Ranking Job Offers for Candidates: learning hidden knowledge from Big Data

Marc Poch[1], **Núria Bel**[1], **Sergio Espeja**[2], **Felipe Navío**[2]

[1]Universitat Pompeu Fabra
Roc Boronat 138, 08018 Barcelona Spain
{marc.pochriera, nuria.bel}@upf.edu
[2]Jobandtalent Inc.
Eloy Gonzalo 27, 3, 28010 Madrid
{sergio.espeja, felipe.navio}@jobandtalent.com

Abstract

This paper presents a system for suggesting a ranked list of appropriate vacancy descriptions to job seekers in a job board web site. In particular our work has explored the use of supervised classifiers with the objective of learning implicit relations which cannot be found with similarity or pattern based search methods that rely only on explicit information. Skills, names of professions and degrees, among other examples, are expressed in different languages, showing high variation and the use of ad-hoc resources to trace the relations is very costly. This implicit information is unveiled when a candidate applies for a job and therefore it is information that can be used for learning a model to predict new cases. The results of our experiments, which combine different clustering, classification and ranking methods, show the validity of the approach.

Keywords: multilingual data, e-recruiting, LDA clustering methods, ranking methods

1. Introduction

Before the appearance of job board websites, labour market was mostly a local business: job agencies assisted employers finding profiled workers all in the same geographical location or in a particular specialized industry. The number of vacancies and candidates allowed for a manual, high quality selection process. The World Wide Web and job board websites have dramatically changed the dimensions of the task: they are handling an enormous number of job seekers and vacancies descriptions and in a very dynamic way. Job boards offer services to filter vacancies with keywords for job seekers to reduce the search space, and have devised powerful database search engines to assist human experts in the process of selecting the right candidates for a particular vacancy. But these facilities can hardly cope with the problem. Currently, human assisted selection processes cannot cope with the number and diversity of candidate profiles and vacancies, which are a constant stream of information. Language Technology can play a role in handling this big amount of information as to offer to each job seeker a selected number of vacancy descriptions, ranked according to his or her profile. The work we present here addresses this task.

In particular, our work has explored the possibility of using first a supervised learning method to build a model of the job market by learning from actual candidate applications to particular vacancies. After clustering job offer descriptions, the task is approached as a classification task where job clusters are the classes in which new job seekers are classified. Afterwards, we use a ranking method to order the job offers in the cluster.

The ultimate goal of the learning exercise is to handle the problem of differences in the way the information is expressed both in candidate profiles and vacancy descriptions. Besides the variation of the names of degrees, professions and job descriptions, there can be a mismatch in the way of making reference to particular skills. For example, a Candidate with "COBOL" programming skills may not match a Job Offer requiring "AS/400" (server that can be programed in C, RPG, COBOL...) skills. These hidden relations are unveiled when a candidate applies for a particular job. Our assumption is that an implicit relation is validated when several candidates sharing a particular skill apply for a job that does not contain explicitly this skill. Our proposal is an attempt to learn the hidden non-explicit relations between skills and job descriptions.

The data being used comes from Jobandtalent.com, a job board web site in which job seekers fill in a form with their profile information and they receive suggestions of appropriate job offers.The description of the job offer is a vacancy full description in free text. The information supplied by users can contain free text and in general is very scarce and non-accurate. A further problem is that candidate free text usually includes text in different languages, English and Spanish as in this example where text in Spanish is in bold: "**Curso de** Corporate Law and Governance **en la Universidad** London School of Economics" ('Course on Corporate Law and Governance at the University London School of Economics') User interaction information generated on Jobandtalent web site and stored in a database is used to learn the model.

2. State of the Art

First attempts for automatically selecting candidates for a particular job (Vega, 1990) were based on keyword search and content-based filtering techniques. Probably this is still the main approach in actual real world systems. These systems are supported with curriculum parsing engines that extract particular information from more or less structured text. After the success of job board websites, where job

seekers and job advertisers try to find each other, a number of research works focus on how to automate the task of relating the appropriate candidate to a job offer. In most cases, the task is approached in terms of Information Retrieval systems that try to find the "right" information and some of them use recommendation techniques worked out for relating products with customers.

A number of works have addressed candidate-job matching as a recommendation task. However, more recently several works have included statistical inference to efficiently handle all the information contained in profiles and job descriptions. Recommender systems are based on similarity measurements.

The so called Content-based Filtering approaches recommend new objects based on object similarities with objects selected in the past. The so called Collaborative Filtering approaches utilize user profile similarities to make recommendations. Rafter et al. (2000) used knowledge about similar user previous behaviour and used a cluster-based collaborative filtering approach to exploit transitive relationships between user profiles. Keim (2007) combined both sources of information, similarity of behaviour and profile similarity, using Probabilistic Latent Semantic Analysis techniques to model selection decisions. The task was to assess the probability that a recruiter rates a particular candidate with a binary value: "qualified" or "not-qualified" or, in the other direction, the probability that a job candidate rates a particular job in also two categories. Published results reported a 86% accuracy in classifying relevant candidates and 88% when classifying relevant and irrelevant jobs. However, experiments were based on a reduced experimental scenario with 30 students rating 100 job offers.

Malinowski et al. 2006 and 2008 also included information about user preferences. User preferences are obtained as trust information and used to predict new matches. The evaluation results were also based on a very reduced experimental setting, because of lack of real data, which, as authors themselves recognize, require of further validation in a real scenario.

Lu et al. (2013) applied graph modeling and afterwards a modified PageRank algorithm to extract top-most ranked job-candidate relations. In order to build the graph, several relations are considered: profile similarity, stated candidate preferences with respect to employers, candidate application for particular offers, and stated preferences as traced by "favorite", "like" and "visit" options in the web site. For this authors, the crucial step is the content-based similarity computing, that they solve by using Latent Semantic Analysis tools. In the evaluation experiment, they used a data-set containing 7000 candidates, 400 employers and 8000 job offers, although information about interaction with different website preference options was only available for 9% of candidates. Precision results for the task of assigning jobs to a candidate, taking into account only top 10 recommendations, ranges from 70% to 50%, outperforming two baselines: one using profile similarity and one using a Collaborative Filtering system. But evaluation was done manually and only for three candidates.

Interestingly, Lu et al. reported that the content-based approach required a lot of work such as profile formatting, job/employer title canonicalization, the creation of particular industry taxonomies, and equivalent lists, among other tasks and resources. Besides the problem of getting these resources, the main problem of the Content-based and Collaborative filtering approaches is that similarity computations reduce matching to explicitly mentioned information and, as mentioned, there can be differences that make similarity comparison difficult. For instance, Mimno and Mccallum (2008) reported that 85% of job titles in a 9,722 resume corpus occur only once, showing the difficulties that are met for computing similarity.

Our approach overcomes the similarity assessment problem training a classification model that relates candidate profiles with job offers based on available information about the actual interaction behaviour of candidates and recruiters. The main hypothesis is that it is possible to learn from the interaction data the relation between candidate skills and vacancy descriptions reducing the problem of linguistic variation and domain based implicit knowledge.

3. Description of the data

For our experiments, the following data was available: (i) a set of 80k candidate profiles, (ii) a set of 30k vacancy descriptions and (iii) a sample of 1M matching relations and (iv) a test-set made by a human expert who selected from one to a maximum 5 job vacancy descriptions for 126 candidates. Note that the human expert selection was in terms of "best options" and not "all possible options".

The candidate profiles (i) were made by actual users filling a form at the website that describe basic general profiles like languages spoken. The user can also write in free text his or her degrees, a personal description and other issues related to the cv. User profile and personal descriptions differ very much, some being made only with a few keywords, and other being more than 20 lines of free text describing personality aspects and experience. As mentioned, one of the most striking characteristics of these texts is that they can mix different languages. For our experiments we have taken the description and other general form fields like languages spoken. The vacancy descriptions (ii) we have worked with are the free text vacancy description provided by recruiting companies, which look as the usual advertising notes published in newspapers. Note that both types of users can also mark options from a list of "Area of Activity" (AofA) names that are meant to describe in a horizontal way the experience of candidates and business activities of the recruiting company: "On-line Marketing, "Finances", are examples of this field values. For candidates this information was considered useless after observing that a significant number of users tend to mark all options, and therefore was ignored. For job descriptions it was used mainly for job clustering, as we will see later.

As for (iii), the matching relations, we worked with real data from user behavior: job seekers that have applied for a particular vacancy and candidates that have been reviewed by a recruiting user for a particular vacancy. Both interactions are considered the same in our experiments. This "matching" information was used for training the classifiers. Finally (iv), the test-set was used for evaluation as

a gold-standard and we evaluated against it resulting precision and coverage as well as the position of the human-selected job descriptions in the final ranked list. For (i) and (ii), which include free text, a first pre-processing step is in charge of cleaning and correcting character encoding, if not in utf-8. Later a sentence-based language detector is applied (Shuyo, 2010)[1]. Language identification is crucial for the PoS tagger and lemmatizer (Padró and Stanilovsky, 2012)[2] applied afterwards. Stop words are then filtered and only lemmas are used for building a bag-of-words vector representation of both candidate profiles and job offer descriptions.

4. System description: training

This section describes the steps involved in creating all necessary models and processes that are required to rank job offers for a new candidate. An schema of the training models process is presented in Figure 1 on page 4.

4.1. Job Offers clustering

As said before, the field Area-of-Activity (AofA) that recruiting users could mark when entering data in the job board was a possible solution to cluster similar job descriptions. Job clustering was required to approach the assignment task in terms of classification. Each candidate will be classified for one or more classes of jobs according to the model learnt from the matching information. Nevertheless, two other clustering methods have been used in our experiments: k-means (Shindler et al., 2011) and Topic Models using Latent Dirichlet Allocation (LDA) (Blei, 2012). For the k-means clustering, job offers have been represented in a vector space and used to obtain different centroids. All jobs end up being assigned by minimum distance to one of the k centroids. As for Topic Models, job offers are represented in a vector space where each component represents a topic. For the experiments, we used GibbsLDA++(Phan and Nguyen, 2007) implementation of LDA, which was designed to obtain topic models of Big Data structures. Unlike k-means, where all jobs are easily assigned to a cluster by the vector distance to all the centroids, Topic Models do not deliver a single topic a vector belongs to. In Topic Models, each document is represented as the probabilities of belonging to a number of topics. That is, each job offer is assigned a probability for each of a number of possible topics. The number of topics is set a priori (Blei, 2012). To be able to define clusters of job offers, a threshold has been experimentally set. All jobs that exceed this threshold are considered to belong to that Topic cluster.

4.2. Users feedback: assign candidates to jobs clusters

After all job offers have been assigned to a cluster in the previous step, each candidate (from training data) is assigned to a job cluster along the matching information. Jobandtalent.com records and stores user-job board interaction information. The possible interactions are when the candidate applies for a particular job offer; when the recruiting user looks at the profile of a particular candidate. Each one of this ”interactions“ is what we called a matching. With all this feedback we can establish a relation between each candidate and one or more job clusters. Since every job offer belongs to a job cluster defined in the previous step, the candidate can be associated to a job cluster via the matching relations. All job clusters for which there exists a matching are considered to be associated with a candidate.

4.3. Training classifiers

Once the relation between candidates and job clusters is set, we use this data to train classification models. The goal is to have predictive models that assign new candidates to one or some job clusters. We chose Support Vector Machines (Joachims, 1999) (SVM) to make the classification models. LIBLINEAR(Fan et al., 2008) was chosen because it has been specially designed for document classification with millions of instances and features. This Big Data capability make it the best option for the experiments and the expected growing data that is foreseen by Jobandtalent.
The set used as training data initially had around 80,000 features after stop words were removed. In order to reduce this large dimensionality, Chi Square feature selection has been used to select the features for each classifier. The different clustering methods produce different clusters. Thus, there is a different average number of selected features. Table 1 shows the average number of selected features (avg. features) for each classifier using the Chi Square method and the vocabulary size (vocabulary).

clustering method	vocabulary	avg. features
AofA	80113	2407
k-means	80113	2085
LDA	80113	3034

Table 1: Chi Square feature selection

Candidates in the training data are represented as feature vectors and associated to different corresponding job clusters. For each job cluster, a binary classifier is trained and will be used to predict if a new candidate belongs or not to that job cluster (a candidate may belong to different Job Clusters). To train the classifier for a job cluster, all candidates with a matching with one of the job offers represented by it are used as positive examples and the rest as negative ones. To evaluate each classifier, 300 vectors are reserved to be used as test-set.

5. System description: ranking job offers for a new candidate

After a new candidate is assigned to a cluster, the system ranks all the job offers in this cluster with the aim of highlighting those that are more appropriate.The whole process for new candidates is presented in Figure 2 on page 5.

5.1. New candidate classification: predicting the job cluster

First, the candidate profile is cleaned and the same text preprocessing explained for the training phase is applied: lan-

[1] a Java detection library that can detect 53 different languages

[2] Freeling is an open-source Suite of Language Analyzers in which performance has been optimized.

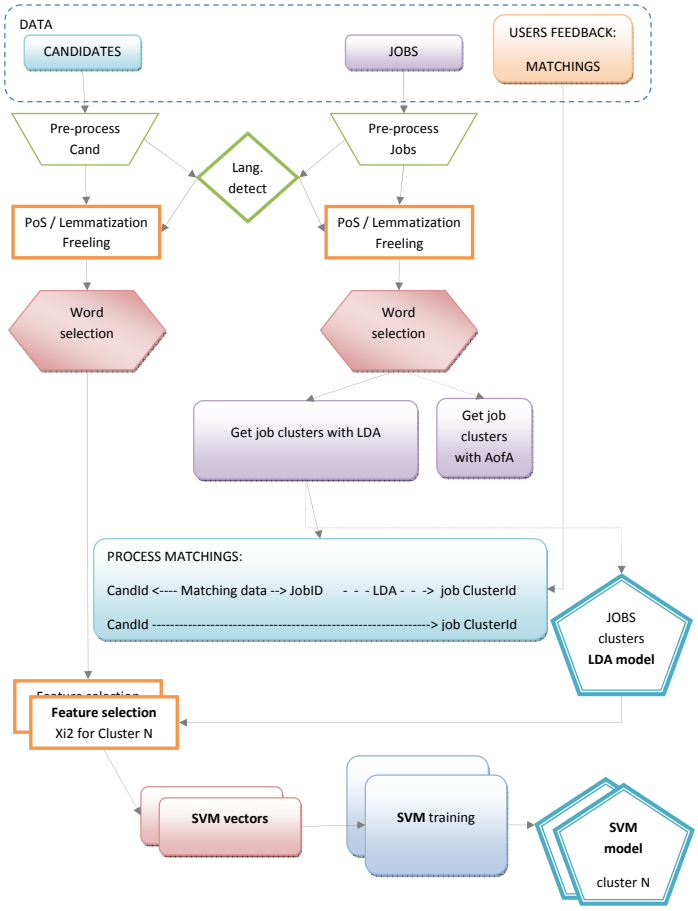

Figure 1: Training Models

guage detection, PoS tagging, lemmatization and word filtering. Because feature selection is class-based, a different candidate vector has to be made for each classifier. The SVM library is executed in prediction mode deciding for the new candidate whether it belongs to each job cluster or not. Finally, a list of Job Clusters is obtained for this new Candidate.

5.2. Ranking job offers in cluster for the new candidate

Once the appropriate job clusters are found for a new candidate, the system ranks the jobs in the cluster. If the candidate has been assigned more than one cluster, the jobs in all assigned clusters are ranked together. In the experiments, three ranking methods have been used: BM25, cosine similarity and Naive Bayes (Zhao et al., 2006).

5.2.1. BM25 ranking

BM25 is a bag-of-words retrieval function used to rank a set of documents taking into account the query terms, document lengths and term frequencies but not the term relations within the query or the documents. Given a query q, the score for each document b, in a collection of documents D, can be calculated with the following formula:

$$S_{BM25}(q,d) = \sum_{t \epsilon q} \left\{ \frac{tf_{t,d} \cdot log(\frac{N - df_t + 0.5}{df_t + 0.5})}{0.5 + 1.5 \frac{dl_d}{dl_{avg}} + tf_{t,d}} \right\}$$

where $tf_{t,d}$ is term frequency of word t in document d, df_t is document frequency of word t in collection D, N is the number of documents in D, dl_d is the length of document d, and dl_{avg} is the average length of documents in D. In our experiments jobs are seen as documents that are being ranked depending on their relevance for each query or candidate.

5.2.2. Cosine similarity ranking

The cosine similarity is also used as a ranking method. Given a candidate, its cosine similarity is calculated with respect to each job.

Given two vectors of n attributes, q (query or candidate) and d (document or job), the cosine similarity, $cos(\theta)$, is represented as:

$$cos(\theta) = \frac{q \cdot d}{||q|| \cdot ||d||} = \frac{\sum_{i=0}^{n} q_i \cdot d_i}{\sqrt{\sum_{i=0}^{n} (q_i)^2} \cdot \sqrt{\sum_{i=0}^{n} (d_i)^2}}$$

The resulting values can be ordered to produce the ranking for a candidate.

5.2.3. Naive Bayes ranking

For the Naive Bayes ranking (NB), the same matching information used for the cluster classifiers is now used to learn a classifier where each job is a class. The candidates represent instances to be classified into the corresponding category. Our experiments followed Zhao et al. (2006)

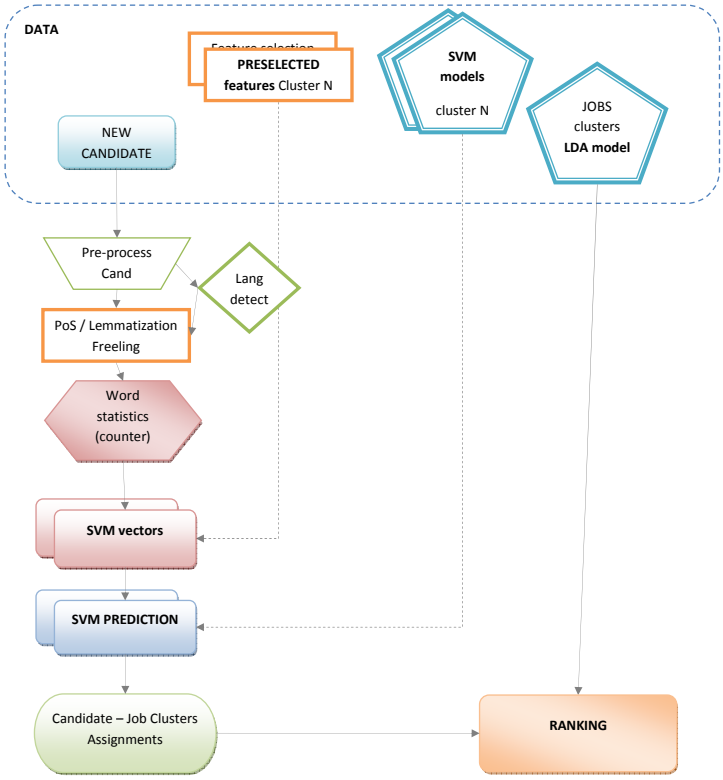

Figure 2: New Candidate

where the score of a document d (a job offer) with respect to a query q (a candidate), formed by a set of words t_i, can be calculated by

$$S_{NB}(q,d) = log\left\{\frac{p(d|q)}{p(\overline{d}|q)}\right\}$$

where $\overline{d}$ consists of all documents except d or, in our case, all job offers to be ranked except job d.
The conditional probabilities can be calculated with

$$p(d|q) = p(d)\cdot\prod_{t\epsilon q} p(t|d) = p(d)\cdot\prod_{t\epsilon q}\left\{\frac{tf_{t,Qd}+1}{tf_{Qd}+|V_Q|}\right\}$$

$$p(\overline{d}|q) = p(\overline{d})\cdot\prod_{t\epsilon q} p(t|\overline{d}) = p(\overline{d})\cdot\prod_{t\epsilon q}\left\{\frac{tf_t - tf_{t,Qd}+1}{tf_Q - tf_{Qd}+|V_Q|}\right\}$$

where Qd is the subset of queries in Q associated with document d, $tf_{t,Qd}$ is the term frequency of word t in Qd, tf_t is the term frequency of t in Q, tf_{Qd} is the total term frequency in Qd, tf_Q is the total term frequency in Q, and $|V_Q|$ is the number of unique words in Q.
These formulas may not be computable depending on the data size and its characteristics. Some of the probabilities may be very low and this could cause and underflow situation (the computer cannot represent such numbers). To help prevent this situation we used the following properties of logarithm: $log(x*y) = log(x) + log(y)$ and $log(x^y) = y*log(x)$ to rewrite the score formula as follows:

$$S_{NB}(q,d) = log\left\{\frac{p(d|q)}{p(\overline{d}|q)}\right\} =$$

$$= log\left\{p(d|q)\right\} - log\left\{p(\overline{d}|q)\right\} =$$

$$= log\left\{p(d)\right\} + \sum_{t\epsilon q} log\left\{\frac{tf_{t,Qd}+1}{tf_{Qd}+|V_Q|}\right\}$$

$$-log\left\{p(\overline{d})\right\} - \sum_{t\epsilon q} log\left\{\frac{tf_t - tf_{t,Qd}+1}{tf_Q - tf_{Qd}+|V_Q|}\right\}$$

6. Experiments

We conducted twelve different experiments combining the three different clustering methods and the three ranking methods in addition to the rankings without clustering (all test jobs ranked for each candidate). The following sections will explain the data used in the experiments, the evaluation methods and the results obtained.

6.1. Train data

Train data consists of a set of candidates, jobs and matchings. As presented in Table 2, there are around 80k candidates and 30k jobs. The train data also has 1M matchings. A matching defines an association between a candidate and a job. It can be positive or negative: when a candidate applies for a job the matching is positive.

	Elements	Words	Vocabulary
Candidates	77879	4.47 M	80113
Jobs	28520	4 M	47887

Table 2: Train data table

6.2. Test data

The test data is presented in Table 3. For each of the 389 test candidates, a set within the 126 test job offers must be selected and ranked. For each candidate, the best possible job offers have been manually selected by a human expert and will be used for evaluating precision and recall, and the position in the ranked list finally delivered. Each candidate can have more than one manually selected job offer. These selected associations between a candidate and a job offer will be called expected matchings. There are 658 expected matchings in the test data.

	Elements	Words	Vocabulary
Candidates	389	29624	4554
Jobs	126	19255	3284

Table 3: Test data table

7. Evaluation

Table 4 in page 7 presents the results for the twelve experiments combining the three clustering methods and the three ranking methods. Also the results without clustering are presented. The results show the precision and the coverage for each experiment to evaluate how well the classifiers are performing. The table also shows the expected or correct job position average (*pos. avg.*): The closer to position one the job selected by the human expert is, the better. The position standard deviation (*pos. dev.*) is also presented, as well as the ranking average length (*len. avg.*). The ranking average length is the average size of the ranked lists of job offers given to candidates and ultimately depends on the number of clusters assigned to a candidate, and the size of each cluster.

The results show that LDA is giving the best clustering results and, at the same time, the size of the ranked lists it provides is also the smallest (which is one of the goals). Regarding the ranking methods, BM25 and cosine similarity are very close. The best results are obtained using the Naive Bayes ranking. The correct position average and deviation allow us to estimate if the ranked list can even be cut while still containing the selected job.

Precision results on Table 5 in page 6 show precision in the form p@n or precision at N. The resulting ranked lists are cut at N first ranked jobs and evaluated again. The table shows that LDA clustering and Naive Bayes ranking get the best results.

P@10 are specially important results since this is the threshold that is considered to be key regarding user experience for Jobandtalent. It is considered that a user has a very low interest in job offers listed after position 10.

clustering	ranking	p@1	p@10	p@30	p@All
aofa	bm25	4.64	20.22	40.16	62.84
kmeans	bm25	4.96	24.24	45.73	71.07
lda	bm25	4.95	26.65	50.82	74.18
aofa	cosine	4.64	18.85	40.71	62.84
kmeans	cosine	4.41	22.87	45.45	71.07
lda	cosine	4.95	29.67	53.02	74.18
aofa	Bayes	3.83	31.15	50.82	62.84
kmeans	Bayes	6.37	36.29	58.45	71.47
lda	**Bayes**	**8.26**	**40.22**	**65.01**	**74.38**

Table 5: Experiment results: precision@

In order to further assess the results, we manually inspected the adequacy of the 10 best-ranked job offers for 20 randomly chosen candidates obtained with the LDA-SVM-Bayes system. For this evaluation, we compared suggested job offers with the matching data-set, that is, the actual interest of users. But we also assessed the adequacy of the rest of suggestions as this is the segment of suggestions where we expected to see the validity of our hypothesis: that the market model could have been effective in discovering implicit knowledge and market tendencies.

In average, each candidate received 1.9 job offers that were in the test-set used as gold-standard, and thus they are clearly adequate. About 3.8 of the suggestions could not be considered appropriate because there were unmatched requirements (for instance requiring engineers and the candidate being a lawyer), and the rest shows the capacities of the system: most of these about 4 other suggestions for each candidate are traineeships or internships with a broad range of profiles required, as in this case: “Buscamos recién titulados y estudiantes de último curso de diversos sectores, interesados en colaborar con nosotros a través de un programa de beca.“ (”We are looking for recent graduates and final year students from various sectors interested in collaborating with us through a scholarship program”).

The suggestions considered as wrong also include, in fact, suggestions that could be considered a side-effect of the approach as they usually are the most assigned vacancies: ticket sales, marketing online, etc. As in what respects multilinguality, in the assessed results there is a crosslingual effect so that candidates who have written the profile in Spanish get suggestions written in English, validating the assumption that information should not be processed separately.

8. Conclusion and future work

We have presented a system for suggesting a ranked list of appropriate vacancy descriptions to job seekers in a job board web site. Our contribution is to have used supervised classifiers for learning implicit relations between candidate profiles and job descriptions. These implicit relations are basically linguistic variation in naming skills or profession names and crosslingual relations. Moreover implicit relations we want to capture also include domain-based knowledge (for instance the relation between AS/400 and COBOL). Other systems can only discover these relations if adhoc resources were used. The results obtained in

clustering	ranking	precision	coverage	pos. avg.	pos. dev.	len. avg.
aofa	bm25	62.84	99.74	28.51	26.14	83.23
kmeans	bm25	71.07	98.97	27.42	26.42	73.34
lda	bm25	74.18	98.97	24.78	23.69	67.25
none	bm25	100.00	100.00	35.12	33.22	126.00
aofa	cosine	62.84	99.74	27.79	25.03	83.23
kmeans	cosine	71.07	98.97	28.12	25.02	73.34
lda	cosine	74.18	98.97	23.69	22.75	67.25
none	cosine	100.00	100.00	36.11	32.59	126.00
aofa	Bayes	62.84	99.74	17.34	16.35	83.23
kmeans	Bayes	71.47	97.43	15.93	16.07	73.32
lda	**Bayes**	**74.38**	**98.71**	**13.87**	**13.90**	**67.25**
none	Bayes	100.00	100.00	21.46	21.20	126.00

Table 4: Experiment results

our experiments demonstrate the validity of the approach. For our test-set of 126 candidates, the best system delivered in position 13, in average, the job offer selected by a human expert and with a precision of 74.38%. A manual inspection to assess the quality of the top 10 ranked job offers for 20 candidates, confirmed that, in average, a new candidate will find more than a half of the suggested vacancies appropriate to his or her profile. While these results show the validity of the approach, we expect to improve the selected top 10 list by performing a better processing of the job and candidate descriptions as well as the linear combination of the different ranking algorithms and models.

9. References

David M. Blei. 2012. Probabilistic topic models. *Commun. ACM*, 55(4), April.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Thorsten Joachims. 1999. Advances in kernel methods. chapter Making large-scale support vector machine learning practical, pages 169–184. MIT Press, Cambridge, MA, USA.

Tobias Keim. 2007. Extending the applicability of recommender systems: A multilayer framework for matching human resources. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, HICSS '07, pages 169–, Washington, DC, USA. IEEE Computer Society.

Yao Lu, Sandy El Helou, and Denis Gillet. 2013. A recommender system for job seeking and recruiting website. In *Proceedings of the 22Nd International Conference on World Wide Web Companion*, WWW '13 Companion, pages 963–966, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee.

David Mimno and Andrew Mccallum. 2008. Modeling career path trajectories.

Lluís Padró and Evgeny Stanilovsky. 2012. Freeling 3.0: Towards wider multilinguality. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey, May. ELRA.

Xuan-Hieu Phan and Cam-Tu Nguyen. 2007. Gibbslda++: A c/c++ implementation of latent dirichlet allocation (lda).

Rachael Rafter, Keith Bradley, and Barry Smyth. 2000. Automated collaborative filtering applications for online recruitment services. In Peter Brusilovsky, Oliviero Stock, and Carlo Strapparava, editors, *Adaptive Hypermedia and Adaptive Web-Based Systems*, volume 1892 of *Lecture Notes in Computer Science*, pages 363–368. Springer Berlin Heidelberg.

Michael Shindler, Alex Wong, and Adam Meyerson. 2011. Fast and accurate k-means for large datasets. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *NIPS*, pages 2375–2383.

Nakatani Shuyo. 2010. Language detection library for java.

Jose Vega. 1990. Semantic matching between job offers and job search requests. In *COLING*, pages 67–69.

Min Zhao, Hang Li, Adwait Ratnaparkhi, Hsiao-Wuen Hon, and Jue Wang. 2006. Adapting document ranking to users' preferences using click-through data. In Hwee Tou Ng, Mun-Kew Leong, Min-Yen Kan, and Dong-Hong Ji, editors, *AIRS*, volume 4182 of *Lecture Notes in Computer Science*, pages 26–42. Springer.