pipeline. As such, model training was highly optimised in terms of both the efficiency of the implementation as well the hyperparameters affecting the speed and convergence of training. Simultaneously, the other two steps saw a substantial effort being put into optimising Spark code [145].

Extra time was also spent monitoring the correctness of the incorporation of sequential and temporal recommendations into the final set of recommendations presented on the platform. Finally, further few weeks were later dedicated to identifying and confirming the discrepancy between the produced recommendations and the recommendations visible to the end users, ultimately leading to our inability to report results from the rerun of the online experiment, referenced above.

As such, we estimate that the total time consumption of the experiment amounted to $12-15$ months.

Appendix B. **Supplementary data**

Table B.1: Grid Search hyperparameter values. Dropout keep probability denotes values considered for Dropout of the user, item and temporal embedding vector (64 combinations).Embedding sizes were considered up to the dataset-specific upper limit, presented in Table 4.2

Hyperparameter	Values
Dropout keep probability	{0, 0.5, 0.8, 1.0}
Trainable initial state	{True, False}
Learning rate	{0.1, 0.01, 0.001, 0.0001}
Temporal embedding d	{0, 1, 2}
Temporal embedding f	{sigmoid, tanh, ReLU}
Embedding width	{50, 100, 200, 500, 1000}
Initialisation range	{0.2, 0.1, 0.01}
Sampling softmax	{True, False}

Table B.2: Full dataset offline evaluation ANOVA p-values of the model factor and R^2.

Statistic	Recall@20	MRR@20	UserRecall@20	UserMRR@20
p-value	0.0100	0.0683	0.0002	0.0366
R^2	0.48	0.30	0.71	0.37

Table B.3: ANOVA p-values of the significance of the model factor and R^2 for offline evaluation on distinct time gap duration groups.

Statistic	Recall@20	MRR@20	UserRecall@20	UserMRR@20
p-value	< 0.0001	0.0001	< 0.0001	< 0.0001
R^2	0.37	0.24	0.43	0.30

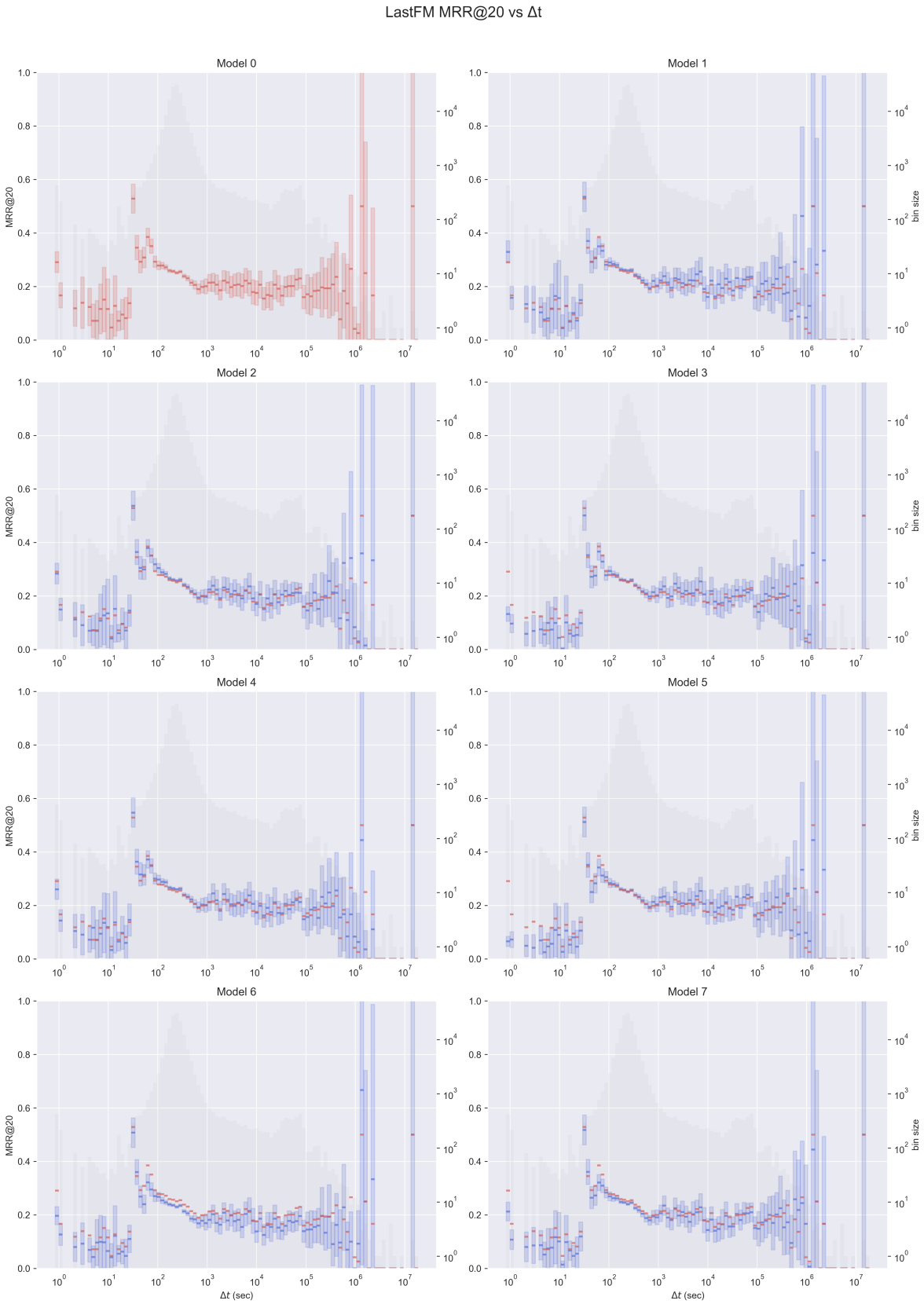


Figure B.1: MRR@20 and the associated 95% confidence intervals of models 0 – 7 for varying time gap duration time on *LastFM* dataset. Performance averaged over individual interactions falling into exponentially expanding bins. Model 0 presented in red, *DeepTimeDelta* models presented in blue. Model 0 MRR@20 included alongside that of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

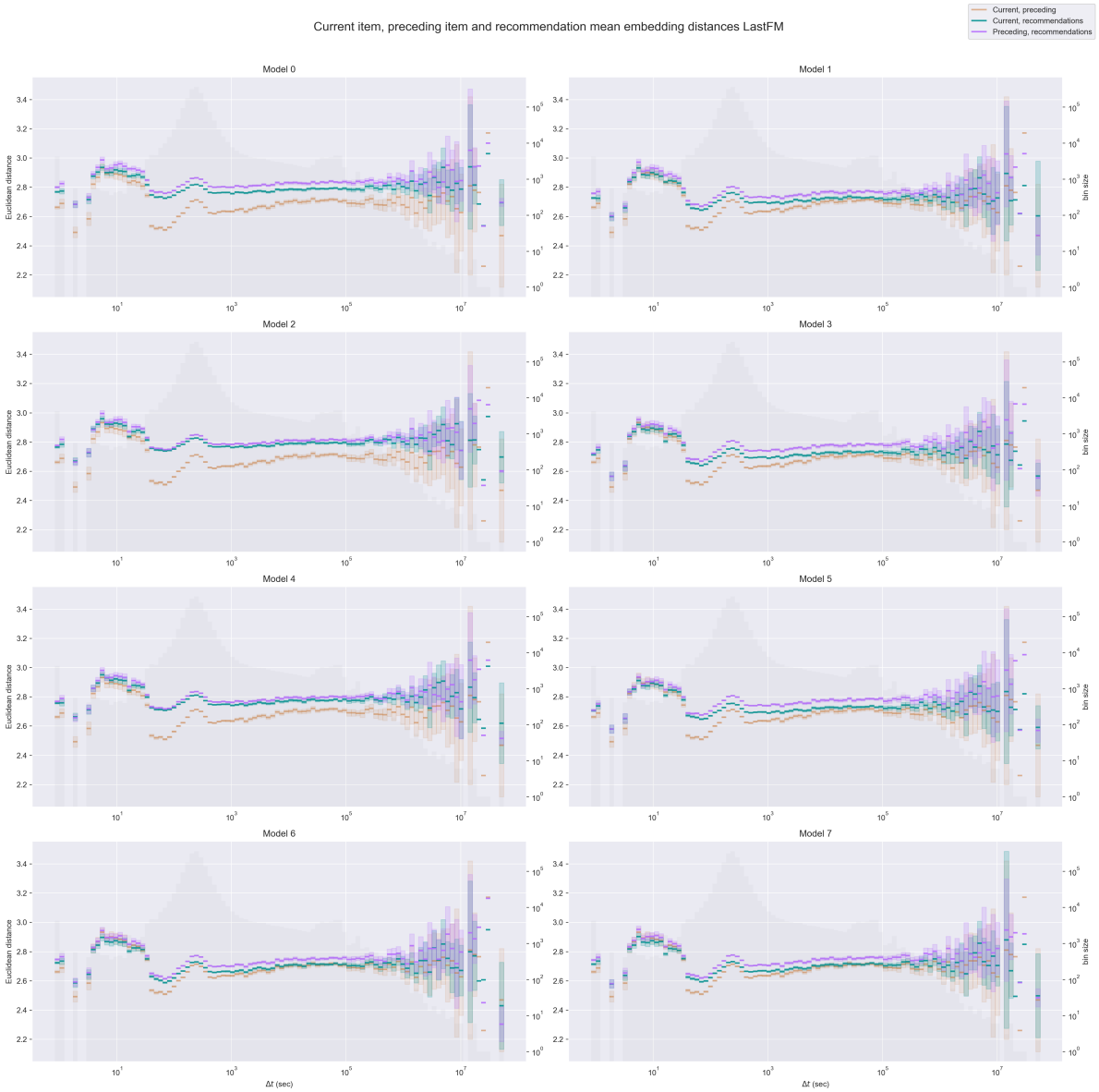


Figure B.2: $|i^{(T-1)} - \bar{p}|_2$, $|i^{(T)} - \bar{p}|_2$ of models 0 – 7, as well as $|i^{(T)} - i^{(T-1)}|_2$ with their associated 95% confidence intervals for varying time gap duration on *LastFM* dataset. Distances averaged over individual interactions falling into exponentially expanding bins. Number of interactions assigned to each bin shown in grey.

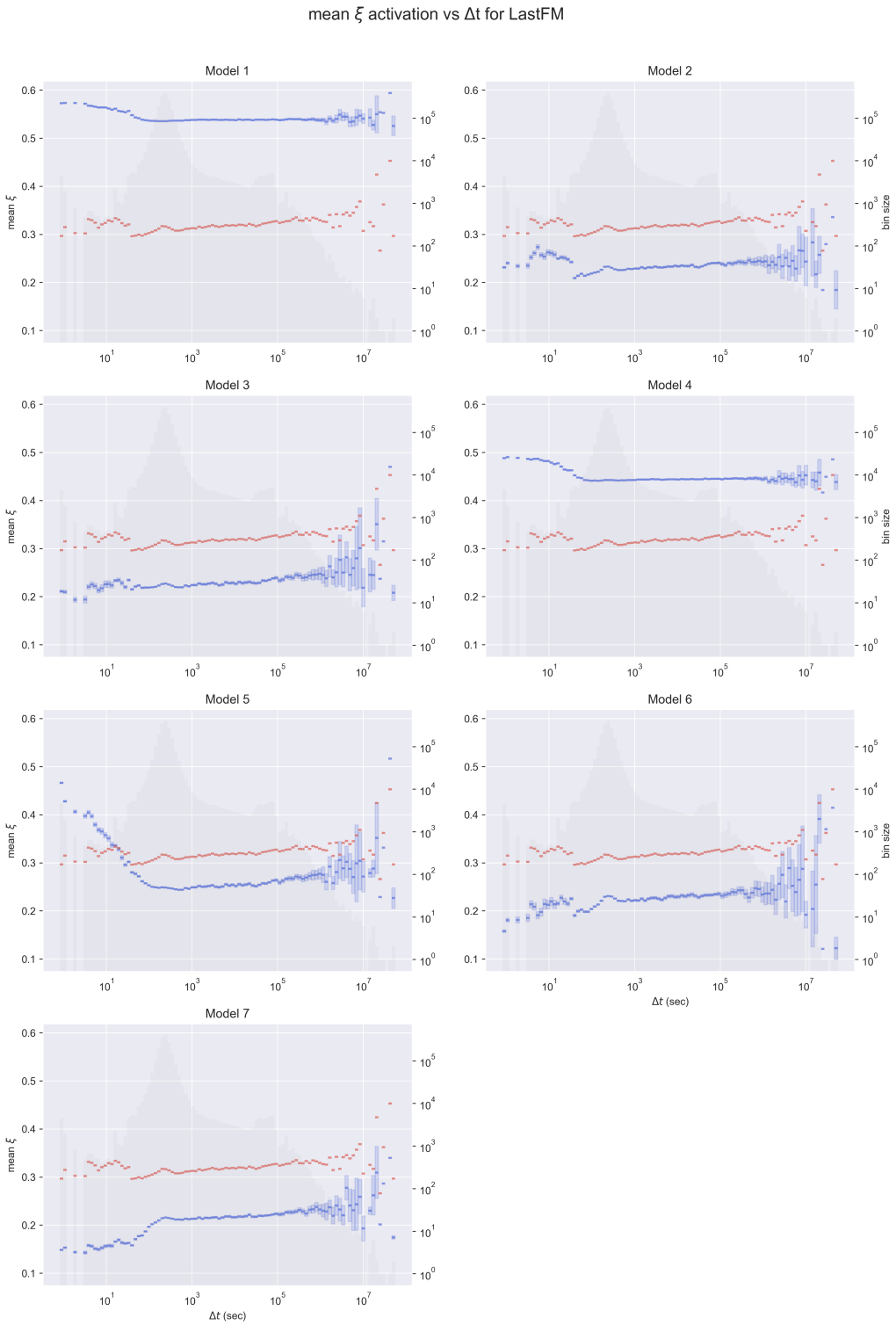


Figure B.3: Mean ξ with its associated 95% confidence interval of models 1 – 7 for varying time gap duration on *LastFM* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

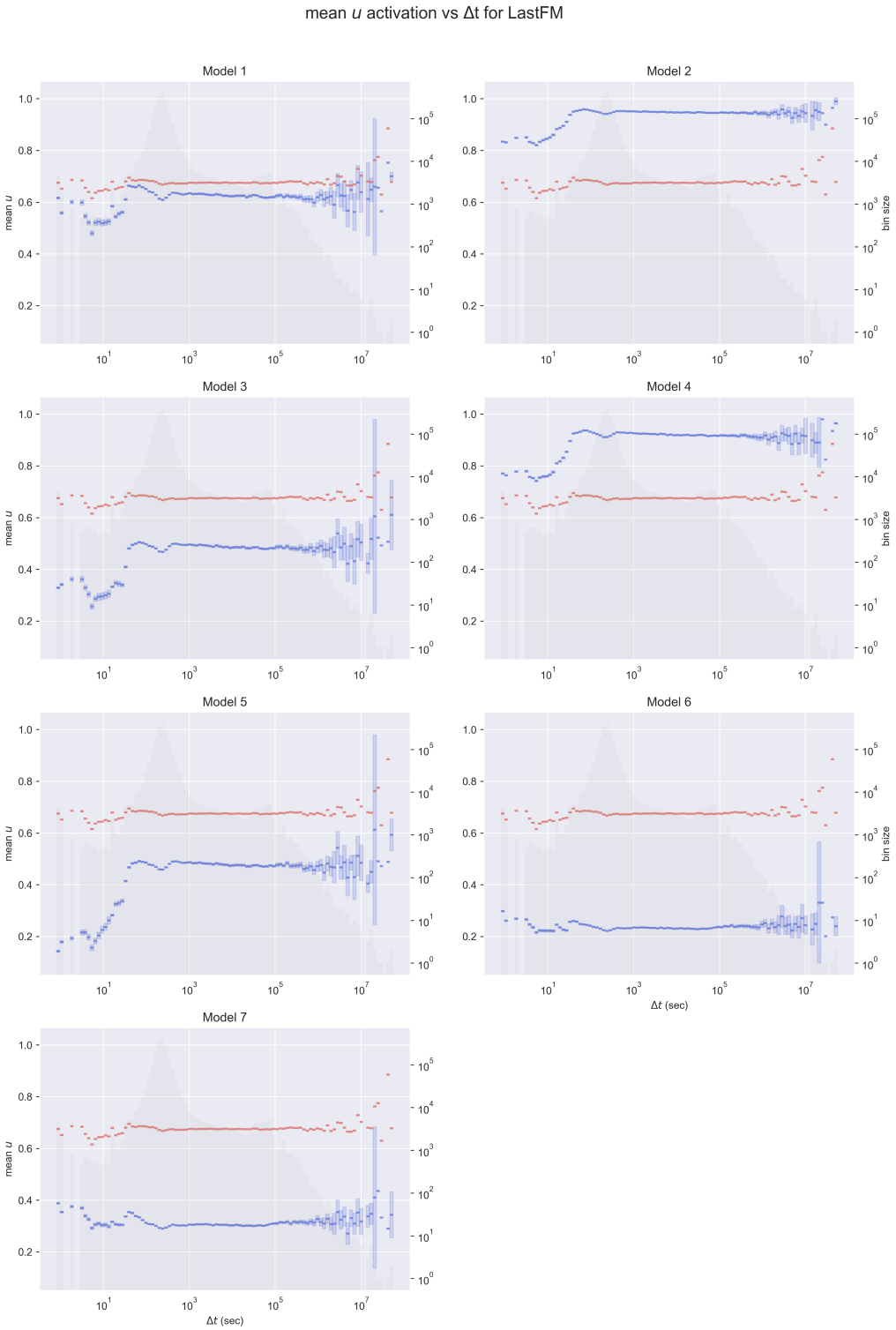


Figure B.4: Mean **u** with its associated 95% confidence interval of models 1 – 7 for varying time gap duration on *LastFM* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

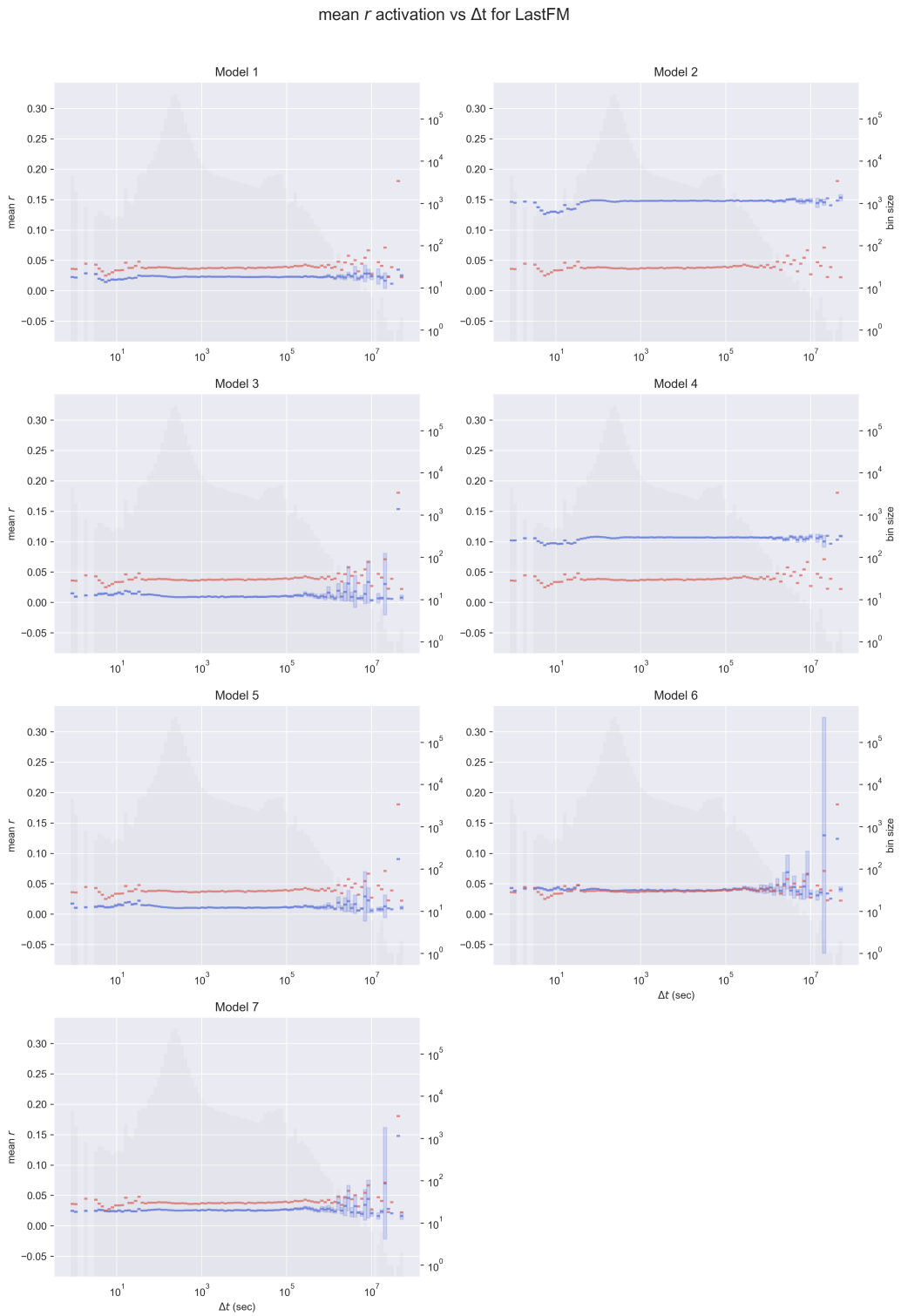


Figure B.5: Mean **r** with its associated 95% confidence interval of models 1 – 7 for varying time gap duration on *LastFM* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

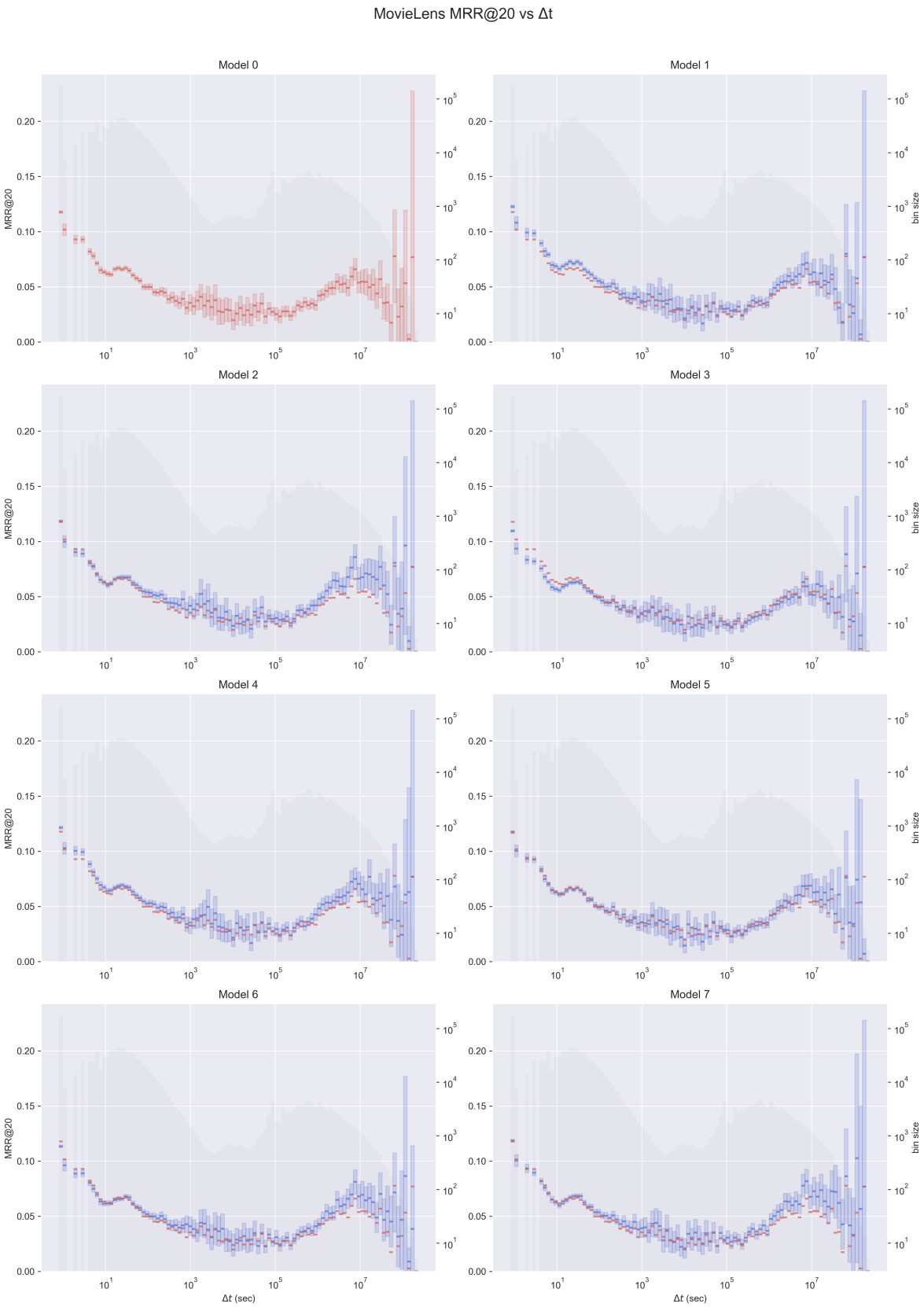


Figure B.6: MRR@20 and the associated 95% confidence intervals of models 0 – 7 for varying time gap duration on *MovieLens* dataset. Performance averaged over individual interactions falling into exponentially expanding bins. Model 0 presented in red, *DeepTimeDelta* models presented in blue. Model 0 MRR@20 included alongside that of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

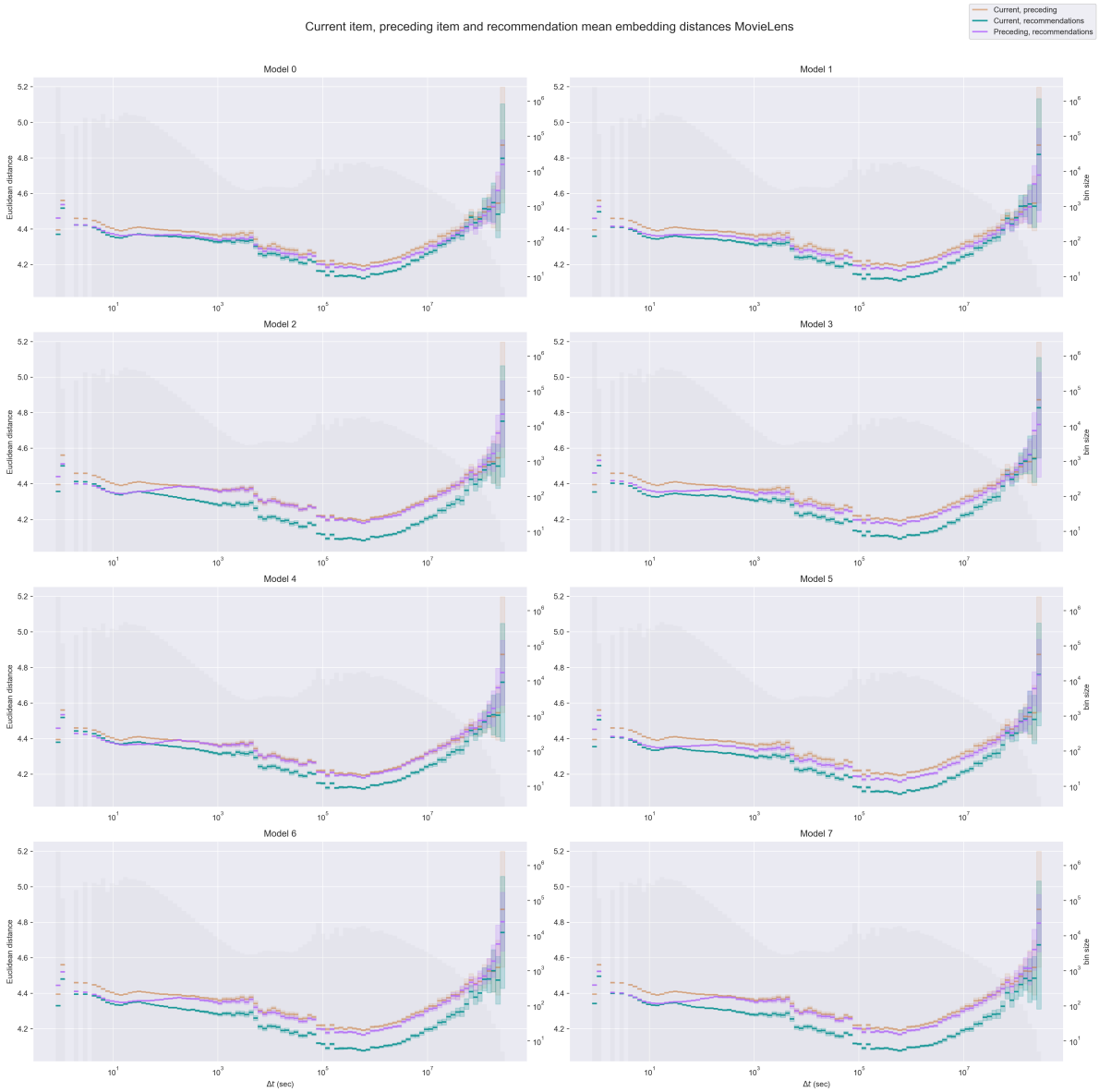


Figure B.7: $|A-P|_2$, $|B-P|_2$ of models $0-7$, as well as $|B-A|_2$ with their associated 95% confidence intervals for varying time gap duration on *MovieLens* dataset. Distances averaged over individual interactions falling into exponentially expanding bins. Number of interactions assigned to each bin shown in grey.

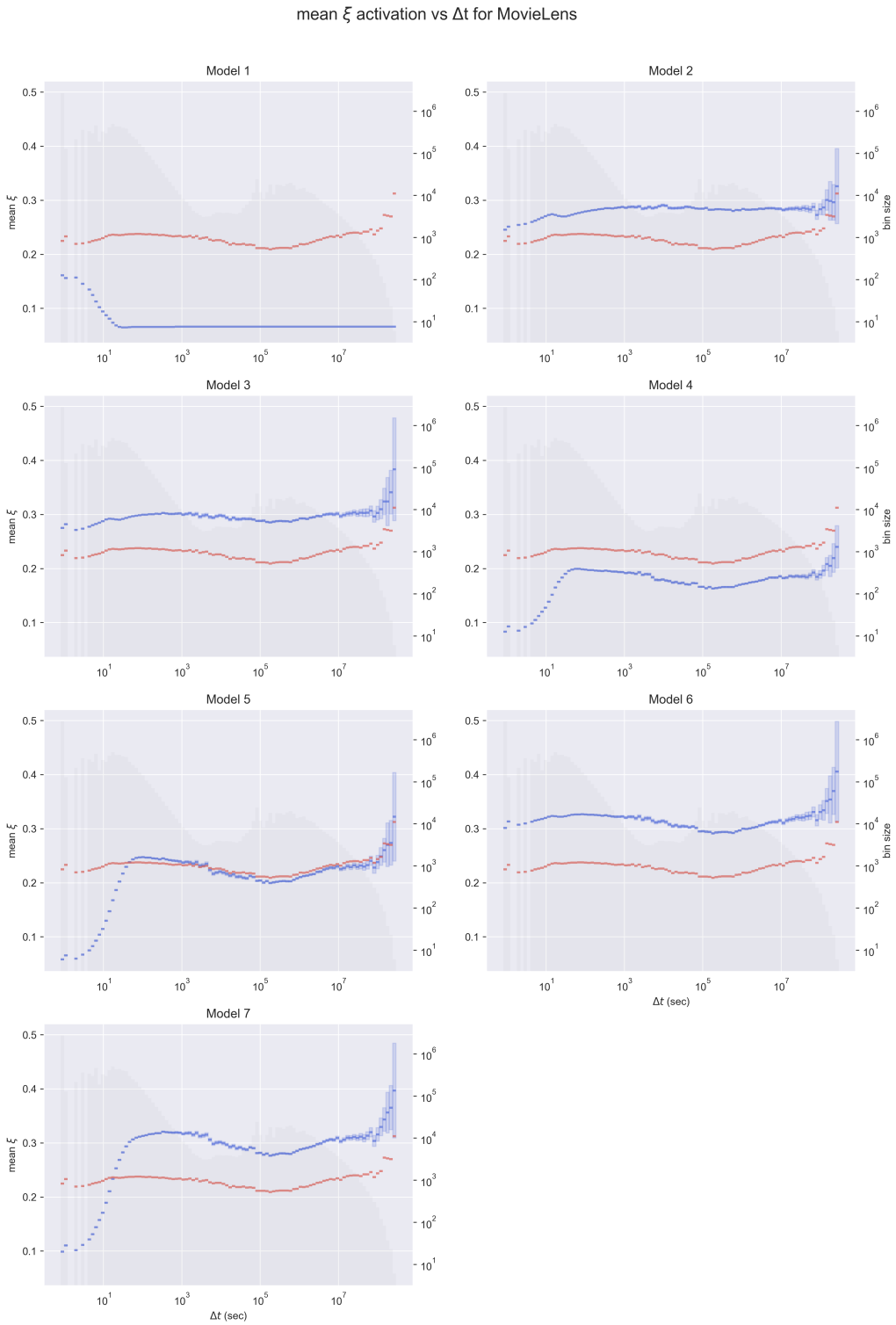


Figure B.8: Mean $\boldsymbol{\xi}$ with its associated 95% confidence interval of models 1 – 7 for varying time gap duration on *MovieLens* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

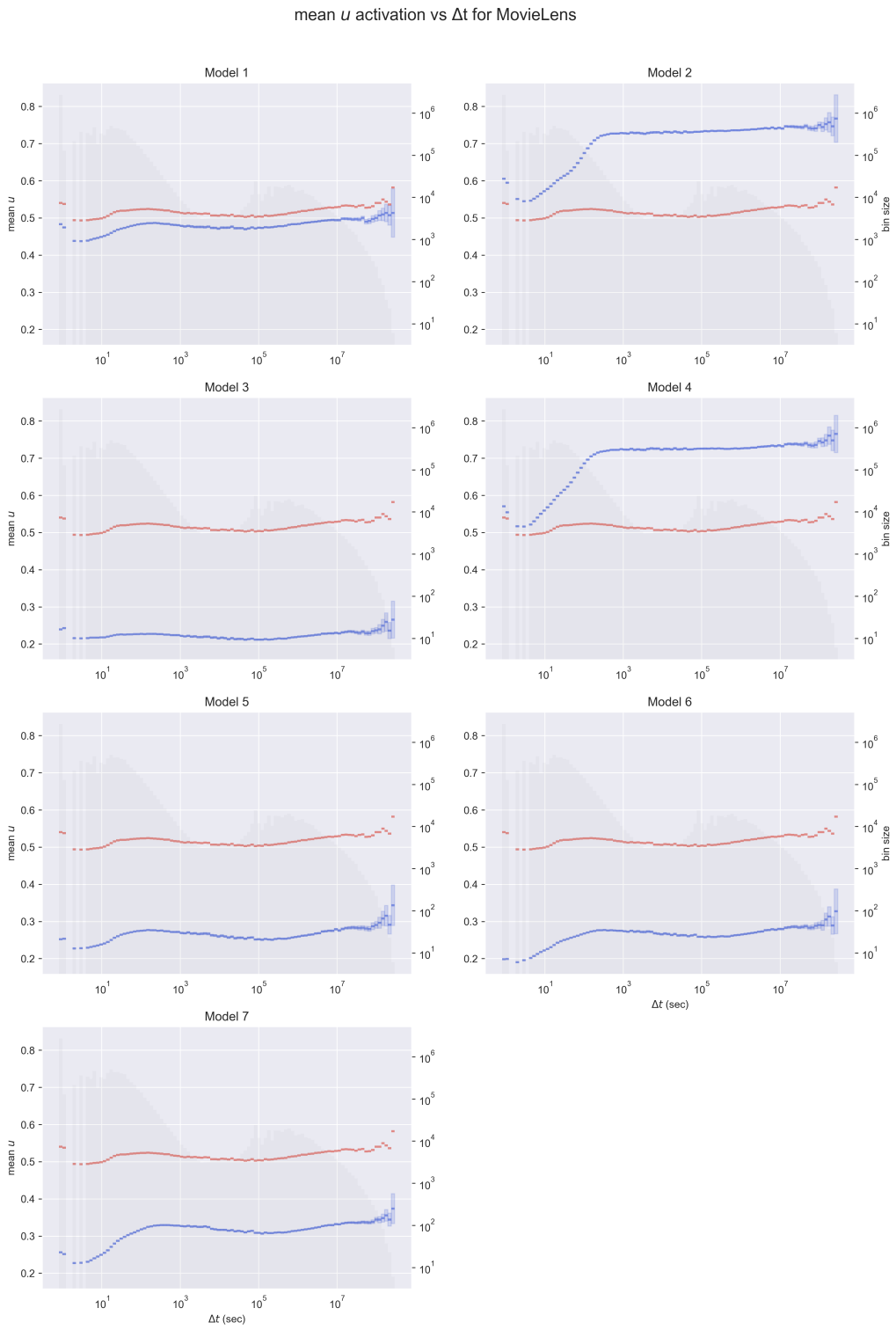


Figure B.9: Mean **u** with its associated 95% confidence interval of models 1 – 7 for varying time gap duration on *MovieLens* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

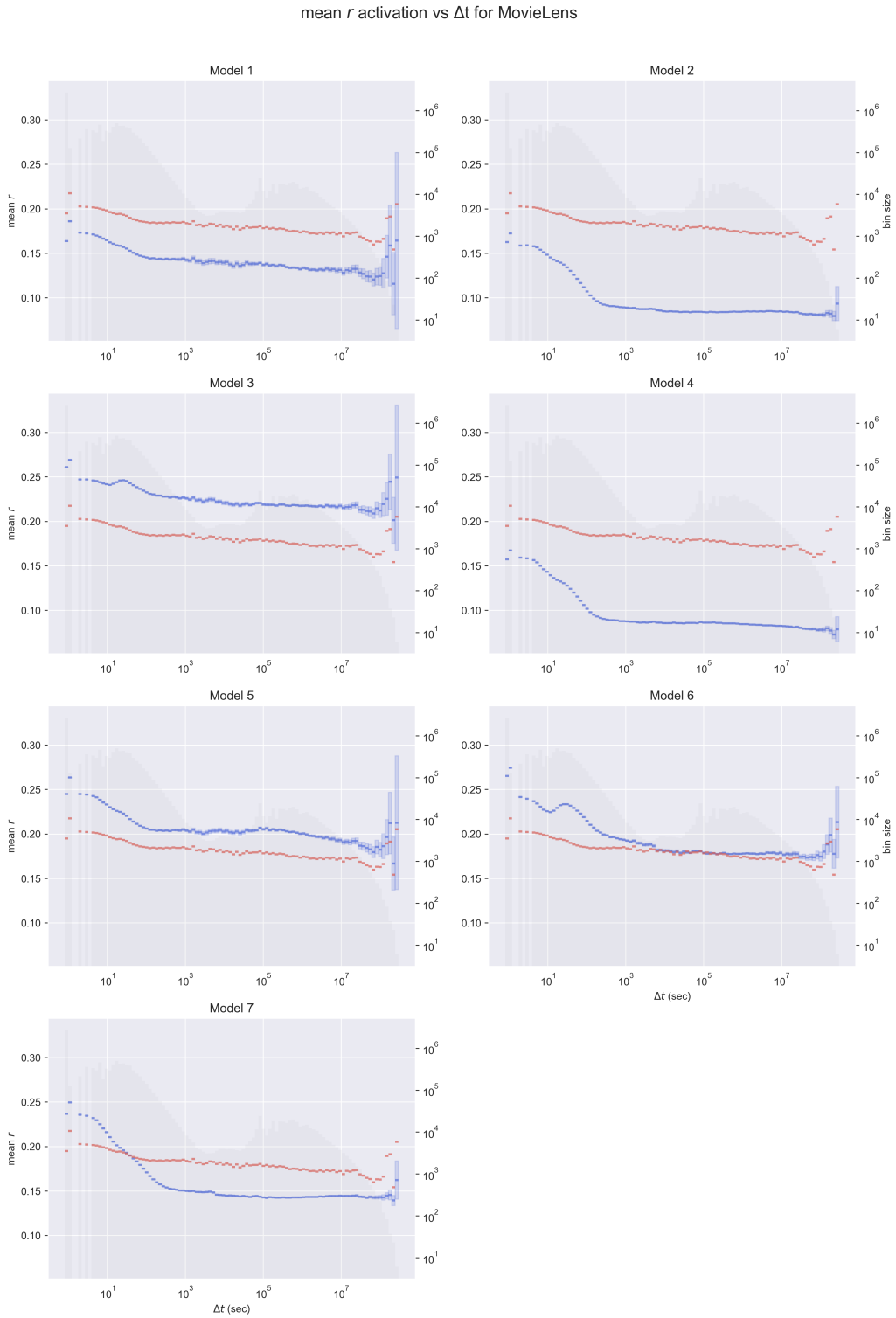


Figure B.10: Mean **r** with its associated 95% confidence interval of models 1 – 7 for varying time gap duration on *MovieLens* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity. Number of interactions assigned to each bin shown in grey.

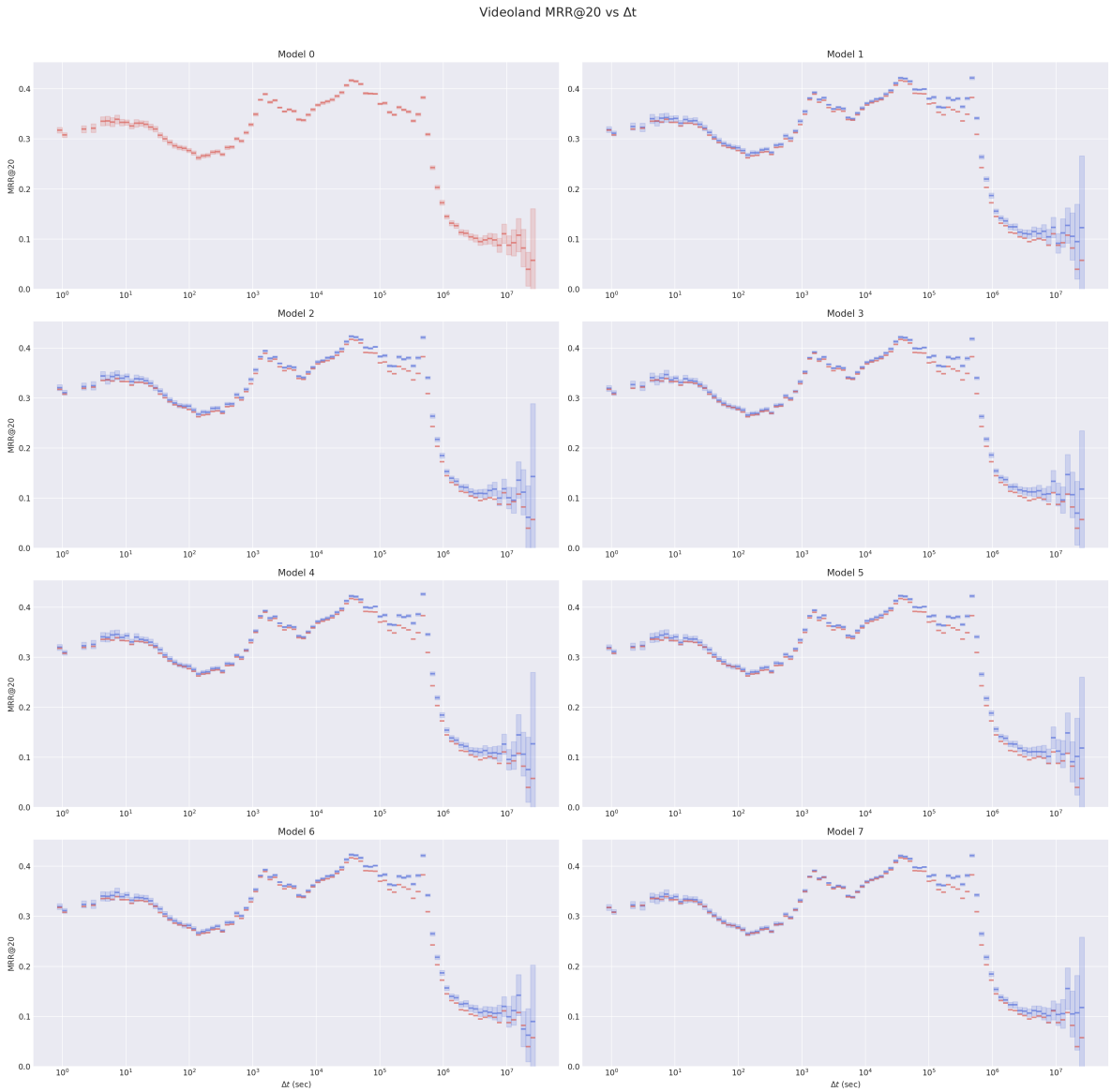


Figure B.11: MRR@20 and the associated 95% confidence intervals of models 0 – 7 for varying time gap duration on *Videoland* dataset. Performance averaged over individual interactions falling into exponentially expanding bins. Model 0 presented in red, *DeepTimeDelta* models presented in blue. Model 0 MRR@20 included alongside that of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity.

Figure B.12: $|A-P|_2$, $|B-P|_2$ of models $0-7$, as well as $|B-A|_2$ with their associated 95% confidence intervals for varying time gap duration on *Videoland* dataset. Distances averaged over individual interactions falling into exponentially expanding bins.

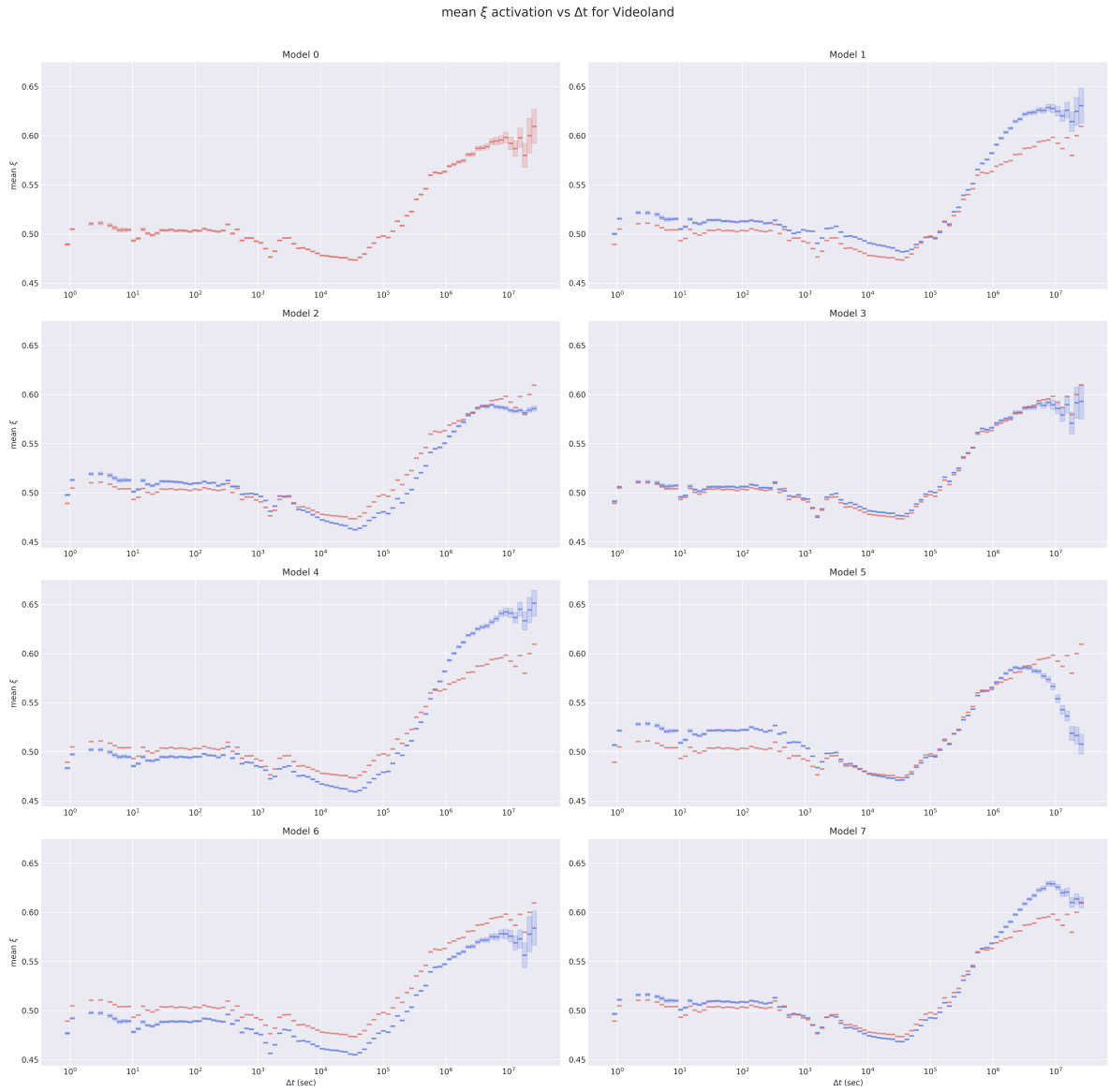


Figure B.13: Mean $\boldsymbol{\xi}$ with its associated 95% confidence interval of models $0-7$ for varying time gap duration on *Videoland* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity.

Figure B.14: Mean **u** with its associated 95% confidence interval of models 0 – 7 for varying time gap duration on *Videoland* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity.

Figure B.15: Mean **r** with its associated 95% confidence interval of models 0 – 7 for varying time gap duration on *Videoland* dataset. Mean activations averaged over individual interactions falling into exponentially expanding bins. *DeepTimeDelta* models presented in blue. Model 0 mean activations presented in red and included alongside those of *DeepTimeDelta* for additional perspective, with model 0 confidence intervals omitted for clarity.

Table B.4: ANOVA p-values of the significance of the model factor and R^2 for offline evaluation on distinct $\overline{\Delta}t_u$ user groups.

Statistic	Recall@20	MRR@20	UserRecall@20	UserMRR@20
p-value	< 0.0001	< 0.0001	< 0.0001	< 0.0001
R^2	0.44	0.27	0.52	0.34

Table B.5: ANOVA p-values of the significance of the model factor and R^2 for offline evaluation on distinct mainstreamness user groups.

Statistic	Recall@20	MRR@20	UserRecall@20	UserMRR@20
p-value	< 0.0001	< 0.0001	< 0.0001	< 0.0001
R^2	0.48	0.43	0.44	0.40

Bibliography

[1] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: A system for large-scale machine learning. pages 265–283, 2016. URL https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf.

[2] Gediminas Adomavicius and Alexander Tuzhilin. *Context-Aware Recommender Systems*, pages 191–226. Springer US, Boston, MA, 2015. ISBN 978-1-4899-7637-6. doi: 10.1007/978-1-4899-7637-6_6. URL https://doi.org/10.1007/978-1-4899-7637-6_6.

[3] Shun-ichi Amari and Si Wu. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks*, 12(6):783–789, 1999.

[4] Suhrid Balakrishnan and Sumit Chopra. Collaborative ranking. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining*, WSDM '12, page 143–152, New York, NY, USA, 2012. Association for Computing Machinery. ISBN 9781450307475. doi: 10.1145/2124295.2124314. URL https://doi.org/10.1145/2124295.2124314.

[5] Dana H Ballard. Modular learning in neural networks. In *AAAI*, pages 279–284, 1987.

[6] Linas Baltrunas and Xavier Amatriain. Towards time-dependant recommendation based on implicit feedback. *Proceedings of the Third ACM Conference on Recommender Systems*, 01 2009.

[7] Linas Baltrunas, Bernd Ludwig, and Francesco Ricci. Matrix factorization techniques for context aware recommendation. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 301–304, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0683-6. doi: 10.1145/2043932.2043988. URL http://doi.acm.org/10.1145/2043932.2043988.

[8] Thanasis G Barbounis, John B Theocharis, Minas C Alexiadis, and Petros S Dokopoulos. Long-term wind speed and power forecasting using local recurrent neural network models. *IEEE Transactions on Energy Conversion*, 21(1):273–284, 2006.

[9] Christine Bauer and Markus Schedl. Global and country-specific mainstreaminess measures: Definitions, analysis, and usage for improving personalized music recommendation systems. *PLOS ONE*, 14 (6):1–36, 06 2019. doi: 10.1371/journal.pone.0217389. URL https://doi.org/10.1371/journal.pone.0217389.

[10] Robert Bell, Yehuda Koren, and Chris Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. pages 95–104, 2007.

[11] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 1171–1179, 2015.

[12] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1): 289–300, 1995.

[13] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13:281–305, February 2012. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=2188385.2188395.

[14] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=944919.944937.

[15] Mathieu Blondel, Akinori Fujino, Naonori Ueda, and Masakazu Ishihata. Higher-order factorization machines. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3351–3359. Curran Associates, Inc., 2016. URL http://papers.nips.cc/paper/6144-higher-order-factorization-machines.pdf.

[16] Dirk Bollen, Bart P. Knijnenburg, Martijn C. Willemsen, and Mark Graus. Understanding choice overload in recommender systems. pages 63–70, 2010. doi: 10.1145/1864708.1864724. URL http://doi.acm.org/10.1145/1864708.1864724.

[17] Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.

[18] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, November 2002. ISSN 0924-1868. doi: 10.1023/A:1021240730564. URL http://dx.doi.org/10.1023/A:1021240730564.

[19] Robin Burke. The adaptive web. chapter Hybrid Web Recommender Systems, pages 377–408. Springer-Verlag, Berlin, Heidelberg, 2007. ISBN 978-3-540-72078-2. URL http://dl.acm.org/citation.cfm?id=1768197.1768211.

[20] Pedro G. Campos, Fernando Díez, and Iván Cantador. Time-aware recommender systems: A comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction*, 24(1-2):67–119, February 2014. ISSN 0924-1868. doi: 10.1007/s11257-012-9136-x. URL http://dx.doi.org/10.1007/s11257-012-9136-x.

[21] O. Celma. *Music Recommendation and Discovery in the Long Tail*. Springer, 2010.

[22] Kai Chen, Yi Zhou, and Fangyan Dai. A lstm-based method for stock returns prediction: A case study of china stock market. In *2015 IEEE international conference on big data (big data)*, pages 2823–2824. IEEE, 2015.

[23] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, DLRS 2016, pages 7–10, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4795-2. doi: 10.1145/2988450.2988454. URL http://doi.acm.org/10.1145/2988450.2988454.

[24] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014. doi: 10.3115/v1/d14-1179. URL http://dx.doi.org/10.3115/v1/D14-1179.

[25] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pages 191–198, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4035-9. doi: 10.1145/2959100.2959190. URL http://doi.acm.org/10.1145/2959100.2959190.

[26] Hanjun Dai, Yichen Wang, Rakshit Trivedi, and Le Song. Deep coevolutionary network: Embedding user and item features for recommendation, 2016.

[27] Marco Degemmis, Pasquale Lops, and Giovanni Semeraro. A content-collaborative recommender that exploits wordnet-based user profiles for neighborhood formation. *User Modeling and User-Adapted Interaction*, 17(3):217–255, July 2007. ISSN 0924-1868. doi: 10.1007/s11257-006-9023-4. URL http://dx.doi.org/10.1007/s11257-006-9023-4.

[28] Li Deng, Geoffrey Hinton, and Brian Kingsbury. New types of deep neural network learning for speech recognition and related applications: An overview. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8599–8603. IEEE, 2013.

[29] Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[30] Christian Desrosiers and George Karypis. *A Comprehensive Survey of Neighborhood-based Recommendation Methods*, pages 107–144. Springer US, Boston, MA, 2011. ISBN 978-0-387-85820-3. doi: 10.1007/978-0-387-85820-3_4. URL https://doi.org/10.1007/978-0-387-85820-3_4.

[31] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[32] Yi Ding and Xue Li. Time weight collaborative filtering. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 485–492, New York, NY, USA, 2005. ACM. ISBN 1-59593-140-6. doi: 10.1145/1099554.1099689. URL http://doi.acm.org/10.1145/1099554.1099689.

[33] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. Sequential user-based recurrent neural network recommendations. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17, pages 152–160, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4652-8. doi: 10.1145/3109859.3109877. URL http://doi.acm.org/10.1145/3109859.3109877.

[34] Nan Du, Yichen Wang, Niao He, and Le Song. Time-sensitive recommendation from recurrent user activities. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pages 3492–3500, Cambridge, MA, USA, 2015. MIT Press. URL http://dl.acm.org/citation.cfm?id=2969442.2969629.

[35] Olive Jean Dunn. Multiple comparisons among means. *Journal of the American statistical association*, 56(293):52–64, 1961.

[36] Charles W Dunnett. A multiple comparison procedure for comparing several treatments with a control. *Journal of the American Statistical Association*, 50(272):1096–1121, 1955.

[37] Peter Emerson. The original borda count and partial voting. *Social Choice and Welfare*, 40(2):353–358, Feb 2013. ISSN 1432-217X. doi: 10.1007/s00355-011-0603-9. URL https://doi.org/10.1007/s00355-011-0603-9.

[38] Katayoun Farrahi, Markus Schedl, Andreu Vall, David Hauger, and Marko Tkalcic. Impact of listening behavior on music recommendation. 2014.

[39] Ronald Aylmer Fisher. Statistical methods for research workers. In *Breakthroughs in statistics*, pages 66–70. Springer, 1992.

[40] Simon Funk. Netflix update: Try this at home. 2006. URL https://sifter.org/simon/journal/20061211.html.

[41] F. A. Gers and J. Schmidhuber. Recurrent nets that time and count. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, volume 3, pages 189–194 vol.3, July 2000. doi: 10.1109/IJCNN.2000.861302.

[42] Jennifer Golbeck. Generating predictive movie recommendations from trust in social networks. In *Proceedings of the 4th International Conference on Trust Management*, iTrust'06, pages 93–104, Berlin, Heidelberg, 2006. Springer-Verlag. ISBN 3-540-34295-8, 978-3-540-34295-3. doi: 10.1007/11755593_8. URL http://dx.doi.org/10.1007/11755593_8.

[43] Miha Grčar, Blaž Fortuna, Dunja Mladenič, and Marko Grobelnik. knn versus svm in the collaborative filtering framework. pages 251–260, 2006.

[44] Klaus Greff, Rupesh K. Srivastava, Jan Koutnik, Bas R. Steunebrink, and Jurgen Schmidhuber. Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, Oct 2017. ISSN 2162-2388. doi: 10.1109/tnnls.2016.2582924. URL http://dx.doi.org/10.1109/TNNLS.2016.2582924.

[45] Weiyu Guo, Shu Wu, Liang Wang, and Tieniu Tan. Personalized ranking with pairwise factorization machines. *Neurocomput.*, 214(C):191–200, November 2016. ISSN 0925-2312. doi: 10.1016/j.neucom.2016.05.074. URL https://doi.org/10.1016/j.neucom.2016.05.074.

[46] F. Maxwell Harper and Joseph A. Konstan. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, 5(4):19:1–19:19, December 2015. ISSN 2160-6455. doi: 10.1145/2827872. URL http://doi.acm.org/10.1145/2827872.

[47] Gerald Häubl and Valerie Trifts. Consumer decision making in online shopping environments: The effects of interactive decision aids. *Marketing Science*, 19(1):4–21, February 2000. ISSN 1526-548X. doi: 10.1287/mksc.19.1.4.15178. URL http://dx.doi.org/10.1287/mksc.19.1.4.15178.

[48] Ruining He and Julian McAuley. Fusing similarity models with markov chains for sparse sequential recommendation. *2016 IEEE 16th International Conference on Data Mining (ICDM)*, Dec 2016. doi: 10.1109/icdm.2016.0030. URL http://dx.doi.org/10.1109/ICDM.2016.0030.

[49] Jonathan L Herlocker, Joseph A Konstan, and John Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 241–250, 2000.

[50] Christoph Hermann. Time-based recommendations for lecture materials. In Jan Herrington and Craig Montgomerie, editors, *Proceedings of EdMedia + Innovate Learning 2010*, pages 1028–1033, Toronto, Canada, June 2010. Association for the Advancement of Computing in Education (AACE). URL https://www.learntechlib.org/p/34759.

[51] Balázs Hidasi and Domonkos Tikk. Fast als-based tensor factorization for context-aware recommendation from implicit feedback. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 67–82. Springer, 2012.

[52] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pages 241–248, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4035-9. doi: 10.1145/2959100.2959167. URL http://doi.acm.org/10.1145/2959100.2959167.

[53] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks, 2015.

[54] Will Hill, Larry Stead, Mark Rosenstein, and George Furnas. Recommending and evaluating choices in a virtual community of use. page 194–201, 1995. doi: 10.1145/223904.223929. URL https://doi.org/10.1145/223904.223929.

[55] Y. Hochberg and A. C. Tamhane. *Multiple Comparison Procedures*. John Wiley & Sons, Inc., USA, 1987. ISBN 0471822221.

[56] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[57] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural networks*, 4(2): 251–257, 1991.

[58] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *2008 Eighth IEEE International Conference on Data Mining*, pages 263–272. Ieee, 2008.

[59] Sheena Iyengar and Mark Lepper. When choice is demotivating: Can one desire too much of a good thing? *Journal of personality and social psychology*, 79:995–1006, 01 2001. doi: 10.1037/0022-3514.79.6.995.

[60] Amir H Jadidinejad, Craig Macdonald, and Iadh Ounis. Unifying explicit and implicit feedback for rating prediction and ranking recommendation tasks. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 149–156, 2019.

[61] Michael Jahrer and Andreas Töscher. Collaborative filtering ensemble for ranking. In *Proceedings of the 2011 International Conference on KDD Cup 2011-Volume 18*, pages 153–167, 2011.

[62] Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. On using very large target vocabulary for neural machine translation. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015. doi: 10.3115/v1/p15-1001. URL http://dx.doi.org/10.3115/v1/P15-1001.

[63] How Jing and Alexander J. Smola. Neural survival recommender. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM '17, pages 515–524, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4675-7. doi: 10.1145/3018661.3018719. URL http://doi.acm.org/10.1145/3018661.3018719.

[64] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Trans. Inf. Syst.*, 25(2), April 2007. ISSN 1046-8188. doi: 10.1145/1229179.1229181. URL http://doi.acm.org/10.1145/1229179.1229181.

[65] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. Unbiased learning-to-rank with biased feedback. *CoRR*, abs/1608.04468, 2016. URL http://arxiv.org/abs/1608.04468.

[66] Christopher C Johnson. Logistic matrix factorization for implicit feedback data. *Advances in Neural Information Processing Systems*, 27, 2014.

[67] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206. IEEE, 2018.

[68] Komal Kapoor, Mingxuan Sun, Jaideep Srivastava, and Tao Ye. A hazard based approach to user return time prediction. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 1719–1728, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2956-9. doi: 10.1145/2623330.2623348. URL http://doi.acm.org/10.1145/2623330.2623348.

[69] Komal Kapoor, Karthik Subbian, Jaideep Srivastava, and Paul Schrater. Just in time recommendations: Modeling the dynamics of boredom in activity streams. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM '15, pages 233–242, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3317-7. doi: 10.1145/2684822.2685306. URL http://doi.acm.org/10.1145/2684822.2685306.

[70] Alexandros Karatzoglou, Linas Baltrunas, and Yue Shi. Learning to rank for recommender systems. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 493–494, 2013.

[71] Andrej Karpathy. Cs231n winter 2016: Lecture 10: Recurrent neural networks, image captioning, lstm. https://www.youtube.com/watch?v=yCC09vCHzF8, February 2016.

[72] Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and understanding recurrent networks. 2015.

[73] Shah Khusro, Zafar Ali, and Irfan Ullah. Recommender systems: issues, challenges, and research opportunities. pages 1179–1189, 2016.

[74] Jaehun Kim, Julián Urbano, Cynthia C. S. Liem, and Alan Hanjalic. One deep music representation to rule them all? a comparative analysis of different representation learning strategies. *Neural Computing and Applications*. ISSN 1433-3058. doi: 10.1007/s00521-019-04076-1. URL https://doi.org/10.1007/s00521-019-04076-1.

[75] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[76] Bart P. Knijnenburg and Martijn C. Willemsen. Evaluating recommender systems with user experiments. pages 309–352, 2015. doi: 10.1007/978-1-4899-7637-6_9. URL https://doi.org/10.1007/978-1-4899-7637-6_9.

[77] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. Grouplens: Applying collaborative filtering to usenet news. *Commun. ACM*, 40(3):77–87, March 1997. ISSN 0001-0782. doi: 10.1145/245108.245126. URL https://doi.org/10.1145/245108.245126.

[78] Yehuda Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 426–434, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-193-4. doi: 10.1145/1401890.1401944. URL http://doi.acm.org/10.1145/1401890.1401944.

[79] Yehuda Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 447–456, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-495-9. doi: 10.1145/1557019.1557072. URL http://doi.acm.org/10.1145/1557019.1557072.

[80] Yehuda Koren. The bellkor solution to the netflix grand prize. 2009.

[81] Yehuda Koren and Robert Bell. *Advances in Collaborative Filtering*, pages 77–118. 01 2015. ISBN 978-1-4899-7636-9. doi: 10.1007/978-1-4899-7637-6_3.

[82] Jan Koutník, Klaus Greff, Faustino Gomez, and Jürgen Schmidhuber. A clockwork rnn, 2014.

[83] William H. Kruskal and W. Allen Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, 47(260):583–621, 1952. doi: 10.1080/01621459.1952.10483441.

[84] Miklós Kurucz, András A Benczúr, and Károly Csalogány. Methods for large scale svd with missing values. In *Proceedings of KDD cup and workshop*, volume 12, pages 31–38. Citeseer, 2007.

[85] John Lafferty and Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 111–119, 2001.

[86] Neal Lathia, Stephen Hailes, and Licia Capra. Temporal collaborative filtering with adaptive neighbourhoods. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09, pages 796–797, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-483-6. doi: 10.1145/1571941.1572133. URL http://doi.acm.org/10.1145/1571941.1572133.

[87] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. Context-aware sequential recommendation. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 1053–1058. IEEE, 2016.

[88] Babak Loni, Alan Said, Martha Larson, and Alan Hanjalic. 'free lunch'enhancement for collaborative filtering with factorization machines. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 281–284, 2014.

[89] Babak Loni, Martha Larson, and Alan Hanjalic. Factorization machines for data with implicit feedback. *arXiv preprint arXiv:1812.08254*, 2018.

[90] P. Lops, M. de Gemmis, and G. Semeraro. *Content-based Recommender Systems: State of the Art and Trends*, page 73. 2011. doi: 10.1007/978-0-387-85820-3_3.

[91] Augusto Q Macedo, Leandro B Marinho, and Rodrygo LT Santos. Context-aware event recommendation in event-based social networks. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 123–130, 2015.

[92] Tariq Mahmood and Francesco Ricci. Improving recommender systems with adaptive conversational strategies. In *Proceedings of the 20th ACM Conference on Hypertext and Hypermedia*, HT '09, pages 73–82, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-486-7. doi: 10.1145/1557914.1557930. URL http://doi.acm.org/10.1145/1557914.1557930.

[93] Henry B Mann and Donald R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60, 1947.

[94] John H McDonald. *Handbook of biological statistics*, volume 2. sparky house publishing Baltimore, MD, 2009.

[95] Sean M. McNee, Istvan Albert, Dan Cosley, Prateep Gopalkrishnan, Shyong K. Lam, Al Mamunur Rashid, Joseph A. Konstan, and John Riedl. On the recommending of citations for research papers. pages 116–125, 2002. doi: 10.1145/587078.587096. URL `http://doi.acm.org/10.1145/587078.587096`.

[96] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[97] Andriy Mnih and Ruslan R Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2008.

[98] Omar Moling, Linas Baltrunas, and Francesco Ricci. Optimal radio channel recommendations with explicit and implicit feedback. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 75–82, 2012.

[99] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[100] David Nichols. Implicit rating and filtering. ERCIM, 1998.

[101] Xia Ning and George Karypis. Slim: Sparse linear methods for top-n recommender systems. In *2011 IEEE 11th International Conference on Data Mining*, pages 497–506. IEEE, 2011.

[102] Christopher Olah. Understanding lstm networks. `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`, August 2015.

[103] Elias Pampalk, Tim Pohle, and Gerhard Widmer. Dynamic playlist generation based on skipping behavior. In *ISMIR*, volume 5, pages 634–637, 2005.

[104] Ashif Panakkat and Hojjat Adeli. Recurrent neural network for approximate earthquake time and location prediction using multiple seismicity indicators. *Computer-Aided Civil and Infrastructure Engineering*, 24(4):280–292, 2009.

[105] Eli Pariser. *The filter bubble: What the Internet is hiding from you*. Penguin UK, 2011.

[106] Denis Parra, Alexandros Karatzoglou, Xavier Amatriain, and Idil Yavuz. Implicit feedback recommendation via implicit-to-explicit ordinal logistic regression mapping. *Proceedings of the CARS-2011*, page 5, 2011.

[107] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. 2007:5–8, 2007.

[108] Wenjie Pei, Jie Yang, Zhu Sun, Jie Zhang, Alessandro Bozzon, and David M.J. Tax. Interacting attention-gated recurrent networks for recommendation. *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management - CIKM '17*, 2017. doi: 10.1145/3132847.3133005. URL `http://dx.doi.org/10.1145/3132847.3133005`.

[109] Gianluca Pollastri, Darisz Przybylski, Burkhard Rost, and Pierre Baldi. Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles. *Proteins: Structure, Function, and Bioinformatics*, 47(2):228–235, 2002.

[110] Nornadiah Mohd Razali, Yap Bee Wah, et al. Power comparisons of shapiro-wilk, kolmogorov-smirnov, lilliefors and anderson-darling tests. *Journal of statistical modeling and analytics*, 2(1):21–33, 2011.

[111] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[112] Steffen Rendle. Factorization machines. In *2010 IEEE International Conference on Data Mining*, pages 995–1000. IEEE, 2010.

[113] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 452–461. AUAI Press, 2009.

[114] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 811–820, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-799-8. doi: 10.1145/1772690.1772773. URL http://doi.acm.org/10.1145/1772690.1772773.

[115] Francesco Ricci, Lior Rokach, and Bracha Shapira. *Introduction to Recommender Systems Handbook*, pages 1–35. Springer US, Boston, MA, 2011. ISBN 978-0-387-85820-3. doi: 10.1007/978-0-387-85820-3_1. URL https://doi.org/10.1007/978-0-387-85820-3_1.

[116] David R Roberts, Volker Bahn, Simone Ciuti, Mark S Boyce, Jane Elith, Gurutzeta Guillera-Arroita, Severin Hauenstein, José J Lahoz-Monfort, Boris Schröder, Wilfried Thuiller, et al. Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography*, 40(8):913–929, 2017.

[117] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[118] Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of the 25th international conference on Machine learning*, pages 880–887, 2008.

[119] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA, 2001. ACM. ISBN 1-58113-348-0. doi: 10.1145/371920.372071. URL http://doi.acm.org/10.1145/371920.372071.

[120] Markus Schedl and Christine Bauer. Introducing global and regional mainstreaminess for improving personalized music recommendation. pages 74–81, 2017. doi: 10.1145/3151848.3151849. URL http://doi.acm.org/10.1145/3151848.3151849.

[121] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, and Lexing Xie. Autorec: Autoencoders meet collaborative filtering. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 111–112, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3473-0. doi: 10.1145/2740908.2742726. URL http://doi.acm.org/10.1145/2740908.2742726.

[122] Hanhuai Shan and Arindam Banerjee. Generalized probabilistic matrix factorizations for collaborative filtering. In *2010 IEEE International Conference on Data Mining*, pages 1025–1030. IEEE, 2010.

[123] Guy Shani, David Heckerman, and Ronen I. Brafman. An mdp-based recommender system. *J. Mach. Learn. Res.*, 6:1265–1295, December 2005. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1046920.1088715.

[124] Samuel Sanford Shapiro and Martin B Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, 1965.

[125] Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating "word of mouth". pages 210–217, 1995.

[126] Yue Shi, Martha Larson, and Alan Hanjalic. Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Comput. Surv.*, 47(1):3:1–3:45, May 2014. ISSN 0360-0300. doi: 10.1145/2556270. URL http://doi.acm.org/10.1145/2556270.

[127] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

[128] Jonathon Shlens. A tutorial on principal component analysis. *CoRR*, abs/1404.1100, 2014. URL http://arxiv.org/abs/1404.1100.

[129] Yang Song, Ali Mamdouh Elkahky, and Xiaodong He. Multi-rate deep learning for temporal recommendation. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, pages 909–912, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4069-4. doi: 10.1145/2911451.2914726. URL http://doi.acm.org/10.1145/2911451.2914726.

[130] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[131] Student. The probable error of a mean. *Biometrika*, pages 1–25, 1908.

[132] Xiaoyuan Su, Taghi M. Khoshgoftaar, Xingquan Zhu, and Russell Greiner. Imputation-boosted collaborative filtering using machine learning classifiers. In *Proceedings of the 2008 ACM Symposium on Applied Computing*, SAC '08, pages 949–950, New York, NY, USA, 2008. ACM. ISBN 978-1-59593-753-7. doi: 10.1145/1363686.1363903. URL http://doi.acm.org/10.1145/1363686.1363903.

[133] Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert MÃžller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8(May):985–1005, 2007.

[134] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. Investigation of various matrix factorization methods for large recommender systems. pages 553–562, 2008.

[135] Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. pages 565–573, 2018.

[136] Nava Tintarev and Judith Masthoff. A survey of explanations in recommender systems. In *2007 IEEE 23rd international conference on data engineering workshop*, pages 801–810. IEEE, 2007.

[137] Roberto Torres, Sean M. McNee, Mara Abel, Joseph A. Konstan, and John Riedl. Enhancing digital libraries with techlens+. In *Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '04, pages 228–236, New York, NY, USA, 2004. ACM. ISBN 1-58113-832-6. doi: 10.1145/996350.996402. URL http://doi.acm.org/10.1145/996350.996402.

[138] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

[139] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1235–1244, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3664-2. doi: 10.1145/2783258.2783273. URL http://doi.acm.org/10.1145/2783258.2783273.

[140] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, page 403–412, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450336215. doi: 10.1145/2766462.2767694. URL https://doi.org/10.1145/2766462.2767694.

[141] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. Learning to rank with selection bias in personal search. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, pages 115–124, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4069-4. doi: 10.1145/2911451.2911537. URL http://doi.acm.org/10.1145/2911451.2911537.

[142] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. Recurrent recommender networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM '17, pages 495–503, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4675-7. doi: 10.1145/3018661.3018689. URL http://doi.acm.org/10.1145/3018661.3018689.

[143] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. Attentional factorization machines: Learning the weight of feature interactions via attention networks. *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, Aug 2017. doi: 10.24963/ijcai.2017/435. URL http://dx.doi.org/10.24963/ijcai.2017/435.

[144] Dingqi Yang, Daqing Zhang, Vincent W Zheng, and Zhiyong Yu. Modeling user activity preference by leveraging user spatial temporal characteristics in lbsns. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(1):129–142, 2014.

[145] Matei Zaharia, Reynold S. Xin, Patrick Wendell, Tathagata Das, Michael Armbrust, Ankur Dave, Xiangrui Meng, Josh Rosen, Shivaram Venkataraman, Michael J. Franklin, Ali Ghodsi, Joseph Gonzalez, Scott Shenker, and Ion Stoica. Apache spark: A unified engine for big data processing. *Commun. ACM*, 59 (11):56–65, October 2016. ISSN 0001-0782. doi: 10.1145/2934664. URL https://doi.org/10.1145/2934664.

[146] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.

[147] S. Zhang, W. Wang, J. Ford, F. Makedon, and J. Pearlman. Using singular value decomposition approximation for collaborative filtering. In *Seventh IEEE International Conference on E-Commerce Technology (CEC'05)*, pages 257–264, July 2005. doi: 10.1109/ICECT.2005.102.

[148] Shuai Zhang, Yi Tay, Lina Yao, and Aixin Sun. Next item recommendation with self-attention. *arXiv preprint arXiv:1808.06414*, 2018.

[149] Zhe Zhao, Lichan Hong, Li Wei, Jilin Chen, Aniruddh Nath, Shawn Andrews, Aditee Kumthekar, Maheswaran Sathiamoorthy, Xinyang Yi, and Ed Chi. Recommending what video to watch next: A multitask ranking system. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, pages 43–51, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-6243-6. doi: 10.1145/3298689.3346997. URL http://doi.acm.org/10.1145/3298689.3346997.

[150] Chang Zhou, Jinze Bai, Junshuai Song, Xiaofei Liu, Zhengchao Zhao, Xiusi Chen, and Jun Gao. Atrank: An attention-based user behavior modeling framework for recommendation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[151] Yu Zhu, Jinghao Lin, Shibi He, Beidou Wang, Ziyu Guan, Haifeng Liu, and Deng Cai. Addressing the item cold-start problem by attribute-driven active learning. *IEEE Transactions on Knowledge and Data Engineering*, page 1–1, 2019. ISSN 2326-3865. doi: 10.1109/tkde.2019.2891530. URL http://dx.doi.org/10.1109/tkde.2019.2891530.