Using Navigation to Improve Recommendations in Real-Time

Chao-Yuan Wu
Computer Science, UT Austin
cywu@cs.utexas.edu

Christopher V. Alvino
Netflix, Los Gatos, CA
calvino@netflix.com

Alexander J. Smola
CMU, Pittsburgh, PA
alex@smola.org

Justin Basilico
Netflix, Los Gatos, CA
jbasilico@netflix.com

ABSTRACT

Implicit feedback is a key source of information for many recommendation and personalization approaches. However, using it typically requires multiple episodes of interaction and roundtrips to a recommendation engine. This adds latency and neglects the opportunity of immediate personalization for a user *while* the user is navigating recommendations.

We propose a novel strategy to address the above problem in a principled manner. The key insight is that as we observe a user's interactions, it reveals much more information about her desires. We exploit this by inferring the within-session user intent *on-the-fly* based on navigation interactions, since they offer valuable clues into a user's current state of mind. Using navigation patterns and adapting recommendations in real-time creates an opportunity to provide more accurate recommendations. By prefetching a larger amount of content, this can be carried out entirely in the client (such as a browser) without added latency. We define a new Bayesian model with an efficient inference algorithm. We demonstrate significant improvements with this novel approach on a real-world, large-scale dataset from Netflix on the problem of adapting the recommendations on a user's homepage.

1. INTRODUCTION

Recommender systems are particularly important for services with large numbers of items. They rely on the promise that through effective algorithmic use of collected data, they can help the user discover novel content by presenting personalized, relevant items to the user [18]. Recommendation is fundamental to many places, such as news readers [5, 8], blogs [10], homepages [4], search engines [12], and streaming video [11]. Click models are often used to model the viewing behaviors and estimate intrinsic relevance [12, 5].

A key challenge in such approaches is that a user's intention can vary significantly between sessions in ways that cannot be captured by prior knowledge. For instance, movie consumption preferences are context-dependent: whether a viewer watches alone, with friends, with a romantic love interest, or with children. Likewise, they are situation dependent (dedicated viewing vs. background entertainment during a chore), depend on the available time (short episode vs. full movie) and can even depend on mood.

Context-aware recommendation approaches try to address this problem by leveraging supplemental information such as time of day or (geo)location to provide insight about the potential per-session user intent [20, 3]. However, contextual recommendations do not fully address the problem since they only offer a *prior* over the likely attributes using observable context prior to the start of a session. Likewise, the use of multiple viewer profiles such as those offered by Netflix, only captures a small facet of this information.

In short, it is difficult to predict a user's intention when recommendations are typically generated — *before* their session has begun. We argue that the problem of understanding intent is inherent to recommendation, that it is compounded in cold-start situations, and that observable context can only partially mitigate it. We propose a way to overcome this problem by using navigation behavior as instant implicit relevance feedback.

Relevance Feedback in Search

A wide variety of user interaction signals have been considered to understand user intent. For example, [14] studies interactions on touch-enabled devices, [19] studies modeling of mouse cursor movements, and [26] uses text selection events in search. In particular, recent work has studied eye gaze positions and mouse cursor movements, and shows that they are able to indicate user preference or attention [15, 16, 13, 23]. However, all work uses such data to infer *a posteriori* what the user's intent and interests might have been, rather than using it immediately.

In contrast to search systems, understanding the current user intent and using it effectively is even more pertinent in *lean back* recommendation experiences (e.g. Android TV, Playstation, Apple TV, Roku) where the primary mode of interaction is to navigate and select items. In contrast, in search, the user can rewrite queries if the results did not match their current intent, e.g. by specializing "apple" to "apple tree". The analysis of query chains [7] has been used extensively to improve relevance. Augmenting a page of recommendations *on the fly* in the way described in this paper gives the recommender system designer a chance to infer the

RecSys '16 September 15-19, 2016, Boston , MA, USA

ACM ISBN 978-1-4503-4035-9/16/09.

DOI: http://dx.doi.org/10.1145/2959100.2959174

current user intent, and thereby guide them towards items that better match their current desires.

Recently, the search engine Surf Canyon offered an extension that is able to alter the result page dynamically based on user's interaction with the page. Based on clicks in previous search queries, the system is able to alter results for *future* queries in the same session. [17] shows that with the dynamic adaptation users consume more results and spend a longer time searching. This strategy is natural for search engines since people can search multiple times for the same goal, and previous queries thus give valuable information.

We push this strategy to its logical conclusion — rather than waiting for a click to expand and clarify results, we can do this *while* the user is still exploring a page, i.e. we propose to dynamically *modify* the result page, based on a user's behavior on that page.

Context in Recommender Systems

Using context is a popular strategy for recommendation. Factorization-based linear models, for example, are among the most effective and popular recommendation approaches [21, 24]. A rich line of work has been proposed in jointly modeling multiple sources of information to alleviate these problems, such as ratings and item meta-data [1], text features [2], and reviews [9].

On the other hand, joint models of content consumption and navigation signals are still lacking in recommender systems. One challenge is that navigation signals are usually very noisy [23]. The data is also usually not publicly available. By working with a proprietary dataset from Netflix containing a large amount of logged navigation events, we are able to show that updating a page of recommendations based on within-session navigation behavior has the capability to help users find items of interest more effectively. Such updates can be done with negligible latency by computing within a browser *without* a roundtrip to a backend server.

Specifically, we propose to supply additional relevant recommendations by reordering rows of items, since each row typically contains a thematically coherent set of recommended content. Many online services use rows of recommendations to organize content and facilitate user navigation [6], though the technique can be adapted to other displays.

Probabilistic user model. Our model jointly captures navigation and consumption through a latent interest variable for the purpose of generating online recommendations based on the current user intent. To the best of our knowledge, this is the first principled attempt to capture both effects jointly.

Hybrid Inference. The algorithm uses both offline inference to learn static user preferences as well as online inference through Expectation Maximization in order to incorporate user navigational information as it becomes available.

Cold-start Experiments. We provide empirical validation that this approach can be effective for updating user recommendations on-the-fly by augmenting the recommendations on the Netflix homepage with better rows of thematically coherent videos. We also present experiments showing how the method is effective in dealing with user cold-start.

We support the model design through insights we obtained from the datasets in question. They make a clear case for advanced session modeling to improve user satisfaction and engagement. Our contributions are both *conceptual* in the form of instant navigation-based relevance feedback and also *statistical* in the form of a novel user interest model. Both are justified experimentally and can be used individually.

Figure 1: The user interface deployed on the Sony PS3 in the Netflix app in 2015. Note the distinct rows of movies in the UI. Each row is thematically or contextually coherent.

2. USER INTERFACE

Netflix is an online video streaming service that provides (e.g. on its homepage) a personalized collection of videos. In this paper, we consider the problem of optimizing the positions of these videos. The specific user interface (UI) considered is illustrated in Figure 1. A page is composed of a list of rows, each of which is composed of a list of videos. Videos in a row are topically or contextually coherent to ease user navigation (e.g. "TV Comedies" or "Popular on Netflix"). The UI has one row in focus at a time. A user can either scroll vertically to see other rows, scroll horizontally to see more videos in a row, or select a video to play. Depending on the device, typically one or more rows are displayed above or below the current selection.

We assume that we have a base recommender system that selects about 40 rows from all possible rows types, and each of which selects 75 personalized candidate videos from our full catalog. Our goal is to optimize the ranking of the rows and also the ranking of videos within each of them. While users navigate within a page, we want to model the navigation signals and perform real-time adaptation online to further improve the ranking of the rows.

While we conduct experiments on a dataset of homepages generated for a specific device UI, our model is completely general and can be directly applied to homepages on many devices, including websites, tablets, smartphones, and smart TVs. This model can also be applied to other applications that have similar structure to the row layout, such as Google Shopping, Google Express, YouTube, and Amazon.com. Also note that it is *not* our goal to design a novel UI layout but rather to demonstrate that implicit feedback from navigation can be incorporated in a principled fashion within a given, real-world UI.

3. MODEL

We now design a statistical model that addresses many aspects of user interaction with an online content recommendation service: First and foremost we need to predict play probabilities (Section 3.1). This is followed by a per-session user intent model (Section 3.2) and finally the full

graphical model (Section 3.3). The ranking procedure is described in Section 3.4. We also discuss some domain-specific user behavior we found (e.g. fatigue) and how to incorporate them for improved accuracy (Section 3.5).

Note that *the choice of the model is largely independent of the function class*. That is, we could use inner product recommenders, factorization machines, deep networks, decision trees or anything else. Instead, *the model is tightly coupled with the UI design*, since we aim to encode the way a user interacts with an application in quantifiable terms. Moreover, the insight of instant implicit feedback can be used in any model where the entire interface does not fit on a single display (e.g. maps, search results, messages) and the model can incorporate navigation data. We use the following symbols below:

Symbol	Definition
t_i	i-th video of a row
r_i	i-th row in a page
ρ_r	Row type of row r
ψ_*	Parameters for play probability model
v_*	Factorization-based parameters for interest model
w_*	Feature-based parameters for interest model
f_t	$\in \mathbb{R}^d_+$, vector of d features of video t
$f_r^{(row)}$	$\in \mathbb{R}^d_+$, vector of d features of row r
$\mathcal{S}$	$\in \{0,1\}$, scroll indicator
$\mathcal{C}$	$\in \{0,1\}$, play indicator
$\mathcal{I}$	$\in \{0,1\}$, interest indicator

3.1 Play Prediction

An accurate estimate of the play probability of each video is essential to optimizing pages full of video recommendations. We begin by considering the problem of estimating play probabilities of videos in one row at a time. Subsequently we generalize this model to incorporate user intent model on multiple rows. Some of our design choices are inspired by the "Fair and Balanced" model of [5]. That is, we use the notion of submodularity to estimate the relevance of any term, given the previously presented results. Note that a function f defined on a set X is submodular if it satisfies

$$f(X \cup \{A\}) - F(X) \leq f(X' \cup \{A\}) - F(X') \text{ for } X' \subseteq X.$$

In other words, the benefit from adding A to X is less than or equal to when we add it to a subset X'. Whenever equality holds throughout, we call f modular (i.e. linear in the parameters). In the context of movies this means that the added benefit from recommending a movie is not determined in isolation but in terms of how novel it is relative to the previous recommendations. This models typical user browsing behaviors to achieve diversity and personalization at the same time. To convert scores into probabilities we employ a logistic transform $\sigma(x) = \frac{1}{1+e^{-x}}$:

$$\mathbb{P}(\mathcal{C}_i = 1) = \sigma(g(t_{0:i}, \Psi)) \tag{1}$$

$$g(t_{0:i}, \Psi) = \langle \psi, f_{t_i} \rangle + \left\langle \tilde{\psi}, q(t_{0:i}) - q(t_{0:i-1}) \right\rangle \tag{2}$$

Here the coordinates of q are given by

$$[q\,(t_{0:i})]_j := h\left(\sum_{k=0}^{i} [f_{t_k}]_j\right)$$

where h is a suitably chosen concave function. The vectors ψ and $\tilde{\psi}$ denote modular and submodular parameters respectively. Each q_j is a submodular function of a set of videos that captures the diminishing returns effect of feature j. Intuitively, we assume users become "tired" of a certain feature if they see it in many videos. Taking this into consideration, maximizing play probability leads to a page that has diversity. Each $\tilde{\psi}_j$ models the strength of the diminishing return in each feature j.

Unlike [5], we have full access to a user's viewing behavior: Since the viewport of our user interface consists of only 5 titles, users can easily glance at all of them. Hence we do not need to model views as latent variables, which can be computationally costly.

For the purpose of personalization and to share statistical strength, we adopt a hierarchical decomposition of parameters: $\psi = \psi_0 + \psi_u + \psi_{\rho,0} + \psi_{\rho,u}$, where the the terms denote shared, user-specific, row-specific, and {row, user}-specific latent factors respectively. We use a similar decomposition for the submodular parameters $\tilde{\psi}$.

3.2 User Intent

While users generally have relatively stable tastes, they often have different intents in different sessions. This unobserved aspect is hard to estimate from general behavior but is much easier once the user interacts with a service. Failing to consider variations in user intent can lead to inaccurate recommendations, even when they match a user's overall taste profile. Here we describe our model that jointly models plays, navigation signals and user intents. A key distinction in this work is the ability to infer and respond to a specific user's preferences without any meaningful penalty in terms of latency and computational cost.

We assume that in each session a user is only interested in some subset of rows. We use $\mathcal{I}_{s,r} \in \{0,1\}$ to indicate a user's interest in row r of session s. As before we use a logistic transform to define a probability:

$$\mathbb{P}(\mathcal{I}_{s,r} = 1 | w, v, v_\rho) = \sigma(g'(r)) \tag{3}$$

$$\text{where } g'(r) = \langle w, f_r \rangle + \langle v, v_\rho \rangle\,. \tag{4}$$

Here $\langle w, f_r \rangle$ is linear with features f_r and parameters $w_r \in \mathbb{R}^n$, and $\langle v, v_\rho \rangle$ is factorization-based with row-type ρ based latent factors v_ρ and v. This captures latent co-consumption patterns that are not captured by features. Again, we decompose $w = w_0 + w_u + w_s$, where w_0 is shared, w_u is user-specific, and w_s is session-specific parameters, and similarly for $v = v_0 + v_u + v_s$.

We focus on horizontal scrolls as navigation signals. Figure 2 shows the play probability of a row, given different numbers of horizontal scrolls on that row. The fact that the play probability is *increasing* as a function of scrolls within a row suggests that scrolls are a rich source of information about user intent. The fact that the biggest increase occurs between zero and one scroll means that a binary scrolling indicator might suffice to capture most of the information about whether a user is interested in a row. We thus introduce $\mathcal{S}_{s,r} \in \{0,1\}$, which indicates whether a row r was scrolled. As before, we use a logistic transform to connect probabilities to scores:

$$\mathbb{P}(\mathcal{S}_{s,r} = 1 | \mathcal{I}_{s,r} = 1, \delta_{\rho_r}) = \sigma(\delta_{\rho_r}) \tag{5}$$

Note that $\mathbb{P}(\mathcal{S}_{s,r} = 1 | \mathcal{I}_{s,r} = 0, \delta_{\rho_r}) = 0$ by definition since users won't click on something they are not interested in ($\mathcal{I}_{s,r} = 0$). We use ρ_r to denote the row type of r and δ_{ρ_r} governs the likelihood of scrolling on each row type.

Similarly, the play probability of the i-th video of r follows

$$\mathbb{P}(\mathcal{C}_{s,r,i} = 1 | \mathcal{I}_{s,r} = 1, \Psi) = \sigma(g(t_{0:i}, \Psi)) \tag{6}$$

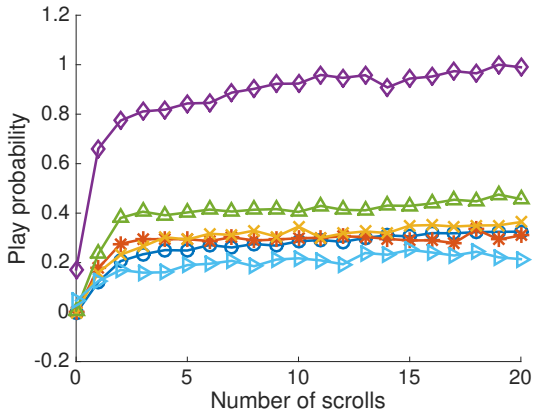

Figure 2: Play probability within a row as a function of videos scrolled to on that row. Each line represents a different row type. Remarkably the conditional play probability *increases* as the user delves into the list. To protect proprietary data we rescaled the numbers $[0, 1]$ so that only the relative magnitudes can be compared.

and likewise $\mathbb{P}(\mathcal{C}_{s,r,i} = 1|\mathcal{I}_{s,r} = 0, \Psi) = 0$. Here $g(t_{0:i}, \Psi)$ is as defined in Equation (2). Again, we assume users only play videos from the rows that they are interested in.

3.3 Graphical Model

Our design choices are best illustrated in the graphical model of Figure 3. Throughout we assume that all the parameters are drawn i.i.d. from normal distributions, since they are merely auxiliary latent variables used to capture the randomness in the observed user behavior. The generative process is as follows:

1. Sample $v_0 \sim \mathcal{N}(0, \lambda_v I)$ and $w_0 \sim \mathcal{N}(0, \lambda_w I)$.
2. For each row type ρ,
   (a) Sample $\delta_\rho \sim \mathcal{N}(0, \lambda_\delta)$.
   (b) Sample $v_\rho \sim \mathcal{N}(0, \lambda_v I)$.
3. For each user u,
   (a) Sample $v_u \sim \mathcal{N}(0, \lambda_v I), w_u \sim \mathcal{N}(0, \lambda_w I)$.
   (b) For each session s of u,
      i. Sample $v_s \sim \mathcal{N}(0, \lambda_v I), w_s \sim \mathcal{N}(0, \lambda_w I)$.
      ii. For each row r with row type ρ:
         A. Sample $\mathcal{I}_{s,r}$ via (3).
         B. Sample $\mathcal{S}_{s,r}|\mathcal{I}_{s,r}$ via (5).
         C. For i-th video in r sample $\mathcal{C}_{s,r,i}$ via (6).

Figure 3 provides the plate notation of this model. For conciseness we set $\Theta_{(*)} := \{v_{(*)}, w_{(*)}\}$. The plates inside a session are understood to be rows, and the plates inside the rows correspond to videos. In other words, we have a principled mapping of the user interface hierarchy directly into a graphical model.

3.4 Online Page Adaptation

Our goal is to optimize both positions of rows on a page and videos within a row such that the user finds interesting content with minimal effort. It follows from (1) that the play probability is maximized when $g(t_{0:i}, \Psi)$ is maximized. Thus, our goal is to find the set of videos that maximizes total gain $\sum_i g(t_{0:i}, \Psi)$. Note that by telescoping with (2),

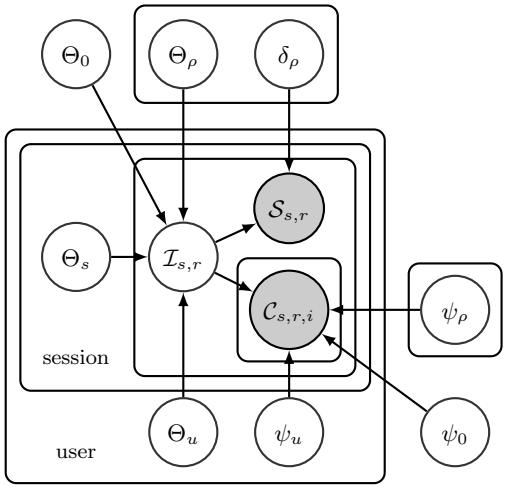

Figure 3: User interaction model. Shaded circles correspond to observed variables, namely play indicators and scroll indicators. For clarity of illustration we omitted the associated hyperparameters from the diagram. Θ models interest and ψ models the play probabilities.

we can see that this sum is a submodular function. Here we use a greedy procedure to optimize this sum:

$$t_i = \operatorname*{argmax}_{t \in \mathcal{T}_r} g(t_{0:i-1} \cap t, \Psi)$$

where $\mathcal{T}_r$ is the set of candidate videos in a row. Properties of submodularity tell us that this greedy procedure guarantees $(1 - \frac{1}{e})$-optimality.[1]

Similarly, to ease navigation we rank the rows according to the probability of play for any of the videos in the row. So the j-th row is selected by

$$r_j = \operatorname*{argmax}_{r \in \mathcal{R}_s, r \notin r_{0:j-1}} \mathbb{P}(\mathcal{I}_r = 1|v, w) \sum_i \mathbb{P}(\mathcal{C}_{r,t_i}|\mathcal{I}_r = 1). \quad (7)$$

where $\mathcal{R}_s$ is the set of rows preselected for session s.

Since we cannot predict user intent beforehand, we infer it on the fly as we observe navigation signals online to update our estimation of session parameters in real time. That is, we first present the page with Θ_s sampled from prior, and keep updating the posterior as we observe more navigation signals for session s. From (7), we can see this in turn updates $\mathbb{P}(\mathcal{I}_r = 1|v, w)$ and thus the rank of the rows. A detailed description about updating v_s and w_s is presented in Section 4.2.

Users expect a video to be fixed in a position once they see it. Similar behavior can also be found in the information retrieval literature: [25] shows that change of search results can hinder users finding what they are looking for. *Therefore, to provide a consistent experience to users, we fix a row once it is seen and only rearrange the rows that a user has yet to see.* This still provides great opportunity for improvement since when a user has not played any videos in the observed portion, the user is likely to consume from the unobserved part. Experiments show that already after

[1]Note that the gain is transformed by a logistic function, hence submodularity of click probabilities is not guaranteed. That said, maximizing the total gain is efficient in practice.

observing navigation signals from only 2 rows, online adaptation can improve the result quality considerably.

We believe that dynamic result adaptation is widely applicable. Firstly, the computation tends to be fairly lightweight and can be carried out on a client device. Hence it does not add any meaningful latency caused by another roundtrip to a server. Moreover, the same strategy could be applied to retrieving and presenting search results, or to games, where the environment is being generated dynamically in accordance to the player's interest and abilities.

3.5 Impression fatigue and repeated plays

Beyond a generic relevance and diversity model for recommendation, we also need to take into account impression fatigue and repeated plays. That is, we need to control for the fact that a user might have seen a recommendation previously or watched the movie before on the service.

We observe significant impression fatigue in our dataset, similar to what is described in [3]. That is, the more a video is presented to a user, generally the less likely the user plays it (when ignoring repeated plays): Repeated display is *not causal* in reducing the play probability, as is evident from the fact that previous plays are a conditioning variable.

As can be seen in Figure 4, most videos have stronger than exponential fatigue effects. It is also worth noting that for less popular videos, e.g. the lowest two curves in Figure 4, repeated impressions initially *do* increase play probability. We conjecture that it is due to "advertisement" effects: as we repeatedly present a less-known video to a user, a user becomes aware of and possibly even interested in the video. For already well-known videos, this effect is relatively minor and dominated by fatigue effects. Also, most curves can be approximated by piecewise linear functions in log-probability space. We thus add the following term to (2):

$$\text{fatigue}(x_t) = \begin{cases} a_t x_t + b_t & \text{if } x_t < k. \\ a_t k + b_t + d_t(x-k) & \text{otherwise} \end{cases} \tag{8}$$

Here $a_t, b_t, d_t \in \mathbb{R}$ control slope, offset and secondary slope of the fatigue function. $k \in \mathbb{Z}^+$ determines the position of the slope change, and x_t is the number of previous impressions. As before we can decompose a_t (similarly for b_t, d_t) into $a_{u,t} + a_{0,t}$ to prevent overfitting.

Also of interest is the high probability of repeated play. This applies to both episodic content (e.g. TV shows), for which many users continue watching subsequent episodes after watching some of them *and*, quite surprisingly, to standalone videos (e.g. movies). See Figure 5 for details.

Nonetheless, the curves clearly form two groups, one for episodic videos (with a higher repeated-play probability) and one for standalone ones. The curves within each group are very similar. This suggests that binary indicators of previous play and episodic videos are sufficient (see Section 5.4).

4. INFERENCE

Given this rather complex statistical model, we need to design efficient inference algorithms for both offline and online adaptation of the model. In particular, the online updates need to be very lightweight to run on consumer devices.

We choose the negative log-posterior of the data such that we can obtain a maximum-a-posteriori (MAP) estimate.

$$L := -\log \mathbb{P}(\mathcal{S}, \mathcal{C}|\text{rest}) + \Omega(\Theta, \Psi, \delta), \tag{9}$$

Here the first term maximizes the likelihood while the second term is a regularizer that penalizes complex models.

$$\log \mathbb{P}(\mathcal{S}, \mathcal{C}|\text{rest}) = \sum_s \sum_r \log \mathbb{P}(\mathcal{S}_{s,r}, \mathcal{C}_{s,r,:}|\text{rest}) \tag{10}$$

$$\mathbb{P}(\mathcal{S}_{s,r}, \mathcal{C}_{s,r,:}|\text{rest}) = \sum_{j\in\{0,1\}} \mathbb{P}(\mathcal{I}_{s,r} = j|\Theta)\mathbb{P}(\mathcal{S}_{s,r}|\mathcal{I}_{s,r} = j, \delta) \prod_i \mathbb{P}(\mathcal{C}_{s,r,i}|\mathcal{I}_{s,r} = j, \Psi) \tag{11}$$

Unfortunately the summation over inspection states is inside the logarithm, rendering the problem non-convex. For tractability, we design an efficient EM procedure using variational inference.

4.1 Offline training

The key to offline training is that if $\mathcal{I}_{s,r}$ is known then the full model decomposes nicely. Thus, we propose a stochastic EM algorithm to efficiently solve this inference problem. In the E-step we estimate $\mathcal{I}_{s,r}$ with all other parameters fixed, and in the M-step we optimize all parameters with $\mathcal{I}_{s,r}$ fixed.

E-step Here we estimate the posterior probability of $\mathcal{I}$ with fixed $\{\Phi, \Theta, \delta\}$. We define

$$\begin{aligned} Q(\mathcal{I}_{s,r})_j :=& \mathbb{P}(\mathcal{I}_{s,r} = j|\mathcal{S}_{s,r}, \mathcal{C}_{s,r,:}) \\ \propto& \prod_i \mathbb{P}(\mathcal{C}_{s,r,i}|\mathcal{I}_{s,r} = j, \Psi)\mathbb{P}(\mathcal{S}_{s,r}|\mathcal{I}_{s,r} = j, \delta) \\ & \mathbb{P}(\mathcal{I}_{s,r} = j|\Theta) \end{aligned} \tag{12}$$

M-step Next we optimize Ψ, Θ, and δ with $\mathcal{I}_{s,r}$ weighted by the posterior $Q(\mathcal{I}_{s,r})_j$. Define

$$J(Q, \Psi, \Theta, \delta) := \sum_{j\in\{0,1\}} Q(\mathcal{I}_{s,r})_j \log \frac{\mathbb{P}(\mathcal{I}_{s,r} = j, \mathcal{S}_{s,r}, \mathcal{C}_{s,r,:})}{Q(\mathcal{I}_{s,r})_j}$$

$$\{\Psi, \Theta, \delta\} = \operatorname*{argmin}_{\Psi,\Theta,\delta} -J(Q, \Psi, \Theta, \delta) + \lambda \|(\Psi, \Theta, \delta)\|_2^2$$

In experiments we use stochastic gradient descent for offline inference. It randomly picks a session from training set and performs a gradient step with gradient

$$\partial_\Psi J = \partial_\Psi \sum_{j\in\{0,1\}} Q(\mathcal{I}_{s,r})_j \log \mathbb{P}(\mathcal{C}_{s,r,:}|\mathcal{I}_{s,r} = j, \Psi)$$

$$\partial_\delta J = \partial_\delta \sum_{j\in\{0,1\}} Q(\mathcal{I}_{s,r})_j \log \mathbb{P}(\mathcal{S}_{s,r}|\mathcal{I}_{s,r} = j, \delta)$$

$$\partial_\Theta J = \partial_\Theta \sum_{j\in\{0,1\}} Q(\mathcal{I}_{s,r})_j \log \mathbb{P}(\mathcal{I}_{s,r} = j|\Theta)$$

We can view the updates with regard to Ψ as videos being weighted by $Q(\mathcal{I}_{s,r})_j$. This matches intuition: a play/non-play decision made on videos from a row that a user is interested in should be more important than a decision made on other rows. For example, when a user is looking for an exciting movie, we should not penalize a video from "Kids' Movies" too much when it is not played. Experiments show that by modeling interests, we are able to obtain a more accurate estimates of play probabilities.

4.2 Online Updates

We now proceed to describe the updates in a per-session context. Note that online does not refer to online learning in the traditional sense but rather to an online response and parameter update due to user behavior in the current

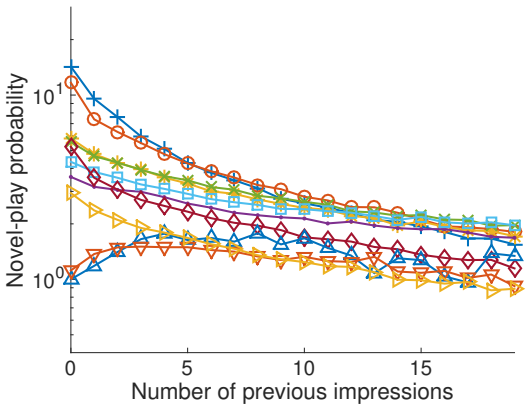

Figure 4: Novel-play probability as a function of non-play impressions for 10 videos. With exception of some rather unpopular videos the log-popularity decreases piecewise linearly. We renormalize them to provide relative magnitudes (the absolute numbers are proprietary).

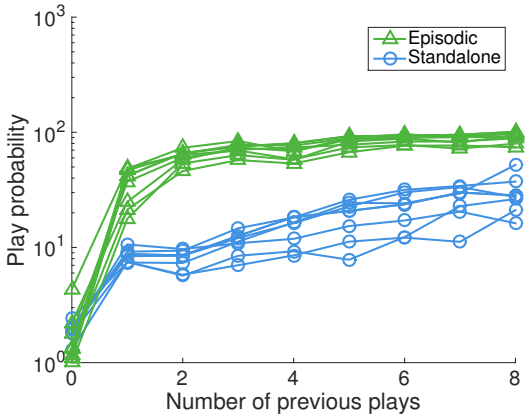

Figure 5: Play probability as function of number of previous plays. *Episodic* refers to content with multiple episodes, *Standalone* defaults to a single video (or movie). As before, probabilities are renormalized to protect proprietary information.

session. When a user scrolls or skips to the next row without scrolling we update the MAP estimate of Θ_s accordingly:

$$\mathbb{P}(\Theta_s|\text{rest}) \propto \prod_{r\in R_{\text{seen}}} \sum_{j\in\{0,1\}} \mathbb{P}(\mathcal{S}_{s,r}|\mathcal{I}_{s,r}=j,\delta) \quad (13)$$
$$\mathbb{P}(\mathcal{I}_{s,r}=j|v_s,w_s,\text{rest})\mathcal{N}(v_s|\lambda_v)\mathcal{N}(w_s|\lambda_w).$$

Here R_{seen} is the set of displayed rows. We adopt a similar EM strategy. The E-step remains unchanged, while in the M-step we make an update with gradient

$$\partial_{\Theta_s} J(Q,\Psi,\Theta,\delta) = \partial_{\Theta_s} \sum_{j\in\{0,1\}} Q(\mathcal{I}_{s,r})_j \log \mathbb{P}(\mathcal{I}_{s,r}=j|\Theta)$$

Once we update our estimation on Θ_s and in turn $\mathcal{I}_{s,r}$, we can then adapt the page. This is computationally inexpensive because we only need to reorder a small (typically $<$ 40) subset of rows. This is easily feasible by most modern browsers and devices. Most of the terms in (12) and (13) can be precomputed, and computing (and updating) $\mathbb{P}(I_{s,r})$ only involves a logarithm and an inner product of sparse feature vectors.

5. EXPERIMENT

We use a real-world dataset from Netflix and show that:

1. Online adaptation improves recommendations at a row level in both mean reciprocal rank (MRR) and Precision@5. (Section 5.2)
2. Our model is able to address cold-start. Even for users with *no* previous data and using *no* context, our model provides personalized recommendations. (Section 5.3)
3. Modeling impression fatigue and repeated play behaviors significantly improves recommendations in both MRR and Precision@5 (Section 5.4).

Comparing the performance with other production systems or other sophisticated models is out of the scope of this paper, since after all, our goal is to investigate the possibility of real-time online updates and to explore new methods to jointly model plays and navigation signals.

5.1 Dataset

The dataset consists of homepage sessions, collected by Netflix for a particular type of playback device from 57,386 distinct profiles pre-computed for members from a single country. For each video on the page, the dataset also contains an indicator about whether the video was displayed to the user, and whether it was played. The dataset consisted of sessions collected over a three-month period in 2015. Sessions from April and May are used for training, while those from June are split into validation (for hyperparameter tuning) and test sets (to report performance).

We are interested in cases where users are navigating the page to discover new content to watch instead of continuing to watch previously watched shows. We thus filter out sessions where users choose from *Continue Watching*, *Recently Watched* or *My List* rows. This leaves approximately 294k training and 59k testing sessions.

Each homepage in the dataset consists of 40 rows that were chosen according to a production personalized recommender system that selects and orders the rows in a personalized fashion for each profile. Thus, for this dataset, the method in this paper already has the space of rows filtered down to a set that are deemed to be relevant. Each row consists of up to 75 videos placed horizontally in the row. The videos are placed in what appears to be a two-dimensional grid, but, in fact, a page is a series of rows on which the user has the ability to scroll individually in a horizontal fashion. When a row comes into the viewport, five videos appear horizontally. In light of this, we only consider a row to be *scrolled on* when a user has viewed more than five videos horizontally in that row.

While a comparison using publicly available datasets would be ideal, there are no such datasets available that form a page of recommendations that are comparable to the one described above. Instead, the purpose of this paper is to present a novel method and show the improvement in relevance metrics with the online adaptation strategy.

5.2 Online Updates

We examine the contribution of the online adaptation

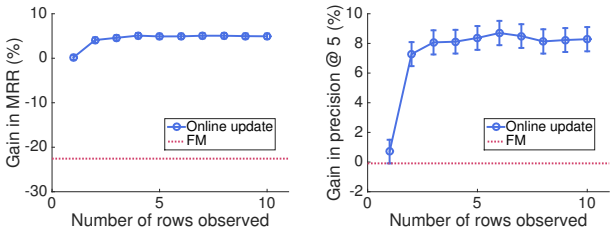

Figure 6: Gains in MRR (left) and Precision@5 (right) as a function of the number of rows visited. We see a clear improvement even just after 2 rows. The Factorization Machine baseline is provided for reference.

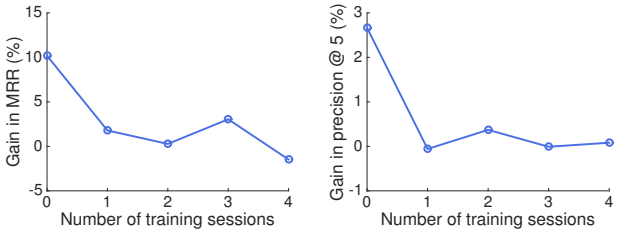

Figure 7: Gain in MRR (left) and Precision@5 (right) compared to offline-only model. For users with few training sessions, the benefit gained from online update is the greatest.

strategy. We first generate pages with the model trained using only previous sessions. During testing, we simulate the online adaptation procedure by looking at one row at a time, and update the page according to navigation signals (scroll or no scroll). To control the causal effects, we must use navigation signals from rows in the order from the original page instead of the reordered page. We thus evaluate using the following procedure: only use navigation signals up to a certain row (the 10th row in experiments) on a page, and only reorder the rows below that row. We then compare the adapted pages with the pages without adaptation.

We evaluate performance by two standard metrics: mean reciprocal rank (MRR) and precision at 5 (P@5). MRR measures how early we place the relevant (played) rows on a page. P@5 measures the fraction of relevant rows in the next 5 rows. Figure 6 shows MRR and P@5 vs. number of rows observed. We see on both metrics that online adaptation provides a significant improvement over non-adapted pages. Not surprisingly, as we observe more rows, online adaptation provides greater improvements. We already see improvements with only 2 rows. This means users only need to scroll 2 rows to have a benefit from this strategy, which is often the case when users are exploring for new videos.

Without a known row ordering baseline, we compared against ordering the rows on the page via Factorization Machines (FMs) using libFM [22]. The FM was trained and tested on the same data as our model. For features we used: user ID and row type identifier; we sampled played rows as positives and unplayed rows within one row of the played rows as negatives. The best FM model was trained with MCMC optimization and latent dimension 15. We show relative MRR and P@5 metrics in Figure 6.

5.3 Coldstart

The coldstart problem is the bane of many recommender systems. It occurs whenever a new user or a new video comes to the service. In such cases, we have relatively little data to infer the preferences of the user or the properties of the video: For new users we often only have basic context information such as device type and location. This makes recommending an initial set of videos exceedingly difficult, yet it is this first impression that matters a lot when it comes to convincing a new user to continue using a service.

Our online adaptation strategy helps alleviate the user coldstart problem. Figure 7 shows MRR and P@5 scores as a function of the number of training sessions available. We see that users with fewer training sessions (coldstart users) benefit most from online adaption. Quite remarkably, it works well even for users with no previous sessions. This opens a new avenue to address the coldstart problem: even when there is *no* previous data and using *no* context information, with online modeling navigation signals, we are able to personalize recommendations within the first page. We are effectively observing the user in a live environment and adjusting recommendations to reduce the amount of navigation needed to find something interesting to watch.

5.4 Fatigue

We discussed the importance of impression fatigue and repeated viewing in Section 3.5. To illustrate the effect we examine the horizontal MRR and P@5 within a row to evaluate the play probability estimates. Relative improvements compared to the model without them are shown in Figure 8. We can see that each provides clear improvement and using a combination is best. This gain is achieved by only using a small number of model parameters.

6. CONCLUSION AND FUTURE WORK

We proposed inferring user intent through in-session navigation information and then updating pages of recommendations based on that intent. To accomplish this, we defined a probabilistic model capable of incorporating current session navigational information as well as logged navigation and consumption data. The model infers interest in candidate rows of recommended items based on homepage navigation to a certain point on the page, and can populate the remaining rows on the page with more relevant content that reflects the current likely intent as predicted by the model.

This is the first work of its kind that performs online updating of recommendations based on user navigation within a single page. Thus, it is the intention of this paper to demonstrate feasibility of such a method, which we do on real-world dataset from Netflix. The model not only improves the relevancy of future recommendations on the page, but also helps to alleviate the user coldstart problem by using navigation information, yielding improved recommendations even within the first page. For users without any offline training sessions, i.e. with online navigation signal alone, we are able to provide personalized recommendations. We argue that this approach addresses the coldstart problem in a more effective manner than traditional approaches, though could be combined with them as well.

This work opens a new avenue for future work on online page adaptation. While we use binary horizontal scrolls in experiments, the online adaptation strategy is completely general. We can easily change the emission function of navigation signals to fit navigation signals in many domains, such as mouse hover, section expansion, or text selection.

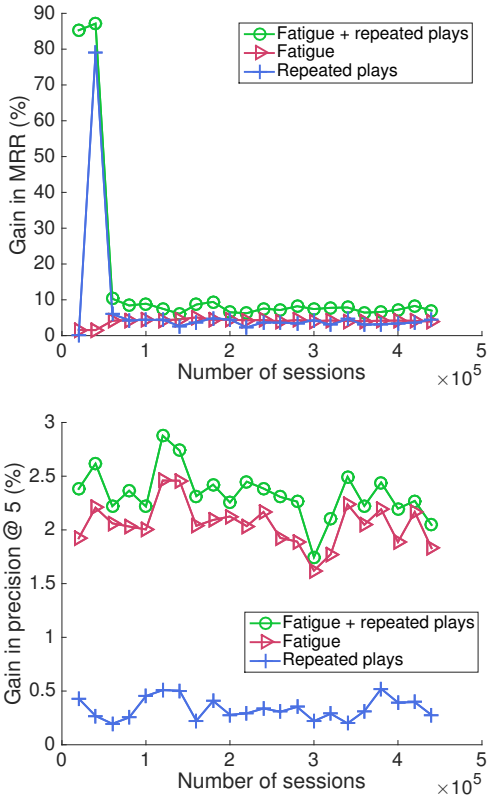

Figure 8: Gain in MRR (Top) and Precision@5 (Bottom) obtained by modeling impression fatigue and repeated plays. Performance compared to the model without modeling impression fatigue and repeated plays.

Extending it to model multiple sources of signals is also straightforward. Also, while we focus on reranking rows, a similar model could also be applied to reranking items within each row. It can likewise be adapted to model other UI designs.

7. REFERENCES

[1] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *KDD*, pages 19–28. ACM, 2009.

[2] D. Agarwal and B.-C. Chen. FLDA: matrix factorization through latent Dirichlet allocation. In *WSDM*, pages 91–100. ACM, 2010.

[3] D. Agarwal, B.-C. Chen and P. Elango. Spatio-temporal models for estimating click-through rate. In *WWW*, pages 21–30. ACM, 2009.

[4] D. Agarwal, B.-C. Chen, P. Elango, N. Motgi, S.-T. Park, R. Ramakrishnan, S. Roy, and J. Zachariah. Online models for content optimization. In *NIPS*, pages 17–24, 2009.

[5] A. Ahmed, C.-H. Teo, S.V.N. Vishwanathan, and A. J. Smola. Fair and balanced: Learning to present news stories. In *WSDM*, pages 333–342, 2012. ACM.

[6] C. Alvino and J. Basilico. Netflix tech blog: Learning a personalized homepage. http://techblog.netflix.com/2015/04/learning-personalized-homepage.html.

[7] P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna. The query-flow graph: model and applications. In *CIKM*, pages 609–618. ACM, 2008.

[8] A. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *WWW*, pages 271–280. ACM, 2007.

[9] Q. Diao, M. Qiu, C.-Y. Wu, A.J. Smola, J. Jiang, and C. Wang. Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS). In *KDD*, pages 193–202. ACM, 2014.

[10] K. El-Arini, G. Veda, D. Shahaf, and C. Guestrin. Turning down the noise in the blogosphere. In *KDD*, pages 289–298. ACM, 2009.

[11] C.A. Gomez-Uribe and N. Hunt. The Netflix recommender system: Algorithms, business value, and innovation. *ACM TMIS*, 6(4), 2015.

[12] F. Guo, C. Liu, A. Kannan, T. Minka, M. Taylor, Y.-M. Wang, and C. Faloutsos. Click chain model in web search. In *WWW*, pages 11–20. ACM, 2009.

[13] Q. Guo and E. Agichtein. Beyond dwell time: estimating document relevance from cursor movements and other post-click searcher behavior. In *WWW*, pages 569–578. ACM, 2012.

[14] Qi Guo, H. Jin, D. Lagun, S. Yuan, and E. Agichtein. Mining touch interaction data on mobile devices to predict web search result relevance. In *SIGIR*, pages 153–162. ACM, 2013.

[15] J. Huang, R. White, and G. Buscher. User see, user point: gaze and cursor alignment in web search. In *SIGCHI*, pages 1341–1350. ACM, 2012.

[16] J. Huang, R. White, and S. Dumais. No clicks, no problem: using cursor movements to understand and improve search. In *SIGCHI*, pages 1225–1234, 2011.

[17] J.Y. Kim, M. Cramer, J. Teevan, and D. Lagun. Understanding how people interact with web search results that change in real-time using implicit feedback. In *CIKM*, pages 2321–2326. ACM, 2013.

[18] J.A. Konstan and J. Riedl. Recommender systems: from algorithms to user experience. *User Model. User-Adapt. Interact.*, 22(1-2):101–123, 2012.

[19] D. Lagun, M. Ageev, Q. Guo, and E. Agichtein. Discovering common motifs in cursor movement data for improving web search. *WSDM*, pg. 183–192, 2014.

[20] H. Lieberman and T. Selker. Out of context: Computer systems that adapt to, and learn from, context. *IBM Systems Journal*, 39:617–632, 2000.

[21] S. Rendle. Factorization machines. In *ICDM*, pages 995–1000. IEEE, 2010.

[22] S. Rendle. Factorization Machines with libFM. *Trans. Intell. Syst. Technol.*, 3(3), pages 57:1–22. ACM, 2012.

[23] K. Rodden, X. Fu, A. Aula, and I. Spiro. Eye-mouse coordination patterns on web search results pages. In *CHI*, pages 2997–3002. ACM, 2008.

[24] R. Salakhutdinov and A. Mnih. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *ICML*, pages 880–887. ACM, 2008.

[25] J. Teevan, E. Adar, R. Jones, and M.A. Potts. Information re-retrieval: repeat queries in yahoo's logs. In *SIGIR*, pages 151–158. ACM, 2007.

[26] R.W. White and G. Buscher. Text selections as implicit relevance feedback. In *SIGIR*, pages 1151–1152. ACM, 2012.