Personalization of Learning Paths in Online Communities of Creators

Mingxuan Sun*
Division of Computer Science and Engineering
Louisiana State University
msun@csc.lsu.edu

Seungwon Yang
School of Library and Information Science
Center for Computation and Technology
Louisiana State University
seungwonyang@lsu.edu

ABSTRACT

In massive online communities of creators (OCOCs), one of the core challenges is to encourage users to learn to create original contents using basic components. Recommending the right learning components at the right time is critical for improving user engagement and has not been fully studied due to the unstructured nature of online communities. To address the problem, we propose in this paper a novel recommendation model which integrates Cox's survival analysis and collaborative filtering. Our model can incorporate factors such as user learning history and social engagements, which provides us insights in improving the personalized service. We apply our method to the user data from Scratch online platform and demonstrate the performance of the model.

1. INTRODUCTION

In recent years, the number of online learning communities (OCOCs) has increased exponentially as evidenced by successful platforms such as Scratch online[1]. These online communities offer flexible learning environment where users can create projects (e.g., games, art designs), share projects, and engage with like-minded users in the community. One of the goals is to foster learning programming concepts through developing and sharing projects among its users based on interactions in the community [11]. Previous studies [7] have found that creating and sharing projects is the gateway to other online social activities including commenting and following. However, only about 29% of Scratch users would like to share their projects and about half of them contribute no more than one project.

One way to improve user engagement is to track users' learning history and recommend contents tailored to each individual. For example, Scratch users learn to create projects by manipulating basic programming blocks such as "goto", "changecolor", and "doIf". Each block is categorized in a certain Computational Thinking (CT) concept [6]. Users are expected to learn CT concepts such as "motion" by manipulating blocks such as "goto", "bounce", and "turn". Users may follow different learning paths over time. Based on programming blocks that each user has used in his/her previous projects, we can recommend particular blocks, concepts, or projects tailored to the individual. For instance, for users who are interested in animation projects with some basic motion blocks such as "goto", the system can recommend projects that have more advanced motion blocks such as "bounce".

In addition to what to recommend, when is a good time to recommend is another important factor to consider since suggesting blocks to users at the right time may influence learning effectiveness and efficiency. For example, if a user is still struggling with basic motion techniques such as "goto", it may not be a good idea to introduce a project or a more advanced programming concept such as "turn" or "direction". Our goal is to alleviate the high dropout rates in the early stage through personalization of the learning path.

In this paper, we propose a model to learn the probability of a user's exposure to a certain learning component at a particular time. The probability of exposure is estimated based on a collaborative filtering model, which recommends the user the items favored by the like-minded. The conditional probability of a user being exposed to a given item at a particular time is modeled by the Cox proportional hazard model from survival analysis.

2. RELATED WORK

Early studies on learning behavior analysis for OCOCs have been based on case-studies evaluating learning process qualitatively [5, 12]. Other attempts [3, 7] have focused on clustering user behaviors based on types and volumes of users' online activities. A recent work by Yang et al. [14] modeled *informal learning trajectories* quantitatively as the growth of cumulative usage of programming blocks by each user.

Personalization approaches that are based on user behaviors have been widely studied in different types of Web services such as e-commerce. In e-commerce, most personalization approaches focus on recommending users the items that have been favored by like-minded users based on their purchase history. Traditional recommendation algorithms are memory based methods including vector similarity and correla-

*Corresponding author.
[1]https://scratch.mit.edu/

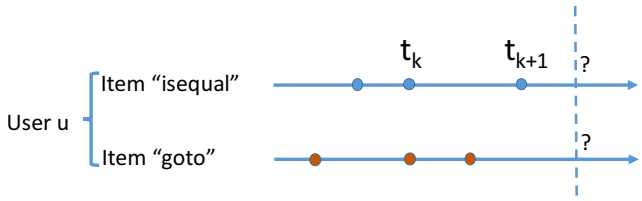

Figure 1: Time-aware recommendation. The occurrence time of user-item interaction is modeled using survival analysis. Our goal is to predict the most desired learning item i at a particular time t for each individual user u.

tion [2]. The state-of-the-art methods including the one that won the Netflix competition [9] are based on matrix factorization. The time factor in personalization services largely affects the user satisfaction of the service [13, 10]. Our contribution in this paper lies in that we incorporate both the Cox model and collaborative filtering to provide personalized recommendation for online learners.

3. METHOD

In OCOCs, users create and share projects consisting of basic items such as programming blocks in Scratch. Each item belongs to a certain category. Based on user-item interaction histories, we would like to suggest items tailored to each user at a particular time. To achieve this goal, we propose to estimate the joint probability $p(u, i, t) = p(t|u, i)p(u, i)$, where $p(u, i)$ is the probability of user u interacting with item i and $p(t|u, i)$ is the conditional probability of user u interacting with item i at time t.

We model the occurrence time t of the event that user u interacts with item i using the Cox model in survival analysis. Survival analysis is used to estimate the probability of the occurrence of an event $p(\text{event in } [t, t + \Delta t])$ such as when a patient fails to survive. In the online learning context, our task is to estimate the probability of the occurrence of exposing to a specific learning block for each user, which is $p(t|(u, i))$. As shown in Figure 1, in the observed sequences of user-item interactions, a user builds a project with a set of items (e.g., "isequal" and "goto") at time t_k. Then item i is used again in another project of the same user at time t_{k+1}. Let x_k be the covariates associated with user u at time t_k. We are interested in predicting the time gap $t_{k+1} - t_k$.

Let $\lambda(t)$ denote the instantaneous rate of event happening at time t following the last event given the covariates x_k, that is $\lambda(t) = P(T = t \mid T \geq t)$. The Cox model assumes that the covariates only affect the magnitude of each individual hazard rates. Formally, for an individual observation with covariates x_k, the hazard at time t is:

$$\lambda(t) = \lambda_0(t) * \exp(x_k^T \beta), \tag{1}$$

where λ_0 is the non-parametric baseline hazard function, x_k is the covariates, and β is the regression coefficient. The log likelihood of observing the occurrences is:

$$\log L = \sum_{k=1}^{K} \left\{ d_k \log \lambda(t_k) - \int_0^{t_k} \lambda(\tau) d\tau \right\}, \tag{2}$$

where d_k is a censor indicator, taking the value one if event occurs at time t_k or the value zero if event does not occur till time t by the end of observation window. The parameters β and the baseline hazard λ_0 can be estimated by maximizing the log partial likelihood with Breslow's approximation [4].

We further estimate the probability $p(u, i)$ of a user favoring a particular item (e.g., block) by adopting collaborative filtering (CF) recommendation algorithms. User interactions contain substantial information to improve recommendation accuracy. For example, in Scratch, users play with a set of programming blocks to develop a project. Therefore, the frequency of each type of block may indicate their preferences. Based on the previous learning history, the system can predict interesting blocks tailored to individual taste. Collaborative filtering methods focus on detecting users with similar preferences and recommending items favored by the like-minded. Algorithms range from similarity based CF methods [2] to matrix factorization based CF methods popularized by the Netflix Prize Competition [9].

Let r_{ui} denote the observed preference of user u for item i, where $u = 1, 2, \ldots, m$ and $i = 1, 2, \ldots, n$. The pairs (u, i) are stored in the set $O = \{(u, i) \mid r_{ui} \text{ is observed}\}$. Since the observed ratings or event frequencies are very sparse, matrix factorization is used to learn latent features of both users and items in a lower dimensional space such that the product of each user-item pair can best approximate the ratings. Specifically, let θ_u and v_i denote latent features for user u and items i, where θ_u and v_i are k-dimensional vectors. The latent features can be estimated by minimizing a prediction loss function between the predicted ratings and true ratings of users. That is,

$$\min_{\Theta, V} \sum_{(u,i) \in O} (r_{ui} - \theta_u^\top v_i)^2, \tag{3}$$

where $\Theta = [\theta_1, \theta_2, \ldots, \theta_m]$ is a $k \times m$ matrix and $V = [v_1, v_2, \ldots, v_n]$ is a $k \times n$ matrix. A gradient descent based method [9] can be used to estimate latent features. The probability of user favoring an item $p(u, i)$ can be generated using a softmax function:

$$p(u, i) = \frac{\exp(r_{ui})}{\sum_{j=1}^{n} \exp(r_{uj})}, \tag{4}$$

4. EXPERIMENTAL RESULTS

We evaluate the model performance through two steps: time-to-return prediction and time-aware recommendation. In the first step, for every user-item interaction (u, i), we estimate the probability of the next occurrence at time t and use the expected value of the time as the predicted time to return. In the second step, for each user u at a particular time t, we rank each item i by the joint probability $p(u, i, t)$ and recommend top-K items. We present the experimental details including data collection, evaluation metrics, and competing baselines.

4.1 Data Collection

We apply our method to user data which was released in spring of 2014 from Scratch online[2]. Users can create a project by programming with basic components called blocks. Each block can be categorized into one or more CT concepts.

[2]https://llk.media.mit.edu/scratch-data/

Table 1: Covariate analysis for CT concept "conditionals". *** p<0.001, ** p<0.01, * p<0.05, . p<0.1

Covariate Name	Coefficient	P-Value
is.remix	0.190556	0.000593 ***
is.self.remix	-0.140119	0.062772 .
is.remixed	0.447668	2.44e-15 ***
like 2 or more	0.226432	0.001440 **
follow 2 or more	0.262599	0.000346 ***
comments 2 or more	0.478668	< 2e-16 ***
conditionals experience	0.332191	1.14e-08 ***
operators experience	-0.074161	0.236036
data experience	-0.157914	0.010259 *

Table 2: Covariate analysis for CT concept "data". *** p<0.001, ** p<0.01, * p<0.05, . p<0.1

Covariate Name	Coefficient	P-Value
is.remix	0.15007	0.018629 *
is.self.remix	-0.18239	0.037235 *
is.remixed	0.52571	3.33e-16 ***
like 2 or more	0.23287	0.005221 **
follow 2 or more	0.20475	0.023510 *
comment 2 or more	0.54465	< 2e-16 ***
conditionals experience	0.03648	0.605257
operators experience	0.14616	0.041800 *
data experience	0.12105	0.076548 .

We adopt the the mapping table from blocks to CT concepts as suggested in [6]. Users are encouraged to share their projects and interact with others by commenting projects, favoring projects, or following other users. For each user, the dataset includes the project details including block usage and timestamps. It also maintains tables of different types of social interactions including user follower-followed relationship and comments. The user history data collected from December 2011 to March 2012 are used to create the training and the testing datasets. Possible spam users who create more than 100 projects in a day are filtered out. The remaining data contains 22415 users and 170 learning blocks with 6 CT concepts. All user records observed during December 2011 to February 2012 are used to train the model through cross-validation and all user records during March 2012 are used for testing.

The following covariates are used to estimate the Cox model. Covariates related to user activity history include the number of days since registration and the gap since last login. User social interaction covariates include the number of projects liked, the number of friends followed, and the number of comments on projects. User project details include the number of projects created, the number of types of blocks, and the number of concepts. We collect user covariates on a daily basis and predict the days till the user's next event. Users who had not been exposed to the event by the end of the time window were censored.

4.2 Performance Evaluation

In the first step "time-to-return prediction", for every block pair (u, i), we estimate the probability of the next occurrence at time t and treat the expected value of the time as the predicted time to return. Since the data are sparse, a direct estimation of a survival model for each block will be noisy. Instead, we train a Cox model for each CT concept using the interactions events of blocks belonging to that concept. To evaluate the performance, we predict the expected time from the learned density function and compute the Rooted Mean Square Error (RMSE) with respect to the true time. We compare the Cox model against the baselines including linear regression and decision tree regression. Smaller RMSE values indicate better performance.

The importance of covariates for predicting each individual user's exposure to CT concepts "conditionals" and "data" are shown in Tables 1 and 2. Both tables show the covariates' names, the regression coefficients and the significance scores. A positive regression coefficient for a variable implies a higher hazard if the value of the variable is high. Both tables show that the regression coefficients for the variable "is.remixed.bool" are positive. It indicates that if a user's project is remixed by others, the hazard rate of observing the user's next event will increase by a factor of $exp(0.190556) - 1$ compared with the baseline hazard. On the contrary, a negative regression coefficient implies a lower hazard, which means the probability of user interacting with the blocks belonging to that concept will be smaller. The value of the coefficient is statistically significant at different significance levels. We only show the covariates with highest significant levels.

As shown in the tables, for both CT concepts "conditionals" and "data", for users who share projects later remixed by others, it is more likely that these users will be back creating projects in the future. Interestingly, users who remix others' projects will be more likely to create projects than those who remix their own projects. In addition, users who like two or more projects, who follow two or more friends, and who have two or more comments are more likely to create and share projects in the future than those who have no social interactions. This implies that social interactions help users to learn and share. In addition, we can see that users who have built blocks in the concept "conditional" are more likely to build blocks falling into the same concept. Interestingly, users who have built blocks in the concepts "operator" and "data" are more likely to build blocks in the concept "data".

We then use the estimated model to predict the time to the next event in each CT concept. Table 3 displays the root mean square error (RMSE) for the return time prediction using the Cox model and baselines, respectively. For concepts "loops", "conditionals", "operators", and "data", the hazard based approach outperforms all the other baselines. For concept "event", the hazard based approach performs very close to linear regression and both of them perform better than the others. All the baselines do not model the underlying temporal patterns in the observed sequences.

For the final step "time-aware recommendation", suppose the testing event of user u occurs at time t, we compute the probability $p(u, i, t)$ of the user favoring an item i at time t for each item i and rank among all items by probability. Ideally, the observed items that the user actually interacts with should appear on top positions. In information retrieval, we focus on the evaluation accuracy on top positions using several standard metrics including precision at k (P@k), Mean Average Precision (MAP) and Normalized Discounted

Table 3: RMSE comparison for user return time prediction. Smaller values indicate better performance.

	Loops	Events	Conditionals	Operators	Data
Linear Regression	9.13	**9.20**	8.94	8.79	8.68
Decision Tree Regression	9.33	9.41	9.13	9.00	8.80
Cox model	**9.04**	9.25	**8.63**	**7.97**	**7.62**

Table 4: Comparison of recommendation accuracy.

	P@1	P@3	P@5	MAP@20	NDCG@20
NMF	0.78	0.70	0.64	0.71	0.67
SurvMF	0.84	0.72	0.64	0.72	0.68

Cumulative Gain at k(NDCG) [8]. We compare with the state-of-the-art baseline non-negative matrix factorization (NMF) [1]. We follow the standard procedure in collaborative filtering to estimate the model using the user data in the training set and evaluate the performance of the prediction in the test set. Specifically, the user records observed before March 2012 are used to train and the user records in March 2012 are used to test. The data contains the rating of each user-block pair, where the rating corresponds to the categorization of event occurrences. The maximum rating is 6 for six or more event occurrences. At the time of the testing event, we compare the ranked list with ground truth. As shown in Table 4, since our method (SurvMF) integrates the survival model into the matrix factorization to capture the temporal dynamics of user-item interaction, it can achieve better performance.

5. CONCLUSIONS AND FUTURE WORK

In this work, we have focused on personalization of learning path in massive online communities of creators. One of the main challenges in online learning is high dropout rates in the early stage due to cognitive overload. To alleviate the problem, we propose a novel model integrating the Cox model and matrix factorization to recommend the right learning contents at the right time. The model can incorporate factors such as user learning history and social engagements. In addition, the latent features learned through matrix factorization further improves the recommendation accuracy. Empirical evaluations on real world data demonstrate the performance of our model.

References

[1] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational statistics & data analysis*, 52(1):155–173, 2007.

[2] J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proc. of the 14th Conference on Uncertainty in Artificial Intelligence*, 1998.

[3] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Visualization of navigation patterns on a web site using model-based clustering. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 280–284. ACM, 2000.

[4] D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 187–220, 1972.

[5] A. Dahotre, Y. Zhang, and C. Scaffidi. A qualitative study of animation programming in the wild. In *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '10, pages 29:1–29:10, New York, NY, USA, 2010. ACM.

[6] S. Dasgupta, W. Hale, A. Monroy-Hernández, and B. M. Hill. Remixing as a pathway to computational thinking. In *ACM Conference on Computer-Supported Cooperative Work and Social Computing*, pages 1438–1449. ACM Press, 2016.

[7] D. Fields, M. Giang, and Y. Kafai. Understanding collaborative practices in the scratch online community: Patterns of participation among youth designers. *To see the world and a grain of sand: Learning across levels of space, time, and scale: CSCL 2013 Conference Proceedings*, 2013.

[8] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.

[9] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data*, 4(1):1–24, 2010.

[10] J. Lehmann, M. Lalmas, E. Yom-Tov, and G. Dupret. Models of user engagement. In *User Modeling, Adaptation, and Personalization*, pages 164–175. Springer, 2012.

[11] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. Silver, and Y. K. B. Silverman. Scratch: programming for all. *Communications of the ACM*, 52(11):60–67, 2009.

[12] C. Scaffidi and C. Chambers. Skill progression demonstrated by users in the scratch animation environment. *Int. J. Hum. Comput. Interaction*, 28(6):383–398, 2012.

[13] J. Wang and Y. Zhang. Opportunity model for e-commerce recommendation: right product; right time. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 303–312. ACM, 2013.

[14] S. Yang, C. Domeniconi, M. Revelle, M. Sweeney, B. Gelman, C. Beckley, and A. Johri. Uncovering trajectories of informal learning in large online communities of creators. In *Proceedings of the Second (2015) ACM Conference on Learning@ Scale*, pages 131–140. ACM, 2015.