Personalizing LinkedIn Feed

Deepak Agarwal, Bee-Chung Chen, Qi He, Zhenhao Hua, Guy Lebanon, Yiming Ma,
Pannagadatta Shivaswamy, Hsiao-Ping Tseng, Jaewon Yang and Liang Zhang
LinkedIn Corporation
Mountain View, CA, USA
{dagarwal,bchen,qhe,zhhua,glevanon,yma,pshivasw,htseng,jeyang,lizhang}@linkedin.com

ABSTRACT

LinkedIn dynamically delivers update activities from a user's interpersonal network to more than 300 million members in the personalized feed that ranks activities according their "relevance" to the user. This paper discloses the implementation details behind this personalized feed system at LinkedIn which can not be found from related work, and addresses the scalability and data sparsity challenges for deploying the system online. More specifically, we focus on the personalization models by generating three kinds of affinity scores: Viewer-ActivityType Affinity, Viewer-Actor Affinity, and Viewer-Actor-ActivityType Affinity. Extensive experiments based on online bucket tests (A/B experiments) and offline evaluation illustrate the effect of our personalization models in LinkedIn feed.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

Keywords

Personalization; Feed Relevance; Large Scale Learning

1. INTRODUCTION

More than 300 million people participate on LinkedIn[1]. By introducing the feed function, LinkedIn dynamically delivers update activities from a user's interpersonal network, ensuring a user always has fresh content to digest at login. Popular update activities include "member shares articles", "company posts jobs", "member connects to member", "member likes another activity", leading to a classification of update activities as (actor, verb, object) triplets.

Due to the large volume of activities in feeds and the fact that the users have limited time to interact with their social network, many users experience information overload. To ensure delivering meaningful content to the user, LinkedIn has introduced a personalized feed that ranks activities according their "relevance" to the user.

[1]https://press.linkedin.com/site-resources/news-releases/2015/linkedin-announces-fourth-quarter-and-full-year-2014-results

KDD'15, August 11 - 14, 2015, Sydney, NSW, Australia.

DOI: http://dx.doi.org/10.1145/2783258.2788614.

Users can play two roles simultaneously in LinkedIn's feed: *actors* who are users that produce feed activities and *viewers* who are users that consume feed activities. The *actor* label also includes other content producer entities such as companies, groups, schools, publishing channels, etc. Classic research work in personalized feed recommendation represents and synthesizes a viewer's profile from three aspects [13, 5]: the viewer-to-actor relationship strength, the correlation of activity content to viewer interests, and the underlying social network. However, personalizing the LinkedIn feed brings new challenges because of the following reasons.

- *Scalability*: Given the vast amount of users who visit LinkedIn at any given time, scalability becomes one of the major concerns for personalization. For each viewer visit, computing personalized scores between the viewer and tens of thousands of candidate activities in **real-time** using features such as viewer-actor profile similarity or past interactions between the viewer and actor is time consuming and may cause latency problems.
- *Data Sparsity*: Not all LinkedIn members have significant historical interaction data. Even for "heavy" viewers, the total amount of interactions is skewed towards a small fraction of highly active actors and popular activity types. Modeling rare events to compute personalized scores is often unreliable and noisy [3].

This paper describes our experiences in addressing these challenges at LinkedIn and our work on the personalized production feed. Our previous work [4] has already shown that a simple feed ranking model that considers activity freshness, feed diversity, and viewer-actor connection relationships significantly outperforms the time feed (reverse chronological ordering of feed activities) and ranking by social popularity in terms of click-through rate (CTR) – the probability of a viewer clicks on an activity. In this paper, we describe our continued modeling efforts on the LinkedIn feed with a focus on personalized models.

We consider three types of signals for personalization in this paper: a) Viewer-ActivityType Affinity that models how likely a viewer tends to interact with an activity type, e.g. does an individual working in IT industry like to click on job recommendations? b) Viewer-Actor Affinity which models how likely a viewer tends to interact with an actor. Note that although people tend to interact more with their close friends, sometimes they also like to interact with other types of actors, such as influencers (e.g. CEO of a company), big companies, or good publishing channels, etc. c) Viewer-Actor-ActivityType affinity which is the most granular signal that models the three-way interactions among viewer, actor and the activity type.

Given the challenge of data sparsity, often simply using users' past interactions in LinkedIn feed is not good enough. Therefore

in this paper, we apply the feature-based approach which contains thousands of cold-start features from peoples' profiles to model the above three affinity signals. Our experimentation in production shows this approach works pretty well in terms of CTR performance, however it raises a scalability challenge as described above: Scoring a sophisticated model with many features for a large scale of candidate activities in real-time with low latency might be technically infeasible.

To address such scalability concern we divide our feature space into two parts: For features that are more dynamic, e.g. time of the day, they still remain in the online scoring system. For the features that are more stable and do not change rapidly over time, e.g. the three affinity scores mentioned above, we pre-generate them in Hadoop offline and then push them to production periodically. The online scoring system only needs to query by the viewer ID and obtain these scores for each (actor, activity-type) pair; no additional scoring needs to be done for generating these scores online, which saves a lot of computational time.

Though ideally the three affinity scores should be jointly trained, it is actually a significant challenge due to the high dimensionality of the parameter space and sparsity of the data. In this paper, we instead adopted a layman's approach by modeling them independently: We train the Viewer-ActivityType affinity first since it has the richest data. Then, the learned Viewer-ActivityType affinity is treated as a single compound feature and is fixed in the downstream actor-level affinity training.

Contributions: In this paper, we make the following contributions:

- We analyze large-scale LinkedIn data and summarize knowledge mined from massive data for personalized models in Section 3.
- We introduce our personalization modeling efforts for Viewer-ActivityType Affinity based on Gamma-Poisson model in Section 4.
- We introduce the Viewer-Actor Affinity modeling and the Viewer-Actor-ActivityType modeling in Section 5.
- We demonstrate how to deploy the pre-computed affinities online for production in Section 6, to tackle the scalability challenge.
- We report on extensive experiments based on online bucket tests (A/B experiments) and offline evaluation in Section 7 to evaluate our personalization models in LinkedIn feed.

2. RELATED WORK

Personalized feeds attracted more user attention compared to the chronologically ordered feed lists [6, 7] or the global relevance ordered feeds [4]. Hence major social networking sites including Facebook and LinkedIn provide users the personalized News feed as the main portal [1].

Early research in personalized feed recommendation is to rank homogeneous tweet feeds in Twitter [12, 2, 11, 13, 16, 26, 22, 10]. This line of work is essentially close to the earlier personalized news article recommendation work [14], except that they used additional social network features like social voting, social tie strength etc. to model the activity relevance. They provided hints on how various features result in various user preferences to the Tweet feeds. For example, [12] reported that content sources, topic interest of users, and social voting on items are three major considerations corresponding to user interests; [11] examined thread length, topic relevance and social tie-strength contribute to the relevance. To implement a personalized recommendation system, [2, 16, 26, 22] enriched user profiles with their activities, text information of tweets and their neighbors through collaborative filtering, and [10] produced different rankers for different sub-communities of a user's social network, where these rankers are described as topics so that the user can rank his/her social feeds by specific topics.

Researchers also studied the personalized recommendation for feeds with heterogeneous activity types [17, 25, 6, 7, 8, 28]. As different activity types support different kinds of actions (e.g., "share", "comment" on activities), some work leveraged user-action interests in the relevance of feed activities [17, 7, 6]. Researchers also investigated the personalized recommendation for heterogeneous enterprise network activity streams [19, 20]. They concluded that the activity stream based user profile is more productive than entity-based user profile for personalizing the stream.

All the above work paved the way for our studies in LinkedIn feed, enlightening our feature exploration and design. Our personalization efforts in computing viewer's affinity scores at activity type level and actor level essentially belong to user-to-content relationships and user-to-user relationships [5]. However, the above work has not been evaluated on top of large-scale production systems. Instead, we always stick to the feasibility of deploying in production when planning personalization models.

Related work on top of LinkedIn feed includes our earlier work [4] and [21]. In [4], we presented an overview of the end-to-end feed recommendation system in LinkedIn, focusing on the infrastructure and generic modeling framework. In [21], various machine learning techniques including linear models on features and latent factor models (matrix factorization and tensor factorization) were experimented in LinkedIn feed data, irregardless of the online deployment.

The novelty of this paper is to present the end-to-end personalization efforts on LinkedIn production feed and online bucket test results. Our product has been deployed in LinkedIn homepage to serve more than 300 million members for more than one year. One effort of this paper is to disclose the implementation details behind this real system, which can not be found from related work.

3. DATA ANALYSIS

In this section we give a brief overview of the LinkedIn feed and show some interesting characteristics of user behaviors on the LinkedIn feed. Beginning by introducing the LinkedIn feed, we show how different users consume LinkedIn content and interact with other users in different ways. The insights that we gain in this section motivate our personalization models that are developed in the following sections.

3.1 The LinkedIn Feed

Every activity in LinkedIn feed is represented as the triple: (actor, verb, object). Accordingly, the *type* of an activity, named as *activity type*, is a triple of (actor type, verb type, object type). Actor types include member, company, school, service, channel. Verb types are comment, connect, join, like, profile change, job change etc. Object types are article, job, member, picture etc. Various combinations of them consist of around 50 activity types on Linked feed. For example, member connects to member, company publishes article, and member updates the profile picture. More details about the taxonomy of LinkedIn feed activities can be found in [4].

Given a viewer, LinkedIn feed displays the activities that happened in the professional network of the viewer, such as activities done by the user's connections, activities done by the companies or the influencers followed by the viewer, and the news articles

that the viewer would be interested in. The goal of LinkedIn feed is to provide a personalized activity stream for each viewer, and there can be many metrics to quantify how good the feed is, such as how often the viewer clicks, how fresh the feed is, or how diverse the feed is [4]. In this paper, among various metrics, we focus on click-through rate (CTR) because click is the most immediate user reaction and is straighforward to quantify.

In the rest of this section, we study how viewers tend to click on activities from the log data. We focus on how the CTR is affected by two different relationships between the viewer and the activity. First is the activity type analysis about how viewers have different levels of interest to different types of activities. Second is the viewer-actor analysis about how viewers interact with activities depending on who the actor is.

3.2 Relationships to activity types

Viewers' responses to different activity types vary for two reasons.

- *Global bias*. Some activity types are inherently more popular than others. Figure 1 shows the CTR of different activity types relative to the average CTR of all types when we serve activities in random orders to the viewers. Each curve corresponds to a particular activity type (we cannot label the curves due to confidentiality). The figure shows that some activity types consistently tend to receive more actions than other types.
- *Personal preference*. Individual viewers have different level of interests to different types of activities. For example, some viewers are heavily interested in building connections (connection activities) while others are interested in information activities such as article sharing. Figure 2 shows a scattered plot of viewers' response to two different activity types. We randomly choose 500 LinkedIn viewers from a random bucket where all activities are randomly ordered, and measure their clicks to two different activity types during one month. We discard the users who have clicked less than 10 times on either types. Each point in Figure 2 corresponds to one of 500 viewers. Not only are the clicks on two activity types not positively correlated to each other, but also we observe a clear division between two groups of viewers, indicating that viewers who are interested in one type are not interested in the other type. In our internal analysis, we could classify two groups of viewers with 70% accuracy simply by their number of connections.

This analysis suggests that, to personalize feed, not only should activity type be used as a feature to remove global bias, but also we must consider individaual affinities to different activity types.

3.3 Relationships to actors

Whether or not they have thousands of contacts in LinkedIn, people are only able to keep up close relationships with a handful of people. Here we study how viewers respond to different actors depending on a few notable features of the viewer-actor pairs.

We look at the past interaction features (e.g., number of past clicks, number of past impressions etc.) between viewer and actor. Figures 3a and 3b illustrate the CTR between viewer and actor as a function of their number of past clicks/impressions. The plots are generated based on a set of random users for a few days on both mobile and desktop platforms. For both platforms, more past clicks between viewer and actor generally translate to higher CTR. However, for impressions, we observe different behaviors. On mobile, a viewer appears to be more tolerant to a repeat showing of the same actor, but on desktop, we observe CTR drop if the same actor showing up multiple times.

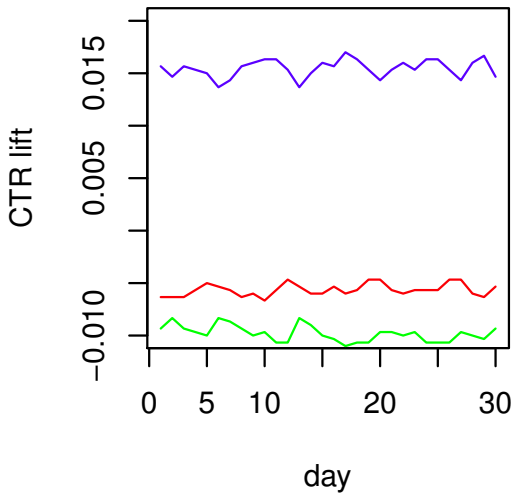


Figure 1: CTR lift of three different acitivity types per day.

Considering the past interaction features are not ubiquitous between viewer and actor, we also look at other features that are always available. Examples of such features include the profile of the viewer and the actor, the interactions in other LinkedIn services outside the feed, and social network features like the number of common connections. In the following we take the number of viewers' connections who have clicked on the actor in the past as the feature for analysis. This feature intuitively makes sense: for example, the more of your connections commented on an article, the more likely you would click on it (Homophily Principle). In fact, researchers have proved that using the connections' responses can improve cold-start recommendation [24]. Figure 4 shows how the CTR between viewer and actor on 4 different activity types positively correlates to the number of the viewer's connections who have clicked on the same actor.

The above two features are just the tip of the iceberg. There are numerous features discoverable in LinkedIn for user-to-user relationships, laying the foundations for the personalization between viewers and actors.

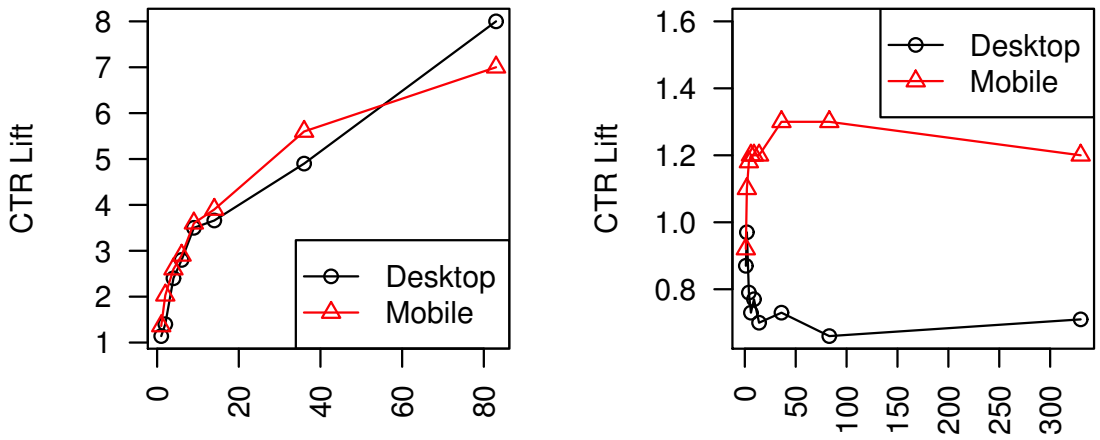
3.4 Discussion

Our data analyses uncover that in LinkedIn feed, not only does the CTR vary on various activity types, but also the CTR should be personalized by actors. In the following sections, we model two kinds of personalized affinities for LinkedIn feed: one is Viewer-ActivityType Affinity that manifests the viewer's topics of interests at the activity type level, and the other is Viewer-Actor Affinity that manifests the viewer's preference to the activity producer. Our personalization strategies are in line with findings of [13] that the source of the activities and the recipient's topics of interest are two major considerations which correspond to user interest.

4. VIEWER-ACTIVITYTYPE AFFINITY

There are about 50 different activity types[2] in LinkedIn feed. In this section, our objective is to compute an affinity score for each

[2]This number varies slightly from time to time as LinkedIn keeps producing new types or retiring old types.



(a) Clicks

(b) Impressions

Figure 3: Past interactions versus relative CTR lift.

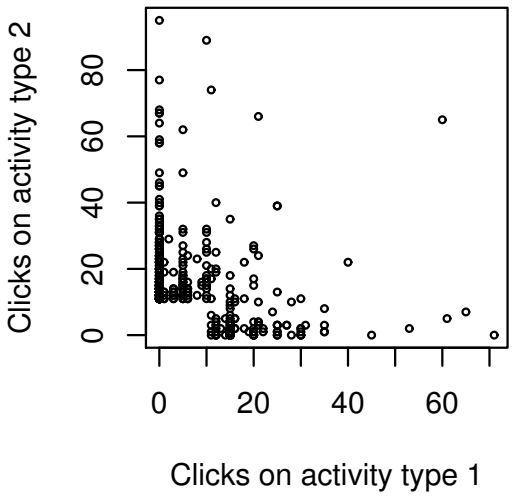


Figure 2: Scatter plot of clicks from 500 random viewers on two different activity types.

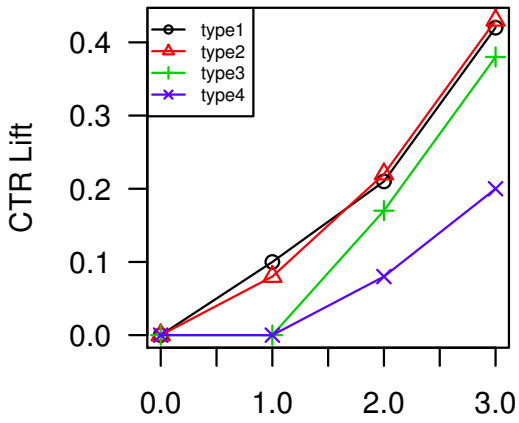


Figure 4: Relations between the relative CTR lift and the number of viewer's connections who have clicked on the same actor.

pair of viewer i and activity type k, which manifests the viewer's topics of interests at the activity type level. In total, we need compute around 300M x 50 affinity scores. We leverage CTR to approximate the affinity, which is defined as:

Definition 1. *Viewer-ActivityType Affinity α_{ik}, the likelihood score of viewer i clicks on the activity type k in LinkedIn feed.*

We describe how we compute the estimator $\hat{\alpha}_{ik}$ for α_{ik}. We are given a set of historical activity impressions $\{e = (y, i, k, m)\}$ where y is viewer i's click to impression m of activity type k. A naive approach to get $\hat{\alpha}_{ik}$ would be to use the CTR of viewer i for type k from a set of historical activity, *i.e.,* we let $\hat{\alpha}_{ik} = S_{ik}/M_{ik}$ where S_{ik} is the number of clicks and M_{ik} is the number of impressions from the viewer i to the activities of type k. However, this CTR is reliable only for large sample sizes, which is not true for computing affinity between viewer and activity type where the majority of viewers are light users with only a few historical impressions. Hence we aim to estimate α_{ik} by applying smoothing via hierarchical probabilistic models.

Denoting by p_e the true CTR for activity impression e with covariates $\mathbf{x}_e$, we assume the following decomposition:

$$p_e = b_e g_{ik}, \tag{1}$$

where b_e is a probability estimate computed from a baseline model which is a function of covariates $\mathbf{x}_e$, and g_{ik} is modeled as the CTR correction factor which is only a function of viewer and activity type pair (i, k) and does not depend on covariates $\mathbf{x}_e$. Logistic regression is a popular choice of the covariate-based baseline models in the literature [27], and is also applied in this paper:

$$\hat{p}_e = \frac{1}{1 + e^{-\boldsymbol{\theta}^T \mathbf{x}_e}} \cdot g_{ik}. \tag{2}$$

The covariates $\mathbf{x}_e$ include features that can remove position bias, remove user interface bias, profile viewers, and profile activities etc. Since the baseline model is based on features at the impression level, we have sufficient statistics to estimate b_e using logistic regression.

Now, given the baseline model, we have the expected number of clicks:

$$E_{ik} = \sum_{e \in i,k} b_e. \tag{3}$$

We assume that $S_{ik}|E_{ik}, g_{ik}$ follows Gamma-Poission distribution $S_{ik}|E_{ik}, g_{ik} \sim \text{Poisson}(E_{ik} g_{ik}), g_{ik} \sim \text{Gamma}(1, \gamma)$, since Poisson assumption is reasonable and has been widely used in rare event estimations [15].

As the correction factor, intuitively the value of g_{ik} depends on the O/E (observed-to-expected) ratio S_{ik}/E_{ik}: if $S_{ik}/E_{ik} > 1$, the baseline model under-estimates the CTR, so $g_{ik} > 1$ can force the expected number of clicks equals to the numebr of observed clicks; if $S_{ik}/E_{ik} = 1$, $g_{ik} = 1$ as no need to correct; if $S_{ik}/E_{ik} < 1$, $g_{ik} < 1$ as the baseline model over-estimates the CTR. According to the Gamma-Poisson conjugate property, the posterior of g_{ik} also follows the Gamma distribution with mean $\frac{\gamma + S_{ik}}{\gamma + E_{ik}}$ and variance $\frac{\gamma + S_{ik}}{(\gamma + E_{ik})^2}$. We simply approximate g_{ik} as the posterior mean of Gamma distribution:

$$\begin{aligned} g_{ik} &= \frac{\gamma + S_{ik}}{\gamma + E_{ik}} \\ &= \frac{\gamma + S_{ik}}{\gamma + \sum_{e \in i,k} \frac{1}{1 + e^{-\boldsymbol{\theta}^T \mathbf{x}_e}}}. \end{aligned} \tag{4}$$

Can we directly use $\hat{p}_e$ to model the personalized affinity α_{ik}? The answer is no because $\hat{p}_e$ is the estimated CTR binded to activity impression e, yet by definition the Viewer-ActivityType Affinity α_{ik} is impression independent. We extract a subset of the covariates $\mathbf{x}_e$ by only keeping (i, k)-dependent features and rename it as $\mathbf{x}_{i,k}$. Finally, the affinity α_{ik} is modeled as:

$$\hat{\alpha}_{ik} = \frac{g_{ik}}{1 + e^{-\boldsymbol{\theta}^T \mathbf{x}_{i,k}}}. \tag{5}$$

There are multiple advantanges of modeling α_{ik} by combining the CTR correction factor with the feature-based CTR. First is that, as we described above, CTR may not be reliable when data is sparse. Smoothing by Gamma-Poisson model mitigates the issues with data sparsity. Second, CTR may contain biases for several reasons: the same activity at different positions attracts different CTRs; the content information of an activity also affects the CTR. We use covariates $\mathbf{x}_e$ to address the effect coming from these factors. Third, it is impractical to discover all possible variates of the CTR function. Computing a baseline model based on covariates $\mathbf{x}_e$ first and then using g_{ik} to model the residue inherited from data is thus easier to implement.

5. VIEWER-ACTOR AFFINITY

5.1 Viewer-Actor Affinity

Many trail-blazing work has studied the strength of social ties in social media for many years [18]. As one kind of social tie strengths, we define

Definition 2. *Viewer-Actor Affinity α_{ij}, the likelihood score of viewer i clicks on any activity produced by actor j in LinkedIn feed.*

The Viewer-Actor Affinity manifests the relationships between the feed recipient (viewer) and activity producer (actor), which have played an important role in scoring feed activity relevance [11]. However, what are the remarkable characteristics of Viewer-Actor Affinity in LinkedIn feed?

Like other commercial social networks, LinkedIn provides a scalable feed for more than 300 millions members to interact with their connections. Given a viewer can see more actors than activity types in feed, the dimension of the Viewer-Actor Affinity is about $40 \sim 50$ times larger than the Viewer-ActivityType affinity on average. Even the number of interactions generated by some viewers may be large, few of them impact a particular actor, making the actor-level interaction data sparse. Due to the absence of activities (zero or a few) between viewer and actor, rate estimates are often unreliable and noisy [3].

Another challenge of Viewer-Actor Affinity is the viewer only interacted with a tiny fraction of actors in the past, comparing to hundreds of millions of LinkedIn entities the viewer can interact. For example, in a one-week testing window, we observed that around half of actors are new to viewers (no past interactions). Hence the Viewer-Actor Affinity heavily suffers from the cold-start problem.

5.2 Viewer-Actor-ActivityType Affinity

We have designed Viewer-ActivityType Affinity and Viewer-Actor Affinity independently. However, an activity is interesting to the viewer often because of both. For example, assuming you like reading articles in general, you may not like reading articles published by sponsors or shared by recruiters. We denote by

Definition 3. *Viewer-Actor-ActivityType affinity α_{ijk}, the likelihood score of viewer i clicks on the activity type k produced by actor j in LinkedIn feed.*

The Viewer-Actor-ActivityType affinity can be seen as a context-aware affinity between viewer and actor, where the context is activity type. It provides a finer-grained affinity between viewer and actor. Undoubtedly, the interaction data of Viewer-Actor-ActivityType are sparser than Viewer-Actor and increase the difficulty of modeling.

5.3 Modeling

Modeling Viewer-Actor Affinities using a counting model like we do for Viewer-ActivityType Affinity is theoretically sound yet heavily suffers from the data sparsity, since the data intensity between viewer and actor is about $40 \sim 50$ times less than the data intensity between viewer and activity type. Modeling zero or few interactions is unreliable and noisy [3]. Hence, we adopt the feature-based regression to model Viewer-Actor Affinities due to simplicity.

Given an activity t from viewer i to actor j on activity type k, to predict the response Y which is 1 if there is a click and 0 otherwise, we use logistic regression where we compute a linear product between a coefficient vector β and a feature vector X_{ijkt}:

$$P(Y = 1) \sim \text{Bernoulli}(\sigma(\beta X_{ijkt})),$$

where σ is the sigmoid function.

We can exclusively classify features in the feature vector X_{ijkt} into three parts: Viewer-Actor features X_{ij}, Viewer-Actor-ActivityType features X_{ijk}, and other activity-dependent features X_t such as the age of the activity and the type of activity which vary on activities and have to be scored online. Though X_{ijk} can be a subset of X_{ij} conceptually, we take it out of X_{ij} for clarity.

We denote the part of β for X_{ij} by β_{ij}, the part of β for X_{ijk} by β_{ijk} and the part of β for X_t by β_t. Accordingly, by linearity of inner product, the score βX_{ijkt} can be decomposed into three parts:

$$\beta X_{ijkt} = \beta_{ij} X_{ij} + \beta_{ijk} X_{ijk} + \beta_t X_t. \quad (6)$$

Since X_{ijk} is about $40 \sim 50$ times sparser than X_{ij}, it may not exist for the activity t. Instead, X_{ij} always exists given the relations between two users are ubiquitous. Our strategy is, by carefully designing indicator features, we can differentiate two scenarios: X_{ijk} exists and X_{ijk} does not exists. If X_{ijk} exists, we name $\hat{\alpha}_{ijk} = \beta_{ij} X_{ij} + \beta_{ijk} X_{ijk}$ as the Viewer-Actor-ActivityType Affinity. No matter X_{ijk} exists or not, we always name $\hat{\alpha}_{ij} = \beta_{ij} X_{ij}$ as the Viewer-Actor Affinity.

5.4 Feature Design and Model Training

When modeling Viewer-Actor Affinities, relation and past interactions between viewer and actor have been shown to be useful [29]. In this subsection, we design a set of past interaction features as "warm-start" features and a set of relation features as "cold-start" features in order to overcome the data sparsity challenge.

Warm-start features: Two nature past interactions from the viewer to the actor are the number of past actions (e..g, clicks, shares, likes, comments, etc.), and the number of past impressions in LinkedIn. Intuitively, more past actions and less past impressions, higher CTR in the future, as shown in Figures 3a and 3b. To enrich the past interaction dynamics, we count the past interactions using multiple time windows. To enrich the past interaction intensity, we extend the time window up to half a year ago. Diverse and long past interactions reduce the data sparsity yet increase the data volume to be processed. This partially leads to our decision of offline computing the affiniy scores later.

The past interaction features can be available for either viewer-actor pairs or viewer-actor-activitytype triples. For example, the number of clicks between viewer and actor in the past 3 months is one feature; and the number of clicks between viewer and actor on "articles shared by the actor" in the past 3 months is another feature.

Cold-start features: The relationships between viewers and actors should be ubiquitous, not limited to viewer-actor pairs with past interactions. Considering the large fraction of new actors who just connected to viewers, we look for cold-start features that enrich relations between viewer and actor.

There are two major sources of cold-start features. One is from the structured member profile where we extract thousands of profile features including education, location, job experiences, and work-related skills etc. We then create millions of binary features by interacting viewer's profile features with actor's profile features. For textual properties like member summaries, we generate binary features by using bag-of-words representations. Top N features are selected by ordering the features using Mutual Information. As shown in Figure 5, we set $N = 2,000$ as the relative CTR lift converges after 2,000 features.

The other source is from the social network underlying the pair of viewer and actor. Graph feature examples considered by us include the number of the viewer's neighbors that took actions on the same actor, the number of common friends between viewer and actor, etc.

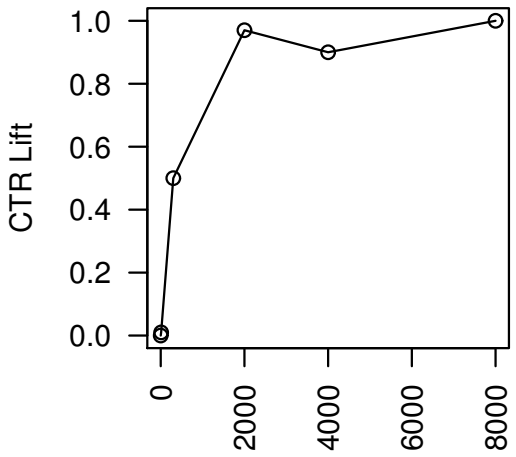


Figure 5: Relative CTR lift over Top N member profile features.

For simplicity, we only consider cold-start features for viewer-actor pairs, not viewer-actor-activitytype triples.

Training large scale logistic regression: From the training data Y and X_{ijkt} that we collect, we aim to learn β using logistic regression. The challenge here is the massive scale of regression, since our training data have billions of activities and we also consider thousands of features for each activity. It is thus not surprising that our regression problem does not fit in a single machine. To overcome this scalability challenge, we solve logistic regression in a distributed fashion, by employing the Alternating Direction Method of Multipliers (ADMM) [9]. ADMM partitions the data and learns a local model from each partition, and then it aggregates all models from partitions and sends back the aggregation results to individual partitions. For more details about how we use ADMM for our regression, refer to [4].

6. AFFINITY DEPLOYMENT FRAMEWORK

In Section 4 and Section 5, we have modeled Viewer-ActivityType Affinity as the CTR estimate and Viewer-Actor Affinities as the partial scores of the CTR estimate respectively. To collectively model all the affinity scores together, we take the following approach: First, we compute Viewer-ActivityType Affinity alone. Then we use Viewer-ActivityType Affinity as one of the baseline features in the joint training of the Viewer-Actor Affinities and the other activity-dependent coefficients. Mathematically, the relevance score used in ranking for viewer i, actor j, activity type k, and activity t, with monotonic logit transformation removed, can be written as

$$\begin{aligned} Y &= \beta_t X_t + \beta_{ik}\alpha_{ik} + \alpha_{ijk}, \text{ if } \alpha_{ijk} \text{ exists;} \\ &= \beta_t X_t + \beta_{ik}\alpha_{ik} + \alpha_{ij}, \text{ else,} \end{aligned} \quad (7)$$

where X_t is the feature vector that corresponds to the more dynamic activity-dependent features, α_{ik} is the Viewer-ActivityType Affinity, α_{ij} is the Viewer-Actor Affinity, α_{ijk} is the Viewer-Actor-ActivityType Affinity, and β_t, β_{ik} are the learned coefficients. Note

Table 1: Offline analysis on the frequency of updating Viewer-Actor Affinities

reward lift at	daily update	hourly update	2-day update
position 1	0.0%	+0.1%	-0.4%

that α_{ijk} or α_{ij} do not have coefficients attached simply because these two are partial results computed from the same model (see Equation (6)). Also, when there is no data to support triple (i, j, k) such that α_{ijk} does not exist, we fall back to α_{ij}.

Ideally the affinities and β should be trained jointly. However, due to the complexity of each affinity scores, high-dimensional feature spaces and data sparsity, we chose this simple yet effective approach for faster model training process. Also, since Viewer-ActivityType affinity has the richest data for warm-start, we choose to model it independently first and treat it as a single compound feature in the model for training Viewer-Actor Affinities.

How to determine the frequency of updating affinities is more an art than a science. We conducted offline analysis in Table 1 for Viewer-Actor Affinities using one week data, and found that increasing the updating frequency from daily to hourly did not change the reward at position 1 [23] significantly (only +0.1% lift). Thus, we choose to update the affinity scores daily in our production pipelines.

Figure 6 depicts the online deployment framework for affinities in LinkedIn feed. The flow starts by the URL request from a viewer i. LinkedIn feed service first fetches all the candidate activities from viewer i's interpersonal social network. Our scoring system then computes the relevance score for each activity and rank activities based on these scores. For each activity t with actor j and activity type k, the online scoring system obtains activity-dependent features X_t, and also fetches affinity scores from Voldemort stores[3] which support distributed key-value storage and queries. The affinity scores are pre-computed offline in Hadoop and updated on the Voldemort store daily. The scoring system finally applies Equation (7) to score and return the top-ranked results to show in people's feed.

Every day we compute tens of billions of Viewer-ActivityType Affinities for all LinkedIn members and all possible activity types (300 millions times 50), and also tens of billions of Viewer-Actor Affinities for both desktop and phone platforms. Note that we only prepare Viewer-Actor Affinities for those pairs with interactions in the past half a year. This strategy largely reduces the dimension of Viewer-Actor Affinities; otherwise, even the offline precomputation is not feasible. Furthermore, for heavy viewers who interacted with hundreds of thousands of actors, we only store top K Viewer-Actor Affinities in the Voldemort store after offline computation. This strategy avoids the latency timeout when serving heavy viewers online. We set $K = 10,000$ such that it can keep the vast majority of the affinity scores and only lost 0.08% reward at position 1 [23] in the offline analysis for one week data.

7. EXPERIMENTAL RESULTS

7.1 Experimental Setup

Online A/B tests: In LinkedIn, bucket tests (A/B tests) are conducted on a regular basis whenever we believe that we have a better model compared to existing models online (typically based on offline analysis). Once we decide to start testing a model, typically we launch it on $2\% \sim 3\%$ random LinkedIn members. We observe the performance of the model for about two weeks before making a decision to ramp the model to a larger population. Most importantly, we consider engagement, typically measured via CTR (click-through rate), as the evaluation metric. In addition, we also consider other metrics such as freshness of the feed based on the model, diversity of the feed, how many times items are repeated to a user etc. as well in making ramp decisions. However, for the purposes of this paper, we only report CTR metrics. Most of the results reported below are from online A/B tests with offline evaluations when appropriate.

[3]http://www.project-voldemort.com/voldemort/

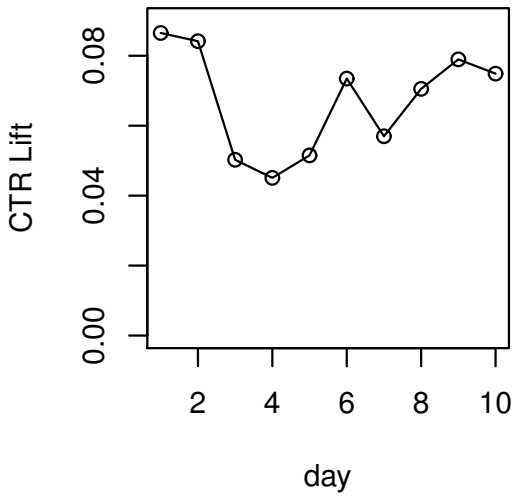


Figure 7: Online CTR lift by Viewer-ActivityType Affinity on desktop.

Offline replay analysis: In LinkedIn, after we train a model, we need to evaluate the model offline before deploying it in production. For this, we use a replay methodology for unbiased offline evaluation of online serving schemes [23], where the relative rewards obtained from two different models are typically indicative of which model would perform better in production.

When launching a new model, we also retrain the model with the recent data. This is because, often new activity types are introduced on the feed and sometimes some types are retired. Typically, we retrain both the baseline model and the new model with recent data, and we then evaluate both models offline to decide whether or not the new model is launchable.

We do not always replace the baseline model in production by the baseline model retrained on the recent data, unless we believe retraining itself will create a big performance gap. Therefore, in some of the A/B experiments below, it is possible that the baseline model was not retrained on recent data but the new model was retrained. In such situations, we also provide the offline comparisons.

7.2 Desktop Experiments

In this section, we show results for the models that were launched to improve personalization on desktop.

7.2.1 Effect of Viewer-ActivityType Affinity

We tested Viewer-ActivityType Affinity since late 2013. The baseline model consists of activity-dependent features such as age

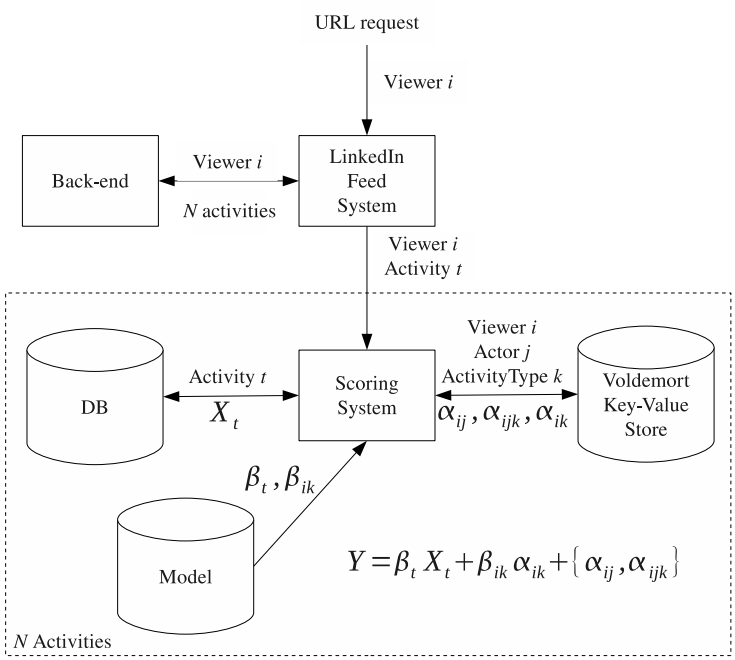


Figure 6: Online deployment framework for affinities.

of an activity, type of an activity etc. and some raw connection strength features between a viewer and an actor. The affinity model applied the Viewer-ActivityType Affinity as a new feature on top of the baseline model. The A/B tests during 10 days between them are shown in Figure 7. The x-axis shows the day index and the y-axis shows the relative CTR lift of the affinity model comparing to the baseline model. Overall, the affinity model achieved a daily CTR lift of about +7.0%. Two models were trained using the same data and tested over the same percentage of LinkedIn members (2%). Hence the comparison is fair except for the bucket bias[4].

7.2.2 *Effect of Viewer-Actor Affinity*

We started testing a model based on Viewer-Actor Affinity since May 2014. Now, the Viewer-ActivityType affinity model in the previous section 7.2.1 becomes the baseline model. The compared affinity model applied the Viewer-Actor Affinity as a new feature on top of the baseline model. The A/B tests between them in Figure 8 showed a CTR lift of about +2.9% on average.

One may wonder why we do not put comparisons in Figure 7 and Figure 8 together in a single plot. They are not numerically comparable simply because two bucket tests were conducted at different time. By the time we started the new bucket tests in May 2014, the baseline model had changed significantly from the Viewer-ActivityType affinity in the previous section 7.2.1. For instance, the new baseline had a few new activity type features, and LinkedIn feed had moved to a paradigm where we always served activities using a relevance model rather than a mixture of relevance feed and real-time feed. However, each bucket test alone is still meaningful because for each bucket test, two compared models have the same test settings and product environment.

Considering the Viewer-Actor Affinity model was trained on newer data, we also have the offline comparison results where both models

[4]Completely removing the bucket bias is impossible because there will always be "noise" inherent to the populations since they're simply not the same people.

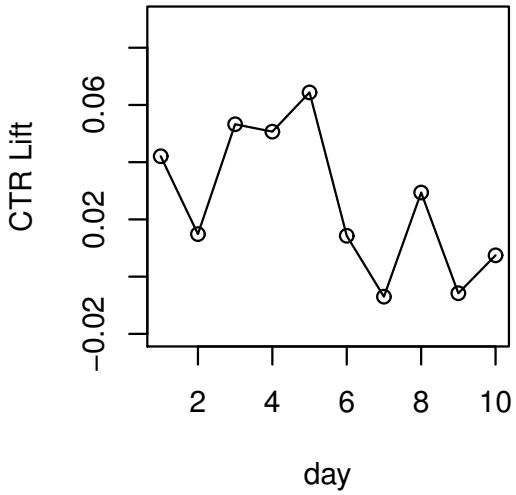


Figure 8: Online CTR lift by Viewer-Actor Affinity on desktop.

were trained and tested on the same data. Comparing to the baseline Viewer-ActivityType Affinity model, the Viewer-Actor Affinity model achieved +3.5% reward lift at position 1 for one week test data, where position 1 has the least position bias in offline replay analysis and is thus the most reliable [23].

7.2.3 *Effect of Viewer-Actor-ActivityType Affinity*

We tested Viewer-Actor-ActivityType Affinity since June 2014. Similarly there were some product design changes since June 2014 in LinkedIn, e.g., showing fresher content on the feed. Thus, we

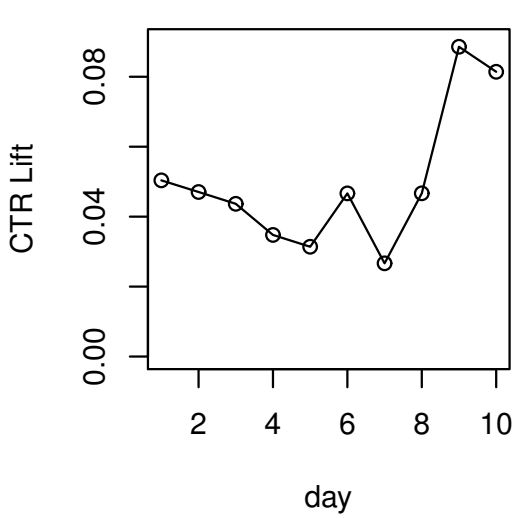


Figure 9: Online CTR lift by Viewer-Actor-ActivityType Affinity on desktop.

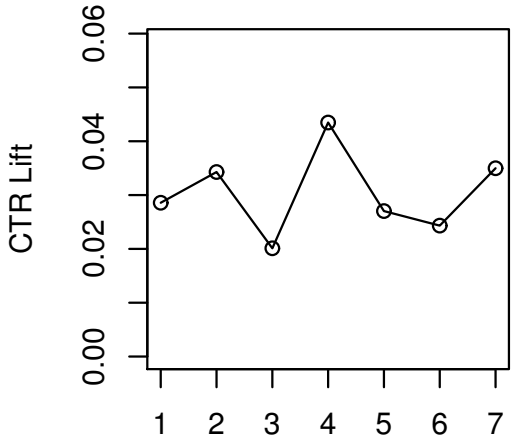


Figure 10: Online CTR lift by Viewer-ActivityType Affinity on mobile.

only look at the comparions in a single bucket test. The baseline model is the Viewer-Actor affinity model in the previous section 7.2.2. The compared affinity model applied the Viewer-Actor-ActivityType Affinity as a new feature on top of the baseline model. The 10-day A/B tests between them in Figure 9 showed an average CTR lift of about +4.3% for each day. During the offline replay analysis where two models were trained and tested on the same data, we found that the Viewer-Actor-ActivityType Affinity model achieved +6.6% reward lift at position 1 for one week test data.

7.2.4 Discussion

Experimental results on desktop are promising – we proved that the more granular the affinity, the higher the CTR through both online bucket tests and offline replay analysis. By treating affinities as a compound feature or partial scores of the final response prediction, we only introduced 3 additional arithmatic operations and 1 additional query to the online production, which contributed to less than 10% of online service time.

7.3 Mobile Bucket Test Results

In this section, we show similar results on mobile. Because of the great successfulness of Viewer-Actor-ActivityType Affinity on desktop, we decided to migrate from Viewer-ActivityType Affinity to Viewer-Actor-ActivityType Affinity directly on mobile. So, mobile results of Viewer-Actor Affinity model are skipped.

7.3.1 Effect of Viewer-ActivityType Affinity

The comparison has been conducted during one week period in April 2014. The baseline model is the simple model without any affinities described in Section 7.2.1. The compared affinity model applied the Viewer-ActivityType Affinity as a new feature on top of the baseline model. Both models were trained on the same data so no need to additionally provide offline replay analysis. The A/B tests between them in Figure 10 showed a CTR lift of about +3% on average.

7.3.2 Effect of Viewer-Actor-ActivityType Affinity

The comparison was conducted during one week period in December 2014. The baseline model is the Viewer-ActivityType Affinity model in the previous section 7.3.1. The compared affinity model applied the Viewer-Actor-ActivityType Affinity as a new feature on top of the baseline model. Both models were trained on the same data so no need to additionally provide offline replay analysis. The A/B tests between them in Figure 11 showed a CTR lift of around +10% on average.

8. CONCLUSION

We studied how to personalize LinkedIn feed in this paper, with a focus on affinity modeling. We generated three kinds of affinity scores offline: Viewer-ActivityType Affinity, Viewer-Actor Affinity, and Viewer-Actor-ActivityType Affinity. All of them are easy to compute, easy to scale, and easy to be deployed in production. We did extensive experiments based on online bucket tests (A/B tests) and offline evaluation to illustrate the effect of these affinities in LinkedIn feed.

Personalizing LinkedIn feed is an open-ended problem given the fact that millions of new members join LinkedIn every month around the world. Our initial study demonstrates that personalization of finer granularity achieves higher CTR, which motivates our future work as below.

- Deepen the personalization to the activity-dependent level, e.g., the affinity between viewer and the content topic of activity.
- Deepen the personalization to the viewer ID level, e.g., each viewer has her own personalization model.

9. REFERENCES

[1] https://www.facebook.com/notes/facebook/new-views-for-your-home-page/162536657130.

[2] F. Abel, Q. Gao, G.-J. Houben, and K. Tao. Analyzing user modeling on twitter for personalized news recommendations. In *UMAP*, pages 1–12, 2011.

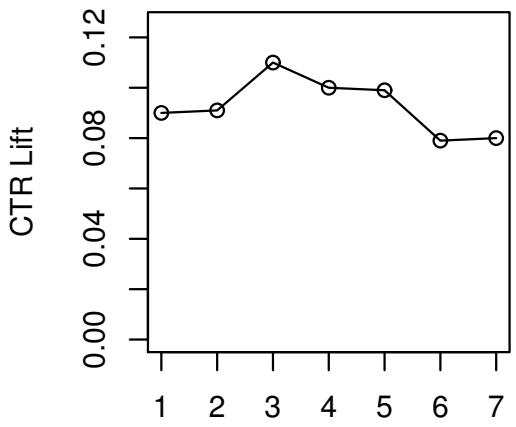


Figure 11: Online CTR lift by Viewer-Actor-ActivityType Affinity on mobile.

[3] D. Agarwal, R. Agrawal, R. Khanna, and N. Kota. Estimating rates of rare events with multiple hierarchies through scalable log-linear models. In *SIGKDD*, pages 213–222, 2010.

[4] D. Agarwal, B.-C. Chen, R. Gupta, J. Hartman, Q. He, A. Iyer, S. Kolar, Y. Ma, P. Shivaswamy, A. Singh, and L. Zhang. Activity ranking in linkedin feed. In *SIGKDD*, pages 1603–1612, 2014.

[5] S. Berkovsky and J. Freyne. Network activity feed: finding needles in a haystack. *LNAI 8940*, pages 21–34, 2015.

[6] S. Berkovsky, J. Freyne, S. Kimani, and G. Smith. Selecting items of relevance in social network feeds. In *UMAP*, pages 329–334, 2011.

[7] S. Berkovsky, J. Freyne, and G. Smith. Personalized network updates: Increasing social interactions and contributions in social networks. In *UMAP*, pages 1–13, 2012.

[8] S. Bourke, M. P. O'Mahony, R. Rafter, and B. Smyth. Ranking in information streams. In *Proceedings of the companion publication of the 2013 international conference on Intelligent user interfaces companion*, pages 99–100, 2013.

[9] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. Technical report, Stanford University, 2010.

[10] M. Burgess, A. Mazzia, E. Adar, and M. Cafarella. Leveraging noisy lists for social feed ranking. In *AAAI*, 2013.

[11] J. Chen, R. Nairn, and E. H. Chi. Speak little and well: Recommending conversations in online social streams. In *SIGCHI*, pages 217–226, 2011.

[12] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. H. Chi. Short and tweet: Experiments on recommending content from information streams. In *SIGCHI*, pages 1185–1194, 2010.

[13] K. Chen, T. Chen, G. Zheng, O. Jin, E. Yao, and Y. Yu. Collaborative personalized tweet recommendation. In *SIGIR*, pages 661–670, 2012.

[14] W. Chu and S.-T. Park. Personalized recommendation on dynamic content using predictive bilinear models. In *WWW*, 2009.

[15] W. DuMouchel and D. Pregibon. Empirical bayes screening for multi-item associations. In *SIGKDD*, pages 67–76, 2001.

[16] G. D. Francisci, A. Gionis, and C. Lucchese. From chatter to headlines: Harnessing the real-time web for personalized news recommendation. In *WSDM*, pages 153–162, 2012.

[17] J. Freyne, S. Berkovsky, E. M. Daly, and W. Geyer. Social networking feeds: Recommending items of interest. In *RecSys*, 2010.

[18] E. Gilbert and K. Karahalios. Predicting tie strength with social media. In *SIGCHI*, pages 211–220, 2009.

[19] I. Guy, I. Ronen, and A. Raviv. Personalized activity streams: Sifting through the "river of news". In *RecSys*, pages 181–188, 2011.

[20] I. Guy, T. Steier, M. Barnea, I. Ronen, and T. Daniel. Swimming against the streamz: search and analytics over the enterprise activity stream. In *CIKM*, pages 1587–1591, 2012.

[21] L. Hong, R. Bekkerman, J. Adler, and B. D. Davison. Learning to rank social update streams. In *SIGIR*, pages 651–660, 2012.

[22] L. Hong, A. S. Doumith, and B. D. Davison. Co-factorization machines: modeling user interests and predicting individual decisions in twitter. In *WSDM*, pages 557–566, 2013.

[23] L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *WSDM*, pages 297–306. ACM, 2011.

[24] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *WSDM*, pages 287–296, 2011.

[25] T. Paek, M. Gamon, S. Counts, D. M. Chickering, and A. Dhesi. Predicting the importance of newsfeed posts and social network friends. In *AAAI*, 2010.

[26] Y. Pan, F. Cong, K. Chen, and Y. Yu. Diffusion-aware personalized social update diffusion-aware personalized social update recommendation. In *RecSys*, pages 69–76, 2013.

[27] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW*, pages 521–530, 2007.

[28] P.-H. Soh, Y.-C. Lin, and M.-S. Chen. Recommendation for online social feeds by exploiting user response behavior. In *WWW*, 2013.

[29] I. Uysal and W. B. Croft. User oriented tweet ranking: a filtering approach to microblogs. In *CIKM*, pages 2261–2264, 2011.