A Graph-Based Approach to Skill Extraction from Text*

Ilkka Kivimäki[1], **Alexander Panchenko**[4,2], **Adrien Dessy**[1,2], **Dries Verdegem**[3], **Pascal Francq**[1], **Cédrick Fairon**[2], **Hugues Bersini**[3] **and Marco Saerens**[1]
[1]ICTEAM, [2]CENTAL, Université catholique de Louvain, Belgium
[3]IRIDIA, Université libre de Bruxelles, Belgium
[4] Digital Society Laboratory LLC, Russia

Abstract

This paper presents a system that performs skill extraction from text documents. It outputs a list of professional skills that are relevant to a given input text. We argue that the system can be practical for hiring and management of personnel in an organization. We make use of the texts and the hyperlink graph of Wikipedia, as well as a list of professional skills obtained from the LinkedIn social network. The system is based on first computing similarities between an input document and the texts of Wikipedia pages and then using a biased, hub-avoiding version of the Spreading Activation algorithm on the Wikipedia graph in order to associate the input document with skills.

1 Introduction

One of the most difficult tasks of an employer can be the recruitment of a new employee out of a long list of applicants. Another challenge of the employer is to keep track of the skills and know-how of their employees in order to direct the right people to work on things they know. In the scientific community, editors of journals and committees of conferences always face the task of assigning suitable reviewers for a tall pile of submitted papers. The tasks described above are example problems of *expertise retrieval* (Balog et al., 2012). It is a subfield of information retrieval that focuses on inferring associations between people, expertise and information content, such as text documents.

In this paper, we propose a method that makes a step towards a solution of these problems. We describe an approach for the extraction of professional skills associated with a text or its author. The goal of our system is to automatically extract a set of skills from an input text, such as a set of articles written by a person. Such technology can be potentially useful in various contexts, such as the ones mentioned above, along with expertise management in a company, analysis of professional blogs, automatic meta-data extraction, etc.

For succeeding in our goal, we exploit Wikipedia, a list of skills obtained from the LinkedIn social network and the mapping between them. Our method consists of two phases. First, we analyze a query document with a vector space model or a topic model in order to associate it with Wikipedia articles. Then, using these initial pages, we use the Spreading Activation algorithm on the hyperlink graph of Wikipedia in order to find articles that correspond to LinkedIn skills and are related or central to the initial pages.

One difficulty with this approach is that it often results in some skills, which can be identified as hubs of the Wikipedia graph, constantly being retrieved, regardless of what the input is. In order to avoid this pitfall, we bias the activation to avoid spreading to general, or popular nodes. We try different measures of node popularity to redirect the spreading and perform evaluative experiments which show that this biasing in fact improves retrieval results.

We have built a web service that enables anyone to test our skill extraction system. The name of the system is `Elisit`, an abbreviation from "Expertise Localization from Informal Sources and Information

*Part of this work has been funded by projects with the "Région wallonne". We thank this institution for giving us the opportunity to conduct both fundamental and applied research. In addition, we thank Laurent Genard and Stéphane Dessy for their contributions for the work.

Technologies" and conveying the idea of trying to elicit, i.e. draw forth latent information about expertise in a target text. According to the best of our knowledge, we are the first to propose such a system and describe openly the method behind it.

2 Related work

The recent review of Balog et al. (2012) gives a thorough presentation of the problems of expertise retrieval and of the methodology used for solving them. They classify these problems in subcategories of *expert retrieval* and *expert profiling*. The former means the task of providing a name of a person who is an expert in a field that is presented as a query, while the latter means assigning expertise to a person, or some other entity based on information that is available of that entity. Recent expertise retrieval research has focused on the TREC enterprise track, which uses the TREC W3C and CERC corpora (Balog et al., 2008). These datasets contain annotated crawls of websites. The task in the TREC enterprise challenge is to build a model that performs expert retrieval and document retrieval based on a set of query topics, which correspond to expertise areas.

Our approach is quite different from the one used in the TREC challenge, as we focus on a fixed list of skills gathered from the LinkedIn website. Thus, we were not able to directly compare our system to the systems participating in the TREC enterprise track. Our problem shares some resemblance with the INEX entity-ranking track (Demartini et al., 2010), where the goal was to rank Wikipedia pages related to queries about a given topic. Our skill retrieval task can also be seen as an entity ranking task, where the entities are Wikipedia pages that correspond to skills.

LinkedIn has developed methods for defining skills and for finding relations between them (Skomoroch et al., 2012). These techniques are used in their service, for example, for recommending job opportunities to the users. The key difference of our technology is that it allows a user to search for skills by submitting an arbitrary text, instead of only searching for skills related to a certain skill. Although expertise retrieval has been an active research topic for some time, there have not been many methods for explicitly assigning particular skills to text content or people producing text content.

Our method consists of two steps. First, we apply a text similarity method to detect the relevant Wikipedia pages. Second, we enrich the results with graph mining techniques using the hyperlink graph of Wikipedia. We have not found a similar combination being applied for skill extraction before, although both parts have been well studied in similar contexts before. For instance, Steyvers et al. (2004) proposed the Author-Topic Model, a graphical model based on LDA (Blei et al., 2003), that associates authors of texts with topics detected from those texts.

Wikipedia has been already used in NLP research both as a corpus and as a semantic network. Its hyperlink graph is a collaboratively constructed network, as opposed to manually crafted networks such as WordNet (Miller, 1995). Gabrilovich and Markovitch (2007) introduced Explicit Semantic Analysis (ESA), where the words of a document are represented as mixtures of concepts, i.e. Wikipedia pages, according to their occurence in the body texts of the pages. The experimental results show that this strategy works very well and outranks, for example, LSA (Landauer and Dumais, 1997) in the task of measuring document similarity. ESA was later extended by taking into account the graph structure provided by the links in Wikipedia (Yeh et al., 2009). The authors of this work used a PageRank-based algorithm on the graph for measuring word and document similarity. This approach was coined *WikiWalk*.

Associating the elements of a text document under analysis with Wikipedia pages involves itself already many problems often encountered in NLP. The process where certain words and multiword expressions are associated with a certain Wikipedia page has been called *Wikification* (Mihalcea and Csomai, 2007). In our work, we take a more general approach, and try to associate the full input text to a set of Wikipedia pages according to different vector space models. The models and the details of this strategy are explained in section 3.3.

The `Elisit` system uses the Spreading Activation algorithm on the Wikipedia graph to establish associations between texts and skills. We chose to use Spreading Activation, as it tries to simulate a cognitive associative memory (Anderson, 1983), and the Wikipedia hyperlink network can be understood as an associative network. The simulation works by finding associations in a network of concepts by spreading pulses of activation from concepts into their neighbours. In the context of NLP, the Spreading Activation algorithm has been traditionally used for word sense disambiguation (Hirst, 1988) and information retrieval (Crestani, 1997). Gouws et al. (2010) have shown that this algorithm, applied to the Wikipedia graph, can also be used to measure conceptual and document similarity.

3 Methodology

In this section, we will explain how the `Elisit` skill extraction system works. We will first explain how the system uses data from Wikipedia and LinkedIn. Then, we will describe the two main components of the system, the `text2wiki` module, which associates a query document with related Wikipedia pages, and the `wiki2skill` module, which aims to associate the Wikipedia pages found by the `text2wiki` module with Wikipedia pages that correspond to skills.

3.1 Wikipedia texts and links

Each page in Wikipedia contains a text that may include hyperlinks to other pages. We make the assumption that there is a meaningful semantic relationship between the pages that are linked with each other and that the Wikipedia hyperlink graph can be exploited as an associative network. The properties of the hyperlink structure of Wikipedia and the nature of the information contained in the links have been investigated by Koolen (2011).

In addition to the encyclopedia pages, Wikipedia also contains, among others, category, discussion and help pages. In our system, we are only interested in the encyclopedia pages and the hyperlinks between them. We are using data downloaded[1] on May 2nd 2012. This dump encompasses 3,983,338 pages with 247,560,469 links, after removal of the redirect pages. The Wikipedia graph consists of a giant Strongly Connected Component (SCC) of 3,744,419 nodes, 4130 SCC's of sizes from 61 to 2 nodes and 228,881 nodes that form their own SCC's.

[1]`http://dumps.wikimedia.org/`

3.2 LinkedIn skills

We gathered a list of skills from the LinkedIn social network[2]. The list includes skills which the users can assign to their profiles. This enables the site to recommend new contacts or open job opportunities to each user. The skills in the list have been generated by an automated process developed by LinkedIn (Skomoroch et al., 2012). The process decides, whether a word or a phrase or a skill suggested by a user is actually a skill through an analysis of the text contained in the user profile pages.

Each LinkedIn skill has its own webpage that contains information about the skill. One piece of information contained in most of these pages is a link to a Wikipedia article. According to Skomoroch et al. (2012), LinkedIn automatically builds this mapping. However, some links are manually verified through crowdsourcing. Not all skill pages contain a link to Wikipedia, but these skills are often either very specific or ambiguous. Thus, we decided to remove these skills from our final list. The list of skills used in the system was extracted from the LinkedIn site in September 2012. After removal of the skills without a link to Wikipedia, the list contained 27,153 skills.

3.3 `text2wiki` module

The goal of the `text2wiki` module is to retrieve Wikipedia articles that are relevant to an input text.

The output of the module is a vector of similarities between the input document and all articles of the English Wikipedia that contain at least 300 characters. There are approximately 3.3 million such pages. We only retrieve the 200 Wikipedia pages that are most similar to the input document. Thus, each input text is represented as a sparse vector $\mathbf{a}(0)$, which has 200 non-zero elements out of 3,983,338 dimensions corresponding to the full list of Wikipedia pages. Each non-zero value $a_i(0)$ of this vector is a semantic similarity of the query with the i-th Wikipedia article. This approach stems from ESA, mentioned above. The vector $\mathbf{a}(0)$ is given as input to the second module `wiki2skill`.

The `text2wiki` module relies on the `Gensim` library (Řehůřek and Sojka, 2010)[3]. In particular,

[2]`http://www.linkedin.com/skills`
[3]`http://radimrehurek.com/gensim`

we have used four different text similarity functions, based respectively on the classical Vector Space Models (VSM's) (Berry et al., 1994), LSA and LDA:

(a) TF-IDF (300,000 dimensions)
(b) LogEntropy (300,000 dimensions)
(c) LogEntropy + LSA (200 dimensions)
(d) LogEntropy + LDA (200 topics)

First, each text is represented as a vector $\mathbf{x}$ in a space of the 300,000 most frequent terms in the corpus, each appearing at least in 10% of the documents (excluding stopwords). We limited the number of dimensions to 300,000 to reduce computational complexity. The models (a) and (b) directly use this representation, while for (c) and (d) this initial representation is transformed to a vector $\mathbf{x}'$ in a reduced space of 200 dimensions/topics. For LSA and LDA, the number of dimensions is often empirically selected from the range $[100 - 500]$ (Foltz, 1996; Bast and Majumdar, 2005). We followed this practice. From the vector representations ($\mathbf{x}$ or $\mathbf{x}'$), the similarity between the input document and each Wikipedia article is computed using the cosine similarity.

Pairwise comparison of a vector of 300,000 dimensions against 3.3 million vectors of the same size has a prohibitive computational cost. To make our application practical, we use an inverted index of `Gensim` to efficiently retrieve articles semantically related to an input document.

3.4 **`wiki2skill`** module

The `wiki2skill` module performs the Spreading Activation algorithm using the initial activations provided by the `text2wiki` module and returns a vector of final activations of all the nodes of the network and a vector containing the activations of only the nodes corresponding to skills.

The basic idea of Spreading Activation is to initially activate a set of nodes in a network and then iteratively spread the activation into the neighbouring nodes. This can actually be interpreted in many ways opening up a wide space of algorithms that can lead to different results. One attempt for an exact definition of the Spreading Activation algorithm can be found in the work of Shrager et al. (1987). Their formulation states that if $\mathbf{a}(0)$ is a vector containing the initial activations of each node of the network, then after each iteration, or time step, or pulse t, the vector of activations is

$$\mathbf{a}(t) = \gamma\mathbf{a}(t-1) + \lambda\mathbf{W}^{\mathsf{T}}\mathbf{a}(t-1) + \mathbf{c}(t), \quad (1)$$

where $\gamma \in [0, 1]$ is a *decay factor* which controls the conservation of activation during time, $\lambda \in [0, 1]$ is a *friction factor*, which controls the amount of activation that nodes can spread to their neighbors, $\mathbf{c}(t)$ is an *activation source vector* and $\mathbf{W}$ is a weighted adjacency matrix, where the weights control the amount of activation that flows through each link in the network. In some cases, iterating eq. (1) leads to a converged activation state, but often, especially when dealing with large networks, it is more practical to set the number of pulses, T, to some fixed, low number.

As already stated, this formulation of Spreading Activation spans a wide space of different algorithms. In particular, this space contains many random walk based algorithms. By considering the case where $\gamma = 0, \lambda = 1, \mathbf{c}(t) = \mathbf{0}$ and where the matrix $\mathbf{W}$ is row-stochastic, the Spreading Activation model boils down to a random walk model with a transition probability matrix $\mathbf{W}$, where $\mathbf{a}(t)$ contains the proportion of random walkers at each node when the initial proportions are given by $\mathbf{a}(0)$. When the situation is changed by choosing $\mathbf{c}(t) = (1 - \lambda)\mathbf{a}(0)$, we obtain a bounded Random Walk with Restart model (Pan et al., 2004; Mantrach et al., 2011).

Early experiments with the first versions of the algorithm revealed an activation bias towards nodes that correspond to very general Wikipedia pages (e.g. the page "ISBN", which is often linked to in the References section of Wikipedia pages). These nodes have a high input degree, but are often not relevant for the given query. This problem is often encountered when analysing large graphs with random walk based measures. It is known that they can be dominated by the stationary distribution of the corresponding Markov Chain (Brand, 2005).

To tackle this problem, we assign link weights according to *preferential transition probabilities*, which define biased random walks that try to avoid hub nodes. They have been studied e.g. in the context of stochastic routing of packages in scale-free networks (Fronczak and Fronczak, 2009). These weights are given by

$$w^*_{ij} = \frac{\pi_j^\alpha}{\sum\limits_{k:(i,k)\in E} \pi_k^\alpha}, \tag{2}$$

where π_j is a *popularity index* and α is a *biasing parameter*, which controls the amount of activation that flows from node i to node j based on the popularity of node j. For the popularity index, we considered three options. First, we tried simply the input degree of a node. As a second option, we used the PageRank score of the node (Page et al., 1999) which corresponds to the node's weight in the stationary distribution of a *random surfer* that surfs Wikipedia by clicking on hyperlinks randomly. As a third popularity index, we used a score based on the HITS algorithm (Kleinberg, 1999), which is similar to PageRank, but instead assigns two scores, an authority score and a hub score. In short, a page has a high authority score, if it is linked to by many hub pages, and vice versa. In the case of HITS, the popularity index was defined as the product of the authority and hub scores of the node. When $\alpha = 0$, w_{ij} is equal for all links leaving from node i, but when $\alpha < 0$, activation will flow more to less popular nodes and less to popular nodes. We included the selection of a suitable value for α as a parameter to be tuned along with the rest of the spreading strategy in quantitative experiments that are presented in section 5.2. These experiments show that biasing the activation to avoid spreading to popular nodes indeed improves retrieval results.

We also decided to investigate whether giving more weight to links that exist in both directions would improve results. The Wikipedia hyperlink graph is directed, but in some cases two pages may contain a link to each other. We thus adjust the link weights w_{ij} so that $w_{ij} = \delta w^*_{ij}$ if $(j, i) \in E$ and $w_{ij} = w^*_{ij}$ otherwise, where $\delta \geq 1$ is a *bidirectional link weight*. With large values of δ, more activation will flow through bidirectional links than links that exist only in one direction. After this weighting, the final link weight matrix $\mathbf{W}$ is obtained by normalizing each element with its corresponding row sum to make the matrix row-stochastic. This makes the model easier to interpret by considering random walks. However, in a traditional Spreading Activation model the matrix $\mathbf{W}$ is not required to be row-stochastic. We plan to investigate in the future, how much the normalization affects the results.

The large size of the Wikipedia graph challenges the use of Spreading Activation. In order to provide a usable web service, we would need the system to provide results fast, preferably within fractions of seconds. So far, we have dealt with this issue within the `wiki2skill` module by respresenting the link weight matrix $\mathbf{W}$ of the whole Wikipedia graph using the sparse matrix library `SciPy`[4]. Each iteration of the Spreading Activation is then achieved by simple matrix arithmetic according to eq. (1). As a result, the matrix $\mathbf{W}$ must be precomputed from the adjacency matrix for a given value of the biasing parameter α and the bidirectional link weight δ when the system is launched. Thus, they cannot be selected separately for each query from the system. Currently, the system can perform one iteration of spreading activation within less than one second, depending on the sparsity of the activation vector. Our experiments indicate that the results are quite stable after five spreading iterations, meaning that we normally get results with the `wiki2skill` module in about one to three seconds.

4 The `Elisit` skill extraction system

The `Elisit` system integrates the `text2wiki` and the `wiki2skill` modules. We have built a web application[5] which lets everyone try our method and use it from third-party applications. Due to this web service, the `Elisit` technology can be easily integrated into systems performing skill search, email or document analysis, HR automatization, analysis of professional blogs, automatic meta-data extraction, etc. The web interface presents the user the result of the skill extraction (a list of skills) as well as the result of the `text2wiki` module (a list of Wikipedia pages). Each retrieved skill also contains a link to the corresponding Wikipedia page.

Figure 1 presents an example of results provided by the `Elisit` system. It lists skills extracted from the abstract of the chapter *Support vector machines and machine learning on documents* from

[4] `http://www.scipy.org/`

[5] GUI: `http://elisit.cental.be/`; RESTful web service: `http://elisit.cental.be:8080/`.

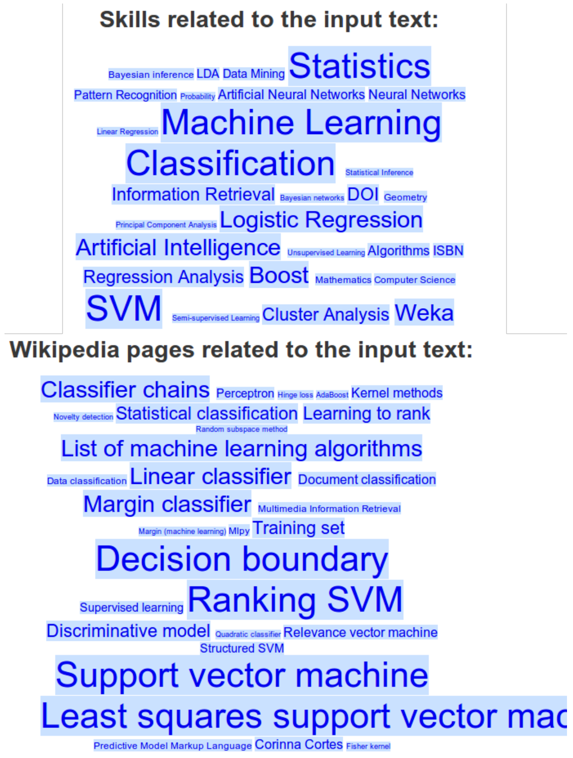
Skills related to the input text:
Bayesian inference LDA Data Mining Statistics
Pattern Recognition Probability Artificial Neural Networks Neural Networks
Linear Regression Machine Learning
Classification Statistical Inference
Information Retrieval Bayesian networks DOI Geometry
Principal Component Analysis Logistic Regression
Artificial Intelligence Unsupervised Learning Algorithms ISBN
Regression Analysis Boost Mathematics Computer Science
SVM Semi-supervised Learning Cluster Analysis Weka
Wikipedia pages related to the input text:
Classifier chains Perceptron Hinge loss AdaBoost Kernel methods
Novelty detection Statistical classification Learning to rank
Random subspace method
List of machine learning algorithms
Data classification Linear classifier Document classification
Margin classifier Multimedia Information Retrieval
Margin (machine learning) Mlpy Training set
Decision boundary
Supervised learning Ranking SVM
Discriminative model Quadratic classifier Relevance vector machine
Structured SVM
Support vector machine
Least squares support vector mac
Predictive Model Markup Language Corinna Cortes Fisher kernel

Figure 1: Skills extracted from a text about text document categorization.

Introduction to Information Retrieval by Manning et al. (2008). As one can observe, the Wikipedia pages found by the `text2wiki` module represent many low-level topics, such as "Desicion boundary", "Ranking SVM" or "Least square SVM". On the other hand, the skills retrieved after using the `wiki2skill` module provide high-level topics relevant to the input text, such as "SVM", "Machine Learning" or "Classification". These general topics are more useful, since a user, such as an HR manager, may be confused by too low-level skills.

5 Experiments & results

5.1 Evaluation of the `text2wiki` module

In order to compare the four text similarity functions, we collected $p = 200,000$ pairs of semantically related documents from the "See also" sections of Wikipedia articles. A good model is supposed to assign a high similarity to these pairs. However, since the distribution of similarity scores depends on the model, one cannot simply compare the mean similarity $\bar{s}$ over the set of pairs. Thus, we used a z-score as evaluation metric. The z-scores are computed as

$$z = \frac{\bar{s} - \hat{\mu}}{\sqrt{\hat{\sigma}^2/p}} \qquad (3)$$

where $\hat{\mu}$ and $\hat{\sigma}$ are sample estimates of mean and standard deviation of similarity scores for a given model. These sample estimates have been calculated from a set of 1,000,000 randomly selected pairs of articles. Table 1 presents the results of this experiment. It appears that more complex models (LSA, LDA) are outperformed on this task by the simpler vector space models (TF-IDF, LogEntropy). This can be just a special case with this experimental setting and perhaps another choice of the number of topics could give better results. Thus, further meta-parameter optimization of LSA and LDA is one approach for improving the performance of the `text2wiki` module.

Model	z**-score**
TF-IDF	8459
LogEntropy	4370
LogEntropy + LDA	2317
LogEntropy + LSA	2143

Table 1: Comparison of different text similarity functions on the Wikipedia "See also" dataset.

5.2 Evaluation of the `wiki2skill` module

In order to find the optimal strategy of applying Spreading Activation, we designed an evaluation protocol relying on *related skills* listed on each LinkedIn skill page. These are automatically selected by computing similarities between skills from user profiles (Skomoroch et al., 2012). Each skill page contains at most 20 related skills.

For the evaluation procedure, we choose an initial node i, corresponding to a LinkedIn skill, and activate it by setting $\mathbf{a}(0) = \mathbf{e}_i$, that is a vector containing 1 in its i-th element and zeros elsewhere. Then, we compute $\mathbf{a}(T)$ with some spreading strategy and for some number of steps T, filter out the skill nodes and rank them according to their final activations. To measure how well the related skills are represented in this ranked list of skills, we use Precision at 1, 5 and 10, and R-Precision to evaluate the accuracy of the first ranked results and Recall at 100 to see how well the algorithm manages to activate all of the related skills.

There are many LinkedIn skills that are not well represented in the Wikipedia graph, because of ambiguity issues, for instance. To prevent these anomalies from causing misguiding results, we selected a fixed set of 16 representative skills for the evaluation. These skills were “Statistics”, “Hidden Markov Models”, “Telecommunications”, “MeeGo”, “Digital Printing”, “OCR”, “Linguistics”, “Speech Synthesis”, “Classical”, “Impressionist”, “Education”, “Secondary Education”, “Cinematography”, “Executive producer”, “Social Sciences”, “Political Sociology”.

Developing a completely automatic optimisation scheme for this model selection task would be difficult because of the number of different parameters, the size of the Wikipedia graph and the heuristic nature of the whole methodology. Thus, we decided to rely on a manual evaluation of the results.

Exploring the whole space of algorithms spanned by eq. (1) would be too demanding as well. That is why we have so far tested only a few models. In the preliminary experiments that we conducted with the system, we observed that using a friction factor λ smaller than one had little effect on the results, and thus we decided to always use $\lambda = 1$. Otherwise, we experimented with three models, which we will simply refer to as models 1, 2 and 3 and which we define as follows

- model 1: $\gamma = 0$ and $\mathbf{c}(t) = \mathbf{0}$;
- model 2: $\gamma = 1$ and $\mathbf{c}(t) = \mathbf{0}$;
- model 3: $\gamma = 0$ and $\mathbf{c}(t) = \mathbf{a}(0)$.

In model 1, activation is not conserved in a node but only depends on the activation it has received from its neighbors after each pulse. In contrast, the activation that a node receives is completely conserved in model 2. Model 3 corresponds to the Random Walk with Restart model, where the initial activation is fed to the system at each pulse. Models 1 and 2 eventually converge to a stationary distribution that is independent of the initial activation vector. This can be beneficial in situations where some of the initially activated nodes are noisy, or irrelevant, because it allows the initial activation to die out, or at least become lower than the activation of other, possibly more relevant nodes. With Model 3, the initially activated nodes remain always among the most activated nodes, which is not necessarily a robust choice.

The outcomes of the experiments demonstrated that model 2 and model 3 perform equally well. Indeed, these models are very similar, and apparently their small differences do not affect the results much. However, model 1 provided constantly worse results than the two other models. Thus, we decided to use model 3, corresponding to the Random Walk with Restart model, in the system and in selecting the rest of the spreading strategy.

We also evaluated different settings for the link weighting scheme. Here, we faced a startling result, namely that increasing the bidirectional link weight δ all the way up to the value $\delta = 15$ kept improving the results according to almost all evaluation measures. This would indicate that links that exist in only one direction do not convey a lot of semantic relatedness. However, we assume that this is a phenomenon caused by the nature of the experiment and the small subset of skills used in it, and not necessarily a general phenomenon for the whole Wikipedia graph. In our experiments, the improvement was more drastic in the range $\delta \in [1, 5]$ after which a damping effect can be observed. For this reason, we decided to set the bidirectional link weight in the `Elisit` system to $\delta = 5$.

We observed a similar phenomenon for the number of pulses T. Increasing its value up to $T = 8$ improved constantly the results. However, again, there was no substantial change in the results in the range $T \in [5, 8]$. In the web service, the number of pulses of the spreading activation can be determined by the user.

In addition to the parameters discussed above, the link weighting involves the popularity index π_j and the biasing parameter α. An overview of the effect of these two choices can be seen in Table 2, which presents the results with the different evaluation measures. These results were obtained by setting parameters as described earlier in this section. First, we can see from this table that using negative values for α in the weighting improves results compared to the natural random walk, i.e. the case $\alpha = 0$. This indicates that our strategy of biasing the spreading of activation to avoid popular nodes indeed improves the results. We can also see

	Pre@1			**Pre@5**			**Pre@10**			**R-Pre**			**Rec@100**		
α	d_{in}	PR	HITS	d_{in}	PR	HITS	d_{in}	PR	HITS	d_{in}	PR	HITS	d_{in}	PR	HITS
0	0	0	0	0.119	0.119	0.119	0.156	0.156	0.156	0.154	0.154	0.154	0.439	0.439	0.439
-0.2	0	0	0	0.206	0.238	0.206	**0.222**	0.216	0.213	0.172	0.193	0.185	0.469	0.469	0.494
-0.4	0	0	0	0.225	**0.263**	0.169	0.203	0.200	0.150	0.185	**0.204**	0.148	0.503	0.498	0.476
-0.6	0	0	**0.063**	0.238	0.225	0.119	0.200	0.197	0.141	0.186	0.193	0.119	0.511	0.517	0.418
-0.8	0	0	0	0.213	0.181	0.075	0.191	0.197	0.113	0.171	0.185	0.109	0.515	**0.524**	0.384
-1	0	0	0	0.169	0.156	0.063	0.178	0.197	0.091	0.154	0.172	0.097	0.493	0.518	0.336

Table 2: The effect of the biasing parameter α and the choice of popularity index on the results in the evaluation of the `wiki2skill` module.

that using Pagerank as the popularity index provided overall better results than using the input degree, which again yielded better results than using HITS. Thus, biasing according to the input connections of nodes seems more preferable than biasing according to co-citation or co-reference connections. The low scores with Precision@1 are understandable, because of the low number of positives (at most 20 related skills) in comparison to the total number of skills (over 27,000). In the `Elisit` system, we use the Pagerank score as the popularity index and set the value of the biasing parameter to $\alpha = -0.4$.

5.3 Evaluation of the whole `Elisit` system

We adapted the evaluation procedure used for the `wiki2skill` module, described in the previous section, in order to test the whole `Elisit` system. This time, instead of activating the node of a given skill, we activated the nodes found by the `text2wiki` module when fed with the Wikipedia article corresponding to the skill. We run the Spreading Activation algorithm with the setup presented in the previous section. To make the evaluation more realistic, the initial activation of the target skill node is set to zero (instead of 1, i.e. the cosine of a vector with itself).

The system allows its user to set the number of initially activated nodes. We investigated the effect of this choice by measuring Precision and Recall according to the related skills, and by looking at the average rank of the target skill on the list of final activations. However, there was no clear trend in the results when testing with 1-200 initially activated nodes. Nevertheless, we have noticed that using more than 20 initially activated nodes rarely improves the results. We must also emphasize that the choice of the number of initially activated nodes depends on the query, especially its length.

We also wanted to compare the different VSM's of the `text2wiki` module when using the whole `Elisit` system. We did this by comparing Precision and Recall at different ranks w.r.t. the related skills of the target skill found on LinkedIn. Thus, this experiment combines the experiments introduced in sections 5.1, where the evaluation was based on the "See also" pages, and 5.2, where we used a set of 16 target skills and their related skills. Table 3 reports the Precision and Recall values obtained with the different VSM's. These values result from an average over 12 different numbers of initially activated nodes. They confirm the conclusion drawn from the experiment in section 5.1, namely that the LogEntropy and TF-IDF models outperform LSA and LDA models for this task.

VSM	**Pre@1**	**Pre@5**	**Pre@10**	**R-Pre**	**Rec@100**
TF-IDF	0.042	**0.231**	**0.214**	0.190	0.516
LogEntropy	**0.068**	0.216	0.212	**0.193**	**0.525**
LogEnt + LSA	0.042	0.180	0.181	0.163	0.491
LogEnt + LDA	0.089	0.193	0.174	0.159	0.470

Table 3: Comparison of the different models of the `text2wiki` module in the performance of the whole `Elisit` system.

6 Conclusion and future work

We have presented a method for skill extraction based on Wikipedia articles, their hyperlink graph, and a set of skills built by LinkedIn. We have also presented the `Elisit` system as a reference implementation of this method. This kind of a system has many potential applications, such as knowledge management in a company or recommender systems of websites. We have demonstrated with examples and with quantitative evaluations that the system indeed extracts relevant skills from text. The evaluation experiments have also allowed us to compare and finetune different strategies and parameters of the system. For example, we have shown that using a bias to avoid the spreading of activation to popular nodes of the graph improves retrieval results.

This work is still in progress, and we have many goals for improvement. One plan is to compute link weights based on the contents of linked pages using their vector space representation in the `text2wiki` module. The method and system proposed in the paper could also be extended to other languages. Finally, our methodology can potentially be used to different problems than skill extraction by substituting the LinkedIn skills with a list of Wikipedia pages from another domain.

References

John R Anderson. 1983. A spreading activation theory of memory. *Journal Of Verbal Learning And Verbal Behavior*, 22(3):261–295.

Krisztian Balog, Paul Thomas, Nick Craswell, Ian Soboroff, Peter Bailey, and Arjen P De Vries. 2008. Overview of the trec 2008 enterprise track. Technical report, DTIC Document.

Krisztian Balog, Yi Fang, Maarten de Rijke, Pavel Serdyukov, and Luo Si. 2012. Expertise retrieval. *Foundations and Trends in Information Retrieval*, 6(2-3):127–256.

Holger Bast and Debapriyo Majumdar. 2005. Why spectral retrieval works. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 11–18. ACM.

Michael W. Berry, Susan T. Dumais, and Gavin W. O'Brien. 1994. Using linear algebra for intelligent information retrieval. Technical Report UT-CS-94-270.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, March.

Matthew Brand. 2005. A random walks perspective on maximizing satisfaction and profit. *Proceedings of the 2005 SIAM International Conference on Data Mining*.

Fabio Crestani. 1997. Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review*, 11(6):453–482.

Gianluca Demartini, Tereza Iofciu, and Arjen P De Vries. 2010. Overview of the inex 2009 entity ranking track. In *Focused Retrieval and Evaluation*, pages 254–264. Springer.

Peter W Foltz. 1996. Latent semantic analysis for text-based research. *Behavior Research Methods, Instruments, & Computers*, 28(2):197–202.

Agata Fronczak and Piotr Fronczak. 2009. Biased random walks in complex networks: The role of local navigation rules. *Physical Review E*, 80(1):016107.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJCAI'07: Proceedings of the 20th international joint conference on Artifical intelligence*, pages 1606–1611, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Stephan Gouws, G-J van Rooyen, and Herman A. Engelbrecht. 2010. Measuring conceptual similarity by spreading activation over wikipedia's hyperlink structure. In *Proceedings of the 2nd Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources*, pages 46–54, Beijing, China, August. Coling 2010 Organizing Committee.

Graeme Hirst. 1988. Semantic interpretation and ambiguity. *Artificial Intelligence*, 34(2):131–177.

Jon M. Kleinberg. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632.

Marijn Koolen. 2011. *The Meaning Of Structure: the Value of Link Evidence for Information Retrieval*. Ph.D. thesis, University of Amsterdam, The Netherlands.

Thomas K. Landauer and Susan T. Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psych. review*, 104(2):211.

Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge.

Amin Mantrach, Nicolas van Zeebroeck, Pascal Francq, Masashi Shimbo, Hugues Bersini, and Marco Saerens. 2011. Semi-supervised classification and betweenness computation on large, sparse, directed graphs. *Pattern Recognition*, 44(6):1212–1224.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In Mário J. Silva, Alberto H. F. Laender, Ricardo A. Baeza-Yates, Deborah L. McGuinness, Bjørn Olstad, Øystein Haug Olsen, and André O. Falcão, editors, *CIKM*, pages 233–242. ACM.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. *Technical Report 1999-0120, Computer Science Department, Stanford University*.

Jia-Yu Pan, Hyung-Jeong Yang, Christos Faloutsos, and Pinar Duygulu. 2004. Automatic multimedia cross-modal correlation discovery. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 653–658. ACM.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May. ELRA.

Jeff Shrager, Tad Hogg, and Bernardo A Huberman. 1987. Observation of phase transitions in spreading activation networks. *Science*, 236(4805):1092–1094.

Peter N Skomoroch, Matthew T Hayes, Abhishek Gupta, and Dhanurjay AS Patil. 2012. Skill customization system, January 24. US Patent App. 13/357,360.

Mark Steyvers, Padhraic Smyth, Michal Rosen-Zvi, and Thomas Griffiths. 2004. Probabilistic author-topic models for information discovery. In *Proceedings of the 10th ACM International Conference on Knowledge Discovery and Data Mining*. ACM Press.

Eric Yeh, Daniel Ramage, Christopher D. Manning, Eneko Agirre, and Aitor Soroa. 2009. Wikiwalk: Random walks on wikipedia for semantic relatedness. In *Graph-based Methods for Natural Language Processing*, pages 41–49. The Association for Computer Linguistics.