Recurrent Coevolutionary Latent Feature Processes for Continuous-Time Recommendation

Hanjun Dai*, Yichen Wang*, Rakshit Trivedi, Le Song
College of Computing, Georgia Institute of Technology
{yichen.wang, hanjundai, rstrivedi}@gatech.edu, lsong@cc.gatech.edu

ABSTRACT

Matching users to the right items at the right time is a fundamental task in recommender systems. As users interact with different items over time, users' and items' feature may drift, evolve and co-evolve over time. Traditional models based on static latent features or discretizing time into epochs can become ineffective for capturing the fine-grained temporal dynamics in the user-item interactions. We propose a coevolutionary latent feature process model that accurately captures the coevolving nature of users' and items' feature. We use a recurrent neural network to automatically learn a representation of influences from drift, evolution and co-evolution of user and item features. We develop an efficient stochastic gradient algorithm for learning the model parameters which can readily scale up to millions of events. Experiments on diverse real-world datasets demonstrate significant improvements in user behavior prediction compared to state-of-the-arts.

1. INTRODUCTION

Online social platforms and service websites, such as Reddit, Netflix and Amazon, are attracting thousands of users every minute. Effectively recommending the appropriate service items is a fundamentally important task for these online services. By understanding the needs of users and serving them with potentially interesting items, these online platforms can improve the satisfaction of users, and boost the activities or revenue of the sites due to increased user postings, product purchases, virtual transactions, and/or advertisement clicks [31, 10].

As the famous saying goes "You are what you eat and you think what you read", both users' interests and items' semantic features are dynamic and can *evolve* over time [21, 3]. The interactions between users and service items play a critical role in driving the evolution of user interests and item features. For example, for movie streaming services, a long-time fan of comedy watches an interesting science fiction movie one day, and starts to watch more science fiction movies in place of comedies. Likewise, a single movie may also serve different segment of audiences at different times. For example, a movie initially targeted for an older generation may become popular among the younger generation, and the features of this movie need to be redefined.

Another important aspect is that users' interests and items' features can *co-evolve* over time, that is, their evolutions are intertwined and can influence each other. For instance, in online discussion forums, such as Reddit, although a group (item) is initially created for political topics, users with very different interest profiles can join this group (**user** $\rightarrow$ **item**). Therefore, the participants can shape the actual direction (or features) of the group through their postings and responses. It is not unlikely that this group can eventually become one about education simply because most users here concern about education (**item** $\rightarrow$ **user**). As the group is evolving towards topics on education, some users may become more attracted to education topics, and to the extent that they even participate in other dedicated groups on education. On the opposite side, some users may gradually gain interests in sports groups, lose interests in political topics and become inactive in this group. Such coevolutionary nature of user-item interactions raises very interesting questions on how to model them elegantly and how to learn them from observed interaction data.

Nowadays, user-item interaction data are archived in increasing temporal resolution and becoming increasingly available. Each individual user-item iteration is typically logged in the database with the precise time-stamp of the interaction, together with additional context of that interaction, such as tag, text, image, audio and video. Furthermore, the user-item interaction data are generated in an *asynchronous* fashion in a sense that any user can interact with any item at any time and there may not be any coordination or synchronization between two interaction events. These types of event data call for new representations, models, learning and inference algorithms.

Despite the temporal and asynchronous nature of such event data, for a long-time, the data has been treated predominantly as a static graph, and fixed latent features have been assigned to each user and item [25, 4, 2, 11, 22, 30, 31]. In more sophisticated methods, the time is divided into epochs, and static latent feature models are applied to each epoch to capture some temporal aspects of the data [21, 19, 29, 5, 13, 3, 24, 19, 29, 12, 16, 27]. For such epoch-based methods, it is not clear how to choose the epoch length parameter due to the asynchronous nature of the user-item interactions. First, different users may have very different

*Authors have equal contributions.

DLRS '16, September 15 2016, Boston, MA, USA
 ISBN 978-1-4503-4795-2/16/09…$15.00
DOI: http://dx.doi.org/10.1145/2988450.2988451

time-scale when they interact with those service items, making it very difficult to choose a unified epoch length. Second, it is not easy for the learned model to answer fine-grained time-sensitive queries such as when a user will come back for a particular service item. It can only make such predictions down to the resolution of the chosen epoch length. Most recently, [10] proposed a low-rank point process based model for time-sensitive recommendations from recurrent user activities. However, it still fails to capture the heterogeneous coevolutionary properties of user-item interactions. [28] modeled the coevolutionary property with an scalable convex algorithm. However, the mutual dependency between user and item is modeled as a linear embedding, which might not be expressive enough.

In the deep learning community, [26] proposed collaborative deep learning, a hierarchical Bayesian model that jointly performs learning for the content features and collaborative filtering for the ratings matrix. This method considers interaction data as static graph and also does not capture latent coevolutionary properties of user-item interactions. [15] applied recurrent neural network based approach to recommender systems. Specifically, they adopt item-to-item recommendation approach but use session based data with temporal ordering to capture influences of past interactions in particular session. However, it does not consider evolving and co-evolving features of users and items interacting with each other, partly because it is designed for the scenario where user information is not available. Finally, our work is inspired from newly proposed recurrent marked temporal point process framework [9] that builds a connection between Recurrent Neural Network (RNN) and Point Processes. However, [9] focuses on the task of next event prediction given a sequence of past events for an entity and is only designed for one-dimension point process. Significant generations and extensions are needed for the recommendation system setting.

In this paper, we combine RNN with fine-grained modeling of continuous-time user-item interactions and propose a coevolutionary latent feature process. It is designed specifically to take into account the asynchronous nature of event data, and the co-evolution nature of users' and items' latent features. Our model assigns an evolving latent feature process for each user and item, and the co-evolution of these latent feature processes is considered using two parallel components:

- (**Item** $\rightarrow$ **User**) A user's latent feature is determined by the latent features of the items he interacted with. Furthermore, the contributions of these items' features are captured by a nonlinear embedding.
- (**User** $\rightarrow$ **Item**) Conversely, an item's latent features are determined by the latent features of the users who interact with the item. Similarly, the contribution of these users' features is also modeled as a nonlinear embedding.

Besides the two sets of intertwined latent feature processes, our model can also take into account the presence of potentially high dimensional observed context features and links the latent features to the observed context features using a low dimensional projection. Despite the sophistication of our model, we show that the model parameter estimation can be efficiently solved by the stochastic gradient algorithm. Finally, the coevolutionary latent feature processes can be used for down-streaming inference tasks such as the return-time prediction. We evaluate our method over synthetic and real world datasets, verifying that our method can lead to significant improvements in user behavior prediction compared to previous state-of-the-arts.

2. BACKGROUND ON TEMPORAL POINT PROCESSES

A temporal point process [6, 7] is a random process whose realization consists of a list of discrete events localized in time, $\{t_i\}$ with $t_i \in \mathbb{R}^+$ and $i \in \mathbb{Z}^+$. Equivalently, a given temporal point process can be represented as a counting process, $N(t)$, which records the number of events before time t. An important way to characterize temporal point processes is via the conditional intensity function $\lambda(t)$, a stochastic model for the time of the next event given all the previous events. Formally, $\lambda(t)dt$ is the conditional probability of observing an event in a small window $[t, t+dt)$ given the history $\mathcal{H}(t)$ up to t, *i.e.*,

$$\lambda(t)\mathrm{d}t := \mathbb{P}\left\{\text{event in } [t, t+\mathrm{d}t) | \mathcal{H}(t)\right\} = \mathbb{E}[\mathrm{d}N(t) | \mathcal{H}(t)]$$

, where one typically assumes that only one event can happen in a small window of size $\mathrm{d}t$, *i.e.*, $\mathrm{d}N(t) \in \{0, 1\}$. Then, given a time $t' \geqslant t$, we can also characterize the conditional probability that no event happens during $[t, t')$ and the conditional density that an event occurs at time t' as $S(t') = \exp(-\int_t^{t'} \lambda(\tau)\,\mathrm{d}\tau)$ and $f(t') = \lambda(t')\,S(t')$ respectively [1]. The function form of the intensity $\lambda(t)$ is often designed to capture the phenomena of interests, like the Poisson process [20], Hawkes process [14], and Self-correcting process [17].

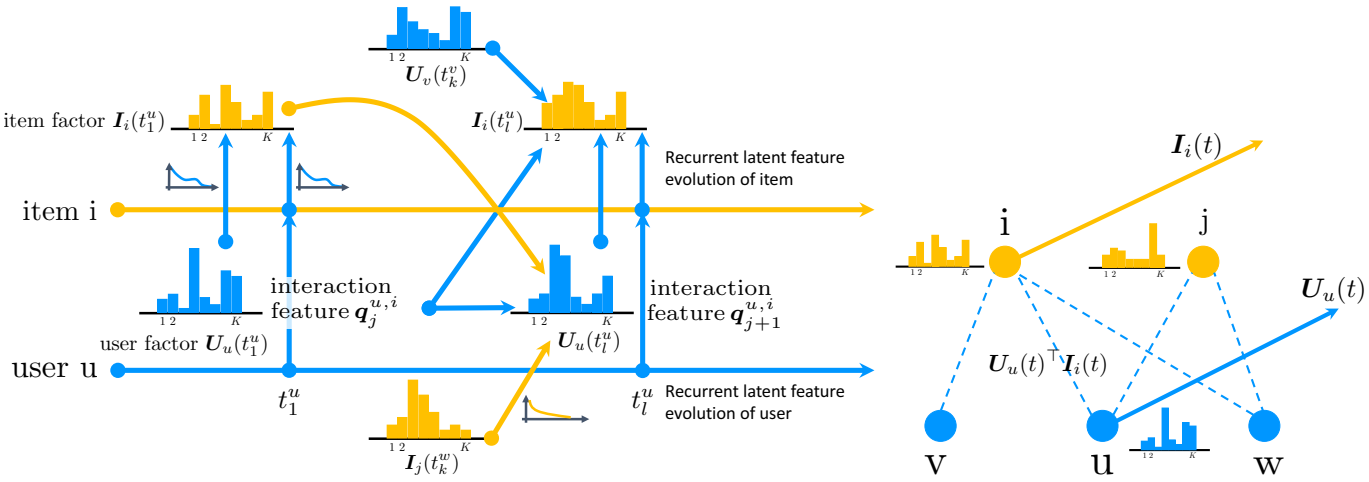
The function form of the intensity $\lambda(t)$ is often designed to capture the phenomena of interests. One commonly used form is Hawkes processes, whose intensity models the excitation between events, *i.e.*, $\lambda(t) = \mu + \alpha \sum_{t_i \in \mathcal{H}(t)} \kappa_\omega(t - t_i)$, where $\kappa_\omega(t) := \exp(-\omega t)\mathbb{I}[t \geqslant 0]$ is an exponential triggering kernel, $\mu \geqslant 0$ is a baseline intensity independent of the history. Here, the occurrence of each historical event increases the intensity by a certain amount determined by the kernel κ_ω and the weight $\alpha \geqslant 0$, making the intensity history dependent and a stochastic process by itself.

3. RECURRENT COEVOLUTIONARY LATENT FEATURE PROCESSES

In this section, we present the generative framework for modeling the temporal dynamics of user-item interactions. We first explicitly capture the co-evolving nature of users' and items' latent feature. Then, based on the compatibility between the users' and items' latent feature, we model the user-item interactions by a temporal point process and parametrize the intensity function by the compatibility.

3.1 Event representation

The input consists of all users' history events: $\mathcal{O} = \{\mathcal{S}^u\}$. For each user u, $\mathcal{S}^u = \{e_k^u\}$ denotes a sequence of events, with $e_k^u := (i_k, \boldsymbol{q}_k^{u,i_k}, t_k^u)$. It represents that u interacts with item i_k at time t_k^u and generates an interaction feature $\boldsymbol{q}_k^{u,i_k} \in \mathbb{R}^D$. For instance, the interaction feature can be a textual message delivered from the user to the chatting-group or a review of the hotel. Furthermore, it can also be unobservable if the data only contains the temporal information. Similarly, for each user i, we denote $\mathcal{S}^i = \left\{e_k^i | i_k = i\right\}$ as the collection of interaction events across all the users. We set each element in $\mathcal{S}^i$ as $e_k^i := (u_k, \boldsymbol{q}_k^{i,u_k}, t_k^i)$.



(a) Co-evolving latent feature processes.

(b) Temporal interactions.

Figure 1: Model illustration for user u (blue) and item i (yellow). (a) User features and item features influence each other and co-evolve over time. At time t_l^u, the latent feature $I_i(t_l^u)$ of item i (in yellow) is influenced by the users' feature ($\boldsymbol{U}_u(t_l^u)$, $\boldsymbol{U}_v(t_k^v)$), and the cumulative past interaction features ($\boldsymbol{q}_j^{u,i}$, $\boldsymbol{q}_{j+1}^{u,i}$) that captured by RNN. Conversely, $\boldsymbol{U}_u(t_l^u)$ is influenced by the item features ($\boldsymbol{I}_i(t_1^u)$,$\boldsymbol{I}_j(t_k^w)$), and by past interaction feature $\boldsymbol{q}_j^{u,i}$. (b) The temporal interacting process where the tendency that a user will interact with an item depends on the compatibility (inner product) of their latent features. u is more likely to interact with i than j.

3.2 Latent feature processes

Given m users and n items, we associate latent features $\boldsymbol{U}_u(t) \in \mathbb{R}^K$ with each user u and $\boldsymbol{I}_i(t) \in \mathbb{R}^K$ with each item i. These features represent the subtle properties which cannot be directly observed, such as the interests of a user and the semantic topics of an item. Specifically, we model the *drift, evolution, and co-evolution* of $\boldsymbol{U}_u(t)$ and $\boldsymbol{I}_i(t)$ as follows a piecewise constant function of time and has jumps only at event times. Specifically, we have define:

User latent feature process. For each user u, we formulate $\boldsymbol{U}_u(t)$ at time t_k^u as:

$$\boldsymbol{U}_u(t_k^u) = \sigma\Big(\underbrace{\boldsymbol{W}_1(t_k^u - t_{k-1}^u)}_{\text{drift}} + \underbrace{\boldsymbol{W}_2 \boldsymbol{U}_u(t_{k-1}^u)}_{\text{self evolution}} + \underbrace{\boldsymbol{W}_3 \boldsymbol{I}_{i_k}(t_k^u-)}_{\text{co-evolution: item feature}} + \underbrace{\boldsymbol{W}_4 \boldsymbol{q}_k^{u,i_k}}_{\text{evolution: interaction feature}} \Big) \tag{1}$$

Item latent feature process. For each item i, we specify $\boldsymbol{I}_i(t)$ at time t_k^i as:

$$\boldsymbol{I}_i(t_k^i) = \sigma\Big(\underbrace{\boldsymbol{V}_1(t_k^i - t_{k-1}^i)}_{\text{drift}} + \underbrace{\boldsymbol{V}_2 \boldsymbol{I}_i(t_{k-1}^i)}_{\text{self evolution}} + \underbrace{\boldsymbol{V}_3 \boldsymbol{U}_{u_k}(t_k^i-)}_{\text{co-evolution: item feature}} + \underbrace{\boldsymbol{V}_4 \boldsymbol{q}_k^{i,u_k}}_{\text{evolution: interaction feature}} \Big) \tag{2}$$

where $t-$ means the time just before time t, $\boldsymbol{W}_4, \boldsymbol{V}_4 \in \boldsymbol{R}^{K \times D}$ are the embedding matrices mapping from the explicit high-dimensional feature space into the low-rank latent feature space and $\boldsymbol{W}_i, \boldsymbol{V}_i \in \boldsymbol{R}^{K \times K}, i = 1, 2, 3$ are weights parameters, σ is the nonlinear activation function, such as commonly used ReLU or Sigmoid. Figure 1 summarizes the basic setting of our model.

Here both the user and item's feature processes are piecewise constant functions of time and only updated if an interaction event happens. Next we discuss the rationale of each term in detail.

Time drift. The first term is defined based on the time difference between inter events. It allows the basic features of users (*e.g.*, a user's self-crafted interests) and items (*e.g.*, textual categories and descriptions) to smoothly drift through time. Such changes of basic features normally are caused by external influences.

Self evolution. The current user feature should also be influenced by its feature at the earlier time.

Evolution with interaction features. Users' and items' features can evolve and be influenced by the characteristics of their interactions. For instance, the genre changes of movies indicate the changing tastes of users. The theme of a chatting-group can be easily shifted to certain topics of the involved discussions. In consequence, this term captures the influence of the current interaction features to the changes of the latent user (item) features.

User-item coevolution. Users' and items' latent features can mutually influence each other. This term captures the two parallel processes:

- (**Item → User**) A user's latent feature is determined by the latent features of the items he interacted with. At each time t_k, the latent item feature is $\boldsymbol{I}_{i_k}(t_k^u)$. In our model, we capture both the temporal influence and feature of each history item as a latent process.
- (**User → Item**) Conversely, an item's latent features are determined by the latent features of the user who just interacts with the item.

Therefore, we have incorporated both of the exogenous and endogenous influences into a single model. First, each process evolves according to the respective exogenous base temporal user (item) features. Second, the two processes also interdepend on each other due to the endogenous influences from the interaction features and the entangled latent features. We present our model in the most general form and the specific choices of $\boldsymbol{U}_u(t), \boldsymbol{I}_i(t)$ are dependent on applications. For example, if no interaction feature is observed, we drop the third term in (1) and (2).

3.3 User-item interactions as temporal point processes

For each user, we model the recurrent occurrences of user u's interaction with all items as a multi-dimensional temporal point process. In particular, the intensity in the i-th dimension (item i) is:

$$\lambda^{u,i}(t) = \exp\Bigg(\underbrace{\eta^{u,i}}_{\substack{\text{long-term}\\ \text{preference}}} + \underbrace{\boldsymbol{U}_u(t)^\top \boldsymbol{I}_i(t)}_{\substack{\text{compatibility}\\ \text{preference}}} + \underbrace{\alpha^{u,i}(t - t_k^u)}_{\substack{\text{short-term}\\ \text{preference}}} \Bigg), \tag{3}$$

where $\boldsymbol{\eta} \in \mathbb{R}^{m \times n}$ is a baseline intensity matrix. The rationale of this formulation is fourfold. First, instead of discretizing the time, we explicitly model the timing of each event occurrence as a continuous random variable, which naturally captures the heterogeneity of the temporal interactions between users and items. Second, the base intensity $\eta^{u,i}$ represents the long-term preference of user u to item i, independent of the history. Third, the tendency for user u to interact with item i at time t depends on the compatibility of their instantaneous latent features. Such compatibility is evaluated through the inner product of their time-varying latent features. Moreover, $\alpha^{u,i}(t-t_k^u)$ emphasizes the current influence of event at t_k^u. Finally, the enclosing exp function guarantees the intensity to be positive and well-defined.

Remark. Our model inherits the merits from classic content filtering, collaborative filtering, and the most recent temporal models. For the cold-start users having few interactions with the items, the model adaptively utilizes the purely observed user (item) base properties and interaction features to adjust its predictions, which incorporates the key idea of feature-based algorithms. When the observed features are missing and non-informative, the model makes use of the user-item interaction patterns to make predictions, which is the strength of collaborative filtering algorithms. However, being different from the conventional matrix-factorization models, the latent user and item features in our model are entangled and able to co-evolve over time. Finally, the general temporal point process ingredient of the model makes it possible to capture the dynamic preferences of users to items and their recurrent interactions, which is more flexible and expressive.

Note [10] proposed a point process based model that assumes each user-item pair as an *independent one-dimension* Hawkes process. This might be realistic, since one can not interact with two items at the same time, and it ignores the influence of past interactions with other items to the user's and item's feature. Hence it fails to capture the coevolutionary properties of user-item interactions.

Finally, our model is generative with rich theoretical support in point process [6, 7, 8, 1]. One can further learn arrival time distributions of future events based on this probabilistic framework. *Time prediction* is important with huge benefits, *e.g.*, for most on-line stores, accurate prediction of the returning time of customers can help to improve stock management and products display and arrangement. As for web company, like Google and Facebook, it can have potential impact to display ads. If we can predict when users will come back, we can make the existing ads bidding much more economic, allowing marketers to bid on time slots.

4. PARAMETER LEARNING

Having presented our model, in this section, we propose an efficient framework to learn the parameters.

Given a collection of events $\mathcal{O}$ recorded within a time window $[0, T)$, we estimate the parameters using maximum likelihood estimation of all events. The joint negative log-likelihood [8] is:

$$\ell = -\sum_{\mathcal{S}^u \in \mathcal{O}} \left\{ \sum_{e_k^u} \log\left(\lambda^{u,i_k}(t_k^u)\right) - \sum_{i=1}^{n} \int_0^T \lambda^{u,i}(\tau)\, d\tau \right\} \tag{4}$$

This objective is highly non-convex, so we seek to use stochastic gradient descent (SGD) with mini-batch for training. Specifically, we adopt the Adam Optimizer [**?**], and use gradient clip to avoid gradient explosion. The Back Propagation Through Time (BPTT) is a classical way to train a Recurrent Neural Network. However, in our model, all the events are more or less related to each other. That is to say, we are essentially train a RNN on network structured dataset. To make the back propagation tractable, one typically needs to do truncation during training.

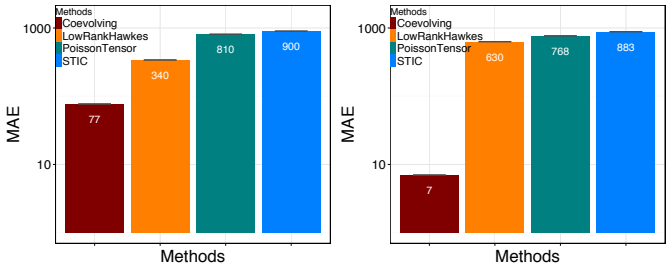
In our work, we first order all the events globally and then do mini-batch training in a sliding window fashion. Each time when we do feed forward and back propagation, we take the consecutive events within current sliding window to build the computational graph. Thus the truncation is done on the global timeline. Ordering the events globally also allows us to keep the user and item latent features that could be used for the future mini-batch training.

5. EXPERIMENTS

We evaluate our model on both synthetic and real-world datasets. For each sequence of user activities, we use all the events up to time $T \cdot p$ as the training data, and the rest events as the testing data, where T is the observation window. We report the results on the task of *time prediction*: we predict the time when a testing event will occur between a given user-item pair. We numerically compute the conditional density [1] of the next event time of the given user-item pair and use the expectation as our estimation. We report the Mean Absolute Error (MAE) between the predicted and true time. Furthermore, we also report the relative percentage of the prediction error with respect to the entire testing time window.

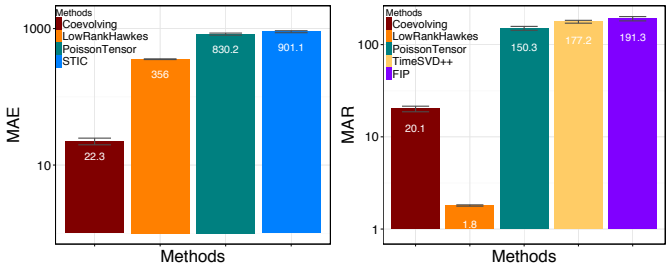
5.1 Competitors

- **PoissonTensor** [5]: Poisson Tensor Factorization has been shown to perform better than factorization methods based on squared loss [19, 29] on recommendation tasks. The performance for this baseline is reported using the average of the parameters fitted over all time intervals.
- **LowRankHawkes** [10]: This is a low rank point process based model which assumes user-item interactions to be independent of each other and does not capture the co-evolution of user and item features.
- **STIC** [18]: it fits a semi-hidden markov model to each observed user-item pair and is only designed for time prediction.
- **TimeSVD++ [21] and FIP [30]**: These two methods are only designed for explicit ratings, the implicit user feedbacks (in the form of a series of interaction events) are converted into the explicit ratings by the respective frequency of interactions with users.



(a) Synthetic (b) Yelp

Figure 2: Prediction results on synthetic and Yelp dataset



(a) IPTV MAE (b) IPTV MAR

Figure 3: Prediction results on IPTV dataset

5.2 Results

Synthetic Data. We simulate interaction data for 1,000 users and 1,000 items. We generate a total of $758K$ interactions using Ogata's thinning algorithm [23]. **Yelp Data.** We use the data provided by Yelp Dataset challenge Round 7. It contains reviews for various businesses from October, 2004 to December, 2015. There are a total number of 292K reviews between 1,000 users and 47,924 businesses with 34,508 text features. **IPTV Data.** IPTV contains 7,100 users' watching history of 436 TV programs in 11 months (Jan 1 - Nov 30 277 2012), with 2,392,010 events, and 1,420 movie features (including 1,073 actors, 312 directors, 22 278 genres, 8 countries and 5 years).

Figure 2 shows that our method has significant improvements for time prediction on synthetic dataset and is two magnitude better than other baselines on the yelp challenge dataset. It means our method accurately captures the pattern between user and item interactions.

We further did experiments on the item recommendation and time prediction jointly on the IPTV dataset. Figure 3 shows that we got the second best item prediction performance using the Mean Average Rank (MAR) metric. This result is slightly worse than the LowRankHawkes, but still significantly better than other methods. Also, for the time prediction, our method outperformed others significantly.

6. DISCUSSION

We have proposed an efficient framework for modeling the co-evolution nature of users' and items' latent features. It is a generative model designed for modeling and understanding user's online behaviors, which is different from prior work that only focuses on the prediction task in the recommender system. Moreover, the user and item's evolving and co-evolving processes are captured by the RNN. We demonstrate the superior performance of our method on the time prediction task. For item recommendation, one can train another classifier according to the compatibility between user and item features. Future work includes extending to other applications such as modeling dynamics of social message groups, and understanding peoples' behaviors on Q&A sites.

7. REFERENCES

[1] O. Aalen, O. Borgan, and H. Gjessing. *Survival and event history analysis: a process point of view*. Springer, 2008.

[2] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In J. Elder, F. Fogelman-Soulié, P. Flach, and M. Zaki, editors, *KDD*, pages 19–28. ACM, 2009.

[3] L. Charlin, R. Ranganath, J. McInerney, and D. M. Blei. Dynamic poisson factorization. In *RecSys*, 2015.

[4] Y. Chen, D. Pavlov, and J. Canny. Large-scale behavioral targeting. In J. Elder, F. Fogelman-Soulié, P. Flach, and M. J. Zaki, editors, *KDD*, pages 209–218. ACM, 2009.

[5] E. C. Chi and T. G. Kolda. On tensors, sparsity, and nonnegative factorizations. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1272–1299, 2012.

[6] D. Cox and V. Isham. *Point processes*, volume 12. Chapman & Hall/CRC, 1980.

[7] D. Cox and P. Lewis. Multivariate point processes. *Selected Statistical Papers of Sir David Cox: Volume 1, Design of Investigations, Statistical Methods and Applications*, 1:159, 2006.

[8] D. Daley and D. Vere-Jones. *An introduction to the theory of point processes: volume II: general theory and structure*, volume 2. Springer, 2007.

[9] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, and L. Song. Recurrent marked temporal point processes: Embedding event history to vector. In *KDD*. ACM, 2016.

[10] N. Du, Y. Wang, N. He, and L. Song. Time sensitive recommendation from recurrent user activities. In *NIPS*, 2015.

[11] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan. Collaborative filtering recommender systems. *Foundations and Trends in Human-Computer Interaction*, 4(2):81–173, 2011.

[12] P. Gopalan, J. M. Hofman, and D. M. Blei. Scalable recommendation with hierarchical poisson factorization. *UAI*, 2015.

[13] S. Gultekin and J. Paisley. A collaborative kalman filter for time-evolving dyadic processes. In *ICDM*, pages 140–149, 2014.

[14] A. G. Hawkes. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1):83–90, 1971.

[15] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk. Session-based recommendations with recurrent neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[16] B. Hidasi and D. Tikk. General factorization framework for context-aware recommendations. *Data Mining and Knowledge Discovery*, pages 1–30, 2015.

[17] V. Isham and M. Westcott. A self-correcting pint process. *Advances in Applied Probability*, 37:629–646, 1979.

[18] K. Kapoor, K. Subbian, J. Srivastava, and P. Schrater. Just in time recommendations: Modeling the dynamics of boredom in activity streams. In *WSDM*, 2015.

[19] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Recsys*, pages 79–86. ACM, 2010.

[20] J. F. C. Kingman. *Poisson processes*, volume 3. Oxford university press, 1992.

[21] Y. Koren. Collaborative filtering with temporal dynamics. In *KDD*, 2009.

[22] Y. Koren and J. Sill. Ordrec: an ordinal model for predicting personalized item rating distributions. In *RecSys*, 2011.

[23] Y. Ogata. On lewis' simulation method for point processes. *IEEE Transactions on Information Theory*, 27(1):23–31, 1981.

[24] J. Z. J. L. Preeti Bhargava, Thomas Phan. Who, what, when, and where: Multi-dimensional collaborative recommendations using tensor factorization on sparse user-generated data. In *WWW*, 2015.

[25] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In W. Cohen, A. McCallum, and S. Roweis, editors, *ICML*, volume 307, pages 880–887. ACM, 2008.

[26] H. Wang, N. Wang, and D.-Y. Yeung. Collaborative deep learning for recommender systems. In *KDD*. ACM, 2015.

[27] X. Wang, R. Donaldson, C. Nell, P. Gorniak, M. Ester, and J. Bu. Recommending groups to users using user-group engagement and time-dependent matrix factorization. In *AAAI*, 2016.

[28] Y. Wang, N. Du, R. Trivedi, and L. Song. Coevolutionary latent feature processes for continuous-time user-item interactions. In *NIPS*, 2016.

[29] L. Xiong, X. Chen, T.-K. Huang, J. G. Schneider, and J. G. Carbonell. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *SDM*, pages 211–222. SIAM, 2010.

[30] S.-H. Yang, B. Long, A. Smola, N. Sadagopan, Z. Zheng, and H. Zha. Like like alike: joint friendship and interest propagation in social networks. In *WWW*, 2011.

[31] X. Yi, L. Hong, E. Zhong, N. N. Liu, and S. Rajan. Beyond clicks: Dwell time for personalization. In *RecSys*, 2014.