PROSPECT: A system for screening candidates for recruitment

Amit Singh
IBM Research, India
amising3@in.ibm.com

Catherine Rose
IBM Research, India
rosecatherinek@in.ibm.com

Karthik Visweswariah
IBM Research, India
v-karthik@in.ibm.com

Enara Vijil
IBM Research, Watson
ecvijil@us.ibm.com

Nandakishore Kambhatla
IBM Research, India
kambhatla@in.ibm.com

ABSTRACT

Companies often receive thousands of resumes for each job posting and employ dedicated screeners to short list qualified applicants. In this paper, we present PROSPECT, a decision support tool to help these screeners shortlist resumes efficiently. Prospect mines resumes to extract salient aspects of candidate profiles like skills, experience in each skill, education details and past experience. Extracted information is presented in the form of facets to aid recruiters in the task of screening. We also employ Information Retrieval techniques to rank all applicants for a given job opening. In our experiments we show that extracted information improves our ranking by **30%** there by making screening task simpler and more efficient.

Categories and Subject Descriptors

H.3.3 [**Information storage and retrieval**]: Information search and retrieval

General Terms

Algorithms, Design, Experimentation, Management

Keywords

resume search, resume information extraction

1. INTRODUCTION

Hiring the right talent is a challenge faced by all companies. This challenge is amplified by the high volume of applicants if the business is labor intensive, growing and faces high attrition rates. One example of such a business is IT services run out of growth markets. In a typical services organization, professionals with varied technical skills and business domain expertise are hired and assigned to projects to solve customer problems. In the past few years, IT services including consulting, software development, technical support and IT outsourcing has witnessed explosive growth, especially in growth markets like India and China. For instance, according to a NASSCOM (National Association of Software and Services Companies of India) study, the total number of IT and IT enabled services professionals in India has grown from 284000 in 1999-2000 to over 1 million in 2004-2005 [18]. More recent estimates suggest that this industry employs more than 2 million professionals in India alone. For organizations in the IT Services business, growth in business is synonymous with growth in the number of employees and recruitment is a key function.

Hiring large number of IT professionals in growth markets poses unique challenges. Most countries in growth markets have large populations of qualified technical people who all aspire to be part of the explosive growth in the IT Services industries. Thus, a job posting for a Java programmer can easily attract many tens of thousands of applications in a few weeks. Most IT Services companies are inundated with hundreds of thousands of applicants. For example, Infosys, one of the largest IT Outsourcing companies in India, received more than 1.3 million job applications in 2009. However, only 1% of them were hired[1].

To give the context for our work, we now give an overview of a typical recruitment process. This is illustrated in Figure 1. The process starts when a business unit decides to hire employees to meet its business objectives. The business unit creates a job profile that specifies the role, job category, essential skills, location of the opening and a brief job description detailing the nature of work. It might also specify the total work experience that the prospective employee should possess, along with the desired experience level for each skill. The job openings are advertised through multiple channels like on–line job portals, newspaper advertisements, etc. Candidates who are interested to apply for the job opening uploads their profile through a designated website. The website typically provides an on–line form where the candidate enters details about her application like personal information, education and experience details, skills, etc. We call this Candidate Meta–data. The candidates can also upload their resumes through the website. The objec-

[1]http://www.infosys.com/newsroom/infosys-in-the-news/infosys-u/pages/index.aspx

tive of allowing the candidate to enter meta–data in an on–line form is to capture the information in a more structured format to facilitate automated analysis. However, real life experience suggests that most candidates do not specify a lot of information in the on–line forms and hence Candidate Meta–data is often incomplete.

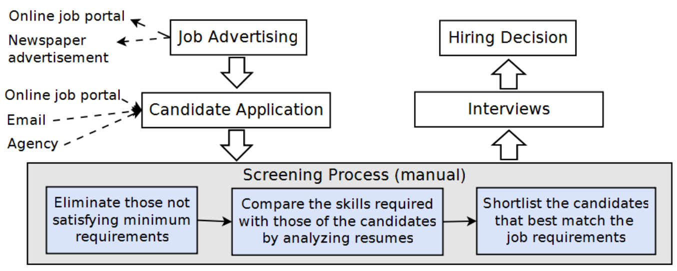

Online job portal
Newspaper advertisement
Job Advertising
Hiring Decision
Online job portal
Email
Agency
Candidate Application
Interviews
Screening Process (manual)
Eliminate those not satisfying minimum requirements
Compare the skills required with those of the candidates by analyzing resumes
Shortlist the candidates that best match the job requirements

Figure 1: Recruitment process with manual screening

Once the applications of prospective candidates are received, they are subjected to careful scrutiny by a set of dedicated screeners. This screening process is crucial because it directly affects the quality of the intake and hence, the company profits. The screeners typically proceed as below:

1. Understand the requirement for the job opening, in terms of the skills that are mandatory and those that are optional but preferable, the experience criteria if any, preference for the location of the candidate etc. Also, note the kind of work that will be performed as part of the job role.

2. Look through each of the applications, and reject those who do not have the minimum years of experience or the skills required for the job.

3. Out of the remaining candidates, find the best match for the job. This requires the recruiter to read the resume in detail and compare it with the job profile. Since the number of candidates who can be interviewed is limited, the recruiter has to make a relative judgment on the candidates.

The top few candidates who are shortlisted during the screening, undergo further evaluation in the form of interviews, written tests, group discussions etc. The feedback from these evaluation processes is used to make the final hiring decision.

In this paper, our focus is the screening task which is a tedious and time consuming task. A candidate's profile is multifaceted and often the various attributes in his profile are not directly comparable with others. For example, a candidate A may be highly experienced in the technical area but may not have the desired industry domain expertise. Another candidate B might have the required domain expertise, but may be slightly less experienced in the technical area than candidate A. Yet another candidate, C, might be much more versatile, possessing skills in a large number of technical areas and business domains, but lacking the required mastery of a specific technical area. The screener has to go through the resume of each applicant and quickly decide whether to shortlist the candidate or not. However, ideally, the screener should not make this decision without looking at the resumes of other candidates who have applied for the job. If the screener defers the decision, it is difficult to later come back and make a decision, especially when there are thousands of candidates who have applied for the same job. The screeners are also under immense pressure to hire people quickly, since the rapid growth of labor intensive companies is critically dependent on this. Often, because of these time pressures, screeners rely on small samples of the applicant pools or ranking provided by third party hiring firms to screen and shortlist candidates. In some cases, the best candidate for a job might not have been looked at by a screener!

In this paper, we present a decision support system (dubbed Prospect) for helping screeners shortlist better candidates faster.

The key challenge in building a system to aid screening is that job requirements and resumes are written in natural language and job requirements often contain complex constraints (e.g. "at least 6 years of J2EE experience"). It is clear that keyword matching of job requirements to resumes does not suffice to give a good ranking of candidates. In addition to improved ranking by weighting skill words more than others, Prospect allows screeners to quickly select candidates that meet specified requirements via the user interface. Thus, we are able to achieve a more uniform screening process, lower cost to the organization, and identify better candidates.

This paper is organized as follows: In the next section, we outline and contrast our work with related work in retrieval on resumes. This is followed by Section 3 which gives the architecture and an overview of our systems features. Section 4 describes the information extraction techniques we use to mine important information useful in the screening process and gives experimental results for accuracies of extraction of various attributes. Section 5 describes our experiments on the retrieval method used for ranking candidates for jobs. Finally, we conclude by discussing some experiences of screeners using the system, and feedback from them that can guide future work.

2. RELATED WORK

There have been several attempts to automate various aspects of the recruitment process [16]. For example, [14] suggests using techniques like collaborative filtering to recommend candidates matching a job. [27] describes a method that uses relevance models to bridge the vocabulary divide between job descriptions and resumes. In their method, related job descriptions are identified by matching a given candidate job description with a database of job descriptions. Then, resumes matching those job descriptions are used to capture vocabulary that is not explicitly mentioned in the job descriptions. These methods assume the existence of manually labeled relevance rankings. On the other hand, our method does not assume the presence of relevance rankings for training. In [9, 11], collaborative filtering measures are combined with content based similarity measures for better ranking. However, most of these studies are performed on synthetic data and not on real world unstructured resumes and job descriptions.

There is a body of work that takes into account individual preferences and team dynamics for staffing [17, 7]. In our work, we do not look at teaming aspects and decisions are made solely based on the skill match of the candidate to the job description.

There is also work focusing exclusively on information extraction from resumes [28]. In contrast to previous work, we describe an entire system for aiding screeners, using a combination of information extracted from resumes and retrieval techniques to help the screeners accomplish their tasks faster. In particular, we show that we can improve retrieval performance using information extracted from resumes. Also unlike them we extract experience for each skill.

3. DESCRIPTION OF THE PROSPECT SYSTEM

As described in Section 1, the manual screening process is highly cumbersome and inefficient. In this section, we describe the features of our system that facilitates a more effective screening process.

For each job profile, the system automatically ranks candidates based on the similarity between job profile and the resume of the candidate. The screener can refine this ranking by adding multiple search and filtering criteria. The filtering criteria can be based on the information in the candidate meta–data as well as on the information that is automatically extracted from the resumes. As new conditions are entered by the screener, Prospect refines the ranking to reflect the new constraints. This refining procedure is repeated until the screener is satisfied with the system generated ranking. Subsequently, the screener can shortlist the top candidates in the ranked list for interviews, and reject in bulk, the remainder of the list.

Figure 2 gives a screenshot of the user–interface of Prospect. As can be noted from the screenshot, our system presents a multi–faceted view of the candidates thus helping the screeners quickly gauge the range of candidate profiles available. It also highlights the key aspects of the candidate profiles and displays relevant snippets of their resume.

In the sections below, we describe the features in Prospect that enable efficient screening of candidates, followed by a detailed system architecture .

3.1 Ranking of Candidates

In our initial ranking, we are looking for candidates that have a broad match of skills mentioned in the job description. One way to achieve this is by computing a similarity score between the job description and the resume of a candidate. In this paper we experimented with using various adhoc retrieval methods to provide a similarity score between the job description and the resume of a candidate. One issue we face is that the job description is long and usually contains lot of terms that are not important to match. We found that the retrieval method that performed the best for our task (TFIDF scoring model from Lucene) could be enhanced by skills extracted from the job description. Details on our experiments are reported in Section 5.

3.2 Full text search on resumes

Recruiters can enter search terms in the search box provided in the system interface. This modifies the search criteria and those candidates whose resume mentions these search terms will have a higher rank.

3.3 Filtering using facets

A facet is a filter on a candidate attribute, which when chosen, will enforce the condition that, the candidates in the list should have the selected value for that particular attribute. For example, in the facet for education, if the screener chooses 'Masters Degree', then, the candidate's list is refreshed to show only those candidates who have 'Masters Degree'.

In addition to displaying the values, we also display the number of candidates against each of them. This gives the distribution of the values for the facet, on the current set of candidates. It helps the recruiters get a broad idea of the skills, education levels etc. of the candidates.

In many cases, the job role requires the candidates to have a minimum level of experience with a tool or skill. To enable the recruiters to filter candidates based on the minimum experience for a skill, Prospect extracts skills mentioned in the resumes and its corresponding experience level. The extraction process is detailed in Section 4.

Prospect provides the following facets:

- Source of the application (specifies whether the application was received through an agency, career portal, employee referral etc.)
- Highest Degree (Bachelor's, Master's, Ph.D., Diploma etc.)
- Relevant and Total work experience in years
- City of the applicant's residence
- Status of the applicant (specifies the status of application, like, Screened, Rejected, Shortlisted for Interview, etc.)
- Skill filter (specifies skills along with corresponding experience in months)

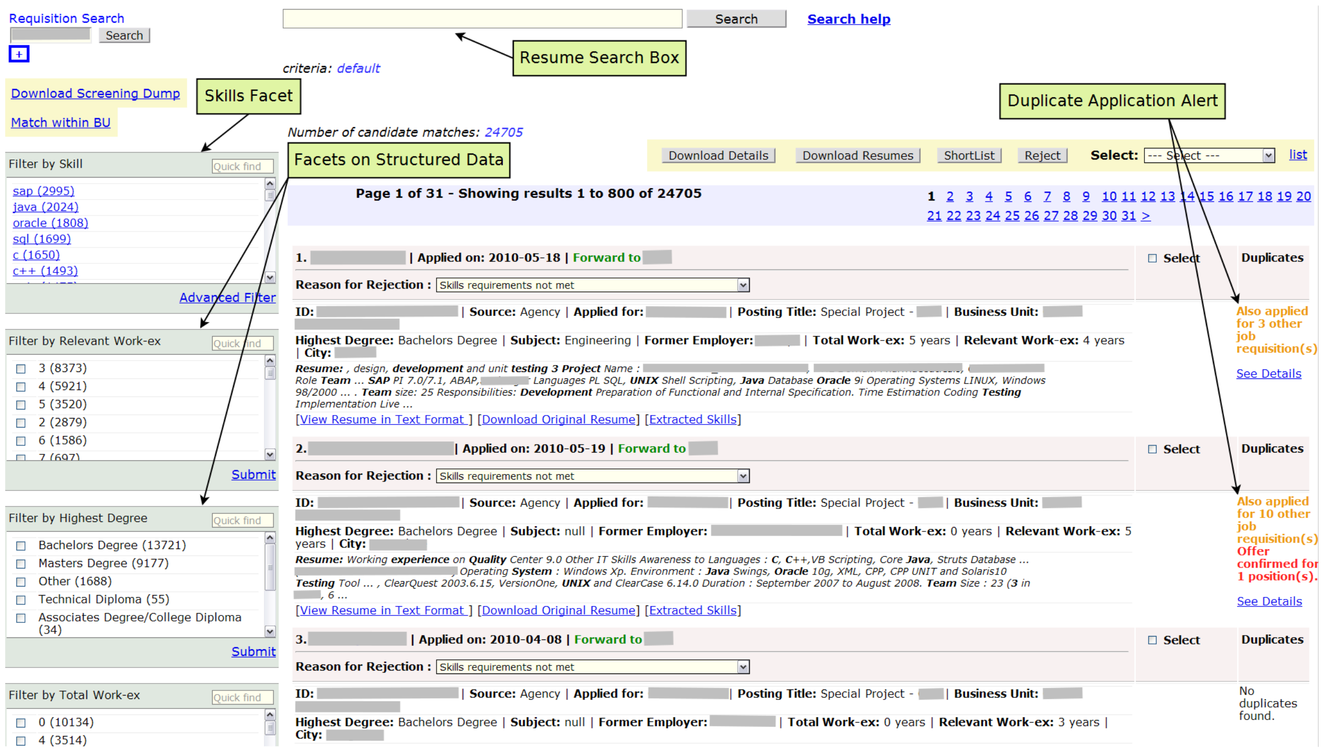
3.4 Highlighting No–Hire Companies and Non Accredited Universities

Most positions require that the candidate's educational qualification is from an accredited university. If the candidate has graduated from a non-accredited university, then he or she may not be eligible to apply for the position and should be rejected. Similarly, there might be restrictions on hiring candidates from a set of specific organizations because of recruiting policies. A typical example is when the candidate is working for a close competitor and has signed a non-compete clause with his current organization.

To assist the recruiters in this task, Prospect highlights such candidates. This is achieved by maintaining a list of no–hire companies and non-accredited universities in the system. Information from this list is compared with the candidate information and matching candidates are highlighted. The recruiters can choose to reject these candidates without going through their profile in detail.

3.5 Duplicate Applicants

Typically, in large organizations, there are several hundred job openings that are active at any point of time. Candidates can apply to multiple job openings in parallel. The screener has to ensure that if a candidate has already been offered a position in the organization, or has been rejected earlier for the same job position, he should not be considered again. In Prospect, we detect duplicate applications


Requisition Search [redacted] Search

Download Screening Dump

Match within BU

Filter by Skill

sap (2995)
java (2024)
oracle (1808)
sql (1699)
c (1650)
c++ (1493)

Advanced Filter

Filter by Relevant Work-ex

3 (8373)
4 (5921)
5 (3520)
2 (2879)
6 (1586)
7 (697)

Submit

Filter by Highest Degree

Bachelors Degree (13721)
Masters Degree (9177)
Other (1688)
Technical Diploma (55)
Associates Degree/College Diploma (34)

Submit

Filter by Total Work-ex

0 (10134)
4 (3514)

criteria: *default*

Number of candidate matches: 24705

Page 1 of 31 - Showing results 1 to 800 of 24705

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 >

1. [redacted] **| Applied on: 2010-05-18 | Forward to** [redacted]

Reason for Rejection : Skills requirements not met

ID: [redacted] | **Source:** Agency | **Applied for:** [redacted] | **Posting Title:** Special Project - [redacted] | **Business Unit:** [redacted]

Highest Degree: Bachelors Degree | **Subject:** Engineering | **Former Employer:** [redacted] | **Total Work-ex:** 5 years | **Relevant Work-ex:** 4 years | **City:** [redacted]

Resume: *, design,* ***development*** *and unit* ***testing 3 Project*** *Name :* [redacted] *Role* ***Team*** *...* ***SAP*** *PI 7.0/7.1, ABAP,* [redacted] *Languages PL SQL,* ***UNIX*** *Shell Scripting,* ***Java*** *Database* ***Oracle*** *9i Operating Systems LINUX, Windows 98/2000 ... .* ***Team*** *size: 25 Responsibilities:* ***Development*** *Preparation of Functional and Internal Specification. Time Estimation Coding* ***Testing*** *Implementation Live ...*

[View Resume in Text Format] [Download Original Resume] [Extracted Skills]

Select | Duplicates

Also applied for 3 other job requisition(s)

See Details

2. [redacted] **| Applied on: 2010-05-19 | Forward to** [redacted]

Reason for Rejection : Skills requirements not met

ID: [redacted] | **Source:** Agency | **Applied for:** [redacted] | **Posting Title:** Special Project - [redacted] | **Business Unit:** [redacted]

Highest Degree: Bachelors Degree | **Subject:** null | **Former Employer:** [redacted] | **Total Work-ex:** 0 years | **Relevant Work-ex:** 5 years | **City:** [redacted]

Resume: *Working* ***experience*** *on* ***Quality*** *Center 9.0 Other IT Skills Awareness to Languages :* ***C, C++****,VB Scripting, Core* ***Java****, Struts Database ...* [redacted] *Operating* ***System*** *: Windows Xp. Environment :* ***Java*** *Swings,* ***Oracle*** *10g, XML, CPP, CPP UNIT and Solaris10* ***Testing*** *Tool ... , ClearQuest 2003.6.15, VersionOne,* ***UNIX*** *and ClearCase 6.14.0 Duration : September 2007 to August 2008.* ***Team*** *Size : 23 (****3*** *in* [redacted]*, 6 ...*

[View Resume in Text Format] [Download Original Resume] [Extracted Skills]

Select | Duplicates

Also applied for 10 other job requisition(s)
Offer confirmed for 1 position(s).

See Details

3. [redacted] **| Applied on: 2010-04-08 | Forward to** [redacted]

Reason for Rejection : Skills requirements not met

ID: [redacted] | **Source:** Agency | **Applied for:** [redacted] | **Posting Title:** Special Project - [redacted] | **Business Unit:** [redacted]

Highest Degree: Bachelors Degree | **Subject:** null | **Former Employer:** [redacted] | **Total Work-ex:** 0 years | **Relevant Work-ex:** 3 years | **City:** [redacted]

Select | Duplicates

No duplicates found.

Figure 2: Screenshot of Prospect Screening UI

and alert the recruiter by highlighting corresponding candidate profiles. We also provide information about the status of other applications submitted by the candidate.

3.6 Automatic Skills Suggestion

Occasionally, the job descriptions are not specified in detail. In such cases, it becomes difficult for the recruiter to screen profiles. This problem is compounded when the recruiter is not well versed in the job domain. In such cases, automatically suggesting additional skills that are relevant to the job description will help them.

Prospect provides an automatic skill suggestion feature, which is based on the hypothesis that most of the candidates who have applied for the job have the relevant skills. The number of candidates possessing a skill is observed to be a good estimate of its relevance to the job role. Though this approach is simplistic, we found that it works well in practice, especially when the number of applicants for a job is large.

One could also use techniques like KL divergence [12] to suggest skills that differentiate various job profile.

3.7 System Architecture

Figure 3 outlines the architecture of Prospect. It comprises of the following main components: Batch Processor, Query Processor and Resume Matcher.

New applications are first processed by the Batch Processor. It stores the candidate meta–data in the Master DB and extracts data from the resumes, which is saved in Extracted DB. This information is used by the Query Processor and the Resume Matcher to provide a ranked candidate list for a given user query.

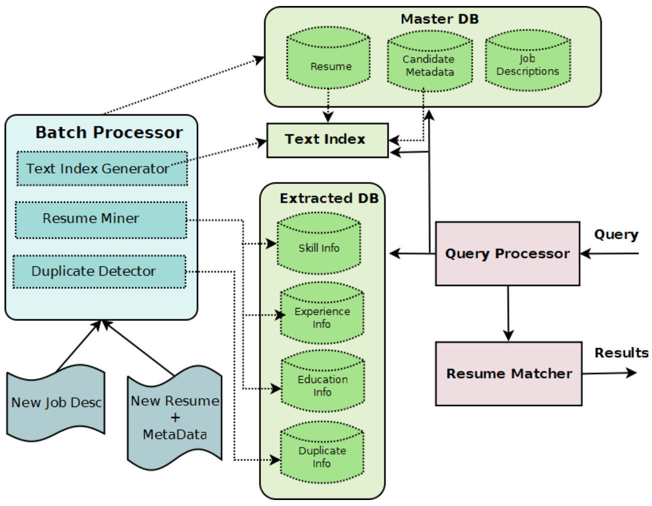


Figure 3: Prospect: System Architecture

3.7.1 Batch Processor

This component executes a nightly job, which processes all resumes that have been received during the day. It consists of the following modules:

1. Text Index Generator: This component creates a full text index to enable text search on candidate resumes. Resumes are converted to plain text from document formats like Microsoft Word, PDF, etc. We use Lucene [1], an open source Java text search engine library to create a full text index over resumes.

2. Resume Miner: This is the core component of Prospect. Resume Miner extracts information from candidate resumes and stores it in a DB2 database (Extracted DB). This is detailed in Section 4 .

3. Duplicate Detector: As described in Section 3.5, candidates applying to multiple job profiles are very common. Detecting these duplicate profiles cannot be achieved by matching candidate meta–data alone. Because, in some cases, especially when candidate applies to the same job profile multiple times, the candidate plays the role of an adversary to the screening system. The candidate might apply by using different identifying information (for example, multiple email ids) and may provide incomplete and/or inconsistent information in the meta–data. However, modifications to resumes are relatively minor. We use shingling [4], a technique commonly used to identify duplicate web pages (like mirror sites), to identify resumes that are very similar. Candidates whose resumes are found to be duplicates are highlighted by the system.

3.7.2 Query Processor and Resume Matcher

The Query Processor is responsible for interacting with various components like Text Index, Database and Resume Matcher. It takes as input, the Search Criteria and returns a ranked list of candidates as result. It comprises of query parser which parses user submitted query strings and converts it to a representative format suitable for Prospect. The query is then issued to text index and database by Query Processor. The results obtained are aggregated and submitted to Resume Matcher. Resume Matcher is the component which ranks the candidates based on extent of resume match with the job description. This ranked list is finally displayed to the user.

4. RESUME MINER

To support the features of Prospect described in Section 3, we extract various pieces of information from the unstructured resumes. Information Extraction (IE) is the important task of deriving structured information from unstructured and semi–structured sources. Over the years, IE capabilities have been successfully demonstrated across various tasks and domains [21]. As resumes can be written in a variety of styles, we prefer to use statistical data driven techniques rather than knowledge based techniques that use regular expressions, dictionaries and rule matching. In this section we describe the techniques used in Prospect to extract information from resumes.

4.1 Conditional Random Fields

Conditional Random Fields (CRFs) are state-of-the-art probabilistic models for information extraction. CRFs were first proposed by [13] for segmenting and labeling sequence data. A CRF is an undirected graphical model that defines a single log-linear distribution over label sequences given an observation sequence. In the special case in which the elements of the designated label sequence are linked by edges in a linear chain, CRFs make a first-order Markov independence assumption among output nodes. In this case CRFs can be thought of as conditionally-trained Hidden Markov Models.

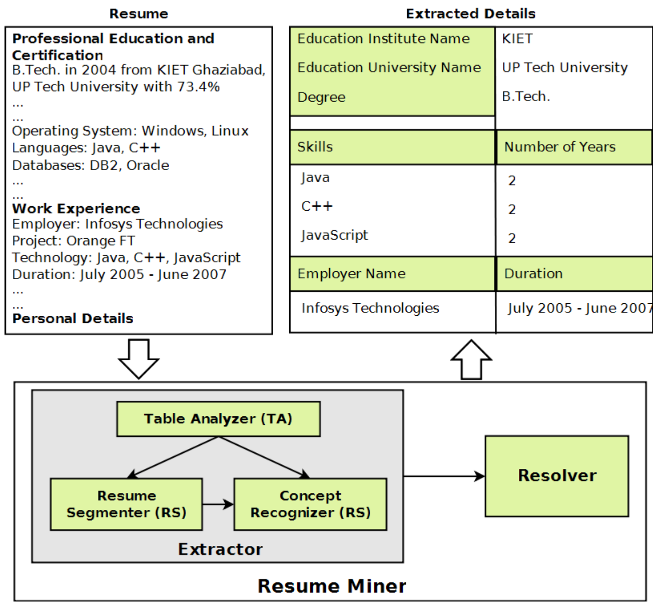


Figure 4: Extraction Model

Formally, let $x = \{x_i\}_{i=1}^n$ and $y = \{y_i\}_{i=1}^n$ be sequences of input and output variables respectively, where n is the length of sequence, $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$ where $\mathcal{X}$ is the finite set of the input observations and $\mathcal{Y}$ is the output label state space. Then, a first-order linear-chain CRF is defined as:

$$p(y|x) = \frac{1}{\mathcal{Z}(x)} \prod_{i=1}^{n} \Psi_i(x, y) \tag{1}$$

$$\mathcal{Z}(x) = \sum_{y \in \mathcal{Y}} \prod_{i=1}^{n} \Psi_i(x, y) \tag{2}$$

$$\Psi_i(x, y) = \exp\left(\sum_j \lambda_j t_j(y_{i-1}, y_i, x, i) + \sum_k \mu_k s_k(y_i, x, i)\right) \tag{3}$$

where s_k is a state feature function that uses only the label at a particular position, t_j is a transition feature function that depends on the current and the previous label, $\mathcal{Z}(x)$ is a normalization factor and λ_j is a learned weight for each feature function. Equation 1 can also be written as

$$p(y|x, \lambda) = \frac{1}{\mathcal{Z}(x)} \exp\left(\sum_j \lambda_j \mathcal{F}_j(y, x)\right) \tag{4}$$

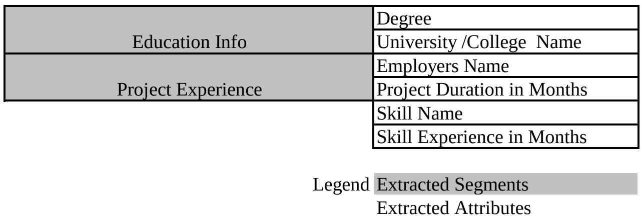
$$\mathcal{F}_j(y, x) = \sum_i f_j(y_{i-1}, y_i, x, i) \tag{5}$$

We train the parameters λ to maximize the conditional log-likelihood on the training data $\mathcal{T}$. The conditional log-likelihood of a set of training instances (x^m, y^m) using parameters λ is given by:

$$\mathcal{L}_\lambda = \sum_m \left[\lambda^T \cdot \sum_j \mathcal{F}_j(y^m, x^m) - \log \mathcal{Z}_\lambda(x^m)\right]. \tag{6}$$

$\mathcal{L}_\lambda$ is concave in λ and Quasi-Newton methods such as L-BFGS [5] can be used to maximize the likelihood.

4.2 Resume Extraction Model



Education Info	Degree
	University /College Name
Project Experience	Employers Name
	Project Duration in Months
	Skill Name
	Skill Experience in Months

Legend ▨ Extracted Segments
Extracted Attributes

Figure 5: Extraction Template.

For our problem of extracting various entities from resumes we use a hierarchical extraction model (Extractor). The rationale being, resumes exhibit a hierarchical structure where related concepts occur in close vicinity and there is a definite order in the overall structure. Once the information is extracted, it is normalized and checked for inconsistencies (Resolver). The Extractor along with the Resolver forms the resume extraction system henceforth referred to as Resume Miner.

Previous work in information extraction from resumes [28] have been based on hidden Markov models (HMM) [20]. Unlike HMMs, the main advantage of the CRF model is that it provides a compact way to integrate complex, overlapping and non-independent feature sets. In addition, CRFs avoid the label bias problem exhibited by HMMs that are based on directed graphical models [13]. A few studies have also shown that CRFs outperform HMMs on a number of real-world sequence labeling tasks [19, 23, 22]. In this work, we have used linear chain CRF to build our extraction models. Figure 4 depicts the outline of our resume extraction model.

Extractor. It consists of three components namely Table Analyzer (TA), Resume Segmenter(RS) and Concept Recognizer(CR). TA is used to analyze and classify information compiled as tables in resume. Output of TA is provided as input to RS and CR as features. RS is responsible for partitioning resume into contiguous blocks of information. CR extracts various aspects of candidate profile using (along with other features) the output of RS. The extraction template in Figure 5 lists the attributes to be extracted.

Table Analyzer (TA) Tables are a natural way of representing information like education and previous employers in resumes. There has been considerable amount of prior work that extracts information from tables [26, 24]. In Prospect, TA is responsible for analyzing tables and extracting information from them. Given a table, it first classifies it into categories like Education Table(ET)[2], Past Employer Table(PET)[3] and Other Table. We use a SVM classifier TabClass for this task. Once the tables are classified we apply another SVM classifier(ColClass) for identifying the type of column information. A majority of Education tables and Past Employer tables were 2–D tables with row as data records and columns as fields. University, Degree, Year, Performance, Specialization, Employer name, Work period and Designation were the column types used. 1–D ET and PET tables (data as rows and no columns) were not subjected to column classification task.

Resume Segmenter (RS) It is responsible for segmenting resumes into predefined, homogeneous, consecutive textual blocks(sections) like Personal Information, Project Information, Educational Information etc. We use CRFs for our segmentation task, the features we use are given in Figure 6.

Concept Recognizer (CR) The segmented resume is then processed by CRF based extractors to mine named entities salient to candidate profile.

Resolver. Data extracted by the Concept Recognizer needs to be normalized. For example, one might encounter the same skill by different names across a resume (e.g. Microsoft Office XP, Office XP, MS Office XP). All these variations of the same skill mention should be mapped to a representative skill (i.e. office xp), so that the experience level across each mention is correctly summed up to give the overall experience level for that skill. Prospect also has to flag candidates having affiliation with education institutes that are not accredited or employers that are in no–hire lists. How ever, the names of the institutes as mentioned in the resumes might not exactly match with the name in the lists. For example in the resume, the candidate may have mentioned "HP India Pvt. Ltd.", as an employer but in the no-hire list the same entity could be referred to as "Hewlett Packard India Pvt. Ltd.". Resolver, a subsystem of Resume Miner is responsible for resolving such cases. It uses string matching techniques [3] to tackle the data reconciliation problem.

4.3 Feature Set

We now describe the various features that the Resume Segmenter and Concept Recognizer modules use for extraction. These features can be broadly categorized as follows:

Lexicon Feature (LEX). To exploit the domain knowledge we used various dictionary based features[4]. Jaro–Winkler [25] distance metric was used for measuring similarity to words in an external dictionary. Some of these lexicons were compilation of attributes to be extracted (skill name, degree, etc.) while others contained keywords for specific blocks (e.g. common keywords for education block.)

Visual Features (VIS). Sections in resumes exhibit certain visual characteristics (layout and font) which can be leveraged to identify them. For example, section headers might follow blank lines, or they might have different font characteristics as compared to a majority of text in the resume. Additionally, certain information like education, skill are mentioned inside tables.

Named Entity Features (NE). Presence of a named entity can be a strong indicator for identifying certain blocks. For example in a resume, a Date is mentioned in places like date of birth, employment date or project date. Similarly in many instances, a Duration[5] occurs inside an experience(project) block.

[2]lists education background

[3]lists past employer's details like designation, employer name, employment duration

[4]Most of the dictionaries were compiled from open knowledge sources like Wikipedia

[5]span of time e.g. 12 months, 12 Sept to 14 Oct

Features		Segmentation			Entity Extaction		
		RS-1	RS-2	RS-3	CR-1	CR-2	CR-3
Text							
1	Starts with a capitalized letter				x	x	x
2	Starts with Digit				x	x	x
3	All characters are digits				x	x	x
4	All characters are capitalized				x	x	x
5	Contains capital and Numeric quantity				x	x	x
6	Word itself				x	x	x
7	Punctuation				x	x	x
8	Numerical data				x	x	x
9	Initials such as A.				x	x	x
Visual							
10	Regex for identifying skills [word : word, word].					x	x
11	Blank Line	x	x	x			
12	is first line of paragraph	x	x	x			
13	Font Style	x	x	x		x	x
14	change in font feature as compared to prev and next line	x	x	x			
15	Binned font sizes (relative to a resume)	x	x	x		x	x
16	Is color of line/text different from majority of resume content	x	x	x		x	x
17	Is the text inside a table	x	x	x		x	x
18	Type of Table (Education or Personal or Experience)	x	x	x		x	x
19	Type of Table column (University, Pass Year, Degree, Specialization, Performance, Past Emp Name, Designation, Work duration)					x	x
20	Is the text part of list (ordered/unordered)	x	x	x		x	x
21	Label to line given by Segmenter (RS)					x	x
Lexicon							
22	List of Skill Names						x
23	List of Designation						x
24	List of Degrees						x
25	List of Specialization						x
26	Words like institution, Labs, etc						x
27	List of Words that occur in Education block		x	x			
28	Keywords for Education block header		x	x			
29	List of Words that occur in Personal block		x	x			
30	Keywords for Personal block header		x	x			
31	List of Words that occur in Project block		x	x			
32	Keywords for Project block header		x	x			
Named Entity							
33	Phone number or zip code			x			
34	Date			x			
35	Time duration			x			
36	Email address			x			
Others							
37	Feature Conjunction	x	x	x	x	x	x

Figure 6: Feature sets we use for Information Extraction.

Text Features (TEXT). This feature captures various attributes of the word itself (see Table 6 for details).

Conjunction of Features (CJN). Feature conjunctions help form non-linear decision boundaries in the original feature space thereby capturing relationships that might not be possible with a linear combination of features. In Resume Segmenter, we use the conjunction of current line with the previous line and the current line with the next line. In the Concept Recognizer, similar token level conjunctions are used.

4.4 Experiment and Analysis

Dataset. For training Resume Segmenter and Concept Recognizer, we annotated around 110 English resumes using GATE (General Architecture for text Engineering) [8]. These resumes were carefully sampled across various job verticals and experience levels, so that learned model was not biased towards a particular job type. More than 98% of resumes were in Microsoft Word format. We used the Apache POI API[6] to parse these documents and extract data and visual (formating and layout) information. Figure 7 summarizes some important corpus statistics. We collected upwards of 7200 annotations from 3 volunteers. Volunteers were told to be as exhaustive as possible and tag all possible instances. This is very important in case of annotating skills, because we need to identify every occurrence of a skill in a resume to be able to calculate the experience level for that skill accurately. Figure 8 summarizes some statistics on inter–annotator agreement for annotation on Skill name. Clearly, a considerable number of disagreements were found in the labeling wherein a segment was tagged by a volunteer while it was not tagged or given a different label by another volunteer. We believe similar cases would exist for other labels(attribute/segment annotations) too. We used 5–fold cross validation to evaluate our extraction system. To analyze the contribution of different kinds of features, we designed experiments with incremental addition of new feature types.

Annotated Field	Total number of Human Annotations
Number of Tagged Resume	110
Number of Job postings used	18
Skill Name	5308
Skill Experience	148
Project Block	411
Project Duration	282
Education Block	86
Education Degree	198
Education University	233
Education Performance	119
Education Pass Year	137
Experience Organization Name	226

Figure 7: Statistics for the corpus of hand annotations.

# Annotation done by more than one person	310
# Annotation with disagreement	41

Figure 8: Inter-annotator agreement on Skill name.

Evaluation Measures. For all tasks, we used standard precision, recall and F1-measure to evaluate experimental results [6]. Precision and recall are macro-averaged across documents and overall F1 computed from average precision and recall. Like [28], we used "Overlap" criteria [15] (match if $>$ 90% overlap) to match ground truth with extracted data.

Table Data Classification. TabClass uses about 100 human annotated tables as examples to learn the classification model. A combination of layout features (number of cells, number of columns, number of rows) and content features (header content, presence of date, presence of duration, table content, lexicon for designation, degree etc.,) achieved an averaged F1 of **0.958**. For identifying the type of column information, ColClass uses content based features similar to TabClass, along with other features like similarity of header cells to a predefined label set (University, Degree, Year, Designation etc.,) and achieved an averaged F1 of **0.941**.

[6]http://poi.apache.org/

Block	Feature Set	Precision	Recall	F1
Education Block	RS-1	0.918	0.860	0.888
	RS-2	0.937	0.881	0.908
	RS-3	0.940	0.902	0.921
Project (Experience) Block	RS-1	0.785	0.730	0.757
	RS-2	0.767	0.771	0.769
	RS-3	0.790	0.780	0.785

Figure 9: Segmentation results with different feature sets.

Segmentation. Figure 9 outlines the effect of features on performance of Resume Segmenter. While VIS alone gives good results, combination of LEX, NE, VIS gives the best result. We found that Features 11, 14 and 18 were most effective in segmentation tasks. A large fraction of resumes had Education information embedded inside a table, so identifying that was relatively easier. On the contrary, Project block appears in variety of formats (a sequence of project blocks, list of past employers each having a list of project block, etc.) and was difficult to extract. This was evident even during manual annotation, as there were several cases when volunteers were not easily able to decide the boundaries of a project block.

Attribute Extraction. Output of Resume Segmenter was used as input features to the Concept Recognizer module. As shown in Figure 10, accuracy results for the Concept Recognizer followed a similar trend to Resume Segmenter. A combination of LEX, TEXT and VIS features gave the best overall F1 for all attribute extractors. Features 1, 5, 10 and 37 were key in identifying Skill names. For Education and past Employer details, features 19, 20 and 21 seemed to matter the most. On analysis we found that most of the incorrect extractions occurred when data was not compiled as tables in resume. Some of the common errors were

- Labeling the client(organization) name for which project was done as the employers name.
- Labeling the project name as the employer's name.
- Mixing up labellings of Degree and University Name.
- Including University address as a part of the name. For example, in the segment "*Universidad Carlos III Madrid, Spain*", "*Universidad Carlos III Madrid*" is incorrectly marked as the university name.)

Skill Experience Level. Generally project blocks contain the duration for which the project was executed. Sometimes this information is mentioned at the past employer level and not at the project level[7]. Any skill that is mentioned inside the project block gets this project experience. For example if Java is mentioned in project–A and project–B having duration of x and y months respectively, then overall experience for Java is $(x+y)$ months. This simplistic rule is used by the recruiters while screening for jobs having specific skill experience level requirement.

Output of Resume Segmenter (project blocks) and Concept Recognizer(skill name), along with project duration information is used to mine experience level information. To

[7]This case can be handled easily

Attribute	Feature Set	Precision	Recall	F1
Skill Name	CR-1	0.801	0.714	0.755
	CR-2	0.828	0.740	0.782
	CR-3	0.867	0.745	0.801
Organization Name	CR-1	0.810	0.540	0.648
	CR-2	0.782	0.590	0.673
	CR-3	0.790	0.601	0.683
University Name	CR-1	0.784	0.500	0.611
	CR-2	0.817	0.600	0.692
	CR-3	0.808	0.630	0.708
Degree	CR-1	0.903	0.549	0.683
	CR-2	0.920	0.639	0.754
	CR-3	0.909	0.770	0.834

Figure 10: Entity extraction results for various feature sets.

evaluate effectiveness of this simple strategy we collected around 148 human judgments, each comprising of a skill and corresponding experience level in months. For collecting these judgments we followed two strategies to nullify any bias due to resume structure.

- Pick a random resume from 110 manually annotated resumes. For each skill in the resume, get human judgments on the experience level.
- Pick a random skill, and pick random resumes having that skill (output of Concept Recognizer). Now get human judgment on experience level for the chosen skill.

For evaluating the performance, we counted the number of instances for which inferred experience level was within a bound of human judgments. Figure 11 outlines the results. X–axis shows the error bounds for example, 10 refers to a bin where inferred level was within 10% of human judged experience level. Y–axis records the number of skills that fall in that bin. We choose this evaluation method because job description generally require skill with range for experience level i.e *2 to 5 years experience in Java*, so just using precision may not indicate utility of extracted data. Detailed analysis highlighted the following reasons of failure:

- Some project blocks had multiple time–intervals mentioned. In some instances incorrect interval was used to calculate duration leading to an error.
- Error in the output of Resume Segmenter and Concept Recognizer cascaded to this task.
- Errors due to imperfect canonicalization of skill names. e.g If skill chosen is *jsp* and it occurs in project–A and project–B with surface form *jsp* and *java server pages* respectively with duration x and y months then experience reported for *jsp* should be $x + y$ and not x alone.
- If multiple projects have overlapping durations and a skill appears in both of them, then overlap duration was counted twice

5. CANDIDATE RANKING EXPERIMENTS

Given a set of candidates that satisfy the filtering conditions, we would like to display candidates that better match

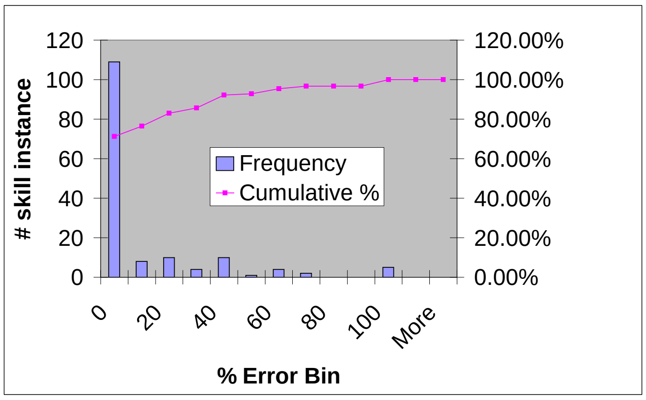


Figure 11: Error histogram for extracting the experience associated with skills.

the job requirements at the top. We create a query q from the job requirements and treat the text of the candidate resumes as documents d and apply standard ad-hoc retrieval techniques to rank the candidates. The different retrieval algorithms we use are Okapi BM25 (from the Lemur toolkit), Kullback–Leibler divergence of language models with Dirichlet smoothing (also from the Lemur toolkit) and the TF-IDF scoring model in Lucene. Since the job descriptions are fairly verbose we also experiment with a retrieval model where certain terms that are important to the quality of the match are weighted up in the TF-IDF scoring model. In our experiments we extracted the skills from the job description using a simple dictionary based extraction method where the dictionary consisted of a set of skills that we extracted using the system described in Section 4 from all the resumes in our system (across all the jobs).

For our experiments to compare the various retrieval methods we used 8 job descriptions and retrieved the top 20 candidate resumes for each method and judged the relevance of these against the job description. The jobs we chose had an average of 2000 candidates per job. Since we were interested in screening we also evaluated the 20 bottom ranked candidates. For the bottom ranked candidates, all of the methods performed equally well and we found no relevant candidates in that set for any of the methods. To compare performance of retrieval methods for the top results returned, we used both NDCG [10] and Precision @ k. Figure 12 summarizes our results and we see that Lucene TF-IDF scoring with boosting of skill terms performs the best. This agrees with results in [2] where it was found that finding and weighting up important concepts in long queries can improve retrieval performance.

While keyword based scoring functions do help in making the job of the recruiter easier, it is clear that they cannot capture requirements such as *"at least 3 years of Java development"*. Prospect allows recruiters to filter candidates based on the number of years of experience they have in certain skills. In the next experiment we compared the quality of the ranking and with and without the filter when the job descriptions contained a requirement in terms of the number of years of experience for one or more skills. We see from the results in Table 13 that the ranking performance is indeed significantly improved (around **30%**) by application of the filter.

	k	Okapi BM25	KL	Lucene Scoring Model	Lucene Scoring Model + Skill Boost
Precision	5	0.18	0.43	0.60	0.70
	10	0.19	0.34	0.59	0.63
	20	0.21	0.36	0.49	0.61
NDCG	5	0.15	0.44	0.58	0.73
	10	0.17	0.37	0.57	0.66
	20	0.19	0.38	0.51	0.64

Figure 12: Comparison of various retrieval methods for ranking candidates

	k	Lucene Scoring Model + Skill Boost	Lucene Scoring Model + Skill Boost + Extracted Years of Experience
Precision	5	0.60	0.80
	10	0.50	0.65
	20	0.48	0.65
NDCG	5	0.68	0.86
	10	0.59	0.73
	20	0.54	0.70

Figure 13: Evaluation of ranking obtained with and without using the extracted years of experience

6. CONCLUSION

In this paper we described Prospect a system that aids in the shortlisting of candidates for jobs. The key technical component that makes Prospect possible is the extraction of various pieces of information from resumes. Since job requirements often specify requirements such as *"at least 3 years of Java development"* we use information extracted from resumes to provide filters that allow screeners to find candidates that match such criteria. This allows us to overcome the limitations inherent in purely keyword based matching.

In piloting Prospect with screeners we estimated that using the tool roughly sped up the screening process by a factor of **20** as compared to manual screening. This speedup can be attributed mainly to a combination of two factors. Firstly, ranking the candidates by match to the job description and use of the filters provided based on various information extracted from the resume allows the screeners to inspect far fewer resumes to shortlist a given number of candidates. Secondly, showing snippets of the resume based on its match to the requirements of the job and based on the information extracted from the resume allows a screener to shortlist or reject candidates much faster than if they had to scan the entire resume.

Despite these benefits, we did receive qualitative feedback that could help improving Prospect. One major area of improvement is in the consideration of three factors that the screeners considered to be important in the ranking function. In order of importance these are:

- Presence of skill mentions in sections describing projects should be weighted higher than mentions in other parts of the resume

- Presence of skill mentions in more recent projects should be weighted higher than those in older projects

- Factoring in the quality of the educational institutions and companies

As future work, we plan to learn to weight these factors (which is part of the information we extract from resumes) using relevance feedback data we collect from the screeners.

7. REFERENCES

[1] Apache Lucene. http://lucene.apache.org, 2009.

[2] M. Bendersky and W. B. Croft. Discovering key concepts in verbose queries. In *Proceedings of SIGIR*, pages 491–498, New York, NY, USA, 2008. ACM.

[3] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg. Adaptive name matching in information integration. *IEEE Intelligent Systems*, 18(5):16–23, 2003.

[4] A. Z. Broder. Identifying and filtering near-duplicate documents. In *COM '00: Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching*, pages 1–10, London, UK, 2000. Springer-Verlag.

[5] R. H. Byrd, J. Nocedal, and R. B. Schnabel. Representations of quasi-newton matrices and their use in limited memory methods. *Math. Program.*, 63(2):129–156, 1994.

[6] S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan-Kauffman, 2002.

[7] V. Chenthamarakshan, K. Dey, J. Hu, A. Mojsilovic, W. Riddle, and V. Sindhwani. Leveraging social networks for corporate staffing and expert recommendation. *IBM Journal of Research and Development*, 56(3), 2009.

[8] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of ACL*, 2002.

[9] F. Farber, T. Weitzel, and T. Keim. An automated recommendation approach to selection in personnel recruitment. *Proceedings of AMCIS*, 2003.

[10] K. Järvelin and J. Kekäläinen. Ir evaluation methods for retrieving highly relevant documents. In *Proceedings of SIGIR*, pages 41–48, New York, NY, USA, 2000. ACM.

[11] T. Keim. Extending the applicability of recommender systems: A multilayer framework for matching human resources. In *HICSS '07: Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, page 169, Washington, DC, USA, 2007. IEEE Computer Society.

[12] S. Kullback. *Information Theory and Statistics*. Dover Publications, 1997.

[13] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[14] S. Laumer and A. Eckhardt. Help to find the needle in a haystack: integrating recommender systems in an it supported staff recruitment system. In *SIGMIS CPR '09: Proceedings of the special interest group on management information system's 47th annual conference on Computer personnel research*, pages 7–12, New York, NY, USA, 2009. ACM.

[15] A. Lavelli, M. E. Califf, F. Ciravegna, D. Freitag, C. Giuliano, N. Kushmerick, and L. Romano. A critical survey of the methodology for ie evaluation. page 16551658, 2004.

[16] I. Lee. An architecture for a next-generation holistic e-recruiting system. *Commun. ACM*, 50(7):81–85, 2007.

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

[18] Nasscom. Knowledge professionals in india. *NASSCOM-Hewitt Survey*, 2005.

[19] D. Pinto, A. McCallum, X. Wei, and W. B. Croft. Table extraction using conditional random fields. In *Proceedings of SIGIR*, pages 235–242, New York, NY, USA, 2003. ACM.

[20] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. pages 267–296, 1990.

[21] S. Sarawagi. Information extraction. *FnT Databases*, 1(3), 2008.

[22] S. Sarawagi and W. W. Cohen. Semi-markov conditional random fields for information extraction. In *NIPS*, 2004.

[23] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of NAACL*, pages 134–141, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[24] Y. Wang and J. Hu. A machine learning based approach for table detection on the web. pages 242–250, 2002.

[25] W. Winkler. Matching and record linkage. 1995.

[26] Y. Yang and W.-S. Luk. A framework for web table mining. pages 36–42, 2002.

[27] X. Yi, J. Allan, and W. B. Croft. Matching resumes and jobs based on relevance models. In *Proceedings of SIGIR*, pages 809–810, New York, NY, USA, 2007. ACM.

[28] K. Yu, G. Guan, and M. Zhou. Resume information extraction with cascaded hybrid model. In *Proceedings of ACL*, pages 499–506, Morristown, NJ, USA, 2005. Association for Computational Linguistics.