Bayesian Multi-View Models for Member-Job Matching and Personalized Skill Recommendations

Abhinav Maurya, Rahul Telang
Heinz College of Information Systems and Public Policy
Carnegie Mellon University, Pittsburgh, PA – 15213
Email: {amaurya,rtelang}@andrew.cmu.edu

***Abstract*—Inefficiencies in the labor market such as friction in matching members to jobs and the existence of skill gaps in various sectors of the economy are considered to be major problems facing economies today. The central premise of our work is that increasing the productivity of a member of the workforce (and thereby of the economy as a whole) crucially depends on identifying and recommending skills whose acquisition will yield the highest utility gains for that member. To this end, we develop novel unsupervised Bayesian multi-view models *BayesMatch* and *NpBayesMatch* which can match members to other similar members as well as relevant jobs using shared features. The matching step is followed by a skill recommendation step *SkillR* which makes demand-based skill recommendations to members. Our extensive quantitative evaluation using a rich dataset comprised of member profiles and job postings from LinkedIn suggests that skill recommendations made by our algorithm are highly correlated with skills demanded in heldout future jobs compared to those made by traditional collaborative filtering algorithms that do not utilize information about skill demand. This indicates that either members of the workforce do not have skills demanded by jobs or do not have enough information about which are the best skills to signal for competing in the labor market.**

I. Introduction

The exchange of skilled services is at the heart of the digital economy. In a rapidly evolving job market, many sophisticated skills gain quick popularity and become desired by employers for rising niche careers. For example, according to `indeed.com`, the demand for "deep learning" and "IoT" increased by 36,738.71% and 5,075.88% respectively, while that for "Java Programming" and "Cloud Computing" declined by 53.31% and 40.16% respectively between 2012 and 2016. Thus, individuals in the workforce need to continuously learn new skills to be relevant and competitive at work.

However, identifying which skills one should acquire can be a challenge in the absence of information about skill demand in the kind of jobs one wishes to be employed in. As a result, individuals make suboptimal decisions about the skills they choose to acquire. These micro-suboptimalities manifest in the larger economic picture as *skill gaps* in the labor market – a phenomenon characterized by a divide between the training of the workforce and the skills demanded by jobs.

Issues such as lack of information surrounding the decision of skill acquisition and pervasive skill gaps in labor markets are not limited to employees and corporations. US Department of Labor spends hundreds of millions of dollars each year in programs that train unemployed and displaced workers. Thus, equipping the workforce with employable skills that can improve their employment outcomes is also a huge public policy challenge.

A first step toward closing the skill gaps in labor markets consists of professionals identifying the optimal skills to acquire – skills which are heavily demanded by jobs but not yet adequately supplied by the workforce. However, it is virtually impossible for any individual to measure the expected benefits and potential costs of skill acquisition effectively. A natural solution to this problem is to harness the large-scale data associated with online professional social networks and job listing sites such as LinkedIn, Indeed.com, Google for Jobs, XING, etc. to help members make informed decisions about skill acquisition.

The crux our research is that increasing the productivity of a member of the workforce crucially depends on identifying skills whose acquisition will yield the highest utility gains for the member given their current skill set. In short, we want to quantify the additional utility that acquisition of a skill brings to an individual given his existing skill set, and use it as a criterion to recommend skills that members of a professional social network should acquire.

In this paper, we develop novel unsupervised Bayesian multi-view models *BayesMatch* and *NpBayesMatch* using Dirichlet prior [1] and Dirichlet process [2] clustering respectively to match members to other similar members as well as relevant jobs. Matching a member to other members serves to identify recommendable skills that the member can acquire feasibly since they have already been acquired by other similar members. The identification of jobs relevant to a member serves to identify the skills demanded by these jobs that the member lacks. Subsequently, we develop a personalized skill recommendation system *SkillR* based on the demand of skills in the jobs most relevant to the member.

A. Summary of Contributions

- We investigate the use of shared views in Bayesian multi-view clustering models for matching records of two different types with shared attributes. In our particular applications, these two types of records correspond to members of a professional networking site and open job listings on the site, and an example of a shared attribute is the college major of a member and that desired by a job.

- We propose skill recommendation mechanisms based on the demand of skills in the labor market. Our findings suggest that such demand-based skill recommendations outperform traditional collaborative filtering-based mechanisms in preparing workers for the jobs of the future.

Section II describes the overall skill recommendation system. In section III, we describe our parametric and non-parametric Bayesian models for matching members and jobs. Section IV describes our skill scoring and recommendation methodology. Section V describes the rich dataset from LinkedIn which we use to evaluate our methodology. Section VI describes our evaluation approach and results. Section VII provides an overview of related literature. Section VIII concludes with a discussion of potential avenues for future work.

II. Overall Algorithm

Figure (1) shows a schematic diagram of the proposed skill recommendation system. Our skill recommendation algorithm consists of two stages. Firstly, using a joint probabilistic model for member-job clustering, we match members to other similar members as well as jobs relevant to them. In the second stage, for each member within a cluster, we first identify recommendable skills that they can feasibly acquire and then rank the recommendable skills based on their demand in the jobs relevant to the member.

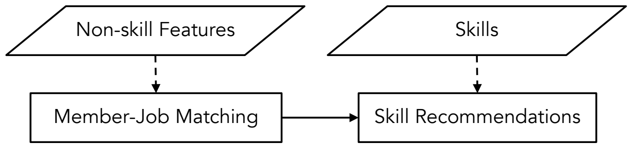


Fig. 1: A schematic diagram of our skill recommendation system. In the first stage, the system jointly clusters similar members and jobs using *BayesMatch* or *NpBayesMatch*. Skills associated with members or jobs are not used in the matching phase. In the second stage *SkillR*, job demand-based skill scores are used to recommend skills to members.

Stratification: Before the member-job matching and skill recommendation steps, we stratify both members and jobs based on clean, low cardinality features. Such stratification (also called "static blocking" in statistical literature) has been employed in [3] to reduce the computational overhead of matching all record pairs by employing a heuristic partitioning of records. Specifically, we use the "city" and "industry" features to stratify members and jobs. The geographic location and industry has been used in previous literature for static blocking when matching members and jobs [4]. Stratification parallelizes the dataset into non-overlapping shards, such that each shard contains the members and jobs for a particular city and industry. Any further steps can be executed in parallel among the shards using frameworks for distributed processing such as Apache Spark [5].

Symbol	Explanation
$N^{(m)}$	Number of members
$N^{(j)}$	Number of jobs
$N^{(s)}$	Number of unique skills
K	Number of clusters in parametric clustering or initialized clusters in nonparametric clustering per stratum
C	Total number of clusters after BayesMatch and NpBayesMatch
$F^{(m)}$	number of member-specific features
$F^{(j)}$	number of job-specific features
$F^{(c)}$	number of features common to both members and jobs
α	Dirichlet hyperparameter in parametric clustering or GEM hyperparameter in nonparametric clustering
$\boldsymbol{\theta}^{(m)}$	global multinomial parameter for distribution over member clusters
$\boldsymbol{\theta}^{(j)}$	global Multinomial parameter for distribution over job clusters
$\beta_i^{(m)}$	Dirichlet hyperparameter for distribution over i^{th} member-specific feature
$\beta_i^{(j)}$	Dirichlet hyperparameter for distribution over i^{th} job-specific feature
$\beta_i^{(c)}$	Dirichlet hyperparameter for distribution over i^{th} common feature
$\boldsymbol{\phi}_{ki}^{(m)}$	parameter for k^{th} cluster's Multinomial distribution over the i^{th} member-specific feature
$\boldsymbol{\phi}_{ki}^{(j)}$	parameter for k^{th} cluster's Multinomial distribution over the i^{th} job-specific feature
$\boldsymbol{\phi}_{ki}^{(c)}$	parameter for k^{th} cluster's Multinomial distribution over the i^{th} common feature
$z_i^{(m)}$	cluster assignment for i^{th} member
$z_i^{(j)}$	cluster assignment for i^{th} job
$f_i^{(m)}$	a member's i^{th} member-specific feature
$f_i^{(j)}$	a job's i^{th} job-specific feature
$f_i^{(c)}$	a member/job's i^{th} common feature

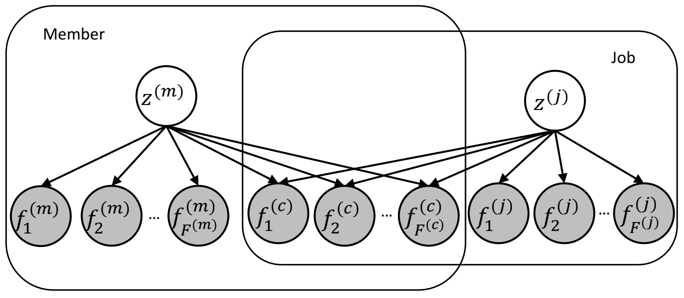
TABLE I: Notation used in this paper

III. Matching Members to Jobs

Each stratum created by static blocking is still very diverse in terms of the members and jobs included in it. For example, a stratum consisting of members from software industry and belonging to San Francisco might consist of software engineers, data scientists, cloud infrastructure architects, software company recruiters, etc. The jobs included in the stratum will also have a similar variety corresponding to the diverse roles in the industry. Additionally, a stratum from a particular city and industry can also be massive consisting of millions of members and jobs. In order to make skill recommendations, we need to further partition the stratum into clusters such that each cluster has a fairly homogeneous mixture of members and jobs. This is the goal of our current step.

Before we outline the Bayesian model that underlies *BayesMatch* and *NpBayesMatch*, we look at some alternative co-clustering algorithms that could be used but did not satisfy the requirements of our problem:-

- **K-Means Clustering:** A naive approach would be to use a popular clustering algorithm like K-Means on members and jobs separately, and then use a cluster similarity metric to match clusters. However, K-Means handles unstructured, missing data poorly, and missing data is



Member
Job
$z^{(m)}$
$z^{(j)}$
$f_1^{(m)}$ $f_2^{(m)}$ ... $f_{F^{(m)}}^{(m)}$
$f_1^{(c)}$ $f_2^{(c)}$ ... $f_{F^{(c)}}^{(c)}$
$f_1^{(j)}$ $f_2^{(j)}$ ... $f_{F^{(j)}}^{(j)}$

Fig. 2: A multi-view clustering perspective for matching members and jobs. Members have member-specific features $f^{(m)}$ (of which there are $F^{(m)}$), jobs have job-specific features $f^{(j)}$ (of which there are a total of $F^{(j)}$), and members and jobs share common features denoted by $f^{(c)}$ (there are $F^{(c)}$ such features). Common features in each cluster are assumed to be generated from common distributions (shared views) which leads to member-job matching.

pervasive in the member profiles as well as job listings. Also, It is not generally necessary that each member cluster has a corresponding cluster of relevant jobs.

- **Propensity Score Matching:** We can use propensity score matching [6] to match members whose covariates are similar but who differ on an acquired skill. However, we would have to run 29 thousand independent high-dimensional logistic regressions (one for each skill) per city and industry which is computationally prohibitive.
- **Deeper Stratification:** We considered a deeper stratification scheme for both members and jobs based on 5 features: city, industry, degree, specialization (i.e. college major), and number of years of experience. Due to the high cardinality of categorical features in our dataset, the size of obtained strata dwindles sharply as we stratify on more features, despite access to enormous amounts of data (Table II).

In order to address the above shortcomings, we designed two novel unsupervised Bayesian multi-view models – *BayesMatch* and *NpBayesMatch* – which match members and jobs using shared views between the two types of records. As shown in figure (2), certain features $f^{(m)}$ are specific to members (e.g. schools attended), certain features $f^{(j)}$ are specific to jobs (e.g. job function), while certain features $f^{(c)}$ are commonly shared between members and jobs (e.g. education degrees and majors acquired by member and those required/desired for a job). Thus, members and jobs share a subset of common attributes, which are used to match members and jobs by placing both in the same latent clusters. The result is a co-partitioning of members and jobs within each stratum. The resulting member-job clusters are such that members in a cluster are similar on all their features $f^{(m)}$ and $f^{(c)}$, jobs in a cluster are similar on all their features $f^{(j)}$ and $f^{(c)}$, and members and jobs in a cluster are matched based on similarity of the common features $f^{(c)}$ and $f^{(c)}$ alone. If

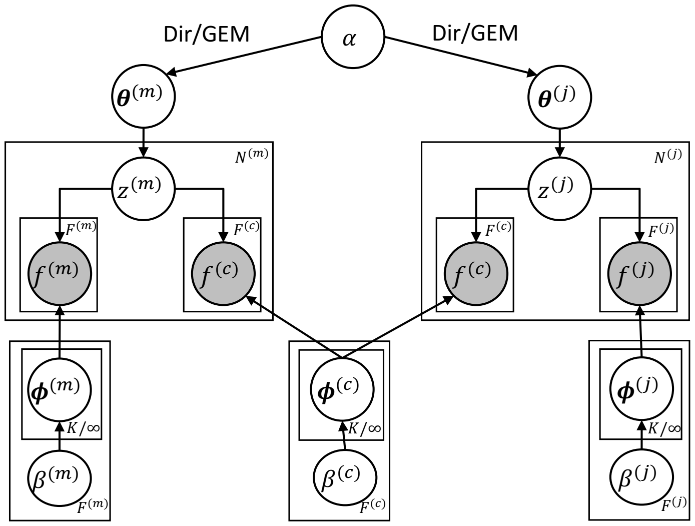


Fig. 3: Graphical model for BayesMatch and NpBayesMatch with $N^{(m)}$ members, and $N^{(j)}$ jobs. The number of clusters is K for BayesMatch and potentially ∞ for NpBayesMatch. Members have member-specific features $f^{(m)}$ (of which there are $F^{(m)}$), jobs have job-specific features $f^{(j)}$ (of which there are a total of $F^{(j)}$), and members and jobs share common features denoted by $f^{(c)}$ (there are $F^{(c)}$ such features). For BayesMatch, $\boldsymbol{\theta}^{(m)}, \boldsymbol{\theta}^{(j)} \sim \text{Dir}(\alpha)$. For NpBayesMatch, $\boldsymbol{\theta}^{(m)}, \boldsymbol{\theta}^{(j)} \sim \text{GEM}(\alpha)$.

a member and a job are assigned to the same cluster, they are considered to be matched, in the sense that the job is likely relevant to the member's career interests, and the skills listed as desirable by the job are worth acquiring for the member.

In our dataset, all features except one are categorical, and therefore modeling their distributions using Dirichlet conjugate priors is natural. The single non-categorical feature – binned number of years of experience – is ordinal with 6 levels, and we have chosen to model it as a categorical feature in the interest of model simplicity.

To perform the described co-partitioning of members and jobs, we employ a joint probabilistic clustering model shown in the plate diagram in figure (3). The data generating process shown in the graphical model is as follows:-

1) For each cluster $k = 1..K$ (where K is the number of clusters)
   a) For i^{th} member-specific feature ($i = 1..F^{(m)}$), sample Multinomial parameter $\boldsymbol{\phi}_{ki}^{(m)} \sim \text{Dir}(\beta_i^{(m)})$. Here, Dir denotes the Dirichlet distribution.
   b) For i^{th} job-specific feature ($i = 1..F^{(j)}$), sample Multinomial parameter $\boldsymbol{\phi}_{ki}^{(j)} \sim \text{Dir}(\beta_i^{(j)})$.
   c) For i^{th} common feature ($i = 1..F^{(c)}$), sample Multinomial parameter $\boldsymbol{\phi}_{ki}^{(c)} \sim \text{Dir}(\beta_i^{(c)})$.
2) Sample the global distribution over clusters for members $\boldsymbol{\theta}^{(m)} \sim \text{Dir}(\alpha)$ for *BayesMatch* or $\boldsymbol{\theta}^{(m)} \sim \text{GEM}(\alpha)$ for *NpBayesMatch*. GEM stands for the stick-breaking

distribution of a Dirichlet process [2].

3) Sample the global distribution over clusters for jobs $\boldsymbol{\theta}^{(j)} \sim \text{Dir}(\alpha)$ for *BayesMatch* or $\boldsymbol{\theta}^{(j)} \sim \text{GEM}(\alpha)$ for *NpBayesMatch*.
4) For each member $i = 1..N^{(m)}$ (where $N^{(m)}$ is total number of members)
   a) Generate i^{th} member's cluster assignment $z_i^{(m)} \sim \text{Cat}(\boldsymbol{\theta}^{(m)})$. Here, Cat denotes the Categorical distribution.
   b) Generate j^{th} member-specific feature $f_{ij}^{(m)}$ using $z_i^{(m)}$: $f_{ij}^{(m)} \sim \text{Cat}(\boldsymbol{\phi}_{kj}^{(m)})$ where $k = z_i^{(m)}$.
   c) Generate j^{th} common feature $f_{ij}^{(c)}$ using $z_i^{(m)}$: $f_{ij}^{(c)} \sim \text{Cat}(\boldsymbol{\phi}_{kj}^{(c)})$ where $k = z_i^{(m)}$.
5) For each job $i = 1..N^{(j)}$ (where $N^{(j)}$ is total number of jobs)
   a) Generate i^{th} job's cluster assignment $z_i^{(j)} \sim \text{Cat}(\boldsymbol{\theta}^{(j)})$.
   b) Generate j^{th} position-specific feature $f_{ij}^{(j)}$ using $z_i^{(j)}$: $f_{ij}^{(j)} \sim \text{Cat}(\boldsymbol{\phi}_{kj}^{(j)})$ where $k = z_i^{(j)}$.
   c) Generate j^{th} common feature $f_{ij}^{(c)}$ using $z_i^{(j)}$: $f_{ij}^{(c)} \sim \text{Cat}(\boldsymbol{\phi}_{kj}^{(c)})$ where $k = z_i^{(j)}$.

The model is symmetric in its treatment of members and jobs. Members and jobs are matched due to common distributions $\boldsymbol{\phi}_{ki}^{(c)}, k = 1..K, i = 1..F^{(c)}$ that generate the common features for both members and jobs. Members and jobs which are similar on the common features end up being matched together into the same cluster.

Since we use conjugate Dirichlet-Multinomial distributions to describe our model, MCMC inference using collapsed Gibbs sampling [7], [2] involves simple mathematical operations and sampling from a categorical distribution. Thus, it is simple to implement and allows us to draw samples from the posterior distribution of model parameters and latent variables.

The conditional distribution for sampling a latent cluster assignment for a member in *BayesMatch* is:

$$p(z_i^{(m)} = k | z_{-i}^{(m)}, z^{(j)}, \alpha, \boldsymbol{\beta}) = (m_{k,-i} + \alpha) \cdot \prod_{x=1}^{F^{(m)}} \frac{(n_{kf_{xv,-i}^{(m)}} + \beta_i^{(m)})}{\sum_{v=1}^{V}(n_{kf_{xv,-i}^{(m)}} + \beta_i^{(m)})} \cdot \prod_{x=1}^{F^{(c)}} \frac{(n_{kf_{xv,-i}^{(c)}} + \beta_i^{(c)})}{\sum_{v=1}^{V}(n_{kf_{xv,-i}^{(c)}} + \beta_i^{(c)})}$$

Here, $m_{k,-i}$ is the count of datapoints currently assigned to cluster k without including the current datapoint. $n_{kf_{xv,-i}^{(m)}}$ is the number of times a member with x^{th} member-specific feature equal to v is assigned to the k^{th} cluster excluding current datapoint, and $\beta_i^{(m)}$ is the Dirichlet hyperparameter corresponding to the x^{th} member-specific feature. $n_{kf_{xv,-i}^{(c)}}$ and $\beta_i^{(c)}$ are defined similarly for common features. The conditional distribution over latent clusters for jobs can be similarly derived [7].

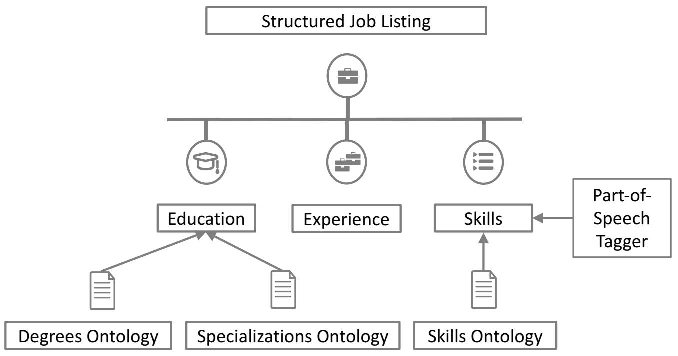


Fig. 4: Extracting structured features from unstructured textual job postings. Curated ontologies for degrees, specializations (college majors), and skills were used to aid in the extraction. A part-of-speech tagger was used to prune spurious extracted skills.

The sampling procedure for *NpBayesMatch* is similar [2]. We sample a new cluster with its own parameters with a prior probability mass $p_{new} = \frac{\alpha}{N^{(m)}+N^{(j)}+\alpha-1}$ and sample one of the old clusters with the remaining prior probability mass $p_{old} = 1 - p_{new}$.

Each of the S strata generates K clusters (where K is an input specified by us). Thus, we obtain a total of $S \cdot K$ clusters after the clustering step. We further filter out some of these clusters so that each cluster contains a minimum number of members and jobs for making skill recommendations. Let C denote the number of clusters finally obtained after cluster filtering.

In summary, our choice of matching procedure can be either parametric (*BayesMatch*) or nonparametric (*NpBayesMatch*). It consists of a stochastic model whose inference produces a co-partitioning of members and jobs.

IV. Scoring and Recommending Skills

Each of the C clusters yields a set of similar members and jobs relevant to them. Before we outline the procedure to recommend skills, we examine the desirable properties that a skill recommendation should have:-

- **Feasibility of Skill Acquisition:** A good skill recommendation system will ensure that the skills it recommends can be acquired by the person by incurring a reasonable cost.
- **Utility of Skill Acquisition:** Not all acquirable skills are worth acquiring. An expert programmer who knows many programming languages will gain little by learning another programming language unless it is heavily demanded by jobs. Therefore, a skill should improve a member's prospects in the job market and make him more attractive for jobs relevant to him.

We tackle these two requirements by adopting a two-step approach to skill recommendation:-

Identifying Recommendable Skills: Consider two similar members A and B within the same cluster. We argue that if member A has already acquired a particular skill s, it cannot be too cost-prohibitive for member B to acquire s. Since A and B are similar on their covariates, they act as each other's counterfactuals for a skill one of them has acquired but the other has not. For each member in a cluster, his set of recommendable skills is the set of all skills that he does not possess but other members in the cluster do. In other words, the set of recommendable skills $S_i^{(r)}$ for the i^{th} member in the cluster is the set difference between the union of skill sets of all members in the cluster and the skill set of the i^{th} member.

Ranking Skill Recommendations: We examine four different skill recommendation mechanisms:

- **SD-ALS:** In this approach, we perform customized skill demand-based collaborative filtering in a cluster using ALS (Alternating Least Squares). We first perform ALS on the job-skill matrix to obtain latent representation of all the skills. To make skill recommendations to a member, we must find her projection on the latent factors discovered by ALS. Let the rank of the ALS decomposition be R. Consider $\boldsymbol{W}$ to be the $(N^{(s)}, R)$-sized matrix whose rows are projections of skills onto the R latent factors learned by ALS. If $\mathcal{I}$ denotes the indices of the skill set $S_i^{(m)}$ of the i^{th} member, then the latent projection $\boldsymbol{p}$ of the member can be found by solving the following least squares optimization:

$$\underset{\boldsymbol{p}}{\operatorname{argmin}} \sum_{i \in \mathcal{I}} (\boldsymbol{W}_{i\cdot} \cdot \boldsymbol{p} - 1)^2 \tag{1}$$

  Here, $\boldsymbol{W}_{i\cdot}$ is the i^{th} row of $\boldsymbol{W}$ corresponding to the latent representation of the i^{th} skill. The problem has a simple solution:

$$\boldsymbol{p} = \frac{\sum_{i \in \mathcal{I}} \boldsymbol{W}_{i\cdot}}{\sum_{i \in \mathcal{I}} \boldsymbol{W}_{i\cdot} \cdot \boldsymbol{W}'_{i\cdot}} \tag{2}$$

  The latent projection $\boldsymbol{p}$ of a member is proportional to the mean of the latent projections of all the skills she has already acquired. This representation can now be used to rank other skills that the member has not yet acquired based on the Euclidean distance of their latent projections from $\boldsymbol{p}$.
- **SS-ALS:** In this approach, we perform traditional collaborative filtering between members and skills in a cluster using ALS-based matrix completion.
- **SD-NMF:** This approach recommends skills using their demand in jobs based on NMF (Non-negative Matrix Factorization) instead of ALS. We first perform NMF on the job-skill matrix to obtain non-negative latent representation of all the skills. Similar to problem 1, the latent projection $\boldsymbol{p}$ of a member can be found by solving the following constrained least squares optimization:

$$\underset{\boldsymbol{p} \geq 0}{\operatorname{argmin}} \sum_{i \in \mathcal{I}} (\boldsymbol{W}_{i\cdot} \cdot \boldsymbol{p} - 1)^2 \tag{3}$$

Statistic	Value
Number of Profiles	414 million
Number of Jobs	58 million
Number of Skills	29 thousand
Number of Companies	7.7 million
Number of Unique Job Titles	98 thousand
Number of Unique Schools	95 thousand
Number of Unique Degrees	158
Number of Unique Majors	7.6 thousand

TABLE II: Dataset Summary Statistics

  The solution $\boldsymbol{p}$ is given as:

$$\boldsymbol{p} = max\left(0, \frac{\sum_{i \in \mathcal{I}} \boldsymbol{W}_{i\cdot}}{\sum_{i \in \mathcal{I}} \boldsymbol{W}_{i\cdot} \cdot \boldsymbol{W}'_{i\cdot}}\right) \tag{4}$$

  Here, the maximum operation is an elementwise maximum that projects $\boldsymbol{p}$ obtained in equation 4 onto its non-negative constraint.
- **SS-NMF:** This approach performs NMF-based collaborative filtering between members and skills to make skill recommendations.

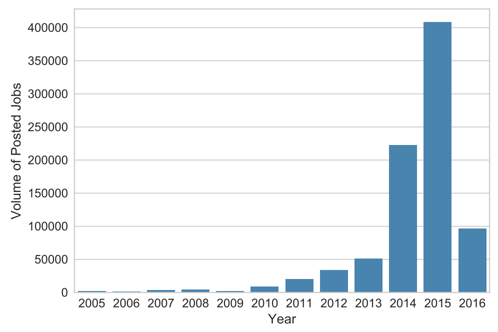
In each of the methods, we pick the top-k skills to recommend ($k = 5$ in our implementation).

V. Data

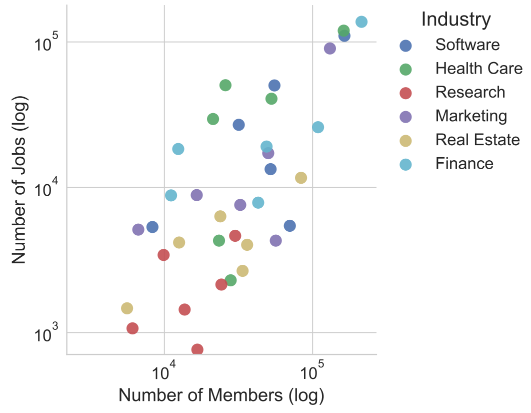
We use a rich dataset of professionals and job postings from LinkedIn to evaluate our skill recommendation system. LinkedIn provided us with large-scale data from the online professional profiles of its members, as well as the jobs posted on the site by recruiters or syndicated by other job posting sites. Some summary statistics of the dataset are presented in table II. With millions of member profiles and job listings, this dataset provides us an opportunity to investigate the nature of industrial skill gaps and their use in making personalized skill recommendations.

From each member's profile, we extract features such as the schools attended, the degrees obtained, past companies worked at and job titles held, total experience, skills acquired, etc. From each job posting, we derive features such as the educational degree and major required, experience level of the job, skills required by the job, etc. A major data engineering challenge we faced was the extraction of features from job postings that are abundantly available but are in the form of unstructured text. Instead of developing sophisticated skill extraction algorithms, we rely on member profiles that are already structured to curate ontologies for extracting features from job postings. We then use the curated ontologies as well as NLP techniques such as POS tagging [8] to extract features from job descriptions as shown in figure (4).

In our dataset, each profile and job is annotated with a city (derived from an associated zipcode) and an industry. We chose to focus on data from a diverse set of cities and industries listed in table III for the purpose of our analysis.



(a) Job Volume by Year



(b) Number of Members/Jobs

Fig. 5: Fig (a): Number of jobs by year in the dataset. Job volume available for only part of the year 2016. Fig (b): Number of members versus jobs in various strata.

INDUSTRIES	CITIES
COMPUTER SOFTWARE	SAN FRANCISCO
HOSPITAL AND HEALTH CARE	NEW YORK
FINANCIAL SERVICES	SEATTLE
REAL ESTATE	PITTSBURGH
ACADEMIC RESEARCH	MUMBAI
MARKETING AND ADVERTISING	NEW DELHI

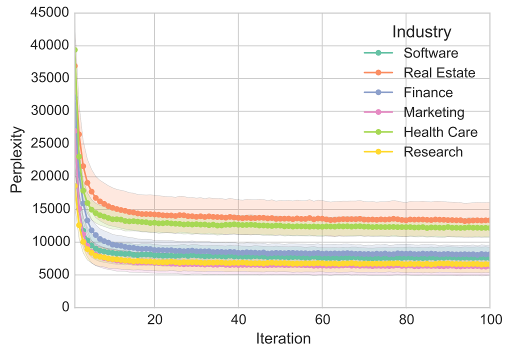
TABLE III: Industries and cities included in the evaluation.

VI. EVALUATION

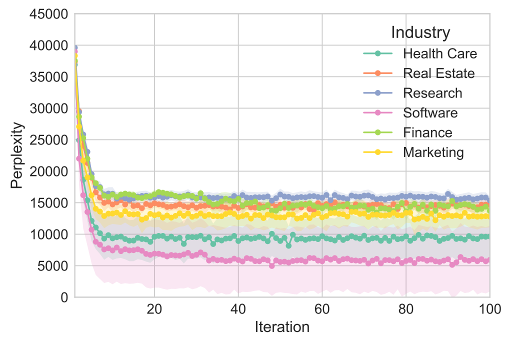
Designing and implementing matching and recommendation systems involves many experimental considerations. In this section, we describe the design choices we made in our algorithmic implementation, and discuss qualitative and quantitative results obtained on our rich, real-world dataset from a major online professional social networking and job listing site.

A. Experimental Setup

After selecting profiles and jobs that belong to the cities and industries listed in table III and extracting structured features from each member profile and job posting, we stratify members and jobs into partitions using the city and industry fields.

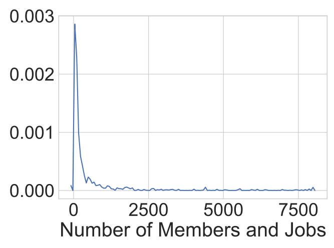


(a) BayesMatch

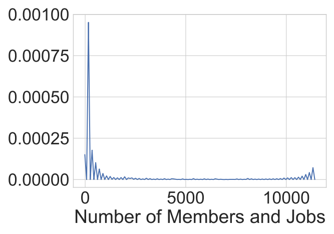


(b) NpBayesMatch

Fig. 6: Perplexity of BayesMatch and NpBayesMatch on heldout data. Each curve corresponds to an industry. This serves as a diagnostic measure to ensure that the model is a good fit for the data.



(a) BayesMatch



(b) NpBayesMatch

Fig. 7: Kernel Density Plot of cluster size (number of members and jobs).

The number of members and jobs in various strata is shown in figure (5b). The average number of members and jobs in a stratum is $47,840.50$ and $23,678.97$ respectively. We then run the member-job matching procedure within each stratum. For both *BayesMatch* and *NpBayesMatch*, we imposed a weak Gamma prior on the α hyperparameter: $\alpha \sim \text{Gamma}(1.0, 0.1)$. Following usual conventions for Dirichlet priors, we set the Dirichlet hyperparameter β for each feature as $\frac{1}{|V|}$ where V is the cardinality of the feature. The number of clusters within each stratum is heuristically initialized to $\frac{(\#members+\#jobs)}{100}$ where $\#members$ and $\#jobs$ are the number of members and

BayesMatch
Software Engineer
Software Engineer at Google
Computer Software Professional
Senior Software Engineer
Software Developer
Software Engineer at Facebook
Software Engineer at Apple
Director

(a) Cluster I

Manager
Assistant Manager
at Financial Services Professional
Independent Financial Services Professional
Vice President at Morgan Stanley
Associate at Deutsche Bank
Student at NMIMS
Associate

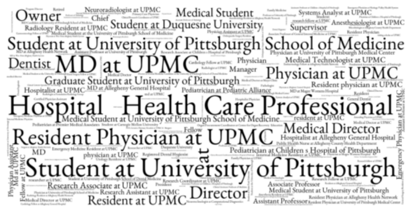
(b) Cluster II

Student at Stanford University
Independent Research Professional
Research Professional
Graduate Student at Stanford University
PhD Candidate at Stanford University
Director

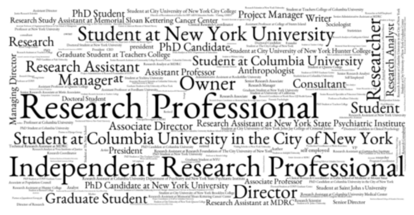
(c) Cluster III

NpBayesMatch
Director
Senior Director
Vice President
VP
Sr Director
Sr Manager
Account Executive
Executive Assistant
Product Manager

(d) Cluster I


Health Care Professional
Student at University of Pittsburgh
MD at UPMC
Hospital
Resident Physician at UPMC
Student at University of Pittsburgh
School of Medicine
Physician at UPMC

(e) Cluster II


Research Professional
Independent Research Professional
Student at New York University
Student at Columbia University in the City of New York
Student at Columbia University
Director
Graduate Student

(f) Cluster III

Fig. 8: Wordclouds of the profile headlines of members from four different example clusters discovered by *BayesMatch* and *NpBayesMatch*. Word size indicates frequency in dataset. These show the homogeneity of discovered clusters despite considerable diversity of careers in each industry.

jobs in the stratum respectively. The number of clusters doesn't change during inference for the parametric matching procedure *BayesMatch*. The nonparametric matching procedure *NpBayesMatch* however will adapt the total number of clusters in a data-driven fashion during model inference. Collapsed Gibbs sampling is run in parallel within each stratum for 100 iterations.

After the completion of the matching phase, we perform skill recommendations within each cluster. In order to evaluate the usefulness of our skill recommendations, we hold out 20% of the most recent jobs in each cluster and use only the remaining 80% of jobs in the recommendation process. In each cluster, we compute the cosine similarity between the empirical distribution of recommended skills and the empirical distribution of skills listed in heldout jobs, as well as the coverage of skill recommendations.

B. BayesMatch and NpBayesMatch Evaluation

Since our matching algorithm is a probabilistic Bayesian model, we examine the model perplexity with respect to the Gibbs iteration on 10% heldout data. The average perplexity within each industry is shown in figure (6). The perplexity decreases rapidly in the beginning before reaching a plateau after around 20 iterations for both *BayesMatch* and *NpBayesMatch*. This rapid convergence indicates that a good clustering is found quickly for our proposed models.

Figure (7) shows Kernel Density Estimation (KDE) plots of the cluster sizes obtained using *BayesMatch* and *NpBayesMatch*, where cluster size denotes the sum of the number of members and jobs in a cluster. The distribution over cluster sizes for *BayesMatch* is unimodal. On the other hand *NpBayesMatch*'s distribution is bimodal, with one mode at low cluster sizes and one mode at high cluster sizes. Thus, *NpBayesMatch* can discover clusters of varies sizes due to its nonparametric nature.

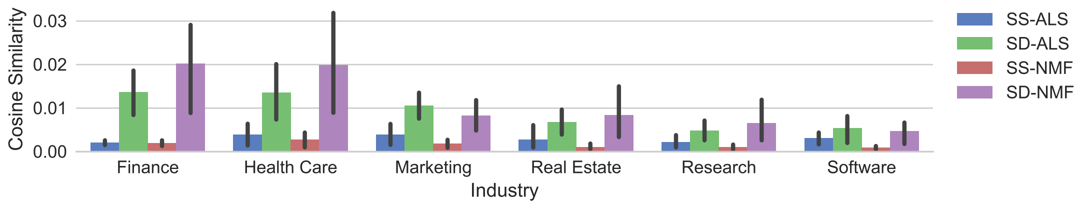
Figure (8) shows word clouds from the profile headlines of members from four different clusters discovered by *BayesMatch* and *NpBayesMatch*. Despite the variety of professionals found in each industry, the clusters discovered are fairly homogeneous in terms of careers of their members. This qualitative assessment of select clusters reveals that *BayesMatch* and *NpBayesMatch* successfully disentangle different professional cohorts requiring disparate skill sets within a single industry.

C. SkillR *Evaluation*

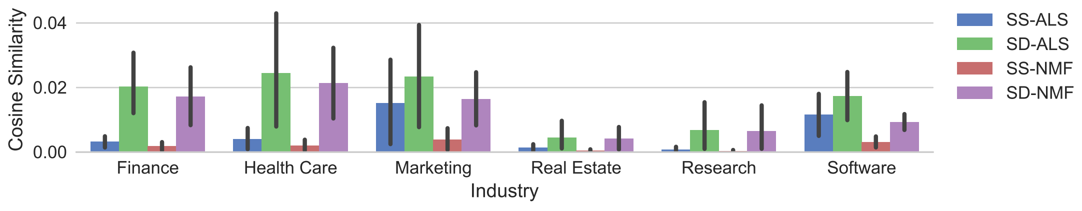
After the matching phase has finished matching members to their relevant jobs, the next stage *SkillR* performs skill recommendations. Here, we investigate both qualitatively and quantitatively the skill recommendations made by the methods described previously in section IV.

Since our skill recommendations are sourced from jobs, traditional measures for evaluating similarity-based collaborative filtering recommendations such as Mean Average Precision or Precision@k are not entirely appropriate for evaluating the recommendations. Instead, we consider if the recommended skills were also demanded in future heldout jobs. If so, members acquiring a recommended skill would be better prepared for future jobs in that they would have already acquired some of the skills that these jobs demanded instead of skills popular among members but not in great demand.

Figure (9) shows the cosine similarity between recommended skills and skills listed in heldout jobs for various industries. To obtain a cosine similarity at the industry level, a weighted average of the cosine similarities for different clusters in the industry is calculated using the number of

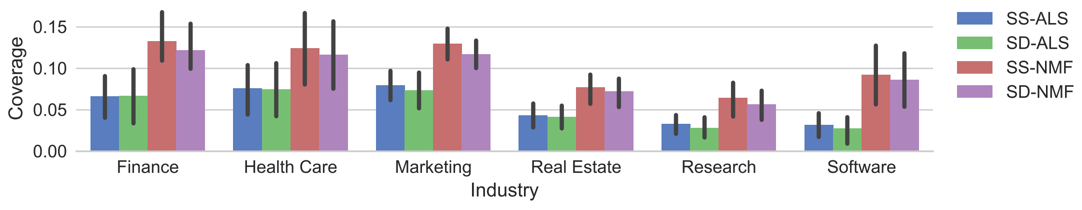


(a) BayesMatch

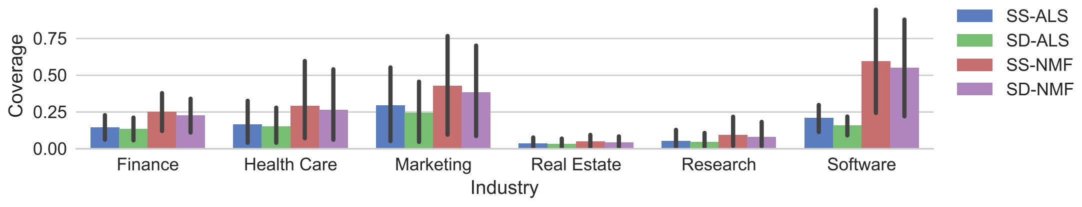


(b) NpBayesMatch

Fig. 9: Cosine similarity between the empirical distribution of recommended skills and heldout job skills under *BayesMatch* and *NpBayesMatch* for various industries.



(a) BayesMatch



(b) NpBayesMatch

Fig. 10: Coverage of various recommendation algorithms under *BayesMatch* and *NpBayesMatch* for various industries.

members in the cluster as the weight. This weighted average for each stratum is shown in figure (9) for the four recommendation mechanisms SS-ALS, SD-ALS, SS-NMF, and SD-NMF discussed in section IV. We observe that the cosine similarity between demand-based skill recommendations and the skills in heldout jobs is consistently higher across various industries than the corresponding cosine similarity for skills recommended by skill supply. The two skill demand-based recommendation approaches SD-ALS and SD-NMF perform comparably on the cosine similarity metric considering the error bars in the graph. Overall, our evaluation with cosine similarity indicates that personalized skill recommendations made by *SkillR* using skill demand can be indicative of skills that remain in high demand in future jobs.

Figure (10) shows the coverage of each of the four algorithms under the two matching mechanisms *BayesMatch* and *NpBayesMatch*. The coverage of corresponding skill supply and skill demand-based recommendations (e.g. SS-ALS and SD-ALS under NpBayesMatch) is comparable. This suggests that both recommendation approaches recommend the same subset of skills from the set of recommendable skills, but the scoring and importance of skills changes in the two types of recommendation.

In figure (11), we show differences between the recommendations made by SS-ALS and SD-ALS within example clusters. It shows wordclouds of the recommended skills for eight of our experimental configurations. The two rows show the recommended skills for two of the studied industries – Computer Software and Financial Services. The two columns on the left show results for *BayesMatch* contrasting SS-ALS and SD-ALS recommendations respectively. The two columns on the right show skill wordclouds for *NpBayesMatch* again

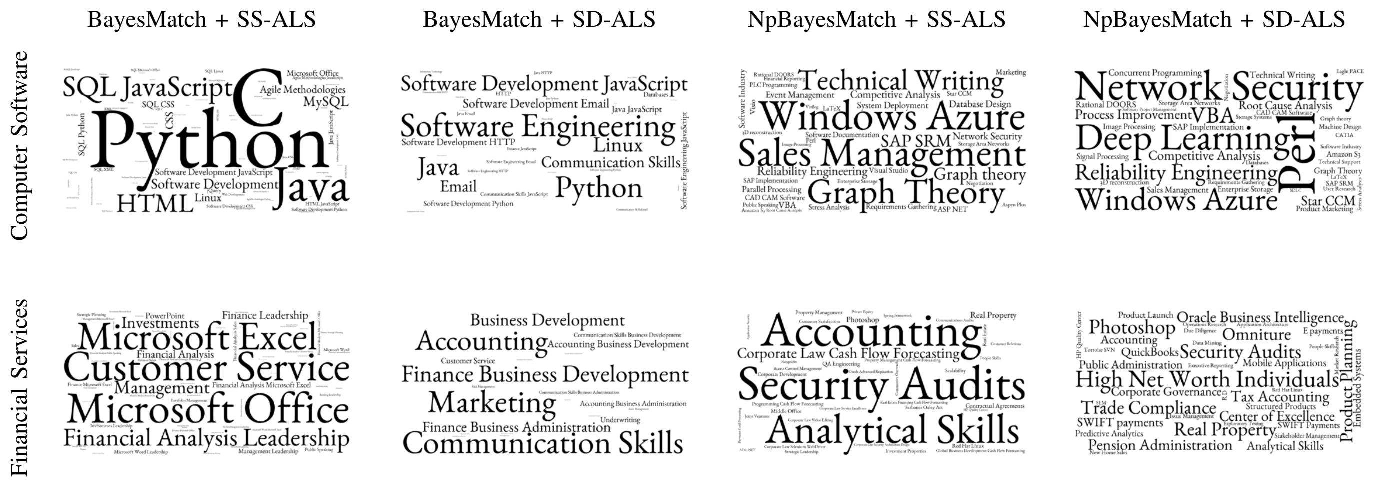


Fig. 11: Word clouds showing skills recommended by SS-ALS and SD-ALS in example clusters in two of the six industries. The first two columns show word clouds for *BayesMatch*, and the last two columns show it for *NpBayesMatch*. Word size indicates frequency of recommendation of a skill in the cluster. Notice that SS-ALS often makes generic recommendations, while SD-ALS recommendations are more precise and diverse.

showing differences in skill recommendations between SS-ALS and SD-ALS. The skill recommendation made by SS-ALS under both *BayesMatch* and *NpBayesMatch* are more generic and predictable compared to the recommendations made by SD-ALS. For example, the recommendations of "Microsoft Office" and "Microsoft Excel" made by SS-ALS in the Finanical Services industry might not be very useful to users since these are not distinguishing skills in the modern workplace. Also worth noting are the recommendations of "Deep Learning," "Network Security," "Reliability Engineering," and "Windows Azure" made by SD-ALS in one of the clusters in the Computer Software industry. It is known that these skills are heavily demanded in the industry, and are therefore valuable recommendations to members but were missed by SS-ALS which looked only at skill supply from members in the industry without considering the demand for skills in the jobs.

Figure (12) shows the empirical relationship between the Jaccard similarity of member-job skill sets in a cluster and the cosine similarity between the empirical distributions of SD-ALS recommended skills and heldout jobs' skills in each cluster. Linear regression curves are overlaid on the plot for each industry. As the cluster member-job skill overlap increases, the cosine similarity of skill recommendations by the SD-ALS procedure decreases for most clusters. This surprising phenomenon indicates that as members and jobs become increasingly similar in terms of skills, there are fewer opportunities to make valuable skill recommendations by looking at skill demand.

VII. Related Work

[9] provides a description of LinkedIn's member-job matching algorithm. However, the system requires supervision which can be costly to acquire especially on big datasets. On the contrary, our matching algorithms *BayesMatch* and *NpBayesMatch* are completely unsupervised and leverage member and job metadata to match them.

Similar to the Joint Confusion (JC) model in [10], our matching models *BayesMatch* and *NpBayesMatch* co-partition records of two different types based on shared Multinomial responses. The difference is that the JC model co-clusters items and evaluators based on shared decision distributions, whereas our models match members and jobs based on their shared features. [10] also employs Dirichlet processes to determine the number of clusters in a data-driven fashion.

[3] proposes a Bayesian approach for deduplicating records by mapping each record to a latent individual; records assigned to the same latent individual are considered to be duplicates. We extend this idea for use with two types of records - the "member" type and the "job" type which share a common subset of features used for matching them.

Our work on partitioning members and jobs for scaling up the recommendation procedure bears resemblance to the partition variant considered in [11]. However, our algorithm runs linearly in the dataset size and converges within a few iterations, unlike the partition algorithm in [11] which runs in polynomial time *after* being provided candidate recommendations.

There is a considerable body of work on partitioning bipartite (weighted) graphs to minimize the weight of edges cut during the partitioning [12], [13]. However, our problem differs from the bipartite graph partitioning problem in that we do not have edges between members and jobs. Partitioning bipartite graphs does not characterize the discovered partitions, whereas *BayesMatch* and *NpBayesMatch* provide a characterization of the discovered latent clusters, and have runtimes linear in the number of members and jobs.

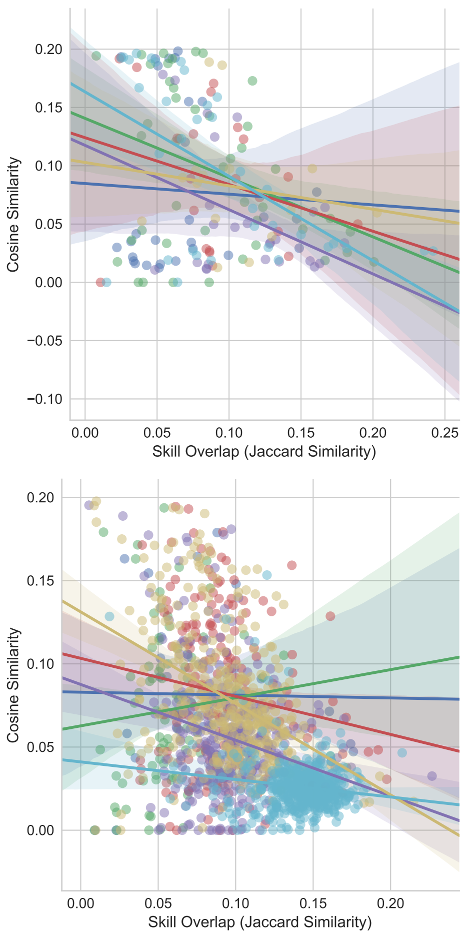


Fig. 12: Dependence of cosine similarity on the Jaccard similarity between the skill sets of members and jobs in a cluster for BayesMatch (upper) and NpBayesMatch (lower). Each datapoint denotes a cluster and each line shows the linear trend for an industry.

VIII. Conclusion

In this paper, we presented *BayesMatch* and *NpBayesMatch* – two novel, Bayesian multi-view member-job matching models, and *SkillR* – a skill recommendation system that exploits the gap between skills that members supply and those that the labor market demands. A potential avenue of future work is to explore the use of spectral methods that provide guaranteed parameter recovery for latent variable models [14].

BayesMatch and *NpBayesMatch* allow a member or job to belong to a single partition only. Extensions to a sparse mixed-membership model will allow us to better model interdisciplinary members and jobs who act as "gray sheep" [15] in our recommendation system. Another direction of considerable promise is to incorporate the behavioral signal of a member's intent to customize the skills being recommended to him.

In the age of information overload, our skill recommendations can help members acquire skills demanded by the labor market and thus make strategic progress in their professions. Educational institutions can keep pace with changing times by identifying and teaching vocationally relevant skills. Organizations can invest in training their employees to remain competitive in a skill-driven economy. Governments can make funding decisions for education and research based on expected future skill demands. Thus, the economic implications of facilitating skill acquisition through personalized skill recommendations are far reaching.

Acknowledgments

The authors would like to thank Alexandra Chouldechova for her insightful feedback on this project.

References

[1] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin, *Bayesian data analysis*. Chapman & Hall/CRC Boca Raton, FL, USA, 2014, vol. 2.

[2] Y. W. Teh, "Dirichlet process," in *Encyclopedia of machine learning*. Springer, 2011, pp. 280–287.

[3] R. C. Steorts, R. Hall, and S. E. Fienberg, "A Bayesian approach to graphical record linkage and de-duplication," *Journal of the American Statistical Association*, 2015.

[4] A. Goyal, "Recommending Jobs You May Be Interested In," *Big Data Innovation Summit 2014, Santa Clara, CA*, 2014, doc: http://www.slideshare.net/AnujGoyal11/big-data-innovationsummit2014. Accessed: October 16, 2016.

[5] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*. USENIX Association, 2012, pp. 2–2.

[6] G. W. Imbens and D. B. Rubin, *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press, 2015.

[7] T. L. Griffiths and M. Steyvers, "Finding scientific topics," *Proceedings of the National academy of Sciences*, vol. 101, no. suppl 1, pp. 5228–5235, 2004.

[8] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, 2003, pp. 173–180.

[9] X. Zhang, Y. Zhou, Y. Ma, B.-C. Chen, L. Zhang, and D. Agarwal, "Glmix: Generalized linear mixed models for large-scale response prediction," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 363–372.

[10] H. Lakkaraju, J. Leskovec, J. Kleinberg, and S. Mullainathan, "A bayesian framework for modeling human evaluations," in *Proceedings of the 2015 SIAM International Conference on Data Mining*. SIAM, 2015, pp. 181–189.

[11] A. Antikacioglu, R. Ravi, and S. Sridhar, "Recommendation Subgraphs for Web Discovery," in *Proceedings of the 24th International Conference on World Wide Web*. ACM, 2015, pp. 77–87.

[12] I. S. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 269–274.

[13] B. Gao, T.-Y. Liu, X. Zheng, Q.-S. Cheng, and W.-Y. Ma, "Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 41–50.

[14] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky, "Tensor decompositions for learning latent variable models," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 2773–2832, 2014.

[15] T. Miranda, M. Claypool, A. Gokhale, T. Mir, P. Murnikov, D. Netes, and M. Sartin, "Combining content-based and collaborative filters in an online newspaper," in *In Proceedings of ACM SIGIR Workshop on Recommender Systems*. Citeseer, 1999.