Improving the discriminative power of inferred content information using segmented virtual profile

Article · October 2014

DOI: 10.1145/2645710.2645738

Improving The Discriminative Power Of Inferred Content Information Using Segmented Virtual Profile

Haishan Liu
LinkedIn Corporation
2029 Stierlin Court
Mountain View, CA, 94043
haliu@linkedin.com

Anuj Goyal
LinkedIn Corporation
2029 Stierlin Court
Mountain View, CA, 94043
agoyal@linkedin.com

Trevor Walker
LinkedIn Corporation
2029 Stierlin Court
Mountain View, CA, 94043
twalker@linkedin.com

Anmol Bhasin
LinkedIn Corporation
2029 Stierlin Court
Mountain View, CA, 94043
abhasin@linkedin.com

ABSTRACT

We present a novel component of a hybrid recommender system at LinkedIn, where item features are augmented by a *virtual profile* based on observed user-item interactions. A virtual profile is generated by representing an item in the user feature space and leveraging the overrepresented user features from users who interacted with the item. It is a way to think about Collaborative Filtering with content features. The core principle is that if the feature occurs with high probability for the users who interacted with an item (henceforth termed as relevant users) versus those who did not (henceforth termed as non-relevant users), then that feature is a good candidate to be included in the virtual profile of the item in question. However, this scheme suffers from the data imbalance problem because observed relevant users are usually an extremely small minority group compared to the whole user base. Feature selection in this skewed setting is prone to noise from the overwhelming non-relevant examples that belong to the majority group. To alleviate the problem, we propose a method to select the most relevant non-relevant examples from the majority group by segmenting users on certain intelligently selected feature dimensions. The resulting virtual profile from the method is called the *segmented virtual profile*. Empirical evaluation on a real-world large scale recommender system at LinkedIn shows that our strategies for segmentation yield significantly better results.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Data Mining

RecSys'14, October 06–10, 2014, Foster City, Silicon Valley, CA, USA.

http://dx.doi.org/10.1145/2645710.2645738.

1. INTRODUCTION

As of today LinkedIn has more than 300 million users. As the largest and most popular professional networking site, LinkedIn presents some unique opportunities and challenges for content discovery and recommendation. It is imperative that users are given an effective and efficient tool to navigate the data deluge. Large scale recommender systems have emerged as a solution to this challenge. Rather than hoping for serendipitous encounters, we have created a hybrid recommender system that incorporates information from a myriad of sources, which presents a smaller pool of relevant items to users from a massive set of candidates.

Since there is no dearth of information about either users or items at LinkedIn, the foundation of the recommender system is a content-based filtering approach. A significant challenge to address while pursuing a content filtering approach is vocabulary dissonance, i.e., the user and item features are not always represented in the same vocabulary. Thus, a feature augmentation method was developed to abstract optimal item features for recommendation in the system, which matches user features in a content filtering setting. Specifically, when several users have interacted with an item that has been deemed relevant for their information need, a "profile" is automatically learned for the item. This allows it to be easily matched (recommended) to other likeminded and potentially interested users. To distinguish from other content features inherently contained within the item, we call such inferred information, derived from user-item interactions, the *virtual profile*.

The creation of the virtual profile can be viewed as a feature selection problem with underpinnings in Information Theory. We have described a method to retrieve relevant terms for the virtual profile by a filter-based feature selection method from labeled examples [10], i.e., the users who are relevant to the item because of positive interaction, and non-relevant users. The learned virtual profile is then added to the vector space representation of the item's original content features, and matched against new users in the system. If a new user matches the augmented item profile adequately, then the item is assumed to be of potential interest to the user and is recommended to the user. The degree to which our approach improves the pure content-based filtering, then, is largely dependent upon the quality of the generated item virtual profile.

More specifically, to create a virtual profile, the algorithm selects a set of features (terms appearing in user profiles) whose presence or absence indicates potential relevance or non-relevance for a user. We use mutual information as the selection criteria. The mutual information values of the selected terms are assigned as weights, indicating the relative importance of the features in predicting relevance of a user. The central idea of this scheme is that if a term occurs with a high probability in the relevant users but with a low probability in the non-relevant users, then it is a good indicator of relevance and should be assigned a high weight in the virtual profile. Other feature selection methods (based on some variation of the probability of occurrence, such as Chi-square [23], Correlation coefficient [14], Odds ratio [13], etc.) can be used, as well.

An item is inferred to fulfill a certain user information-need by its observed interactions with users. However, when the number of users who have interactions with the item is small and the non-relevant users who have no interaction with the item is large, feature selection can become prone to noise. This phenomenon of imbalanced data is a known problem in the study of feature selection commonly encountered in text categorization tasks (see a motivating example on job recommendations in Section 3).

To tackle this problem, we propose a method to obtain more balanced relevant and non-relevant examples by segmenting users on intelligently selected feature dimensions. Specifically, define an item's user segment as the set of users that have some similarity to the item according to some measure. In this study, we show that using a selected set of non-relevant users that belongs to an item's user segment to learn the virtual profile yields better results than using the entire non-relevant user base. Results show that our strategies to select non-relevant users for learning the virtual profile yield significantly better performance. The resulting virtual profile generated by such strategy is called the segmented virtual profile.

The rest of the paper is organized as follows. In Section 2 we briefly review related work and provide necessary background. In Section 3 we describe details of the segmented virtual profile generation process. In Section 4 we present experiment results from both online and offline evaluations. Finally, we conclude the paper in Section 5 and discuss possible future directions.

2. BACKGROUND AND RELATED WORK

This section surveys previous work in recommender systems and feature selection as well as providing necessary background information related to virtual profiles.

2.1 Recommender Systems

The recommender system approaches can be mainly classified into four categories: Collaborative filtering [8, 19], Content-based filtering [6, 17], Knowledge-based [3, 4] and Hybrid algorithms [2].

A fundamental assumption in collaborative filtering (CF) is that if user x and y rate n items similarly, they will rate other items similarly too. The CF techniques are known to suffer from data sparsity and cold start problems [18]. Content-based filtering (CBF) methods are based on a description of the item and a profile of the user's preference and can be treated as a information retrieval or machine learning problem. CBF techniques are sometimes prone to deliver skewed recommendations [17]. Knowledge-based approach attempts to suggest objects based on inferences on user's needs and preferences. The main challenge for these systems is difficulties of knowledge acquisition and knowledge engineering. To avoid certain limitations of content and collaborative filtering systems, hybrid approaches are proposed to combine CF and CBF based techniques. In [2] authors propose to use a taxonomy of recommender systems, where multiple recommenders are arranged to allow execution in a parallel or cascaded topology. The system described in [1] combined the output of multiple collaborative filtering approaches by using a linear combination of weights learned via linear regressions. In [16], the author proposes a hybrid approach where the content based user profiles are used to group similar users which is subsequently used to predict user preferences.

In this paper, we experiment our recommender system in a job recommendation application. There has been good amount of research in the domain of matching job candidates (resumes) with job postings. Simple Jobs recommendation algorithm [11] are based on the boolean filtering techniques that cannot sufficiently capture the complexity of a person-job fit. In [12], authors propose to consider unary attributes such as individual skills, mental abilities and personality that control the fit between the individual and the task to be accomplished, as well as the relational attributes that determine the fit between the individual and the team members. A decision support tool named PROSPECT proposed in [20] mines resumes to extract features of candidate profiles such as skills, education, and experience. It then uses information retrieval techniques to rank applicants for a given job position. In [15], authors exploit all past job transitions as well as the data associated with employees and institutions to predict an employee's next job transition.

2.2 Feature Selection

The creation of virtual profiles is closely related to the study of feature selection. A feature selection algorithm can be seen as the combination of a search technique for proposing new feature subsets, along with an evaluation measure which scores the different feature subsets. The choice of evaluation distinguishes between the three main categories of feature selection algorithms: wrappers, filters and embedded methods [7]. Wrapper methods use a predictive model to score feature subsets. Embedded methods perform feature selection as part of the model construction process. Filter methods produce a feature set which is not tuned to a specific type of predictive model. Filter methods use a proxy measure instead of the error rate to score a feature subset. Common measures include mutual information [7], Chi-square [23], Correlation coefficient [14], Odds ratio [13], inter/intra class distance, and the scores of significance tests for each class/feature combinations [24].

Recent studies have explored the topic of feature selection on imbalanced data. Imbalanced datasets are commonly encountered in text categorization problems. There are often overwhelming numbers of non-relevant training documents especially when analyzing a large collection of categories assigned to small numbers of documents. To overcome this problem, *query zone* [21] has been introduced to select a subset of the most relevant non-relevant documents as the non-relevant training data. The user segment proposed in this paper aims at achieving the same objective by segmenting the user base on intelligently selected feature dimensions. These techniques try to obtain a more balanced relevant and non-relevant training data by under-sampling negative examples. Zheng et al. [25] consider the imbalanced data problem from a different perspective. Instead of balancing the training data, the authors propose a way to find the optimal way to combine positive and negative features according to the imbalanced data.

2.3 Virtual Profile

The creation of a virtual profile can be viewed as a feature selection process [10]. From a total of n user-item interaction features, it aims at selecting a subset with $k(<= n)$ features that gives the maximum information about the item. From a text categorization point of view, the item that we generate a virtual profile for represents a class label for a set of documents (user profiles). We can use mutual information to evaluate the information content of each individual feature with regard to the class label. In accordance with Shannon's information theory, the uncertainty of a document class C as a random variable can be measured as:

$$H(C) = -\sum_{c \in C} P(c) \log P(c) \, ,$$

In this case, a document class is a virtual profile of a target item, while documents are profiles of users who have interacted with the target item. After knowing the occurrence of a feature F, the conditional entropy $H(C|F)$ measures the remaining uncertainty about C:

$$H(C|F) = -\sum_{f \in F} P(f) \sum_{c \in C} P(c|f) \log P(c|f) \, .$$

The mutual information, i.e., the amount of decreased class uncertainty is defined as:

$$I(C;F) = H(C) - H(C|F) = \sum_{c,f} P(c,f) \log \frac{P(c,f)}{P(c)P(f)} \, . \tag{1}$$

To generate virtual profiles, the goal is to find the optimal feature subset, $S \subseteq F$, such that $I(C;S)$ is maximized. We employ a straightforward best individual features strategy (BIF [9]) by making a first-order class dependence assumption (each feature independently influences the class variable). The BIF method evaluates all features individually, sort them and select the best k features based on the mutual information score.

3. METHOD

In this section, we describe details of the segmented virtual profile. We first give a motivating example to illustrate the data imbalance problem that causes the suboptimal quality of features selected in the virtual profile.

3.1 Motivation

Consider a content-based filtering system that recommends jobs to users. Content features from both jobs and users can be abstracted from their textual description into predefined standardized fields, such as location, industry, function, skill, etc., which constitutes their primary profiles. Additionally, a virtual profile can be created for each job based on the observed behavior of users applying the job. Virtual profile aims at selecting indicative features from users that correlate well with the application of the job. In other words, features selected for the virtual profile should be able to distinguish job applicants (relevant users) from non-applicants (non-relevant users). However, since features from job applicants are evaluated against a vast amount of non-applicants, most of whom are completely irrelevant to the job, the resulting virtual profile may not include the real indicative features.

Take a job posting about *professor of economics* for example. If we regard all applicants to this job as the set of relevant users, and all non-applicants as the set of non-relevant users, the term *teaching* from its applicants' skills will score a high mutual information value with regard to the job. It is because the frequency of the term is low for non-relevant users and high for relevant users. Therefore according to such choice of relevant/non-relevant users, *teaching* would appear as a very strong indicator of relevance. However, in reality, this term will not be very good at separating this job from other academic jobs, such as a *professor of history*. In this case, the use of all non-applicants as non-relevant users has boosted the importance of a term that is possibly not very important. Therefore we need a way to constrain the set of non-relevant users to include only those that have some reasonable similarity to the item. For example, if we consider only users from the higher education field, the term teaching would lose its significance but terms such as *economics* and *history* would be able to surface. Following this idea, we define user segments in the next section as a solution for this purpose.

3.2 User Segments

A user segment for a target item is a set of users that have some reasonable similarity to the item. It provides a small set of negative, or non-relevant users for the purpose of selecting more indicative terms in the virtual profile generation process. In practice, a user segment can be simulated by performing "slice and dice" in the vector space model, which is to filter both the item vector and the user vector space on one or multiple dimensions simultaneously. Take job postings as an example. A user segment for one job posting can be users from the same industry to the job. The aim therefore, is to select a set of non-relevant users that relate well to the target item, from the whole corpus of labeled example set, to be used in the virtual profile generation process.

To answer the question of how to select the feature dimension to generate the user segment, we propose the following steps:

1. Construct a labeled pilot dataset with observed positive user-item interactions from a time period as positive examples, and randomly sampled non-interactive user-item pairs as negative examples.

2. Construct features in the above dataset so that each feature represents a similarity between a pair of corresponding values from a user-item pair on the same dimension of their vector space model. For instance, in the job application example, the feature in the dataset can be the similarity between the user industry and the job industry, the similarity between user skills and job skills, user location and job location, and so forth.

3. Select top features based on this dataset, then use the corresponding dimensions from the original user/item-

feature	MI
geo_region	0.034
skill	0.0075
seniority	0.0055
function	0.0055
industry	0.0014

Table 1: Top features selected from the pilot dataset using mutual information. Each row represents an interaction feature between a user and a job.

feature vector space as the candidates to generate user segments.

An example of the pilot dataset constructed from job application data within a two-week period is shown in Table 1, together with the top five features selected from the dataset using mutual information. It is worth pointing out that domain knowledge is a valuable source in deciding the dimension to segment on. For instance, we only use *function* and *industry* features in our experiments reported in Section 4, because the *geo_region* and *skill* features are sparse (too many categories, some of which contains very few users), while the *seniority* feature is coarse (only nine categories in total).

3.3 Mutual Information in User Segments

The class membership C and feature F in Equation 1 are both binary-valued (indicating whether a user-item interaction exists and whether a feature is present). Therefore Equation 1 for calculating the mutual information $I(C;F)$ can be written in the following form:

$$I(C;F) = \sum_{e_f \in \{1,0\}} \sum_{e_c \in \{1,0\}} P_{CF}(e_c, e_f) \log \frac{P_{CF}(e_c, e_f)}{P_C(e_c) P_F(e_f)}, \tag{2}$$

where f is a random variable that takes values $e_f = 1$ (the feature f appeared in a user) and $e_f = 0$ (the feature f does not appear in a user), and c is a random variable that takes values $e_c = 1$ (the user positively interacted with c) and $e_c = 0$ (the user did not positively interact with c). Equation 2 can be solved by the contingency table shown in Table 2, in which the joint probabilities are approximated by maximum likelihood estimation from the data.

	$F(e_f = 1)$	$F(e_f = 0)$
$C(e_c = 1)$	N_{fc}	$N_c - N_{fc}$
$C(e_c = 0)$	$N_f - N_{fc}$	$N - N_c - (N_f - N_{fc})$

Table 2: The contingency table to calculate the regular mutual information between a term and a job.

As a concrete example, the counts in Table 2 have the following meaning if we instantiate each item as a job, and a user-item interaction as a job application.

- N_c: number of people who applied the job
- N_{fc}: number of people who have the feature and applied the job
- N: total number of people who have applied any job
- N_f: number of people who have the feature

To calculate the mutual information with a specific user segment, we can use the contingency table shown in Table 3, which has a slight modification in the calculation of joint probabilities when non-relevant users are involved ($e_c = 0$).

	$F(e_f = 1)$	$F(e_f = 0)$
$C(e_c = 1)$	N_{fc}	$N_c - N_{fc}$
$C(e_c = 0)$	$N_{fs} - N_{fc}$	$N_s - N_c - (N_{fs} - N_{fc})$

Table 3: The contingency table to calculate the mutual information between a term and a job, with negatives chosen from a specific user segment.

The counts in Table 3 have the following meaning in the job application setting.

- N_c: number of people who applied the job
- N_{fc}: number of people who have the feature and applied the job
- N_s: total number of people who are from a specific user segment
- N_{fs}: number of people who are from the specific user segment and have the feature.

3.3.1 *Frequency Threshold*

The mutual information score should be interpreted with care when data is sparse. It is known to attribute high scores to low frequency terms. For instance, in the job posting example, suppose some terms only appear in a small number of applicants to a job but not in the general public. Those terms will have high scores since as far as mutual information is concerned, they correlate well with the event of users applying to the specific job. Therefore we impose two thresholds on term frequency to prevent terms being selected simply because of rarity. They are, namely, the *global frequency threshold* θ_g (i.e., the minimum frequency of a feature in the global feature space across all users), and the *per-item frequency threshold* θ_i (i.e., the minimum frequency of a feature in users who interacted with the specific item). These two thresholds are the parameters of the virtual profile generation process and should be tuned as per application requirement. When the feature or item space is large, aggressive thresholding can be an effective way to provide additional scalability.

4. EXPERIMENTS

Our main goal in the experiment design is to estimate the impact of segmented virtual profile on recommendation performance. In addition, we want to do a comparative performance analysis between segmented and non-segmented virtual profile. Furthermore, since segmented virtual profile can be generated in different ways depending on the dimension it is segmented on, we also want to compare the performances of these different segmented virtual profiles.

4.1 Experiment Design

LinkedIn has a wide variety of recommendation products available on its website. In this study, we experiment with LinkedIn's renowned jobs recommendation engine named as

Table 4: Performance of baseline and non-segmented virtual profile models

Model	Precision			Recall			F1		
	P@1	P@5	P@10	R@1	R@5	R@10	F1@1	F1@5	F1@10
Baseline	.829	.307	.160	.589	.928	.963	.659	.442	.268
Non-segmented Virtual Profile	.833	.309	.161	.590	.931	.965	.661	.445	.269

Table 5: NDCG and MAP of baseline and non-segmented virtual profile models

Model	NDCG			MAP
	NDCG@1	NDCG@5	NDCG@10	
Baseline	.829	.876	.890	.854
Non-segmented Virtual Profile	.833	.879	.893	.858

Jobs You May Be Interested In (JYMBII). JYMBII plays a crucial role in helping LinkedIn members explore relevant jobs. JYMBII leverages the user's profile content as well as the user's activity to recommend relevant jobs. Apart from JYMBII provided recommendations, users can also use the job search functionality to find relevant jobs. We record various kinds of members-jobs interactions in both the JYMBII and Search system.

We extract two kinds of features from entities (users and jobs) in this application domain:

1. **Content Features (Primary Profile):** standardized features extracted from the textual content of the job's posting and the user's profile (e.g., skills, degrees, specialties, industry, function, location, etc.).

2. **Virtual Profile:** A set of features extracted from a job's applicants. These features are in the same vector space as user features, and are intended to be a maximally informative features about a job given its observed interaction with users.

We model the job recommendation problem as a classification problem where the classifier outputs score is $p(j|u)$, which can be ideally interpreted as the probability that a job j is relevant to the given user u. We train a model by using $L2$-regularized logistic regression classifier with different user-job interaction features as input features. The model's output score is used to rank the relevant jobs for a user.

In our experiments, we train two types of models: baseline and virtual profile. The baseline model is trained by using the content features while the virtual profile model is trained by using both content and virtual profile features. The best model (best regularization parameters) under each configuration is selected by optimizing the area under the ROC curve (AUC-ROC).

The training set consists of all the applications (from any of the source, i.e., JYMBII or search) a job has received in a month. Training data is required for two purposes: 1) extracting virtual profiles, and 2) training a classifier. Since the same training data is used to generate the virtual profile feature as well as to train the model, extra care is required in the way we use the training data. If we use the same user-job pair in generating the virtual profile as well as in training the model, the learned model will be highly biased towards the virtual profile features. This is mainly due to the fact that information from a user-job pairs are used twice, in generating features as well as in training the model. To overcome this bias, we partitioned the application data into two equal sized random parts. The first part is used to generate the virtual profile features and the other is used to train the model. As discussed in section 3 and based on our domain knowledge and pilot studies, we use *global frequency threshold* θ_g equals to 2000 and *per-item frequency threshold* θ_i equals to 2 to generate the virtual profile.

Based on the domain knowledge and the approach discussed in section 3, we generate virtual profile segmented on function feature and industry feature. Function features are pre-defined job functions based on the user's current job position for example Engineering, Finance, Sales, and Administrative etc. Industry features are pre-defined industries based on user's current employer for example Banking, Farming, Food Production, Higher Education etc.

Performances of different models are evaluated in both offline and online settings. The next section discusses the results of these evaluations.

4.2 Results

4.2.1 Offline evaluation

In these experiments, we adopt an offline evaluation method similar to the one described in [22]. The job applications data set is split into training set R and test set T. In order to measure different information retrieval measures, we first train the model by using the training set R. Then, for each job i applied by user u in T:

1. We randomly select 1000 additional jobs not applied by the user. We assume that these random jobs are likely to be irrelevant to the user.

2. We calculate the relevance score for the job i and for the additional 1000 drawn jobs.

3. We build a ranked list by ordering all the 1001 items according to their relevance score. Let p denote the rank of the test job i within the list. The best model corresponds to the case where the applied job i precedes all the random items (i.e., $p = 1$).

To compute different information retrieval metrics, we form a top-K recommendation list by picking the top K ranked items from the list. The metrics we calculate include *Precision@K*, *Recall@K*, *F1@K*, *NDCG(Normalized Discounted Cumulative Gain)@K* and *Mean Average Precision (MAP)* measures. We calculated these for $K = \{1, 5, 10\}$.

We conduct two experiments, one to compare the baseline with the non-segmented virtual profile and the other to

Table 6: **Performance of baseline and segmented virtual profile models**

Model	Precision			Recall			F1		
	P@1	P@5	P@10	R@1	R@5	R@10	F1@1	F1@5	F1@10
Baseline	.848	.435	.234	.400	.879	.936	.520	.560	.365
Industry Segmented Virtual Profile	.878	.447	.238	.415	.902	.951	.539	.575	.370
Function Segmented Virtual Profile	.860	.440	.236	.405	.889	.942	.527	.567	.367

Table 7: NDCG and MAP of baseline and segmented virtual profile models

Model	NDCG			MAP
	NDCG@1	NDCG@5	NDCG@10	
Baseline	.848	.850	.876	.833
Industry Segmented Virtual Profile	.878	.877	.899	.860
Function Segmented Virtual Profile	.860	.861	.886	.844

compare with the proposed segmented virtual profile. Since these experiments were conducted on the data collected from different time periods, the baseline model has different numbers in these two experiments.

Table 4 and Table 5 compare the performance between the baseline and the non-segmented virtual profile models. Table 6 and Table 7 compare the performance between the baseline and the segmented virtual profile models.

As can be seen in the tables, the non-segmented virtual profile model performs better than the baseline. Segmented models outperform the baseline by an even bigger margin across all metrics. Between the segmented models, the industry segmented model outperforms the function one. It is worth noting that these gains are considered significant because the baseline we used in the experiments is a sophisticated model currently deployed in the production. The model has gone through numerous rounds of fine tuning, resulting in the use of a comprehensive list of features, including interaction content features, user behavior features, and interest/preference features.

4.2.2 *Online evaluation*

Once we picked the best performing models from the offline evaluation, we deployed them to serve real time online JYMBII recommendation traffic and compare performances through a bucket (A/B) test. We assign a distinct bucket of 5% randomly selected users to each model from the whole user base.

Since industry segmented virtual profile perform better than the function segmented virtual profile in the offline setting, we deployed only industry segmented virtual profile in the online setting. To compare segmented with non-segmented virtual profile, we also deployed the non-segmented virtual profile.

We collected the data from the A/B test for 11 days. From the collected data, we calculate two metrics:

1. **Unique User-Job view pairs:** Total number of (u, j) pairs where user u has viewed the job j recommended to her at least once during the online test.

2. **Unique User-Job application pairs:** Total number of (u, j) pairs where user u has applied to the job j recommended to her at least once within the duration of the online test.

Figure 1 presents results of the test by showing the percentage change in the unique user-job view pairs for the virtual profile models relative to the baseline, on each individual day of the test. Figure 2 presents the percentage change in the unique user-job application pairs for virtual profile models relative to the baseline. As expected, both segmented and non-segmented virtual profile models beat the baseline in the performance. More importantly, the proposed segmented virtual profile model performs better than the non-segmented virtual profile model in both job views and applications.

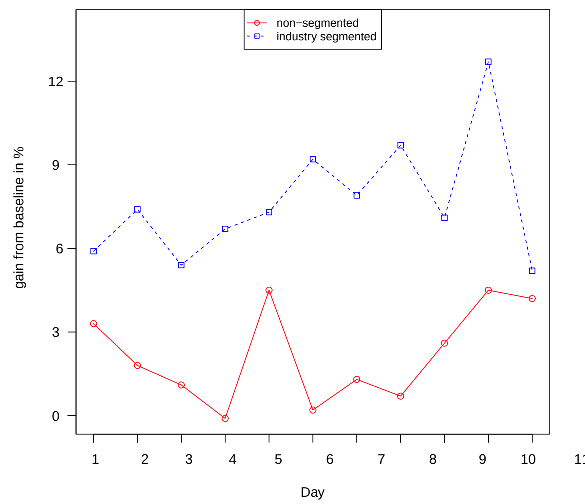

Figure 1: Online Job Views Performance improvement over baseline for virtual profile models

Overall, the industry segmented virtual profile model outperforms the baseline by 10.4% in applications and 6.5% in views (p value $< .0008$). The non-segmented virtual profile model outperformed the baseline by 4.3% in applications and 2% in views (p value $< .003$). Our experiment's main goal is to study the impact of segmented virtual profile on recommendation performance. From these results we can say that segmented virtual profile improves the quality of the job recommender system at LinkedIn.

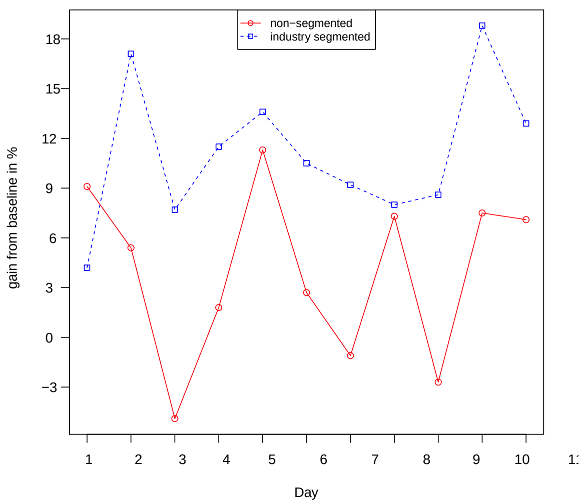

Figure 2: Online Job Application Performance improvement over baseline for virtual profile models

4.3 Analysis

In this section, we conduct a series of in-depth investigation to study the reasons behind the improvements achieved through the segmented virtual profile model, as well as if there is any possible side effect incurred by the model.

As we discuss in the previous section, the segmented virtual profile model performs better in terms of both the metrics in the offline setting, and the volume of job views and applications in the online settings. Virtual profile's better performance may attribute to the following two reasons:

1. Users apply to more jobs when served by the virtual profile model, and/or

2. More users engage in applying to jobs when served by virtual profile model.

The first point can be concluded from the online performance shown in Figure 1 and 2. To verify the second point, we analyze the collected data from the online evaluation. We collect activities of all members in the bucket served with the baseline model and the bucket served with the virtual profile model.

We conduct a paired t-test between the distribution of users who applied to n number of jobs from the two buckets. The paired t-test shows that more users apply for a job in the virtual profile bucket (p value $= .005$). This is indicating that virtual profile model helps jump start users in the application process. In other words, the virtual profile improves the whole job application ecosystem by getting more users to start interacting with the job recommender system.

We further analyze the results to verify if the virtual profile may cause any side effect on the overall job recommendation ecosystem. One possible side effect could be that virtual profile based models give higher preference to jobs having virtual profiles than jobs that do not have virtual profiles. This problem can be seen as a cold start problem. Let us call jobs without virtual profiles (jobs without applicants) as *cold jobs* and jobs with virtual profiles (jobs with applicants) as *warm jobs*. The cold start problem occurs when warm jobs are shown to more users than cold jobs due to the fact that cold jobs has missing virtual profile features.

To verify this we partitioned the jobs into two subgroups, jobs with virtual profiles (warm jobs) and jobs without virtual profiles (cold jobs), and analyze the impact on the impressions (i.e., the showing of the job) and applications separately from the data collected in the virtual profile model bucket.

Results show that, for warm jobs, number of impressions increase by 1.63%, while number of applications increase by 1%. However, for cold jobs, impressions decrease by 7.8%, but applications increase by 6.4%. The increase in applications for cold jobs is seemingly larger than the warm jobs, which can be explained by the fact that the base number of applications for cold jobs is low. However, the increase in number of applications for cold jobs can be attributed to the following two reasons.

1. It is mostly irrelevant non-virtual profile jobs that are getting replaced by the jobs having virtual profiles. A very relevant job is still getting impressed to the user even though it does not yet have the virtual profile.

2. Since virtual profile based model is improving the overall quality of the recommendations, it may result in higher user interactions, which may even help the jobs not having virtual profile.

In other words, although virtual profiles are generated only for warm jobs, both warm jobs and cold jobs are benefited from them. Low-quality cold jobs have a fewer chance to surface in the recommendation because of being replaced by warm jobs with higher feature coverage due to virtual profiles. And overall, users get more engaged after interacting with high-quality jobs. Therefore, in general, the adoption of segmented virtual profile resulted in a more healthy job ecosystem.

5. CONCLUSION AND FUTURE WORK

We presented an improved content feature extension method called segmented virtual profiles. The goal of virtual profiles is to provide a means to tap into rich-content information from one type of entity and propagate features extracted therein to other related entities that may suffer from relative data scarcity. The segmented virtual profile addresses the data imbalance problem in the feature selection process. Totally irrelevant examples in the majority class decrease the signal-to-noise ratio in the feature selection process. The proposed strategies for user segmentation provide a warranty against the data imbalance by constraining the non-relevant example space to a smaller set that only contains the most likely good quality examples. Evaluation on real-world recommender system shows that the segmented virtual profile performs significantly better than baselines.

One possible future direction is to investigate ways to improve label quality within a user segment. Since only observed positive user-item interactions are deemed to be positive examples in feature selection, negative interactions are increasingly vulnerable to mislabeling. This is due to the fact that the user segment consists of users who are *a priori* likely to positively interact with the item in the first place (a user may not have positively interacted with the item simply because she had not seen it yet). Previous research showed that noise in the labeled data can cause serious degradation in feature selection performance [5]. Therefore, one important future direction is to look at how implicit user feedback can be utilized to generate better quality negative labeling within the user segment. Another direction is to generate virtual profiles for new jobs with few applicants. One possible way is to backfill the virtual profiles of a job using those of similar jobs.

6. REFERENCES

[1] R. M. Bell, Y. Koren, and C. Volinsky. The BellKor solution to the Netflix Prize.

[2] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, Nov. 2002.

[3] A. Felfernig. Koba4ms: selling complex products and services using knowledge-based recommender technologies. In *CEC 2005. Seventh IEEE International Conference on E-Commerce Technology*, pages 92–100, July 2005.

[4] A. Felfernig, M. Schubert, M. Mandl, F. Ricci, and W. Maalej. Recommendation and decision technologies for requirements engineering. In *Proceedings of the 2Nd International Workshop on Recommendation Systems for Software Engineering*, RSSE '10, pages 11–15, New York, NY, USA, 2010.

[5] B. Frénay, G. Doquire, and M. Verleysen. Estimating mutual information for feature selection in the presence of label noise. *Computational Statistics & Data Analysis*, 71:832–848, 2014.

[6] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, July 2001.

[7] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182, Mar. 2003.

[8] J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, CSCW '00, pages 241–250, New York, NY, USA, 2000.

[9] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[10] H. Liu, M. Amin, B. Yan, and A. Bhasin. Generating supplemental content information using virtual profiles. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pages 295–302, New York, NY, USA, 2013.

[11] J. Malinowski, T. Keim, O. Wendt, and T. Weitzel. Matching people and jobs: A bilateral recommendation approach. In *System Sciences, 2006. HICSS '06. Proceedings of the 39th Annual Hawaii International Conference on*, volume 6, pages 137c–137c, Jan 2006.

[12] J. Malinowski, T. Weitzel, and T. Keim. Decision support for team staffing: An automated relational recommendation approach. *Decision Support Systems*, 45(3):429–447, June 2008.

[13] D. Mladenic. *Machine learning on non-homogeneous, distributed text data.* PhD thesis, University of Ljubljana, Faculty of Computer and Information Science, 1998.

[14] H. T. Ng, W. B. Goh, and K. L. Low. Feature selection, perceptron learning, and a usability case study for text categorization. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '97, pages 67–73, New York, NY, USA, 1997.

[15] I. Paparrizos, B. B. Cambazoglu, and A. Gionis. Machine learned job recommendation. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 325–328, New York, NY, USA, 2011.

[16] M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13(5-6):393–408, Dec. 1999.

[17] M. J. Pazzani and D. Billsus. The adaptive web. chapter Content-based recommendation systems, pages 325–341. Springer-Verlag, Berlin, Heidelberg, 2007.

[18] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, CSCW '94, pages 175–186, New York, NY, USA, 1994.

[19] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 285–295, 2001.

[20] A. Singh, C. Rose, K. Visweswariah, V. Chenthamarakshan, and N. Kambhatla. Prospect: A system for screening candidates for recruitment. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 659–668, New York, NY, USA, 2010.

[21] A. Singhal, M. Mitra, and C. Buckley. Learning routing queries in a query zone. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '97, pages 25–32, New York, NY, USA, 1997. ACM.

[22] J. Wang, Y. Zhang, C. Posse, and A. Bhasin. Is it time for a career switch? In *Proceedings of the 22Nd International Conference on World Wide Web*, WWW '13, pages 1377–1388, Republic and Canton of Geneva, Switzerland, 2013.

[23] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1-2):69–90, May 1999.

[24] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 412–420, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[25] Z. Zheng, X. Wu, and R. Srihari. Feature selection for text categorization on imbalanced data. *SIGKDD Explor. Newsl.*, 6(1):80–89, June 2004.