An Analysis Framework for Content-based Job Recommendation

Xingsheng Guo, Houssem Jerbi and Michael P. O'Mahony

Insight Centre for Data Analytics[1],
School of Computer Science and Informatics,
University College Dublin, Belfield, Dublin 4, Ireland
{xingsheng.guo, houssem.jerbi, michael.omahony}@insight-centre.org

Abstract. In this paper, we focus on the task of job recommendation. In particular, we consider several personalised content-based and case-based approaches to recommendation. We investigate a number of feature-based item representations, along with a variety of feature weighting schemes. A comparative evaluation of the various approaches is performed using a real-world, open source dataset.

1 Introduction

With the rapid development of the information society, a vast amount of information is now available from many diverse sources. To deal with this information overload problem, recommender systems, which assist users to discover relevant information, products and services, have now been successfully deployed in many domains [1, 2].

In this paper, we consider the task of job recommendation, where suitable jobs are recommended to users based on their past job application history. Recent research has also focused on this task. For example, in [3] jobs are recommended based on a graph constructed from the previously observed job transition patterns of users. These patterns were based on various features relating to employer sector and size, employee experience and education, etc. A supervised machine learning approach was then adopted to recommend suitable new jobs to users. In [4], a graph-based hybrid recommender is proposed which considers both content-based user and job profile similarities and interaction-based activities (e.g., applying to or liking a job). Personalised recommendations of candidates and jobs are then generated using a PageRank-style ranking algorithm. Further, a collaborative filtering approach based on implicit profiling techniques was proposed in [5] to deliver personalized, query-less job recommendations to users. For other work in this area, see [6, 7, 14].

To date, research in the area of job recommendation has not considered an in-depth analysis of traditional content-based or case-based approaches to recommendation. Thus, the core contributions of this work are as follows. Firstly, a number of content-based approaches to job recommendation are proposed. In particular, we consider

[1] The Insight Centre for Data Analytics is supported by Science Foundation Ireland under Grant Number SFI/12/RC/2289.

variations on the unstructured item representation used in traditional content-based recommenders, where jobs are represented as documents in a multi-dimensional feature space. Secondly, case-based approaches [8] are considered, where jobs are represented in a structured manner using a well-defined set of features and feature values. This approach potentially allows for more sophisticated and fine-grained judgements about the similarity between jobs, by computing weighted aggregate similarities between individual feature values. Finally, a hybrid approach to job representation is proposed. A formal description and an evaluation of the various approaches are presented in Sections 2 and 3. Section 4 presents conclusions and future work.

2 Job Recommendation Approaches

In this section, we describe the job representation and the similarity assessment for our proposed content-based and case-based approaches to job recommendation.

2.1 Content-based Recommendation

Our content-based recommender system uses the Vector Space Model (VSM) [9]. In the VSM, each job is represented by a vector in an *n-dimensional* space, where each dimension corresponds to a textual feature from the overall feature-set of the job collection. Let $J = (J_1,J_2,...,J_N)$ denote a set of jobs and $F = (f_1,f_2,...,f_m)$ be the feature-set. Formally, every job J_a is represented as a vector of feature weights, where each weight indicates the degree of association between the job and the feature:

$$J_a = (w_1^a,w_2^a,...,w_m^a), \text{ where } w_i^a \text{ is the weight of feature } f_i \text{ for job } J_a. \quad (1)$$

For feature weighting, we employ the most commonly used term weighting scheme in information retrieval, TF-IDF (Term Frequency-Inverse Document Frequency) [10]. The intuition behind TF-IDF is that a term that occurs frequently in a given document (TF), but rarely in the rest of the corpus (IDF), is more likely to be representative of that document. Each job is considered as a document, where the job features in that document are weighted using TF-IDF as follows:

$$w_i^a = tf\text{-}idf(f_i,J_a) = tf(f_i,J_a) \times idf(f_i) = tf(f_i,J_a) \times \log(N/N_i) \ . \quad (2)$$

where $tf(f_i, J_a)$ *is* the number of occurrences of feature f_i in the content of job J_a, N_i is the number of jobs that contain the feature f_i and N is the total number of jobs.

We characterise jobs using two types of feature sets that are highly informative with regard to the job recommendation task: explicit features that are extracted from job content using different text mining techniques (bag-of-words [10] and entities [11]) and social tags that represent labels which are similar to user-defined tags.

Bag-of-words based Representation. The bag-of-words representation of a job is obtained by applying standard natural language processing operations to its content, such as tokenisation, stop-words removal, and stemming [12] (see Figure 1(*a*)):

$F(J_a) = (t_1, t_2, \dots, t_m)$, where $F(J_a)$ denotes the features of J_a and t_i is a term extracted from the job content. (**3**)

Entity-based Representation. A job content typically includes references to particular objects called *named entities*, such as the names of people, companies, and locations, that can be extracted using a named entity recognition tool. Job features are defined as follows (see Figure 1(*b*)):

$F(J_a) = (e_1, e_2, \dots, e_p)$, where e_i is an entity extracted from the job content. (**4**)

Social Tag-based Representation. In social tagging systems, users can annotate items with their own tags, commonly known as *social tags*, to facilitate their search and classification. For example, users associate tags with videos on YouTube, with Web pages in Delicious, and with scientific papers on CiteULike. These user-defined tags reflect the semantics of the annotated items from a user perspective; hence they represent an informative type of data to describe jobs. While a social tagging platform for job postings is not available, we can describe jobs according to a list of social tags that are automatically inferred from their content using a machine learning approach; *teaching*, *customer service*, *financial economics*, and *software development* are examples of social tags that are extracted from job descriptions using OpenCalais[2]. Hence, a job can be seen as a VSM of social tags that describe the job in a manner similar to human-based tagging (see Figure 1(*c*)):

$F(J_a) = (c_1, c_2, \dots, c_q)$, where c_i is a social tag associated with job J_a. (**5**)

Job Similarity Assessment. For a given feature representation, the similarity between two jobs, J_a and J_b, is computed by the cosine similarity between their vector space representations, as follows:

$$sim(J_a, J_b) = \frac{\sum_{i=1}^{m} w_i^a \times w_i^b}{\sqrt{\sum_{i=1}^{m} (w_i^a)^2} \times \sqrt{\sum_{i=1}^{m} (w_i^b)^2}} \quad , \qquad (\mathbf{6})$$

where w_i^a and w_i^b are the weights of the feature f_i in job J_a and job J_b respectively.

We also used the Jaccard index to assess the similarity between two jobs J_a and J_b that are represented by the sets of features $F(J_a)$ and $F(J_b)$:

$$sim(J_a, J_b) = \frac{|F(J_a) \cap F(J_b)|}{|F(J_a) \cup F(J_b)|} \quad . \qquad (\mathbf{7})$$

Job Recommendation. In order to generate recommendations for a given user, the similarity scores between a candidate job and each job in the user profile (*i.e.* jobs that the user previously applied to) are aggregated to compute the relevance score of the candidate job for that user. Then the top-*k* jobs with the highest scores are returned as recommendations.

---

[2] http://www.opencalais.com

2.2 Case-based Recommendation

In our case-based recommendation approach, each job is described by the distinct set of features, and there is a well-defined set of values each feature may take. With this structured representation of jobs and user profiles, customized similarity measures can be computed to recommend jobs that best match the user profile.

Case Representation. The job collection is seen as a case base and the jobs are then represented according to set of features and feature values. We employ the categories of entities that are extracted from the job content as the *case features* and the related entities as *feature values*. For example, the entity category *position* has the values *business finance manager* and *front line supervisor* (see Figure 1(*d*)). We also define a case representation based on *explicit* features that are selected from the job content (e.g., the required number of *years of experience*, the minimum *education* level, and the job *location*, see Figure 1(*e*)).

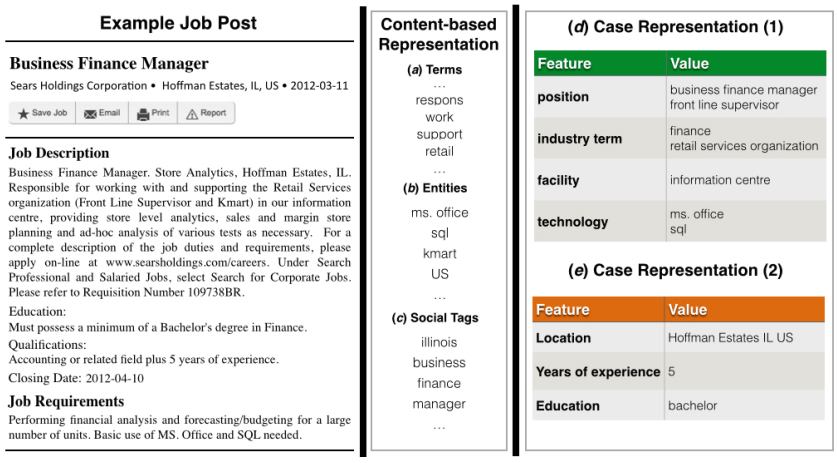


Figure 1. Job representations according to content-based and cased-based approaches.

Similarity Assessment. The similarity between two cases is defined as a weighted sum of the feature-level similarities. Each feature similarity is computed based on a similarity function that is typically specific to the feature data type. Further, each feature is associated with a weight that specifies to what extent it contributes to the overall case similarity assessment. The similarity between two jobs J_a and J_b which are represented by the same vector of features $F(J_a)=F(J_b)=(f_1, \ldots, f_m)$ is defined as:

$$sim(J_a, J_b) = \frac{\sum_{i=1}^{m} w_i \times sim(f_i^a, f_i^b)}{\sum_{i=1}^{m} w_i} \quad , \tag{8}$$

where w_i is the weight of feature f_i, and $\text{sim}(f_i^a, f_i^b)$ is a real number between 0 and 1.

Feature Similarity. For categorical features, the individual feature-level similarity is binary (*i.e.* 1 if jobs share the same feature value, 0 otherwise), whereas the similarity for a numerical feature is calculated using a symmetric similarity measure as follows:

$$sim(f_i^a, f_i^b) = 1 - \frac{|v_i^a - v_i^b|}{\max(v_i^a, v_i^b)}, \quad (9)$$

where v_i^a and v_i^b are the numerical values of feature f_i in jobs J_a and J_b, respectively. Some features are multi-valued. For example, a job *location* is typically specified using a combination of the *city*, *state* and *country* information, and the entity category *position* for a job in a research laboratory might include numerous values such as *software engineer* and *research assistant*. When a feature f_i has multiple values, we apply Jaccard index across the feature values to calculate the similarity as follows:

$$sim(f_i^a, f_i^b) = \frac{|v_i^a \cap v_i^b|}{|v_i^a \cup v_i^b|}, \quad (10)$$

where v_i^a and v_i^b are the sets of values of the feature f_i in jobs J_a and J_b, respectively.

Feature Weighting. We present two schemes to weight the features that describe a job case which are inspired from the information retrieval measures document frequency (DF) and inverted document frequency (IDF).

DF Weighting. The first approach, *normalized DF*, for assigning a weight to a particular feature in the job case is by counting the number of jobs (document frequency) in the case base where the feature appears. The intuition behind *DF* is that a feature that is shared by more cases is more likely to be important from a job requirements perspective. The *normalized DF* feature weighting is defined as:

$$w\text{-}DF_i = \frac{N_i}{N}, \quad (11)$$

where N_i is the number of jobs including the feature f_i and N is the total number of jobs in the case base.

IDF Weighting. In an information retrieval framework, the IDF measure is usually applied to the document profiles to remove common terms that appear in many documents of a collection when selecting representative terms. Our second approach for weighting case features adopts this principle to prioritize distinctive features:

$$w\text{-}IDF_i = \log\frac{N}{N_i}. \quad (12)$$

2.3 Hybrid Case-based Approach

A hybrid recommender system combines different recommendation techniques to produce its output. Several strategies have been proposed to build a hybrid recommender [13]. We are particularly interested in a hybrid recommender based on feature combination, where the bag-of-words feature is combined with well-structured features to provide a single representation. The hybrid approach lets the system avail of the known performance of the bag-of-words based recommendations, while representing jobs according to a well-structured set of features ranging from demographic features (e.g., job location) to specific features for the job recommendation task (e.g., required experience, education level, etc.).

Similarity for the bag-of-words based feature is calculated using TF-IDF based cosine similarity as described in Equations (2) and (6).

3 Evaluation

In this section we present the experimental results of our recommendation approaches. We first describe the dataset and the evaluation methodology and metrics. Then we compare the recommendation performance of our different approaches.

3.1 Dataset and Methodology

We conducted our experiments using a real-world dataset [3] from the online employment website Careerbuilder[4]. The dataset includes information about job postings during 13 weeks in 2012 (e.g., job title, description, requirements, posting date, closing date, etc.), the job seekers (e.g., location, education level and major, management experience, job history, etc.), and the user job applications history.

Our offline evaluation is based on the training/test paradigm. In particular, all job applications are sorted by application date, and all job applications submitted over a particular 6-day window are selected as training data. The test set for each user is formed using the first 5 job applications submitted during the subsequent 3-day period. Here, we only consider users with a minimum of 5 job applications in both the training and test period. In total, there are 843 such users; on average, each user has 14 training set job applications. The candidate recommendation set consists of the union of all job applications in the test period; in total, there are 3,538 such job applications. For each recommendation approach, candidate recommendations are ranked according to their mean similarity to the jobs in each user's training set.

In our evaluation, we compare different variants of our proposed content-based, case-based and hybrid recommendation approaches:

- Content-based recommenders using **b**ag-**o**f-**w**ords and TF-IDF cosine-based similarity (referred to as ***BoW***), **en**tities and Jaccard similarity (***EN***), **s**ocial **t**ags and Jaccard similarity (***ST***);
- Case-based recommenders using **cat**egories of **en**tities (***CAT_EN***), **cat**egories of **en**tities with the **ex**plicit features *years of experience*, *education* and job *location* (***CAT_EN+EX***);
- A hybrid recommender using both the **cat**egories of **en**tities features and the **b**ag-**o**f-**w**ords feature (***CAT_EN+BOW***) as described in Section 2.3;
- A baseline **n**on-**p**ersonalised approach (***NP***), which recommends the most frequently applied to jobs in the test period.

For the content-based approaches, features were extracted from the job title, description, and requirements. After removing stop-words using a stop-words lexicon[5], the remaining terms were stemmed using Porter's algorithm [12]. Entities, social tags and categories of entities were extracted using OpenCalais.

We evaluate the performance of our proposed recommendation approaches by computing the precision and recall at rank position N for $N = 1, 2, 3, 4$, and 5.

---

[3] Dataset available at: https://www.kaggle.com/c/job-recommendation/data

[4] http://www.careerbuilder.com/

[5] Lexicon available at: https://code.google.com/p/stop-words/

3.2 Features and Similarities Distributions

To assess the potential of our recommendation approaches, the features and pairwise similarity distributions of training (12,280) and test (3,538) set jobs are first analysed.

Feature Histograms. Figures 2(*a*) and 2(*b*) report the feature distribution for the content-based and case-based approaches, respectively; i.e. how many jobs have a given number of each feature type. Except for bag-of-words, the distributions follow a power law, where most (few) jobs include few (many) features. For example, 35% of jobs have only 1–5 entities, while only 5% have 20-25 entities. The social tags distribution is narrower; 40% of jobs have less than 10 social tags while only 8% jobs have 15 or more social tags. This indicates that job descriptions and requirements include only a few of the named entities (e.g. names of places and people, etc.) that are found in documents such as news articles and blog posts; hence the need for an ad-hoc entities recognition for the job recruitment domain (and likewise for social tags). Of course, the narrow distribution of these features poses a great challenge for recommendation based on such features only. However, although the distribution of the categories of entities is also narrow (64% of jobs have less than 4 features), we can expect to generate reasonably rich cases based on the bag-of-words features and also based on our hybrid bag-of-words and categories of entities approach (*CAT_EN + BoW*); from Figure 2(*a*), there are 188 bag-of-words features, on average, per job.

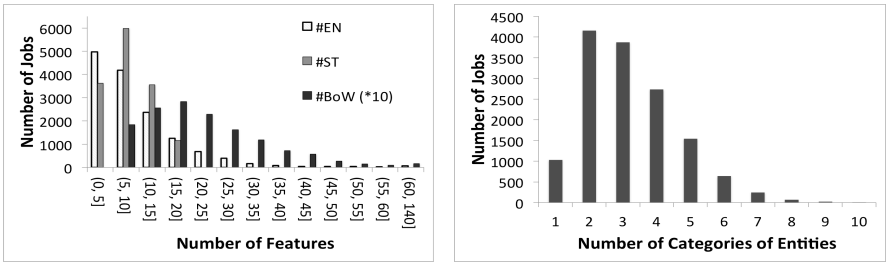
Similarity Histograms. Figure 3 shows histograms of the similarity values between all pairs of jobs for each feature set. Not surprisingly, most pairwise similarities are very low, confirming the narrow distribution of feature quantity. However, we can see that the pairwise similarities calculated based on categories of entities (*CAT_EN*) increase when these features are combined with the bag-of-words feature (*CAT_EN + BoW*). It is also worth noticing that a wider range of similarity values are obtained when categories of entities are combined with explicit features (*CAT_EN + EX*).

3.3 Recommendation Performance

Figure 4(*a*) shows the precision and recall results for our approaches, where the case-based and hybrid approaches attribute equal weight ($w_i = 1$) to all features. It is clear that *BoW* performed best among the content-based approaches. It also considerably outperformed the case-based recommenders (*CAT_EN* and *CAT_EN + EX*). This is expected because the job cases contain only small numbers of categories of entities and because of the low pairwise similarity values between jobs (*cf.* Figures 2(*b*) and 3(*d*)). Moreover, although *CAT_EN + EX* shows a wider range of pairwise similarities between cases (*cf.* Figure 3(*e*)), the generated recommendations are not effective.

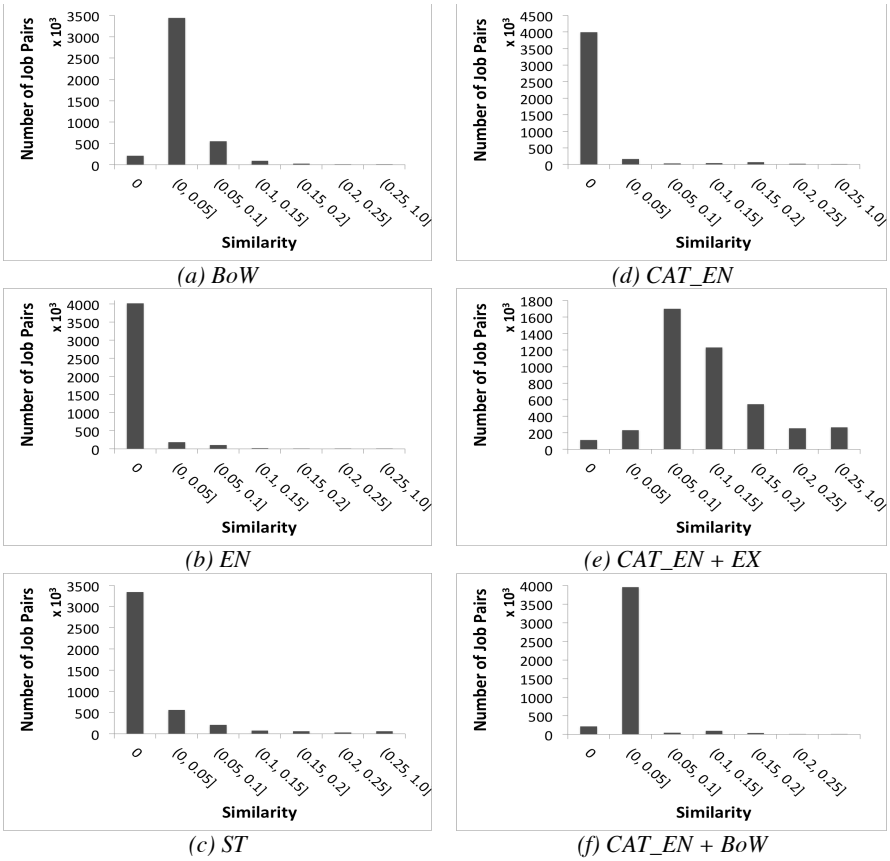
The hybrid approach (*CAT_EN + BoW)* was found to outperform *BoW*, albeit only at position 1 (i.e. for the top-ranked recommendation); thereafter, the *BoW* approach performed better. Nonetheless, this result confirms that the case-based representation, which includes the BoW feature, can boost performance. Moreover, as mentioned above, the hybrid approach (and also the entities and social tags approaches) depends on the quality of the categories of entities extracted by OpenCalais; the development

of an ad-hoc parser for the job domain is left to future work. Finally, it can be seen that all approaches outperformed the non-personalised baseline approach (*NP*).



(a) # of BoW, EN and ST features per job. *(b) # of categories of entities per job.*

Figure 2. Feature distribution histograms.



(a) BoW *(d) CAT_EN*

(b) EN *(e) CAT_EN + EX*

(c) ST *(f) CAT_EN + BoW*

Figure 3. Job similarity histograms.

In the above experiments involving the case-based recommenders, the feature weights in Equation 8 were all set to 1. In what follows, we analyse the performance

of our hybrid approach when different weighting schemes (as defined in Equations 11 and 12) are applied. Figure 4(*b*) shows the precision-recall curve.

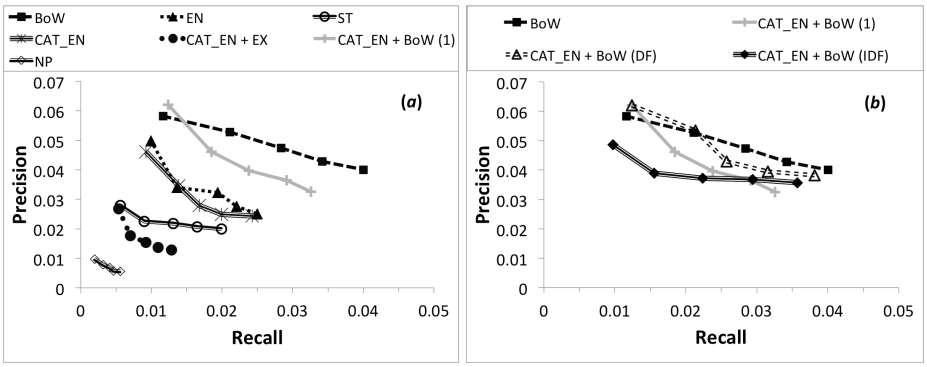


Figure 4. Performance comparison among the different job recommendation (*a*) and feature weighting (*b*) approaches.

It can be observed that weighting features according to their coverage of the case base (*DF*) leads to better case-based recommendations when compared to both *IDF* weighting and the baseline hybrid approach (equal weighting of features). Weighting by *IDF* performs poorly, only beating the baseline at position 5. More interestingly, the hybrid approach using *DF* weighting outperformed the *BoW* approach at positions 1 and 2 (albeit just marginally for the latter). Nevertheless, this gain is useful, given the recognised importance of generating accurate recommendations at the top of recommendation lists.

4 Conclusions and Future Work

We addressed the problem of job recommendation based on user job application history. We proposed different personalised content-based and case-based approaches that use features ranging from extracted features (bag-of-words and entities) to mined features (social tags) and explicit features (job attributes, e.g. job location, years of experience and job education). We also proposed a hybrid approach by combining content features with well-structured features, along with various feature-weighting schemes for the case-based approaches. The experiments that we conducted using a real-world dataset from CareerBuilder showed that our hybrid approach outperforms the other approaches, especially using the *DF* weighting scheme.

In future work we plan to extend our job recommendation framework by considering alternative content-based job representations; for example, by representing jobs based on the particular features of applicants. Moreover, we will apply collaborative-filtering style approaches to the task, and consider ways in which the various approaches may be combined to improve performance.

References

1. Mooney, R.J., Roy, L.: Content-based book recommending using learning for text categorization. Proceedings of the 5th ACM Conference on Digital Libraries, pp. 195–204 (2000)
2. Lee, J.S., Lee, J.C.: Context awareness by case-based reasoning in a music recommendation system. Ubiquitous Computing Systems, Vol. 4836 of Lecture Notes in Computer Science, pp. 45–58 (2007)
3. Paparrizos, I., Cambazoglu, B., Gionis, A.: Machine learned job recommendation. Proceedings of the 5th ACM Conference on Recommender Systems, pp. 325–328 (2011)
4. Yao, L., Helou, S.E., Gillet, D.: A recommender system for job seeking and recruiting website. Proceedings of the 22nd International Conference on World Wide Web Companion, pp. 963–966 (2013)
5. Rafter, R., Bradley, K., Smyth, B.: Automated collaborative filtering applications for online recruitment services. Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems, pp. 363–368 (2000)
6. Hong, W., Zheng, S., Wang, H.: Dynamic user profile-based job recommender system. Proceedings of the 8th International Conference Computer Science and Education, pp. 1499–1503 (2013)
7. Laumer, S., Eckhardt, A.: Help to find the needle in a haystack: Integrating recommender systems in an IT supported staff recruitment system. Proceedings of the Special Interest Group on Management Information System's 47th Annual Conference on Computer Personnel Research, pp. 7-12 (2009)
8. Smyth, B.: Case-based recommendation. The Adaptive Web, Vol. 4321 of Lecture Notes in Computer Science, pp. 342–376 (2007)
9. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Communications of ACM, Vol. 18, No. 11, pp. 613–620 (1975)
10. Salton, G., McGill, J.M.: Introduction to modern information retrieval. McGraw-Hill, Inc. (1986)
11. Nadeauand, D., Sekine, S.: A survey of named entity recognition and classification. Linguistic Investigations, Vol. 30, No. 1, pp. 3–26 (2007)
12. Porter, M.F.: An algorithm for suffix stripping. Readings in Information Retrieval, pp. 313–316 (1997)
13. Burke, R.: Hybrid recommender systems: Survey and experiments. User Modelling and User-Adapted Interaction, Vol. 12, No. 4, pp. 331–370 (2002)
14. Gupta, A., Rothkrantz, L.J.M.: JobScan. Proceedings of the 13th International Conference on Computer Systems and Technologies, pp. 352-359 (2012)