Augmenting content-based rating prediction with link stream features

Tiphaine Viard[1,2], Raphaël Fournier-S'niehotta[1]

[1] *CEDRIC, CNAM, Paris, France* [2] *RIKEN AIP, Tokyo, Japan*

Abstract

While graph-based collaborative filtering recommender systems have been introduced several years ago, there are still several shortcomings to deal with, the temporal information being one of the most important. The new link stream paradigm is aiming at extending graphs for correctly modelling the graph dynamics, without losing crucial information.

We investigate the impact of such link stream features for recommender systems. We design link stream features, that capture the intrinsic structure and dynamics of the data. We show that such features encode a fine-grained and subtle description of the underlying system.

We focused on a traditional recommender system context, the rating prediction on the MovieLens20M movie dataset and the Goodreads book dataset. We input link stream features along with some content-based ones into a gradient boosting machine (XGBoost) and show that it outperforms significantly a sole content-based solution.

These encouraging results call for further exploration of this original modelling and its integration to complete state-of-the-art recommender systems algorithms. Link streams and graphs, as natural visualizations of recommender systems, may offer more interpretability in a time when algorithm transparency is an increasingly important topic of discussion. We also hope that these results will sparkle interesting discussions in the community about the connections between link streams and traditional methods (matrix factorization, deep learning).

Keywords:
link streams, recommender systems, interpretability, temporal networks, graphs

1. Introduction

Collaborative filtering algorithms are at the core of recommender systems research. They rely on finding *similar users* to the user for whom the recommendation is intended, collecting the previous opinions of these users to compute scores for items the given user has not yet rated, and present these items with the best scores. The most widespread modelisation of recommender data is in the form of a matrix where the rows represent the users, the columns represent the items, and one element of the matrix indicates the rating the user has given to the item.

This ensemble of users and items may also be seen as a bipartite graph, where nodes represent users and items, and an edge between a user-node and an item-node represents a rating between the user and the item. Finding similar users in this context is naturally linked to the graph-theoretic notion of *neighborhood*, *i.e.* the user-nodes which share a subset of neighbour item-nodes with a given node.

While recommender systems initially discarded any notion of time, and two ratings given several years apart where considered equal, a body of research has emerged to take this into account. A common solution is to rely on sequences of user-item matrices, with a time step (Δ): one builds a sequence $\{M^k\}_k$ such that for all k, M^k is a user-item matrix, and $M^k_{i,j} \neq 0$ indicates that user i has interacted with item j at least once in $[k, k + \Delta[$.

Studying dynamic graphs traditionally relies on sequences of *snapshot* graphs, similar to the sequences of user-item matrices M^k, with the same shortcomings (loss of information). To overcome these issues, the complex network community has recently come up with *link streams*, also called temporal networks or time-varying graphs depending on the context. The link stream paradigm enables to study jointly the topological structure and the dynamics of interactions. A system where users interact with items over time, like in a recommender system scenario, may then be efficiently modelled as a bipartite link stream.

In this paper, we show that modelling a recommender setting as a link stream produces descriptors that are relevant to a recommendation task, including in a large-scale context.

To test our approach, we use the interactions from the well-known Movie-

lens 20M dataset[1], along with a more recent one, Goodreads[2] [26]. We devise relevant content-based features for both datasets and generate link stream-based features, adapted to the datasets' parameters. We then feed a state-of-the-art machine learning algorithm for recommendation (*XgBoost*) to learn the recommendation task.

We evaluate the relevance of our link-stream features by comparing their performance to a content-based-only baseline, and to state-of-the-art results on these datasets.

We obtain promising results in terms of the RMSE metric, even though our primary objective is not to be directly competitive with state-of-the-art results, but to demonstrate the point of incorporating link stream features into more complex models. Indeed, recommender systems have benefited from much attention in the last 20 years, leading to extremely optimized models, based for example on matrix factorization or deep learning approaches.

These approaches come with the drawback that they are increasingly complex and obscure, leading to effective recommendations that operators have difficulty to explain to their users. On the contrary, our analysis of graph and link stream-based features show that they provide an accurate and fine-grained description of the datasets at hand. This makes an interesting point towards more explainability and justifiability in recommender systems.

The remainder of this paper is organized as follows: in Section 2, we present the recommending context we study ; the bipartite link stream model is formalised in Section 3. We detail the features we devised in Section 4. We present our experimental setup and results in Section 5. We discuss related works in Section 6, before concluding the paper with some stimulating perspectives in Section 7.

2. Problem setting

Given a user u and a movie i, we focus on the task of predicting the rating assigned by u to i on the 0.5 scale from 0 to 5, *i.e.* a *regression* task. Let U be a set of users, and I a set of movies, with $|U| = n$ the number of users and $|I| = m$ the number of movies, and F a set of features such that $|F| = f$.

[1]Grouplens: `https://grouplens.org/datasets/movielens/20m/`.

[2]UCSD Book Graph dataset : `https://sites.google.com/eng.ucsd.edu/ucsdbookgraph/home`.

Our model inputs a $(n \cdot m) \times f$ matrix, and outputs a $n \cdot m$ matrix P, where each element r of P is the predicted rating, between 0 and 5.

The feature set F is composed of a mixture of numerical and categorical variables describing the dataset, and, for each user $u \in U$ and movie $i \in I$, is comprised of three sets: $F(u)$, the set of user-based features, $F(m)$, the set of movie-based features, and $F(u, i)$, the set of interaction-based features. The contents of each of these sets is discussed below in Section 4.

3. The bipartite link stream model

Modelling a recommender system as a *bipartite graph* $G = (U, I, E_G)$ is rather natural: the sets of users U and items I represent the two sets of nodes U and I. The set of edges $E \subseteq U \otimes I$ is composed of interactions between a user and an item, where $\otimes$ is a shorthand notation for a pair of distinct elements [16]. All those sets may be completed by weights, for example a rating, or labels, for example a list of timestamps. However, these solutions are limited by essence; not resorting to weighted or labeled graphs causes important losses of information, while weighted and labeled graphs are complex objects that currently lack the vast array of algorithms required for social network analysis.

A *bipartite link stream* $L = (T, U, I, E_L)$ is defined by a time span T, a set of users U, a set of items I, and a set of links $E_L \subseteq T \times U \otimes I$ [16]. Nodes u and i are linked at time t if $(t, ui) \in E$. We say that (b, e, ui) is a link of L if $[b, e]$ is a maximal interval of T such that u and i are linked at all t in $[b, e]$. See Figure 1 for an illustration.

The usual properties of graphs (neighbourhoods, paths, clustering, etc.) have been generalized to link streams [16], enabling the study of interaction streams with a single modelling structure, and without resorting to snapshots. As with *bipartite graphs*, it is easy to see a recommender system as a *bipartite link stream* $L = (T, U, I, E_L)$.

4. Data and feature engineering

4.1. Datasets

For our evaluation, we focus on two datasets coming from different contexts: a movie rating context (MovieLens) and a book rating context (Goodreads).

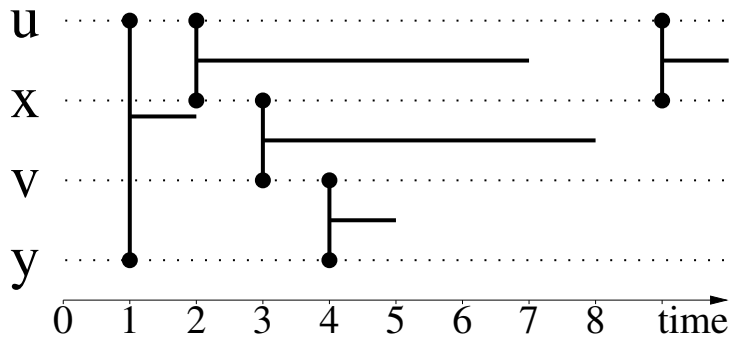


Figure 1: A bipartite link stream $L = (T, U, I, E_L)$ with $T = [0, 10]$, $U = \{u, v\}$, $I = \{x, y\}$, and $E_L = ([2, 7] \cup [9, 10]) \times \{ux\} \cup [1, 2] \times \{uy\} \cup ([3, 8] \times \{vx\} \cup [4, 5] \times \{vy\}$. In other words, the links of L are $(2, 7, ux)$, $(9, 10, ux)$, $(1, 2, uy)$, $(3, 8, vx)$, and $(4, 5, vy)$. We display nodes vertically and time horizontally, each link being represented by a vertical line at its beginning that indicate its extremities, and an horizontal line that represents its duration.

While the first one has been studied extensively in the recommendation literature, the second is very recent. However, they present similar characteristics (time-based ratings, easily usable content-based information, size magnitude) which makes them comparable. Several other datasets generally used for such experiments do not present those suitable characteristics (Epinions, Ciao) or were no longer publicly available (Flixster, Douban). The MovieLens 20M dataset comprises of 20,000,263 interactions, involving 138,493 users and 27,278 movies over the course of 20 years (from January 9^{th}, 1995 to March 31^{st}, 2015). The datasets contains two types of interactions: a user u gives a rating $r \in [0, 5]$ to item (movie) i at time t, or a user u assigns textual tags to item (movie) i at time t. Movies have limited information associated with them, only release year and genres. No demographic information about the users is present, contrary to other MovieLens datasets.

We do not use IMDB identifiers to extract more movie information, which would require NLP[3] to come up with good features.

The Goodreads dataset is a recent extract of the book-rating website *GoodReads* [26]. For scalability reasons, we focus on a specific genre, the Goodreads Children dataset. It contains 10,080,558 interactions between

[3]NLP: Natural Language Processing (tokenisation, lemmatisation)

462,164 users and 122,741 books, over the course of 11 years (from August 30^{th}, 2006 to November 6^{st}, 2017). Each interaction is of the same form: a user u gives a rating $r \in [0, 5]$ to item (book) i at time t. We remove interactions in the dataset that do not have a rating information, or have a rating but state the book has not been marked as read by the user. These interactions account for about 35% of interactions. We end up with 6,393,905 ratings. Each book also has information attached to it, for example the publication year, or some notes about awards and distinctions the book gained.

4.2. Content-based features

From the available information in the MovieLens dataset, we obtained 39 content-based features. Among them, 19 were the result of a n-out-of-one strategy for the genre of the movies (18 possible genres and a "(no genres listed)" column). 10 features code the decade in which the movie was released. To reduce the noise in this feature, we decided to group together all movies that came out before 1940, since they represent a minor fraction of the dataset. Concerning the user-generated tags, while not resorting to NLP methods as stated above, we include two features: the total number of tags (normalized), and, for each pair *(user, movie)*, the number of tags they have in common. The remaining 6 features are the rating mean, median and standard deviation, and the minimum, maximum and number of ratings (normalized). Those last 6 features are computed both for movies and users.

From the Goodreads dataset, we extract 6 features corresponding to the rating mean, median and standard deviation, and the minimum, maximum and number of ratings (normalized), 4 features encode the decade in which the book was released. Just like for the MovieLens dataset, we group books together if they have been published in 1970 or before. There is no genre information, since the data are already restricted to the "children" genre, and so we end up with 10 features.

4.3. Graph and link stream features

Equipped with the link stream model, we devised 21 features, some of them being close to what can be found in other graphs models, others being completely original. We detail them in the following, relying on two structures: a bipartite link stream $L = (T, U, I, E_L)$ and the bipartite graph it induces, $G = (U, I, E_G)$.

Neighbourhood-based features. These features explore the relations between the users and the movies over time. We say that u is a neighbour of

i if there exists at least one interaction between u and i, in the dataset. The degree of a node u in the graph G is simply the number of neighbours of u, *i.e.* $d_G(u) = |\{i : ui \in E_G\}|$. This, however, does not take into account the dynamics of a user's neighbourhood; we focus on the neighbourhood of $u \in U \cup I$ at each time $t \in T$: $d_t(u) = |\{i : \exists (t, ui) \in E\}|$. We describe the evolution of $d_t(u)$ with its mean value, $d(u) = \frac{1}{|T|} \int_t d_t(u)dt = \frac{1}{|T|} \sum_{m \in I} |\{(t, ui) \in E_L\}|$, and its maximum value, $\max(d_t(u))$. Notice that $\max(d_t(u))$ is not necessarily equal to $d_G(u)$.

We also computed the minimum stream degree $\min(d_t(u))$ and the standard deviation of $d_t(u)$, however these two features showed little relevance; we discard them.

We also compute the *assortativity* [19] of each link $ui \in E_G$, $a(ui) = \frac{\min(d(u),d(i))}{\max(d(u),d(i))}$, which is the ratio between the degrees of the nodes. In this context, low values of assortativity typically correspond to famous blockbusters that all users have likely seen.

Inter contact time features. To take into account the dynamics of the link stream, we use the sequence of durations between two links involving u, defined as follows. For each user (resp. movie) $u \in U$ (resp. I), let $t(u) = (t : (t, ui) \in E_L \cap T \times I \otimes \{u\})$ be the ordered sequence of times at which there is a link involving u. Then, the inter-event times sequence is the sequence of the differences between two consecutive elements of $t(u)$, *i.e.* $\tau(u) = (t_{i+1} - t_i)_{i=0}^{|t(u)|-1}$. We describe this sequence, for each user (resp. movie), by its maximum $max(\tau(u))$, minimum $min(\tau(u))$, mean $\mu(\tau(u))$ and standard deviation $\sigma(\tau(u))$.

Clique-based features. As an exploratory approach to find clusters of users and items in the bipartite link stream, we rely on cliques in the link stream model, for lack of an established clustering algorithm. However, enumerating all the maximal cliques is computationally intractable on large data. Plus, some cliques (like stars) do not capture relevant information for recommendation. We then use the methodology described in [25] to sample maximal balanced bipartite cliques, *i.e.* cliques involving approximately the same number of users and items. This kind of object is interesting from a recommender systems point of view: it corresponds to dense subgroups of users all rating a substantial number of items.

Formally, $(U', I', [b, e])$ is a clique in a link stream if all nodes of $U' \subseteq U$ interact with all nodes of $I' \subseteq I$ over $[b, e] \subseteq T$, *i.e.* $\forall t \in [b, e], \forall ui \in U' \otimes I', \exists\ (t, ui) \in E_L$. A clique is maximal if it is included in no other.

From our set of sampled balanced maximal cliques, we computed the following features for each user (resp. movie) u:

- The balancedness of the cliques involving u:

$$\frac{1}{|C_u|} \sum_{(U',I',[b,e])\in C_u} \frac{\min(|U|,|I|)}{\max(|U|,|I|)}$$

- The normalized average duration of the cliques involving u:

$$\frac{1}{C_u} \sum_{(U',I',[b,e])\in C_u} \frac{|[b,e]|}{|T|}$$

- The fraction of cliques containing u:

$$\frac{|\{(U',I',[b,e]) : u \in U \cup I\}|}{|\{(U',I',[b,e]\}|}$$

where $C_u = |\{(U',I',[b,e]) : u \in U \cup I\}|$ is the number of cliques involving node u, a normalizing factor. All these features tend to describe the sampled maximal balanced cliques containing u. Intuitively, nodes belonging to a high fraction of all cliques are typically high-raters.

5. Evaluation setting and recommendation results

5.1. Evaluation Metrics

The main metrics to evaluate the prediction performance of a recommender system are MAE and RMSE [11]. They have been a widespread proxy for effective recommendations in real-world contexts. The **Mean Absolute Error (MAE)** is the average error between elements of the ground truth y and the predicted elements $\hat{y}$:

$$MAE(\hat{y}, y) = \frac{1}{n} \sum_{i=1}^{n} |\hat{y}_i - y_i|$$

For a set of n predictions $\hat{y}$ for which the ground truth y is known, the **Root Mean Squared Error** is defined as:

$$RMSE(\hat{y}, y) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2}$$

The MAE is a simple metric that is readily interpretable, but is less sensitive to outliers than the RMSE, for example. The RMSE is less easy to interpret, however it offers a good tool of comparison to the state-of-the-art, and is more sensitive to outliers, which is interesting in a prediction context.

We also report our results for the *NDCG@k* metric, the Normalized Discounted Cumulative Gain for a ranking of k elements. It is formulated as:

$$NDCG(k) = N \sum_{i=1}^{k} \frac{2^{R(i)} - 1}{\log_2(1+i)},$$

where $R(i)$ is the graded relevance of item i, and N is a normalizing factor, such that a perfect ranking has a *NDCG* value of 1.

5.2. Recommendation results

We perform the recommendation task by feeding our features into *XgBoost* [8], a prominent tree-based framework for learning the probabilities of matching users and items in a recommendation context, using the method from [20]. We tuned *XgBoost* using Bayesian optimization [22] on the hyper parameter space, and obtain optimal results with deep trees and a small learning rate. Optimality is reached with the parameters presented in Table 2. The parameters' meaning in Table 2 are briefly explained in Table 1.

Hyperparameter	**Range**	**Meaning**
α	$[0, 1]$	L_1 regularization on feature weights
η	$[0, 1]$	Learning rate of the algorithm
`min_child_weight`	$[0, \infty[$	Minimum number of (weighted) instances to create new leaf in the boosting tree
`colsample_bytree`	$[0, 1]$	Subsample of features to use in each tree
γ	$[0, \infty[$	Minimum loss reduction required to make new leaf in the boosting tree
`max_depth`	$[0, \infty[$	Maximum depth of the trees
`subsample`	$]0, 1]$	Ratio of the training dataset to subsample before building each tree

Table 1: Brief explanation of *XgBoost*'s hyperparameters. `max_depth` directly controls the complexity of the model, with deep trees more prone to overfitting. The other parameters control the model's tendency to be conservative, and are leveraged to reduce overfitting.

	Movielens 20M	Goodreads
α	0.3649	9.3515
η	0.1	0.1
`min_child_weight`	18.6967	11.5504
`colsample_bytree`	0.9112	0.9541
γ	0.9930	2.7449
`max_depth`	10	10
`subsample`	0.9810	0.9959

Table 2: *XgBoost* parameters which give the best results for our approach, for both datasets.

We performed a 5-fold cross validation on the dataset, and ran the experiment for 10 hours on a machine with 32 8-cores CPUs and 64 GB of RAM. We present the results according to our evaluation metrics for both datasets in Table 3, comparing the performance of our algorithm when using link streams features along with content-based ones or when using only content-based features. We see that adding link stream features lead to significantly better learning performance than using only content-based features (bold values).

Our results on the Movielens dataset are on-par with some of the recent literature [28]. Their authors use a completely different setting (a non-convex matrix factorization approach) to reach a RMSE of 0.785 on the same dataset. To the best of our knowledge, the minimum RMSE (best recommendation score) obtained on the MovieLens 20M dataset was 0.7652 with a deep learning approach [23]. Let us bring attention on the fact that our model is simpler and do not rely on years of model tuning. While we are not competitive with the best results in the literature, we detail some perspectives in Section 7 to close this gap.

The Goodreads dataset having been released in 2018, there is to date no similar baseline to compare our results to.

5.3. Discussion of feature importance

In addition to the classic performance metrics presented above, we evaluate the descriptiveness of the link stream features we devise. Figure 2 shows the relative importance of features as selected by *XgBoost*, with the link stream features indicated in red.

Dataset	Metric	With stream features		
		# iter	train	test
MovieLens 20M	MAE	2134	**0.60234 (± 0.00022)**	**0.63427 (± 0.00012)**
	RMSE		**0.76349 (± 0.00043)**	**0.80954 (± 0.005)**
	NDCG@10		**0.99212 (± 0.015)**	**0.97209 (±0.019)**
Goodreads Children	MAE	998	**0.5299 (± 0.0002)**	**0.5899 (± 0.0003)**
	RMSE		**0.6818 (± 0.0002)**	**0.7561 (±0.0004)**
	NDCG@10		1.0 (± 0.0)	**0.9901 (± 0.019)**

Dataset	Metric	Without stream features		
		# iter	train	test
MovieLens 20M	MAE	2134	0.63421 (± 0.00013)	0.644277 (± 8.6e-05)
	RMSE		0.82682 (± 0.00017)	0.83961 (± 0.00025)
	NDCG@10		0.98212 (± 0.022)	0.94863 (±0.048)
Goodreads Children	MAE	635	0.63343 (± 0.00042)	0.64986 (± 0.0003)
	RMSE		0.7986 (± 0.0005)	0.82652 (± 0.0003)
	NDCG@10		1.0 (± 0.0)	0.9772 (± 0.0454)

Table 3: MAE, $NDCG@10$ and $RMSE$ values for the train and test datasets of both Movielens and Goodreads, with and without link stream or graph features.

We can see that the introduced link stream and graph features are commonly used as split points by the boosting algorithm, which supports the claim that such features are very descriptive of the structure and dynamics of the dataset. For MovieLens, out of 20 graph and link stream features, 14 of them have a non-zero feature importance. More importantly, the top-20 most important splitting features include 13 graph and link stream features.

A similar trend is seen on the Goodreads dataset. See Figure 2 for the details, Figure 2a for Movielens and Figure 2b for Goodreads. Moreover, while it is a bit too early in this work to extrapolate this result, one might notice that the three most important features in this context relate to the *intercontact times*, just before more classic rating-based features. This makes an interesting point to back the claim that link streams and graphs are valuable conceptual tools for recommender systems theory, their features being considered as very relevant descriptors of the underlying recommender system. Indeed, the feature importance score grows with the number of times a feature is selected for splitting. For a feature to be a splitting criterion means it is able to discriminate variations between the data points' behaviours. In other words, the link stream features capture a diversity of behaviours in the

data.

Since graphs and link streams are intuitively visualizable, in terms of explainability this means that one can compute such features and visualize subgraphs or substreams of interest, and from then on explain why such subgraphs are interesting.

6. Related Work

A graph-based vision of collaborative filtering algorithms was introduced, to the best of our knowledge, in 1999 by Aggarwal et al. [1]. Since then, it has been explored in several directions, most of which focused on unipartite graphs which were gathered in a *social* environment, where links between user- or item-nodes are explicit, such as a trust network (see [24] for a review on these *social recommender systems*). Desrosiers and Karypis [9] presented *path-based* methods in a traditional CF setting, which rely on counting shortest paths between nodes to compute their similarity, and *random-walk based* techniques, which evaluate the probability of reaching nodes by a random walk on the graph.

While finding similar users and grouping them into communities is a vast subject in social network analysis, it has been attempted with limited success in the recommendation context. See for instance Bernardes et al. [5] which use the Louvain algorithm in a collaborative filtering framework and obtain state-of-the-art results. There are currently no community detection algorithm in the link streams framework, which calls for theoretical work.

Introducing bipartite networks in a recommending framework was first proposed in [30]: the recommendation problem was presented as a *link prediction* problem, a well-studied subject in the complex networks community. The approach consist in an adequate projection of the bipartite network to embed the information available into the weight of a unipartite graph. While there have been several attempts at defining and finding bipartite communities [2, 18, 4], there are still several challenges to overcome before using them in a recommender system.

Temporal dynamics has a high impact on recommender system performance, mostly because of the *concept drift* effect: rating mean may change individually or globally as time elapses [15]. However, before 2010 and this pioneering paper by Koren, it was mostly ignored as a research avenue. Since then, ACM RecSys challenge top-3 contestants have succeeded in incorporating time-based features into an *XgBoost* prediction framework [20], and a

dedicated workshop took place at the ACM RecSys 2017[4].

Several approaches have been proposed to understand the temporal dynamics of interactions between entities modelled by a graph, bipartite or not. See [17, 7, 13] as general references on the topic. The most classic approach relies on building the graphs for different time slices and study the differences, losing information in the aggregation process. Some recent literature has shed some light on the importance of studying concomitantly dynamics *and* structure. A simple model proposed to add node and/or edge attributes to enclose temporal information [3]. It keeps the ability to use traditional graph tools, some key concepts like density, centrality or neighborhood are rather overlooked. There has been other attempts in modelling jointly those two dimensions, such as the MultiAspect graphs model [27] or the study of *path-based* centralities [21]. The *link streams* model is one of them, aiming at providing tools of wider generality for dynamics structures [16]. The model was used in a network-security study [25], while some graph algorithms are progressively adapted to it [12]. Up to our knowledge, our work is the first attempt at using a time-varying graph model in a recommender system setting.

7. Conclusion and perspectives

Our contribution is a proof-of-concept of incorporating link streams features in a classic recommender system environment. We use two large-scale datasets to explore the contribution they may offer to encode the dynamics of user-item interactions. We obtain a performance a little bit below the state-of-the-art, but relying solely on a limited model focusing on link streams features.

While there are several possible improvements for our content-based features, like entity-based merging of tags (or other NLP work), we would like to focus here on the avenues for future work related to the graph and link stream-based features.

Defining and listing communities of users, or of users and items, is a work to be done with the link stream paradigm. It is still an open issue in the broader time-varying graphs community. Since many recommender systems rely, either implicitly or explicitly on finding similar-minded users,

[4] `https://sites.google.com/edu.haifa.ac.il/tempreasoninginrs/`

establishing a robust model of communities may prove very useful for recommender systems. An avenue to be explored consists in devising an equivalent to the graph-based modularity and to pursue the work on nodes' betweenness centralities and clustering coefficients, which should also give precious information on the dynamic of neighbourhoods.

A recent trend in the deep-learning and recommendation literature has seen the importance of graph-based models being reinforced, with the introduction of Graph Convolutional Networks (GCNs) [6, 14, 29]. They aim at generalizing the very efficient convolutional methods devised for 2D data, which enables the embedding of graph information. However, while parts of the best models rely on random walks on local neighbourhoods, they also use very few graph-based features compared to other data (for instance, only one graph feature versus thousands of visual features in [29]). We are confident that link stream features could be incorporated with limited work into a GCN model. Given GCNs' performance, the current gap between link streams and pure deep learning approaches may be closed.

Besides performance metrics, we think that link streams features and algorithms contribute to improving recommender systems by offering more justifiability of the recommendations, since there is always a rather simple underlying model which can be called upon to explain why an item was proposed to a user. As a drawback, there is the newness of the concept, which still lacks some conceptual tools. We hope our experiment may prolong the fruitful exchanges of ideas between the recommender system and social network analysis communities.

References

[1] C. C. Aggarwal, J. L. Wolf, K.-L. Wu, and P. S. Yu. Horting hatches an egg: A new graph-theoretic approach to collaborative filtering. In *Proc. 5th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pages 201–212. ACM, 1999.

[2] M. J. Barber. Modularity and community detection in bipartite networks. *Phys. Rev. E - Statistical, Nonlinear, and Soft Matter Physics*, 76(6), 2007. ISSN 15393755.

[3] V. Batagelj and S. Praprotnik. An algebraic approach to temporal network analysis based on temporal quantities. *Social Network Analysis and Mining*, 6(1):28, 2016.

[4] S. J. Beckett. Improved community detection in weighted bipartite networks. *Royal Society Open Science*, 3(1), 2016. ISSN 2054-5703.

[5] D. Bernardes, M. Diaby, R. Fournier, F. Fogelman-Soulié, and E. Viennet. A social formalism and survey for recommender systems. *ACM SIGKDD Explorations*, 16(2):20–37, 2015.

[6] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[7] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems*, 27(5):387–408, 2012.

[8] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[9] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*, pages 107–144. Springer, 2011.

[10] J. Elith, J. R. Leathwick, and T. Hastie. A working guide to boosted regression trees. *Journal of Animal Ecology*, 77(4):802–813, 2008.

[11] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

[12] A.-S. Himmel, H. Molter, R. Niedermeier, and M. Sorge. Enumerating maximal cliques in temporal graphs. In *Advances in Social Networks Analysis and Mining (ASONAM), 2016 IEEE/ACM International Conference on*, pages 337–344. IEEE, 2016.

[13] P. Holme. Modern temporal network theory: a colloquium. *The European Physical Journal B*, 88(9):234, Sep 2015. ISSN 1434-6036. doi: 10.1140/epjb/e2015-60657-4.

[14] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[15] Y. Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[16] M. Latapy, T. Viard, and C. Magnien. Stream graphs and link streams for the modeling of interactions over time. *Social Netw. Analys. Mining*, 8(1):61:1–61:29, 2018. doi: 10.1007/s13278-018-0537-7.

[17] N. Masuda and R. Lambiotte. *A Guide to Temporal Networks*. World Scientific, 2016.

[18] T. Murata. Modularities for bipartite networks. *Proceedings of the 20th ACM conference on Hypertext and hypermedia - HT '09*, page 245, 2009.

[19] M. E. Newman. Mixing patterns in networks. *Physical Review E*, 67(2): 026126, 2003.

[20] A. Pacuk, P. Sankowski, K. Węgrzycki, A. Witkowski, and P. Wygocki. Recsys challenge 2016: Job recommendations based on preselection of offers and gradient boosting. In *Proceedings of the Recommender Systems Challenge*, RecSys Challenge '16, pages 10:1–10:4, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4801-0.

[21] I. Scholtes, N. Wider, and A. Garas. Higher-order aggregate networks in the analysis of temporal networks: path structures and centralities. *The European Physical Journal B*, 89(3):61, 2016.

[22] J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[23] F. Strub, R. Gaudel, and J. Mary. Hybrid recommender system based on autoencoders. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, pages 11–16. ACM, 2016.

[24] J. Tang, X. Hu, and H. Liu. Social recommendation: a review. *Social Network Analysis and Mining*, 3(4):1113–1133, 2013.

[25] T. Viard, R. Fournier-S'niehotta, C. Magnien, and M. Latapy. Discovering patterns of interest in ip traffic using cliques in bipartite link streams. In *International Workshop on Complex Networks*, pages 233–241. Springer, 2018.

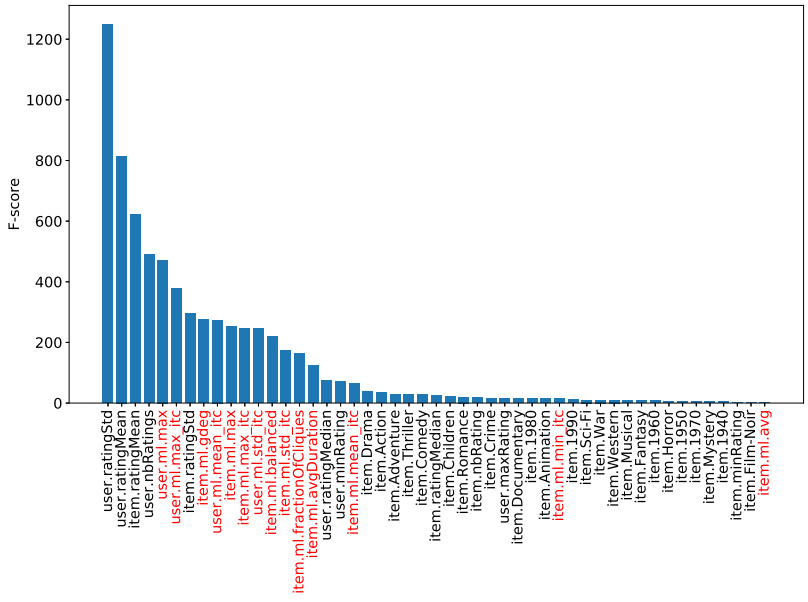
[26] M. Wan and J. McAuley. Item recommendation on monotonic behavior chains. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 86–94. ACM, 2018.

[27] K. Wehmuth, É. Fleury, and A. Ziviani. On multiaspect graphs. *Theoretical Computer Science*, 651:50–61, 2016.

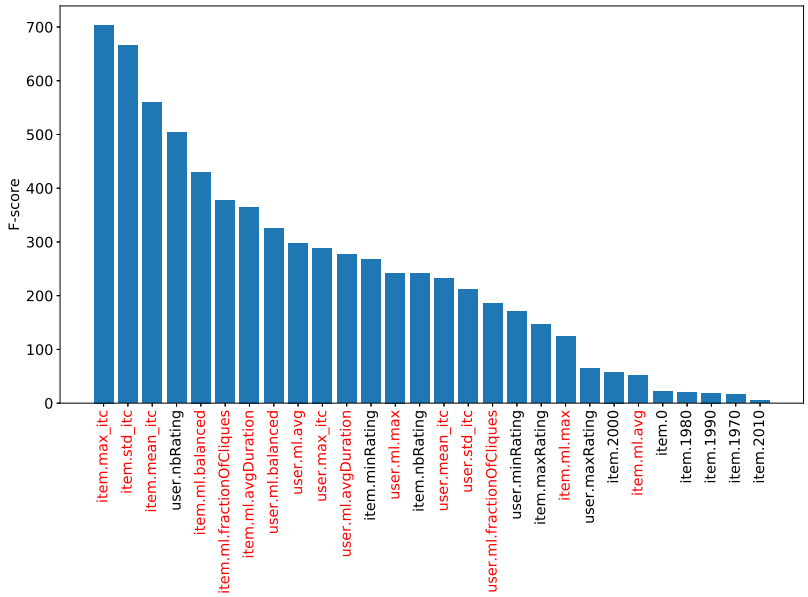
[28] Q. Yao, J. T. Kwok, F. Gao, W. Chen, and T.-Y. Liu. Efficient inexact proximal gradient algorithm for nonconvex problems. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 3308–3314, 2017.

[29] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec. Graph convolutional neural networks for web-scale recommender systems. In Y. Guo and F. Farooq, editors, *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 974–983. ACM, 2018. doi: 10.1145/3219819.3219890. URL `https://doi.org/10.1145/3219819`.

[30] T. Zhou, J. Ren, M. Medo, and Y.-C. Zhang. Bipartite network projection and personal recommendation. *Phys. Rev. E*, 76(4), 2007.



(a) Movielens



(b) Goodreads

Figure 2: Feature importance as selected by *XgBoost*. Link stream features are indicated in red. Feature importance is calculated as *"the number of times a variable is selected for splitting, weighted by the squared improvement to the model as a result of each split, and averaged over all trees" [10].*