Addressing Fast Changing Fashion Trends in Multi-Stage Recommender Systems

Aayush Singha Roy[1,2*] **Edoardo D'Amico**[1,2*] **Aonghus Lawlor**[1,2] **Neil Hurley**[1,2]
[1]University College of Dublin, Dublin, Ireland
[2]Insight Centre for Data Analytics, Dublin, Ireland
{aayush.singharoy, edoardo.damico, aonghus.lawlor, neil.hurley}@insight-centre.org

Abstract

Fashion industry is driven by fashion cycles, in which a fashion item is launched, rises to mainstream appeal and becomes a trend, then diminishes and eventually becomes obsolete. These properties make it critical to incorporate temporal information when adapting a recommendation framework to be employed in the fashion domain. However, an industry standard real-world recommendation architecture entails numerous phases, including data preparation, establishing and training recommender models, filtering and fulfilling revenue-based user needs. The contributions of the presented work are twofold. We first formalise the multi-stage recommendation pipeline by including the time dimension intrinsically present in the fashion data. We then present a study to incorporate explicit fashion domain characteristics into the presented pipeline. Finally, we conduct comprehensive experimentation on a real-world web-scale fashion dataset released by H&M, illustrating how including domain knowledge in the multi-stage framework can lead to significantly improvement on the final recommendation performance.

Introduction

The notion of *trend* is an important driving force in fashion. Many users desire fashion items in line with current trends, but even a dislike of the most current fashion trends can be considered a trend in itself. A recommender system model which learns from past interactions between a user and fashion items, needs to take into account the dynamics of trends in a very precise way. Fashion recommendations in general have underlying characteristics which are different from traditional product recommendations, and these features can be leveraged in order to offer a better experience to end users.

When dealing with real-world problems, it is possible to identify a common architectural pattern used by companies when deploying large-scale multi-stage recommendation systems models which comprises of two major components: A *recommender system model*, capable of generating candidate items, followed by a *re-ranker model*, which generates the final list of recommendations for each user, as proposed to some extent in papers and blogs by NVIDIA (Higley et al. 2022), Google (Covington, Adams, and Sargin 2016), Meta's Instagram (Medvedev, Wu, and Gordon 2019) and Pinterest (Liu et al. 2017a). Although multi-stage recommendation systems are widely utilised in industry, the steps required to design one are not clearly explained in literature. In this paper we start by presenting a general formalisation of multi-stage recommender architectures to deal with temporal data. We then examine fashion trend behaviors, gaining meaningful insights to help refine the pipeline, when dealing with fashion related problems. The system was developed for a Kaggle competition based on a web-scale dataset supplied by the fashion company H&M (H&M 2022). Among 2954 teams, the presented approach achieved the *top 2% final ranking*.

Related Works

Fashion is a type of statement that reflects people's inner perceptions through their outer appearance. It has evolved into a non-verbal mode of communication and an important part of people's outer appearance over time (Barnes and Lea-Greenwood 2010). Consumer clothing preferences and product preference data are now available on the Internet in the form of texts or images, thanks to technological advancements. Because these customer article preference data contain information about people from all over the world, both online and offline fashion retailers are utilizing these platforms to reach billions of Internet users (Guan et al. 2016; Liu et al. 2017b).

Researchers have proposed fashion recommendation systems (FRS) which differ in terms of the filtering strategies used, information gathering and learning procedures, feature extraction methodologies and types of recommendations delivered to users. For instance, in (Xiang et al. 2019; Corbiere et al. 2017) a fashion retrieval system is developed using fashion product clusters and feature similarities, as well as correlation analysis based on individual historical data. Personal wardrobe recommendation algorithms look for similar fashion styles based on previous wardrobe usage. Fashion pairing suggestion systems, also known as fashion coordination systems, are built around the rules of matching various sorts of clothing pieces with stylistic knowledge (Guan et al. 2016; Agrawal et al. 2021).

As time is an important factor in the fashion domain, the recommendation system should reflect the decline in preference for fashion products over time. Previous temporal recommendation studies assume that the intensity of preference

*These authors contributed equally to this work.

Table 1: Dataset Statistics, we report the number of transations along with the number of unique customers and articles composing the historical data, Furthermore we report the number of features available for each entity.

	Count	No. of features
Transactions	31.8 M	3 context features
Articles	106 K	24 item features
Customers	1.3 M	6 customer features

decreases as time passes (Campos, Díez, and Cantador 2014; Ding and Li 2005; Larrain et al. 2015; Lathia et al. 2010). Koren (Koren 2009) presents a methodology for modeling time drifting user preferences in the context of recommender systems. Hong et al (Hong, Li, and Li 2012) provides an empirical study in which they consider both the long-term profile and the short-term preference simultaneously to match the user's interest. This research, in other words, presumes that recent preference-indicating activities, such as clicks or purchases for the same product, reflect stronger preferences than earlier ones. In (Ding, Li, and Orlowska 2006), the authors propose a recency-based collaborative filtering algorithm, using weights for items based on their expected accuracy on the future preferences. Likewise, Huang et al (Huang et al. 2014) improved a tag-based recommender relying on a two-step filtering process which modelled the recency effect of interactions as a linear decay function. However, the concept of decline in preference for fashion products which occurs over time following their release has yet to be studied in the fashion domain. One of the aim of this work is to study the decline in preference of articles over time using the real-world H&M dataset released by the company for a Kaggle competition. [1]

Preliminaries and Problem Formalisation

The goal of the H&M challenge is to develop an algorithm to provide online fashion recommendations to users. To this extent, multimodal data are provided, ranging from historical user transactions to product and user meta-data. We summarise the main statistics of the dataset in Table 1. The transactions are associated with the purchase history of customers spanning the period from the 20th September 2018 to 22nd September 2020. In this period 31.8M transactions associated with 106K different products and 1.3M unique users have been registered. The end goal of the challenge was to predict the items a user would buy in the week following the end of the training data.

Formally, we model the data as follows: let $\mathcal{U}$ be the set of users of size $|\mathcal{U}| = U$ and $\mathcal{I}$ be the set of items of size $|\mathcal{I}| = I$ and the data is split based on weeks $\mathcal{W} = \{1, 2, .., w-1, w\}$. We denote the purchase matrix up to week $t \in \mathcal{W}$, as $\mathrm{Y}^t = \{y^t_{u,i} | \mathrm{u} \in \mathcal{U}, \mathrm{i} \in \mathcal{I}\}$ where:

$$y^t_{u,i} = \begin{cases} 1, & \text{user } u \text{ purchased item } i \text{ before week } t \\ 0, & \text{otherwise} \end{cases}$$

In addition to the purchase history, the meta-data of users and items are provided. In order to leverage this additional information it is possible to construct three different kinds of features: user, item and context features, with the latter being dependent on both the user and the item features. It is crucial to highlight that any such feature vectors might be time dependent; for example, the popularity of an item will change if we analyse it at two separate time points. In order to account for the temporal variation we associate to each feature vector a superscript t, denoting the time instant to which they refer:

$$\text{User features up to week } t : \mathrm{X}^t_{\mathcal{U}} = \{\mathbf{x}^t_u | u \in \mathcal{U}\}$$
$$\text{Item features up to week } t : \mathrm{X}^t_{\mathcal{I}} = \{\mathbf{x}^t_i | i \in \mathcal{I}\}$$
$$\text{Context features up to week } t : \mathrm{X}^t_{\mathcal{C}} = \{\mathbf{x}^t_c | c \in \mathcal{U} \times \mathcal{I}\}$$

The user-item interactions in a particular week $t \in \mathcal{W}$ is defined as $\mathrm{H}^t = \{h^t_{u,i} | \mathrm{u} \in \mathcal{U}, \mathrm{i} \in \mathcal{I}\}$ where:

$$h^t_{u,i} = \begin{cases} 1, & \text{user } u \text{ purchased item } i \text{ in week } t \\ 0, & \text{otherwise} \end{cases}$$

The ultimate aim of the H&M Challenge is to estimate which products are likely to be purchased by each user in the week immediately after the end of the available training data.

Multi-stage Recommender Systems

Multi-stage recommender systems have become increasingly popular in recent years due to their ability to provide more accurate recommendations (Li et al. 2010; Feng, He, and Xu 2015; Zhang et al. 2017). In this approach, a recommender system is used initially to create a list of candidate items, which are then fed to a reranker model to obtain the final recommendations. The reranker model takes into account additional features beyond those used in the initial recommender system, which can include contextual information, user behavior, and item properties. The reason why additional features are used only in the reranker part and not in the initial recommender model is related to the scalability of the pipeline when working with real-world large-scale datasets. The initial recommender model is typically designed to operate on a large number of items and users with limited information about them, making it computationally efficient and scalable. On the other hand, the reranker model can leverage additional features that are more specific to the user and the items, but require more processing power and resources to train. By using a multi-stage recommender system, we can take advantage of the efficiency of the initial recommender system and the accuracy of the reranker model to provide more relevant recommendations to users.

The pipeline architecture of our multi-stage recommendation system, Figure 1, involves using the historical transaction data up to time t, Y_t, as input to the recommendation

[1] https://www.kaggle.com/competitions/h-and-m-personalized-fashion-recommendations

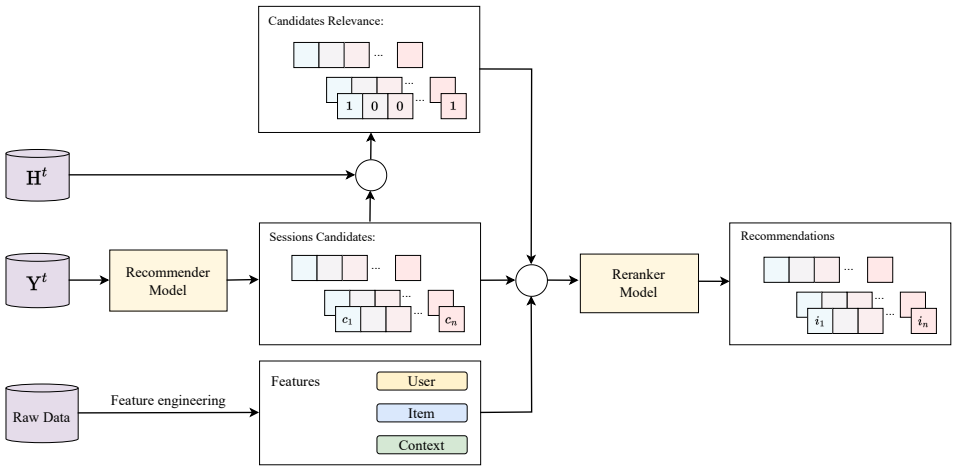


Figure 1: Pipeline Architecture: The historical transaction up to time t, Y^t serves as input to the recommendation model which generates a set of candidate items for each user. Exploiting the purchased made in the holdout week H^t we create the candidates relevance vectors. Finally employing the features vector we create a representation $\mathbf{z}_{u,i}^t$ for every user-item pair generated by the association of every user with the recommended candidates items. After the training phase the re-ranker model is used to retrieve the final recommendation list from the candidate items for every user.

model, which generates for each user $u \in \mathcal{U}$ a set of k items with scores $\mathcal{R}_u$. The purchased items made in the holdout week H_t are then used to create the candidate relevance vectors. Finally, the feature vectors are employed to create a representation for every user-item pair generated by associating every user with the recommended candidate items. After the training phase, the re-ranker model is used to retrieve the final recommendation list from the candidate items for each user. The following sections presents an overview of how Learning-to-rank algorithms can be utilized in the context of recommender systems.

Learning To Rank Overview

Learning-to-rank (LTR) algorithms are supervised methods that were originally developed for the information retrieval task of selecting a set of documents that match a given query. A broad category of such algorithms accomplishes this by learning a function which maps the joint space of queries and documents into the space of document permutations. Due to the factorial growth of permutations, the learning task becomes quickly intractable with the increasing size of the set of documents to rank. This problem is generally addressed by the *score-and-sort* approach which breaks the problem in two parts: the first step is to learn a scoring function that takes a query and a subset of k documents into a vector in $\mathbb{R}^k$, where the i^{th} element is a measure indicating the relevance of the i^{th} document with respect to the given query, and in a second step sorting documents by relevance scores (Liu and others 2009).

When working with recommender systems, we can recover the learning to rank (LTR) problem by interpreting the set of queries as the set of users $\mathcal{U}$ and the set of relevant documents for each user u as the set of k items $\mathcal{R}_u \subset \mathcal{I}$ retrieved by the recommender system model. For our problem, we denote the training data for the re-ranker at time instant t as:

$$\mathrm{D}^t = \left\{ \left(\mathrm{Z}_u^t, \boldsymbol{\psi} \right) | \forall u \in \mathcal{U}, \boldsymbol{\psi} \in \{0,1\}^k \right\} \tag{1}$$

Where Z_u^t is a matrix containing the feature vectors $\mathbf{z}_{u,i}^t$ representing the user-item pairs created by the association of a given user u with the set of k items retrieved from the recommendation model at the given time instant t and $\boldsymbol{\psi}$ is the associated binary relevance vector. We construct the feature vector representing a user item pair (u, i) as the concatenation of the feature vectors related to the user item and context as:

$$\mathbf{z}_{u,i}^t = [\mathbf{x}_u^t; \mathbf{x}_i^t; \mathbf{x}_c^t]$$

Formally the objective of the ranking model is to learn a scoring function $f(\cdot; \Theta) : \mathrm{Z}_u^t \rightarrow \mathbb{R}^k$ where Θ is a set of free parameters learned by minimising any given empirical listwise loss function devised for maximising the utility of the ranking of the of items recommended with respect to a given user. Once trained the model can be used for estimating the relevance scores for any feature vector $\mathbf{z}_{u,i}^t$ representing a user-item pair at a given time instant t. In the following we explain candidate relevance creation for the re-ranker.

Relevance Creation

The creation of relevance holds significant importance in the implementation of a multi-stage approach as it plays a key role in determining the quality of the training data employed by the reranker model. The reranker model, being responsible for generating the final recommendations, is heavily dependent on the quality of this training data. In the following we present the approach used to create the relevance vectors. The first step of the framework is to create session candidates from the recommender model. To do this, the dataset is split into a holdin set Y^t, i.e. the user item interactions up to week t, and a holdout set H^t, i.e. user item interactions in the particular week t. Given a set of user item interactions Y^t

Table 2: The table summarizes the notations introduced to formalise the presented problem reporting symbols with the associated descriptions.

Symbol	Notation	Description
$\mathcal{U}$	$\{u_0, \ldots, u_{U-1}\}$	Set of all users
$\mathcal{I}$	$\{i_0, \ldots, i_{I-1}\}$	Set of all Items
$\mathcal{W}$	$\{1, \ldots, w-1, w\}$	A set of values indicating the weeks in data
Y^t	$\{y^t_{ui} \vert u \in \mathcal{U}, i \in \mathcal{I}\}$	Purchase matrix up to week t
$\mathrm{X}^t_{\mathcal{U}}$	$\{\mathbf{x}^t_u \vert u \in \mathcal{U}\}$	User features up to week t
$\mathrm{X}^t_{\mathcal{I}}$	$\{\mathbf{x}^t_i \vert i \in \mathcal{I}\}$	Item features up to week t
$\mathrm{X}^t_{\mathcal{C}}$	$\{\mathbf{x}^t_c \vert c \in \mathcal{U} \times \mathcal{I}\}$	Context features up to week t
H^t	$\{h^t_{ui} \vert u \in \mathcal{U}, i \in \mathcal{I}\}$	The set of purchase in a week $t \in W$
$\mathcal{R}^t_u$	$\{i_0, \ldots, i_{k-1}\}$	Set of k recommendations for user u
$\mathbf{z}^t_{u,i}$	$[\mathbf{x}^t_u; \mathbf{x}^t_i; \mathbf{x}^t_c]$	Feature vector for user-item pair up to week t
Z^t_u	$\{\mathbf{z}^t_{u,i} \vert \forall i \in \mathcal{R}^t_u\}$	Feature matrix for user u up to week t
D^t	$\left\{\left(\mathrm{Z}^t_u, \boldsymbol{\psi}\right) \vert \forall u \in \mathcal{U}, \boldsymbol{\psi} \in \{0,1\}^k\right\}$	Training data for the reranker

the framework incorporates a recommender model to generate for each user $u \in \mathcal{U}$ a set of k items $\mathcal{R}_u$, known as *session candidates*. The framework's flexibility stems from the fact that any recommender model suitable for the dataset can be used for the candidates generation phase. After generating the session candidates the corresponding relevance are computed leveraging the holdout set. Formally the relevance for the session candidates is 1 if it has been purchased by the user in the holdout set H^t, i.e. $h^t_{u,i} = 1$, otherwise it is 0.

It is worth highlighting that the relevance creation phase may result in a loss of samples. Specifically, in cases where the recommender model fails to provide any relevant items for a given user u, the associated relevance vector becomes *null* and cannot be utilized during the training of the re-ranker model. This can have a significant impact on the overall efficacy of the re-ranker model, as the resulting loss of training data can limit its ability to accurately predict the relevance of candidates items.

The upcoming section of the paper delves into an exploration of key characteristics unique to the fashion domain. Specifically, we aim to identify insights that can be leveraged to customize the multi-stage framework and enhance the performance of the recommendation pipeline. By analyzing these characteristics and incorporating them into our approach, we can develop a more effective and accurate recommendation system that is uniquely tailored to the fashion domain.

Fashion Domain Characteristics

An effective fashion recommender system model should take proper account of the underlying characteristics associated with the domain. It is widely known that many fashion items exhibit trending behaviour, experiencing rapid growth and decline in the volume of transactions over a short period of time. In addition, there are strong *seasonality* effects. Due to the frequent changes in trends, even within the same season but across different years, the catalogue of products for sale is likely to change dramatically. In summary, the temporal context of the data is critical in the fashion domain and an ad-hoc solution should be devised to cope with it.

In the following, we provide an analysis which reveals two critical time-related insights that must be addressed when dealing with the task of generating recommendations for the fashion domain, as well as a method of using them in the proposed pipeline. Finally, we empirically demonstrate how accounting for temporal knowledge improves the framework's performance significantly.

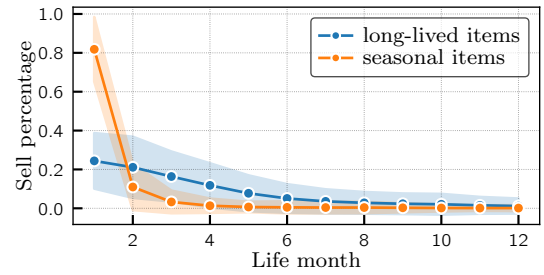
Articles Life

Fashion trends change frequently, resulting in huge fluctuations in the number of sales of certain items (Corbiere et al. 2017; Xiang et al. 2019; Agrawal et al. 2021). To better understand this phenomenon, we first construct for every item i its associated *sales profile* vector $\mathbf{s}_i$. This vector contains the percentage of sales that the item received in each month since it was first introduced to the market, e.g. $\mathbf{s}_i(1) = 0.5$ would indicate that item i generated 50% of its total sales during the first month from its release. More formally:

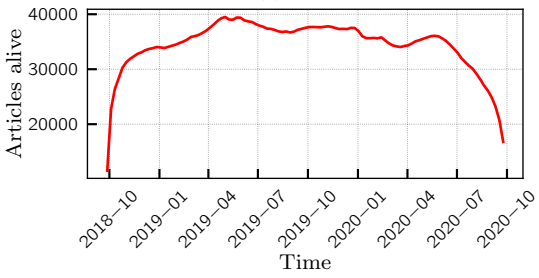
$$\mathbf{s}_i(j) = \frac{sell_i(m_j)}{\sum_{k=0}^{S} sell_i(m_k)}$$

where $sell_i(m_j)$ indicates the number of sells item i received on the j^{th} month from its release and S is a fixed time horizon expressed in months. We can now seek to cluster items with respect to their sales profile to the extent of grouping those items which have similar sales patterns through time.

Applying K-Means clustering on the normalised sale vectors using a time horizon $S = 12$ months and a silhouette analysis, leads to the identification of two distinct clusters. In Figure 2a we report the centroid sell percentage vector for each of the two clusters found. We can clearly see that there are certain items for which after only 4 months the sales drop close to 0. We refer to these items as the *seasonal items*, i.e. the ones which are sold only during a very short time period, which intuitively can be a season. The second cluster has a



(a)



(b)

Figure 2: Figure (a) reports the centroids relative to the clustering of Articles according to the normalised Sales Profiles. In figure (b) is reported the number of alive article week by week from the start to the end of the training data available.

different behaviour for which the sales decrease much more slowly through the months. We refer to this second cluster as *long-lived items*. From this we can say that at every point in time the items that are *alive* are a subset of the full catalogue of items. To verify that, in Figure 2b we report the number of alive items week by week from the start of the available training data. An item is defined alive at a given point in time t if exist a user who has purchased it at a time instant $t' > t$. One can see that on average the number of alive items is around 35k. From this analysis we can infer two different insights, relevant to the fashion domain, that can help in creating better recommendations:

1. **Whitelisting items at prediction time**: Since the alive items are a strict subset of all the items available on the catalogue, we hypothesise that it is not necessary to allow the recommender system to predict *any* item, instead we can predict from the subset of alive items the one with the highest score.
2. **Time-weight interactions:** The second hypothesis that can be made is that due to the fast changing domain we are in, the taste of users also changes very rapidly over time (Larrain et al. 2015; Hong, Li, and Li 2012). Considering all the historical interactions in the same way can lead to the introduction of noise in our data. A possible solution is to give more importance to the most recent interactions, in order to capture the current users fashion preferences.

Experiments

In this section, we conduct a series of experiments in order to empirically validate the previously presented insights.

Table 3: Results obtained using different time weight functions to weight the interaction data.

	$f(\Delta_t)$	MAP @12	MAP @100	MAP @200	Recall @12	Recall @100	Recall @200
Cosine	Constant	0.0115	0.0132	0.0137	0.0286	0.0881	0.1145
	Linear	0.0161	0.0178	0.0186	0.0377	0.0976	0.1268
	Exponential	0.0262	0.0278	0.0281	0.0498	0.0915	0.1252
	Power-Law	**0.0268**	**0.0285**	**0.0289**	**0.0518**	**0.1087**	**0.1385**
RP^3_β	Constant	0.0120	0.0140	0.0143	0.0297	0.0890	0.1179
	Linear	0.0167	0.0189	0.0191	0.0394	0.0988	0.1280
	Exponential	0.0270	0.0287	0.0289	0.0511	0.0921	0.1248
	Power-Law	**0.0272**	**0.0292**	**0.0295**	**0.0534**	**0.1111**	**0.1417**

The evaluation metric used to asses the model performance during the competition on the test week was MAP@12. Although in our experiments, we report the Recall metric along with the MAP for several cutoffs in order to further evaluate the quality of the final recommendations. During the course of the challenge, the recommender model that exhibited superior performance and best results was the standard item-based collaborative filtering (CF) approach based on k-nearest neighbors (Ferrari Dacrema, Cremonesi, and Jannach 2019). This inductive model enabled us to generate recommendations for users who were not included in the training data (D'Amico et al. 2023). To asses the generality of the observations, we perform experiments using two different item-item similarity measures: cosine similarity and $RP^3\beta$ (Christoffel et al. 2015).

LightGBM (Ke et al. 2017) was chosen as the re-ranker model in view of its ability to natively utilize categorical features. In addition to the features provided in the initial dataset, we crafted additional features, that ultimately brought the total number to 65. Since the features are specific to the dataset used in this study, providing a detailed description is beyond the scope of this paper. The aim of the presented work is to present a general framework that can be easily customized to suit different datasets. As the goal of the challenge was to predict the items that would be purchased in the week following the available training data, we performed a local split to use as validation data. The last week of training data was held out to determine the best hyperparameters for the re-ranker. Once the hyperparameters were selected, the model was trained using the entire available dataset.

The code used to produce the experiments presented in the paper is available on Github[2].

Ablation Study

Time-weighted interactions (TW) The interaction matrix of ItemKNN models can be weighted using a time weight function that accounts for how recently an item has been interacted before the computation of the similarity matrix. In Table 3 we report the effect of different weight mechanisms of the user-item interaction matrix. From the results obtained is evident how giving more importance to recent

[2]https://github.com/damicoedoardo/HnMChallenge

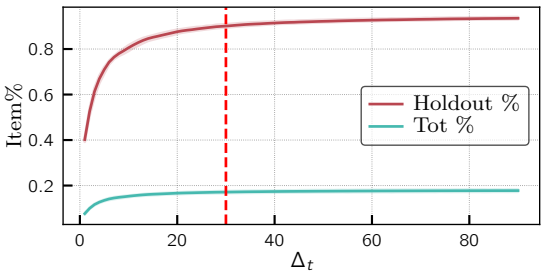


Figure 3: Cumulative percentage of items transacted with respect to the holdout and total item sets over time.

user interactions can benefit the final recommendation performance. Among the difference function tested the best performance are obtained using a power-law function which gives more importance to recent interactions but at the same time is capable of giving almost the same importance to all the past interaction with respect to an exponential.

Whitelisting items (WI) Traditionally, recommender models are not limited in their ability to recommend items from the entire catalogue. However, our analysis, as depicted in Figure 2b, reveals that only a specific subset of items in the catalogue can be deemed alive. By filtering recommendations based on this subset, we may be able to optimize the model's performance. However, selecting the correct subset of items for whitelisting is a nontrivial task.

To address this challenge, we propose an iterative study. Starting from the beginning of the holdout week (test set), we look back in time, day by day, to identify the alive subset by accumulating the items with at least one registered transaction. Using this approach, we can calculate the coverage of the set of whitelisted items over the set of items transacted in the holdout week. This helps us determine how far back in time we need to go to generate a whitelisted set that comprises a certain percentage of the items transacted in the test week. To establish the validity and reliability of our findings, we carried out the investigation over the most recent five weeks of available data. The results were consistent and robust, as illustrated in Figure 3.

Our analysis suggests that, on average, going back 30 days from the start of a given holdout week and selecting the whitelisted items as the ones transacted within this timeframe leads to 90% coverage over the items transacted in the holdout week. It's important to note that only 20% of the total items in the catalogue are responsible for 90% of the transactions in the holdout week. This finding implies that the number of items that can be recommended can be significantly reduced, making it easier for the recommender system to suggest the most relevant items.

In Table 4 we report ablation studies over the recommender model to asses the benefit of the proposed procedure. The results demonstrate that whitelisting alone can improve all metrics for all evaluated cutoffs. Moreover, combining time weighting and whitelisting resulted in better recommendation performance compared to using only the time-weighting mechanism (see Table 3). Finally, Table 5

Table 4: Ablation study on the recommender model.

	WI	TW	MAP @12	MAP @100	MAP @200	Recall @12	Recall @100	Recall @200
Cosine	-	-	0.0115	0.0132	0.0137	0.0286	0.0881	0.1145
	✓	-	0.0136	0.0160	0.0163	0.0346	0.0992	0.1262
	✓	✓	**0.0271**	**0.0290**	**0.0295**	**0.0536**	**0.1121**	**0.1436**
RP^3_β	-	-	0.0120	0.0140	0.0143	0.0297	0.0890	0.1179
	✓	-	0.0141	0.0164	0.0167	0.0354	0.0990	0.1292
	✓	✓	**0.0275**	**0.0298**	**0.0301**	**0.0548**	**0.1173**	**0.1497**

Table 5: Ablation study on the multi-stage framework. WI, TW and R indicate the use of whitelisting items, time-weighted interactions and reranker respectively.

Pipeline	MAP @12	MAP @100	MAP @200	Recall @12	Recall @100	Recall @200
Cosine+R	0.0212	0.0237	0.0255	0.0485	0.0578	0.1042
Cosine+WI+TW+R	**0.0328**	**0.0351**	**0.0357**	**0.0657**	**0.0853**	**0.1376**
Improvement %	54.71	48.10	40.00	35.46	47.57	32.05
RP^3_β+R	0.0231	0.0252	0.0277	0.0512	0.0608	0.1196
RP^3_β+WI+TW+R	**0.0343**	**0.0369**	**0.0373**	**0.0733**	**0.0889**	**0.1402**
Improvement %	48.48	46.42	34.65	43.16	46.21	17.22

presents the results of an ablation study on the full multi-stage pipeline, comparing the benefits of the two domain-driven modifications proposed. The findings indicate that applying the modifications together leads to a significant increase in the overall performance of the pipeline, resulting in a minimum gain of 35.46% and a maximum gain of 54.71%.

Conclusion

In this paper, we highlight two important implications induced by the time in the fashion domain setting: The length of time a product remains in the product catalogue fluctuates as a result of people's constantly changing fashion tastes. At the same time, fashion trends change quickly, reducing the amount of information carried by the data over time making it important to take into account the user's most recent item transactions in order to assess and forecast future trends for customers. In this work we report the first results on the H&M dataset, a real-world large-scale fashion dataset. We present the application of a multi-stage recommender pipeline specifically tailored to deal with the fast changing fashion domain, and empirically validate through an extensive ablation study the modification applied demonstrating how they entail a significant increase in the final recommendation performance.

Acknowledgement

This work was supported by the Science Foundation Ireland through the Insight Centre for Data Analytics under grant number SFI/12/RC/2289_P2.

References

Agrawal, P.; Dayama, P. S.; Saha, A.; and Tamilselvam, S. G. 2021. Coordinated event based wardrobe recommendation. US Patent 11,049,164.

Barnes, L., and Lea-Greenwood, G. 2010. Fast fashion in the retail store environment. *International Journal of Retail & Distribution Management*.

Campos, P. G.; Díez, F.; and Cantador, I. 2014. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction* 24(1):67–119.

Christoffel, F.; Paudel, B.; Newell, C.; and Bernstein, A. 2015. Blockbusters and wallflowers: Accurate, diverse, and scalable recommendations with random walks. In *Proceedings of the 9th ACM Conference on Recommender Systems*, 163–170.

Corbiere, C.; Ben-Younes, H.; Ramé, A.; and Ollion, C. 2017. Leveraging weakly annotated data for fashion image retrieval and label prediction. In *Proceedings of the IEEE international conference on computer vision workshops*, 2268–2274.

Covington, P.; Adams, J.; and Sargin, E. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, 191–198.

Ding, Y., and Li, X. 2005. Time weight collaborative filtering. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, 485–492.

Ding, Y.; Li, X.; and Orlowska, M. E. 2006. Recency-based collaborative filtering. In *Proceedings of the 17th Australasian Database Conference-Volume 49*, 99–107.

D'Amico, E.; Muhammad, K.; Tragos, E.; Smyth, B.; Hurley, N.; and Lawlor, A. 2023. Item graph convolution collaborative filtering for inductive recommendations. In *Advances in Information Retrieval: 45th European Conference on Information Retrieval, ECIR 2023, Dublin, Ireland, April 2–6, 2023, Proceedings, Part I*, 249–263. Berlin, Heidelberg: Springer-Verlag.

Feng, X.; He, J.; and Xu, L. 2015. A hybrid recommender system with enhanced accuracy using a context-aware approach. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*, 2859–2864. IEEE.

Ferrari Dacrema, M.; Cremonesi, P.; and Jannach, D. 2019. Are we really making much progress? a worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM conference on recommender systems*, 101–109.

Guan, C.; Qin, S.; Ling, W.; and Ding, G. 2016. Apparel recommendation system evolution: an empirical review. *International Journal of Clothing Science and Technology*.

Higley, K.; Oldridge, E.; Ak, R.; Rabhi, S.; and de Souza Pereira Moreira, G. 2022. Building and deploying a multi-stage recommender system with merlin. In *Proceedings of the 16th ACM Conference on Recommender Systems*, 632–635.

H&M. 2022. H&m personalized fashion recommendations.

Hong, W.; Li, L.; and Li, T. 2012. Product recommendation with temporal dynamics. *Expert systems with applications* 39(16):12398–12406.

Huang, C.-L.; Yeh, P.-H.; Lin, C.-W.; and Wu, D.-C. 2014. Utilizing user tag-based interests in recommender systems for social resource sharing websites. *Knowledge-Based Systems* 56:86–96.

Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; and Ye, Q. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems* 30:3149–3157.

Koren, Y. 2009. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 447–456.

Larrain, S.; Trattner, C.; Parra, D.; Graells-Garrido, E.; and Nørvåg, K. 2015. Good times bad times: A study on recency effects in collaborative filtering for social tagging. In *Proceedings of the 9th ACM Conference on Recommender Systems*, 269–272.

Lathia, N.; Hailes, S.; Capra, L.; and Amatriain, X. 2010. Temporal diversity in recommender systems. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, 210–217.

Li, C.; Li, Y.; Geng, X.; Liu, W.; and Chen, E. 2010. Towards a multi-stage approach to personalized recommendation. In *2010 IEEE International Conference on Data Mining*, 689–698. IEEE.

Liu, T.-Y., et al. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval* 3(3):225–331.

Liu, D. C.; Rogers, S.; Shiau, R.; Kislyuk, D.; Ma, K. C.; Zhong, Z.; Liu, J.; and Jing, Y. 2017a. Related pins at pinterest: The evolution of a real-world recommender system. In *Proceedings of the 26th international conference on world wide web companion*, 583–592.

Liu, Y.; Gao, Y.; Feng, S.; and Li, Z. 2017b. Weather-to-garment: Weather-oriented clothing recommendation. In *2017 IEEE International Conference on Multimedia and Expo (ICME)*, 181–186. IEEE.

Medvedev, I.; Wu, H.; and Gordon, T. 2019. Powered by ai: Instagram's explore recommender system. *Retrieved June* 17:2022.

Xiang, J.; Zhang, N.; Pan, R.; and Gao, W. 2019. Fabric image retrieval system using hierarchical search based on deep convolutional neural network. *Ieee Access* 7:35405–35417.

Zhang, Z.; Du, Y.; Yang, Y.; Li, J.; and Li, Z. 2017. Deepgbm: A deep learning framework distilled by gbdt for online prediction tasks. In *Proceedings of the 26th International Conference on World Wide Web*, 147–156. International World Wide Web Conferences Steering Committee.