Taxonomy-based Job Recommender Systems On Facebook and LinkedIn Profiles

Mamadou Diaby *†, Emmanuel Viennet *
* Université Paris 13, Sorbonne Paris Cité, L2TI, F-93430, Villetaneuse, France
{mamadou.diaby, emmanuel.viennet}@univ-paris13.fr
† Work4, 3 Rue Moncey, 75009, Paris, France

***Abstract*—This paper presents taxonomy-based recommender systems that propose relevant jobs to Facebook and LinkedIn users; they are being developed by Work4, a San Francisco-based software company and the Global Leader in Social and Mobile Recruiting that offers Facebook recruitment solutions; to use its applications, Facebook or LinkedIn users explicitly grant access to some parts of their data, and they are presented with the jobs whose descriptions are matching their profiles the most.**

In this paper, we use the O*NET-SOC taxonomy, a taxonomy that defines the set of occupations across the world of work, to develop a new taxonomy-based vector model for social network users and job descriptions suited to the task of job recommendation; we propose two similarity functions based on the AND and OR fuzzy logic's operators, suited to the proposed vector model. We compare the performance of our proposed vector model to the TF-IDF model using our proposed similarity functions and the classic heuristic measures; the results show that the taxonomy-based vector model outperforms the TF-IDF model. We then use SVMs (Support Vector Machines) with a mechanism to handle unbalanced datasets, to learn similarity functions from our data; the learnt models yield better results than heuristic similarity measures. The comparison of our methods to two methods of the literature (a matrix factorization method and the Collaborative Topic Regression) shows that our best method yields better results than those two methods in terms of AUC. The proposed taxonomy-based vector model leads to an efficient dimensionality reduction method in the task of job recommendation.

***Keywords*—Job recommendation, Taxonomy, O*NET, Dimensionality reduction, SVM, Facebook, LinkedIn**

I. Motivations

The rapid growth of social networks in recent years has developed a new business: the trade of social networks users data. A social network site is often defined as a web-based service that allows users to construct profiles (public or semi-public), to share connections with other users and to view and traverse lists of connections made by others in the system [1]. Information posted by social network users (personal description, posts, ratings, ...) can be used by recommender systems to determine their interests for items in order to advertise items to them. Recommender systems are often defined as software that elicit the interests or preferences of individual consumers for products, either explicitly or implicitly, and make recommendations accordingly [2]; they have become an active research field for two decades.

This paper presents taxonomy-based recommender systems that extract social network users' interests for job offers and then make recommendations to them accordingly. It is focused on two very popular social networks Facebook and LinkedIn, each counting millions of active users [3, 4]. Due to privacy concerns the proposed recommender systems only use the data that social network users explicitly granted access to.

Our previous work [5] showed that systems that use TF-IDF model together with similarity functions to recommend jobs to Facebook and LinkedIn users perform poorly due to incomplete and noisy data of social network users (especially in Facebook): we need a richer representation of social network users and job descriptions. This leads us to develop a new representation for our users and jobs based on O*NET-SOC[1] taxonomy, an ontology that defines the set of occupations across the world of work; it is being developed under the sponsorship of the US Department of Labor/Employment and Training Administration (USDOL/ETA).

The contributions of this paper are 2-fold:

1) we propose a new representation of social network users and job descriptions suited to the task of job recommendation; this representation is an alternative to the TF-IDF; the proposed representation model is an efficient dimensionality reduction method in the task of job recommendation;
2) we propose two similarity functions based on fuzzy logic's operators adapted to our proposed representation and we show that the use of machine learning together with our proposed representation model yields better results than heuristic similarity measures.

The rest of this paper is organized as follows: we summarize the state-of-the-art of recommender systems and ontologies in the section II; the sections III and IV respectively present the proposed methods and the obtained results and we discuss and conclude in section V.

II. Related Work

In this paper, our proposed methods are related to recommender systems and information retrieval.

A. Recommender Systems

Recommender systems [6, 7] are mainly related to information retrieval [8], machine learning [9], data mining [10] and other research fields beyond the scope of this study. [6] classified recommender systems into two main groups: rating-based systems and preference-based filtering techniques. Rating-based recommender systems focus on predicting the

[1] http://www.onetcenter.org/taxonomy.html.

978-1-4799-2393-9/14/$31.00 ©2014 IEEE

absolute values of ratings that individual users would give to the unseen items while preference-based filtering techniques predict the correct relative order of items for a given user; this paper is focused on rating-based recommender systems. Recommender systems are generally classified into three or four categories [11, 6]: content-based methods, collaborative filtering, demographic filtering systems and hybrid approaches.

Content-based recommender systems [12] use the descriptions of users and items (if they are available) as well as the ratings that users gave to items in the past to define their profiles [6] while recommendations are made in demographic filtering systems by using users personal attributes (age, gender, income, country, survey responses, etc.) [11, 13]. In contrast to content-based systems, collaborative filtering methods use the opinions of community of similar users to recommend items to a the active user [7, 6] while a hybrid recommender system combines two or more types of recommender systems into a single model [11]. Hybrid systems generally yield better results than simple recommendation techniques [6] but are much more complex to design.

B. Data representation and similarity functions

In recommender systems, the textual description of a document (user or item) is generally represented as a vector in which each component has a value that represents the importance of the associated term for the document. This vector is generally constructed using weighting functions and the "bag-of-words" model and by filtering out some terms (stop words or some grammatical categories or the very highest and lowest frequencies words, ...).

A weighting function calculates the importance of a term for a document. We used TF-IDF as weighting function in our previously developed recommender systems [5] (used as baseline methods in this paper). TF-IDF is a combination of a local weighting function (which calculates the importance of a term in a given document) and a global weighting function (which uses the whole corpus to calculate the weight of a given term); it is known to yield good results in information retrieval [8] and is defined as follows:

$$\text{TF-IDF}_{t,d} = \text{TF}_{t,d} \times \text{IDF}_t \tag{1}$$

where $\text{TF}_{t,d} = \frac{f_{t,d}}{\max_k f_{k,d}}$, $\text{IDF}_t = 1 + \log\left(\frac{N}{n_t}\right)$, $f_{t,d}$ is the frequency of the term t in the document d, N is the total number of documents in the corpus, n_t is the number of documents that contain the term t.

Two popular heuristic functions [6, 7] used by recommender systems to compute similarity between users, between items or between users and items are cosine similarity and Pearson Correlation Coefficient (PCC) defined as follows:

$$\cos(u, v) = \frac{\sum_{k=1}^{n} u_k v_k}{\sqrt{\sum_{k=1}^{n} u_k^2}\sqrt{\sum_{k=1}^{n} v_k^2}} \tag{2}$$

$$\text{PCC}(u, v) = \frac{\sum_k (u_k - \overline{u})(v_k - \overline{v})}{\sqrt{\sum_k (u_k - \overline{u})^2}\sqrt{\sum_k (v_k - \overline{v})^2}} \tag{3}$$

where u and v are the vectors of users or items, $\overline{u}$ and $\overline{v}$ the mean values of u and v respectively.

C. Ontologies

Our proposed vector model is related to ontologies and taxonomies. Ontologies have several definitions depending on the context: the term refers to the study of existence in philosophy while in computer science, it refers to representations useful to explain the world(s) as perceived by a given application [14]. They are generally classified into three main categories [15, 14]: formal, terminological and prototype-based ontologies. The different concepts are based on definitions and axioms in formal ontologies while they are defined by typical instances also called prototypes in prototype-based ontologies; in terminological ontologies, they are distinguished by using subtype-supertype relations and describing concepts by labels or synonyms [15]. Taxonomies are collections of entities ordered by a classification scheme and usually arranged hierarchically, this corresponds to the notion of terminological ontologies [14]. There is only one type of relation in taxonomies: "IS-A"or "PART-OF" according to [14]. Several studies reported that the use the ontologies could improve the results of an information retrieval system [16].

D. Performance metrics

This paper is focused on rating-based recommender systems (c.f. section II-A) in which we have only 2 values for ratings (0 and 1); we therefore use AUC (Area Under the Curve of a ROC) as performance metric for our recommender systems since it is suited to this type of problems and does not involve any threshold (as the Precision, Recall and F-measure do). The curve of a ROC (Receiver Operating Characteristic) [17] is obtained by plotting the TP rate (fraction of true positives) as a function of FP rate (fraction of false positives). Theoretically the min and max values of the AUC are respectively 0 and 1 but if the AUC is lower than 0.5 (value for a classifier that randomly assigns the different labels), one can inverse each of the predictions to obtain an AUC greater than 0.5. In our industrial context, we cannot afford to invert the predictions even if the AUC is below 0.5.

III. Job Recommender Systems

This section presents in details our proposed vector model and job recommender systems.

A. Taxonomy-based vector for social network users and jobs

Our Facebook users authorized the Work4's applications to access 5 fields data: *Work*, *Education*, *Quote*, *Bio* and *Interests*; LinkedIn users only authorized 3 fields: *Headline*, *Educations*, *Positions* and our job descriptions have 3 fields: *Title*, *Description*, *Responsibilities*. Our users and jobs are seen as documents; their different fields are sub-documents.

Our previous work [5] showed that the most important fields in the task of job recommendation are: *Work* for Facebook users, *Headline* for LinkedIn users and *Title* for jobs; however, they also showed the use of the TF-IDF model to recommend jobs to our social network users might be limited (missing and noisy data) as a result, we use the information contained in these three important fields to extract new a type of vector for social network users and jobs (that we called O*NET vector) using the O*NET-SOC taxonomy as described in the pseudo algorithm 1 and presented in Fig. 1.

We indexed O*NET database using Elasticsearch[2] and we thus use this library to query the different O*NET occupations (with their relevance scores) related to a document. Since O*NET database currently supports only English language, we test our proposed methods on users and jobs whose languages are English.

Pseudo algorithm 1: Extraction of O*NET vectors.

Input:
doc: a document that represents a user or job
Output:
d_vector: the O*NET vector of doc (a list of couples (O*NET occupation, relevance score))

```
if doc.type = 'Job' then
    Extracting the text in the Title field;
    text ← extract_from_field(doc, sub-field='Title');
else if doc.type = 'LinkedIn' then
    Extracting the text in the Headline or Last position field;
    text ← extract_from_field(doc, sub-field='Headline');
    if text = "" then
        text ← extract_from_field(doc, sub-field='Last position');
else if doc.type = 'Facebook' then
    Extracting the text in the Last position field;
    text ← extract_from_field(doc, sub-field='Last position');
else
    Unknown type of document;
    text ← ""
end
Cleaning the text;
text ← remove_stopwords(text);
Quering the O*NET database: returns a list of couples (occupation, relevance score);
onet_vector ← query_O*NET_dbs(text);
max_relevance ← get_max_relevance(onet_vector);
d_vector = empty list;
foreach (occ, relevance) in onet_vector do
    d_vector ← append(d_vector, (occ, relevance/max_relevance));
end
Return d_vector
```

The following example in which we consider a user who only filled his last position sub-field with "software engineer", shows the difference between our proposed O*NET vector model and the TF-IDF vector model:

- the user's TF-IDF vector could be *[("software", 5.8), ("engineer", 3.2)]*;
- while his O*NET vector will look like *[("Software Developers", 1), ("Aerospace Engineers", 0.97), ("Electrical Engineers", 0.97), ..., ("Software Quality Assurance Engineers and Testers", 0.85), ..., ("Web Developers", 0.52), ..., ("Database Architects", 0.24), ..., ("Avionics Technicians", 0.01)]*.

[2]http://www.elasticsearch.org

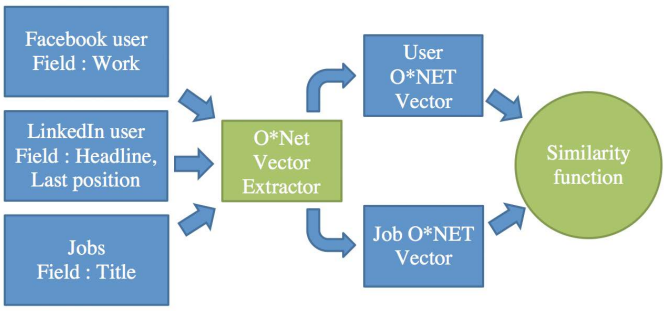

Fig. 1: Scheme of our taxonomy-based job recommender systems.

B. Proposed Recommender Systems

We propose in this paper four heuristic-based and one SVM-based recommender systems using our vector model.

1) Baseline methods: Engine-1 and Engine-2: We use two of our previously developed recommender systems [5] as baseline methods: Engine-1 and Engine-2. The TF-IDF vectors of users and jobs are computed in Engine-1 by assuming that all the fields have the same importance on recommendation scores while we have used the optimal importance of users' and jobs' fields (computed in [5]) when calculating the vectors of documents as weighted sum of the TF-IDF vectors of their fields in Engine-2. We measure the interest of a user for a given job by computing the cosine similarity (see equation (2)) of the user's vector and the vector of the job in the two systems.

2) Classic heuristic similarity-based recommender systems: Engine-3a and Engines-3b: Both the systems use our proposed vector model together with heuristic-similarity functions: Engines-3a and Engines-3b use the cosine similarity (see equation (2)) and PCC (see equation (3)) respectively.

3) Fuzzy logic-based recommender systems: Engine-3c and Engine-3d: We proposed two similarity functions based on some AND and OR fuzzy logic's operators [18] and adapted to our vector models for job recommendation to social network users; among fuzzy logic's operators [19], we use *max* as the OR operator with *min* and *product* as AND operators:

$$\text{fuzzy-sim-1}(u,v) = \max_{k=1}^{K}(u_k.v_k) \tag{4}$$

$$\text{fuzzy-sim-2}(u,v) = \max_{k=1}^{K}\min(u_k,v_k) \tag{5}$$

where K is the total number of O*NET occupations (1,040), u and v are the vectors of a user and a job respectively.

Engine-3c and Engine-3d use the proposed vector model with respectively fuzzy-sim-1 and fuzzy-sim-2 as similarity functions.

4) SVM-based recommender system: Engine-4: We also explored the use of trained statistical models: SVMs (Support Vector Machines) [20] which are known to yield good performance in text categorization [21]. Using the libSVM [22], we therefore apply this supervised learning procedure to our problem; this leads to our fifth recommender system: Engine-4. The input vectors I_{SVM} of the SVM are stated as follows:

$$I_{\text{SVM}}(u,v) = (u,v) = (u_1,\ldots,u_K,v_1,\ldots,v_K)$$

where K is the total number of O*NET occupations (1,040), u and v are respectively the O*NET vectors of a user and job.

We decide to use a linear model (linear SVM): quicker and avoid the over-fitting problem in general; since we are using a linear kernel $K(I_1, I_2) = \langle I_1, I_2 \rangle$ when I_1 and I_2 are two input data of the SVM, if $I_1 = (u_1, v_1)$ and $I_2 = (u_2, v_2)$, we notice that $K(I_1, I_2) = \langle u_1, u_2 \rangle + \langle v_1, v_2 \rangle$; this means that our model uses both similarities between users and between jobs to learn models from data. In order to efficiently handle unbalanced datasets, we use different costs for the two classes: c_0 and c_1 for the label 0 and the label 1 respectively.

IV. Experiments

We use AUC as performance metric for our different the experiments and we compute 95% empirical confidence intervals ([2.5% quantile, 97.5% quantile]) using bootstrapping (100 runs per experiments). We compare our methods to two methods of the literature. The first method is a simple Collaborative Filtering (CF) based on matrix factorization and the second method is the Collaborative Topic Regression (CTR) proposed by [23] (we use the code provided by [23]). For the CTR and CF, we use the $25,000$ terms for users and $25,000$ terms for jobs with the highest TF-IDF weights and the following parameters: $a = 1$, $b = 0.1$, num_factors $= 50$ (number of latent dimensions) and max_iter $= 50$ (max number of iterations); a and b are respectively the confidence parameters for the labels 1 and 0.

A. Description of datasets

We evaluate the performance of our systems on 6 datasets collected by the company Work4. Each entry in our datasets is a 3-tuple (u, v, y) where u and v are the vectors of a given user and job respectively and $y \in \{0, 1\}$ is their associated label; label 1 denotes a matching between the user and the job while the label is 0 when the job does not correspond to the user. Here are the descriptions of the 6 collected datasets:

1) Candidate: users can use Work4's applications to apply to jobs, we assume that users only apply to the jobs that are relevant for them (label =1); this dataset contains applications' data.
2) Feedback: contains the feedback from users of Work4's applications.
3) Random: this dataset contains couples of users - jobs that have been randomly drawn from Work4's databases and manually annotated.
4) Review: it contains recommendations made by Work4's systems that have been manually validated by two different teams of the company.
5) Validation: it contains recommendations made by Work4's systems that have been manually validated by only one team of the company.
6) ALL: a sixth dataset has been artificially created: it is the union of the five previous datasets.

Each job is associated to a job page which generally represents a page of a company, hence jobs from the same job page are generally similar since they are likely from the same company. Table I shows the basic statistics from our datasets. After filtering out stop words using lists of defined by Work4, we obtain a dictionary with *218,533* terms (English, French and other languages' terms); we focus on English and French. We have currently *1,040* O*NET occupations. We can notice that the proposed vector model is a straightforward dimensionality reduction method (from 218,533 to 1,040).

B. Results

1) Heuristic similarity-based methods: In the first series of experiments we compare Engine-3a, Engine-3b, Engine-3c and Engine-3d; Table II shows that Engine-3a slightly outperforms the others; its also suggests that our proposed similarity functions have good a performance comparable to cosine and PCC.

Table II: Comparison between Engines-3a, Engine-3b, Engine-3c and Engine-3d in terms of AUC.

Dataset	Engine-3a	Engine-3b	Engine-3c	Engine-3d
ALL	**0.67**±0.00	0.66±0.00	0.66±0.01	**0.67**±0.01
Validation	**0.80**±0.01	**0.80**±0.01	0.78±0.00	0.76±0.01
Review	**0.74**±0.01	**0.74**±0.01	0.72±0.01	0.72±0.01
Random	0.81±0.20	**0.83**±0.18	0.80±0.20	0.82±0.12
Feedback	**0.69**±0.07	0.67±0.07	0.64±0.09	0.65±0.06

We then compare Engine-3a to our two baseline methods: Engine-1 and Engine-2 (see Fig. 2). We have large confidence intervals for Feedback and Random datasets, this is due to the fact that these datasets are small and unbalanced. We can notice that Engine-3a outperforms the baseline methods on all our datasets; the out-performance is dramatic on the ALL dataset. We can conclude that the use of taxonomy-based vector model clearly and significantly improves our results. Fig. 3 shows the

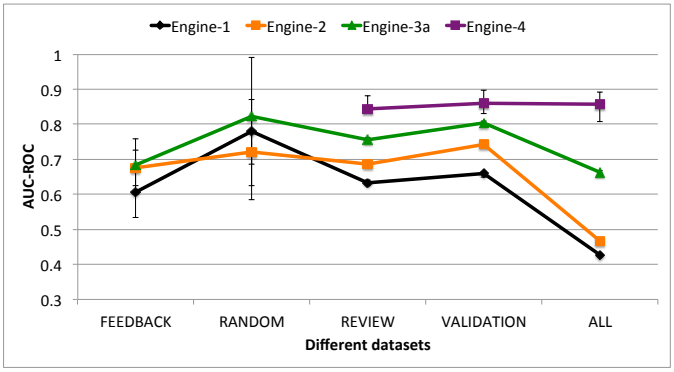

Fig. 2: Comparison between Engine-3a and Engine-4 and the two baseline methods (Engine-1 and Engine-2); for Engine-4, we use a 10-fold cross-validation on the 3 biggest datasets (Review, Validation and ALL).

behavior of Engine-3a for Facebook and LinkedIn users; we can notice that the performance of the system for Facebook users is slightly lower than for LinkedIn users. Comparing the performance of Engine-3a (see Fig. 3) to the performance of Engine-1 (see Fig. 4) and Engine-2 (see Fig. 5) for Facebook and LinkedIn users, we can note that the quality of our Facebook users' data for job recommendation seems lower than the quality of LinkedIn users data using TF-IDF model (see Fig. 4 and 5) on our datasets and that using our vector

Table I: Basic statistics from our datasets: number of entries, proportion of label 0/1, proportion of entries linked to Facebook/LinkedIn users and proportion of entries whose languages are English.

	Datasets					
	ALL	Candidate	Feedback	Random	Review	Validation
Total number of entries	203,990	30,179	246	2,898	32,441	138,226
Proportion of label 0	0.79	0.00	0.35	0.99	0.76	0.96
Proportion of label 1	0.21	1.00	0.65	0.01	0.24	0.04
Proportion of entries linked to a Facebook user	0.38	0.97	0.25	0.76	0.33	0.25
Proportion of entries linked to a LinkedIn user	0.62	0.03	0.75	0.24	0.67	0.75
Proportion of entries whose users' and jobs' languages are English	0.81	0.58	0.76	0.26	0.77	0.88

model, we improve the performance for both social networks and we reduce the difference of quality of data between the two social networks in the task of job recommendation on our datasets.

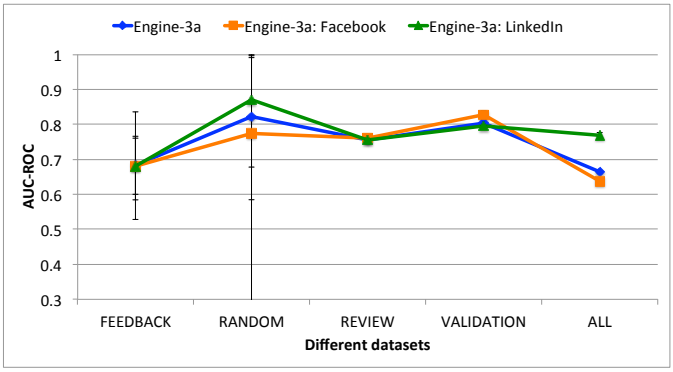

Fig. 3: Detailed results of Engine-3a for Facebook and LinkedIn users.

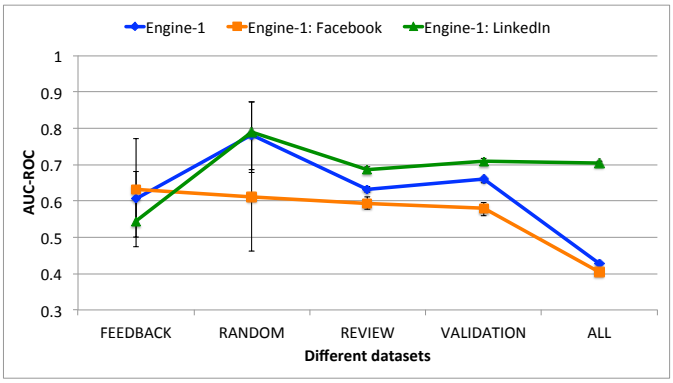

Fig. 4: Detailed results of Engine-1 for Facebook and LinkedIn users.

2) Machine learning-based methods: To learn our SVM linear model (Engine-4), we set the costs of different labels as in [24], then we use the 10-fold cross-validation method (on the 3 biggest datasets since the two smallest ones have not enough entries to obtain significant results using machine learning) to assess the performance of Engine-4 and to compare it to the other methods; we split our datasets into training sets and test sets making sure to have the same distribution of job

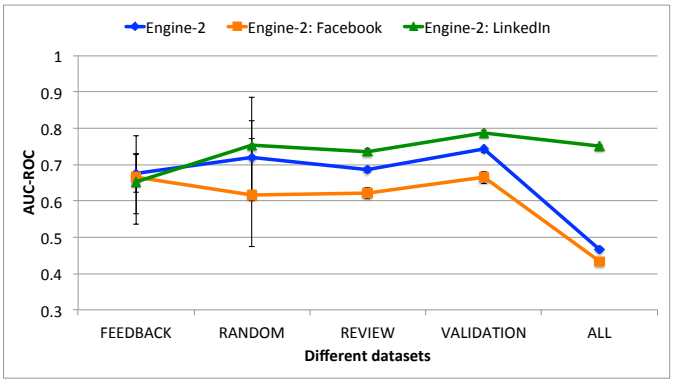

Fig. 5: Detailed results of Engine-2 for Facebook and LinkedIn users.

pages and label 0/1 in the two sets. This procedure of splitting datasets into training and test sets could bias the results since jobs from the same page are similar but this is important for Work4's applications. Fig. 2 shows that Engine-4 outperforms the heuristic-based systems on our 3 biggest datasets; this shows that we are able to learn efficient models from our data to recommend jobs to social network users. Not surprisingly we obtain better results for CTR than for CF (see Table III); we can also note from this table that our Engine-4 outperforms CTR on our two biggest datasets. The main weakness of the CTR is the fact that it only uses the data of relevant jobs for users (one-class Collaborative Topic Regression) while our Engine-4 use both the data of relevant and non-relevant jobs for users.

Table III: Comparison between Engines-3a, Engine-4, CF and CTR in terms of AUC.

Dataset	Engine-3a	Engine-4	CF	CTR
ALL	0.67±0.00	**0.85**±0.02	0.67±0.03	0.79±0.02
Validation	0.80±0.01	**0.85**±0.05	0.73±0.03	0.79±0.03
Review	0.74±0.01	0.84±0.02	0.84±0.02	**0.88**±0.01

V. Discussion and Future Work

We propose a taxonomy-based vector model and two similarity functions suited to our vector model. The first series of experiments conclude that the cosine similarity yields results slightly better than the proposed fuzzy logic-based similarity functions using the proposed vector model (see

Table II) while the second series of experiments show that the use of our taxonomy-based vector model improves the performance of our job recommender systems compared to the TF-IDF and dramatically reduces the difference of quality between our Facebook and LinkedIn data in the task of job recommendation (see Fig. 2, 3, 4 and 5). Our SVM-based recommender systems (Engine-4) outperforms two methods of the literature (CTR and CF) and the proposed heuristic-based recommender systems (see Table III). We recently applied this taxonomy-based vector model to model jobs posted on social networks by our clients in order to predict their audience. In our future work, we will be extending this vector model to other languages different from English and improving it by incorporating metadata about social network users and jobs.

VI. Acknowledgments

This work is supported by Work4, ANRT[3] (the French National Research and Technology Association) and the project Open Food System. Some models proposed in this paper use the O*NET[4] database which is being developed under the sponsorship of the US Department of Labor/Employment and Training Administration (USDOL/ETA). The authors thank Guillaume Leseur, software architect at Work4, Benjamin Combourieu and all the Work4 *Engines* team for providing the code we used to evaluate proposed systems.

References

[1] D. M. Boyd and N. B. Ellison, "Social network sites: Definition, history, and scholarship," *Journal of Computer-Mediated Communication*, vol. 13, no. 1, pp. 210–230, 2008.

[2] B. Xiao and I. Benbasat, "E-commerce product recommendation agents: use, characteristics, and impact," in *MIS Quarterly*, vol. 31, no. 1. Society for Information Management and The Management Information Systems Research Center Minneapolis, MN, USA, March 2007, pp. 137–209.

[3] Facebook. (2014, April). [Online]. Available: http://newsroom.fb.com/company-info/

[4] LinkedIn. (2014, April). [Online]. Available: http://press.linkedin.com/about

[5] M. Diaby, E. Viennet, and T. Launay, "Toward the next generation of recruitment tools: An online social network-based job recommender system," in *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining ASONAM 2013*, 2013, pp. 821–828.

[6] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. on Knowl. and Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[7] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, *Recommender Systems: An Introduction*. Cambridge University Press, 2011.

[8] G. Salton, A. Wang, and C. Yang, "A vector space model for information retrieval," *Journal of the American Society for Information Science*, vol. 18, no. 11, pp. 613–620, Nov. 1975.

[9] L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, and M. A. Rueda-Morales, "Combining content-based and collaborative recommendations: A hybrid approach based on bayesian networks." *Int. J. Approx. Reasoning*, vol. 51, no. 7, pp. 785–799, 2010.

[10] J. Han, "Data mining techniques," *SIGMOD Rec.*, vol. 25, no. 2, pp. 545–, Jun. 1996.

[11] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-Based Systems*, vol. 46, pp. 109–132, 2013.

[12] P. Lops, M. de Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends." in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer, 2011, pp. 73–105.

[13] P. Kazienko, K. Musiał, and T. Kajdanowicz, "Multidimensional Social Network in the Social Recommender System," vol. 41, no. 4, pp. 746–759, Jul. 2011.

[14] C. Biemann, "Ontology learning from text: A survey of methods," *LDV-Forum*, vol. 20, no. 2, pp. 75–93, 2005.

[15] J. Sowa, Nov. 2010. [Online]. Available: http://www.jfsowa.com/ontology/

[16] S. Aseervatham, "Apprentissage à base de noyaux sémantiques pour le traitement de données textuelles," Ph.D. dissertation, Université Paris 13 – Institut Galililée, Villetaneuse, France, 12 2007.

[17] Z. Omary and F. Mtenzi, "Machine learning approach to identifying the dataset threshold for the performance estimators in supervised learning," *International Journal for Infonomics (IJI)*, vol. 3, Sept. 2010.

[18] L. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338 – 353, 1965.

[19] J. L. Castro, "Fuzzy logic controllers are universal approximators." *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 4, pp. 629–635, 1995.

[20] C. Cortes and V. Vapnik, "Support-vector networks," in *Machine Learning*, 1995, pp. 273–297.

[21] T. Joachims, "Text categorization with suport vector machines: Learning with many relevant features," in *Proceedings of the 10th European Conference on Machine Learning*, ser. ECML '98. London, UK, UK: Springer-Verlag, 1998, pp. 137–142.

[22] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27:1–27:27, May 2011.

[23] C. Wang and D. M. Blei, "Collaborative topic modeling for recommending scientific articles," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '11. New York, NY, USA: ACM, 2011, pp. 448–456.

[24] A. Anand, G. Pugalenthi, G. B. Fogel, and P. N. Suganthan, "An approach for classification of highly imbalanced data using weighting and undersampling." *Amino Acids*, vol. 39, no. 5, pp. 1385–91, 2010.

[3] ANRT: Association Nationale de la Recherche et de la Technologie.

[4] http://www.onetcenter.org/taxonomy.html.