DISCOVERING JOB KNOWLEDGE FROM WEB DATA

Intellectual Output 3 (Report)

Job Knowledge Analysis Engine and Visualization Application Programming Interfaces (APIs)

Co-funded by the European Union

Erasmus+
Enriching lives, opening minds.

The DISKOW project has received funding from the Erasmus plus programme under grant agreement no. 2018-1-DE02-KA202-005215

Grant Number	2018-1-DE02-KA202-005215	Acronym	DISKOW
Full Title	Discovering Job Knowledge through Web Analytics towards facilitated mobility of European Professionals and Refugees Career Integration		
Topic	KA2 - Cooperation for Innovation and the Exchange of Good Practices KA202 - Strategic Partnerships for vocational education and training		
Funding scheme	ERASMUS+		
Start Date	01-08-2018		
Duration	36 months		
Project website URL	www.diskow.eu		
Project Coordinator	Gottfried Wilhelm, Leibniz Universitaet Hannover		
Report	IO3 - report		
Intellectual Output	IO3 - From HTML to structured data		
Delivery Month	36		
Version	1.0		
Delivery Date	31/08/2021		
Lead Beneficiary	LUH		

1. Executive summary

In the last few years, online information has become more readily available than ever. In particular, several websites dedicated to job seekers have become increasingly popular, where a large amount of job postings advertising positions in different countries and across multiple sectors can be found. However, the data available online is often unstructured and not standardised, making it impossible to utilise it in order to analyse the dynamics of the labour market. The information extraction part of the DISKOW project therefore aims to collect data from the web, standardise it and finally publish it on the JKB which is described in IO4. The principle behind the design of this process is for it to be as widely applicable as possible, regardless of the website chosen or the language of the job posting. We therefore developed a process that starts by scraping raw HTML data, translates the text to english if needed and extracts relevant information without using any website-specific techniques (such as specific tags). In an effort to add value to the analysis and visualisation of the

data in the JKB we adopted the ESCO standard for skills, and the NACE standard for job sectors. This report will describe in depth the techniques used to extract information, including keyword matching, Word2Vec and more. We will also outline the steps taken after the evaluation process in order to improve our performance in the categories with the lowest accuracy.

2. Crawling and Scraping

As previously mentioned, the web contains a vast amount of unstructured information that the DISKOW project aims to standardise. As such, the first phase of our end-to-end process consists of identifying potential job postings and gathering all the relevant information. This is achieved by utilising a three step process, whose input is a set of URLs, and whose output is a set of job postings in HTML format.

The first step in this process involves identifying the number of available job postings per country. This information is derived by the fact that each URL provided as input is associated with a country. As it is often the case, websites allow the user to browse job postings based on country; it should be noted that this information is used for data balancing purposes and is not utilised in the information extraction phase.

Once this information is known, the URL for each individual job posting is gathered by crawling. This process starts at a given page and gathers the URL of all potential job postings that are linked in the original page. In order to limit the number of crawled job postings, an upper bound of 10000 job postings per country was set. This also results in a reduced imbalance in our dataset, especially with respect to countries that tend to have a significant amount of published job postings, such as Germany or France.

Finally, the content of the web page corresponding to each crawled URL is scraped, extracting its HTML code. All the job postings that only contain a link to another website are filtered, resulting in only samples with a complete text being considered. All valid job postings are saved to an SQLite database that will be preprocessed and then used in the information extraction phase of this project.

3. Preprocessing

All the job postings scraped using the method outlined above are saved in the HTML format, however this format is not directly usable for information extraction purposes, resulting in the data needing further preprocessing. This not only includes stripping the HTML from the job postings and converting them into text, but also other steps which are outlined in this section.

3.1 HTML stripping and chunking

The first step of preprocessing involves extracting all the text data from each job posting; this process is called HTML stripping. The most basic form of stripping involves removing all HTML tags and saving the entire text as plain string data. This however does not exploit the additional information provided by the HTML format. In particular, the text inside an end level children HTML tag can be considered as a text chunk. Subdividing the text into chunks is useful for information extraction as the type of information carried by

Algorithm 1 Algorithm for preprocessing a set of HTML job advertisements

Input: *html_postings* - job postings in HTML format; *split* - whether to split in chunks
Output: *parsed_text* - list of preprocessed text

```
1: parsed_chunks := ∅
2: if split then
3:    all_text := ∅
4:    for posting ∈ html_postings do
5:       all_text := all_text ∪ stripHTML( splitChunks(posting))
6:    end for
7: else
8:    all_text := stripHTML(html_postings)
9: end if
10: for text ∈ all_text do
11:    text := cleanString(text)
12:    if word length of text > 1 then
13:       language := detectLanguage(text)
14:       if language is not english then
15:          text = translateToEng(text)
16:       end if
17:       parsed_text := parsed_text ∪ {text}
18:    end if
19: end for
```

each chunk can be predicted using a classifier. This allows us to apply category-specific extraction techniques at a chunk level rather than on the whole text, a concept that will be expanded later in this report.

As part of this project, both techniques of HTML stripping are used. Whenever cleaning data from the eurojobs.com website, we also extract some information from the job posting associated with some specific HTML tags. This information includes job location, employment type, job title, base salary and job sector. As the goal of the project is to use any HTML page as input we refrained from using this website-specific technique in our approach and opted to leverage this information as ground truth data for evaluation.

3.2 Cleaning Parsed Text

Once the HTML has been stripped from job postings, the text needs a further layer of preprocessing in order to be used for information extraction, called cleaning. The aim of cleaning the text is to standardise it as much as possible and rectify any irregularities that were left from the stripping process. Such irregularities include any HTML tags that were not stripped and extra white spaces. After removing irregularities all accents are removed from the text and special characters are removed. The last step of cleaning involves standardising the text. This includes expanding contractions, making all the text lowercase and converting number words to numeric form. After this initial cleaning takes place on

both the full text and chunked data, some task-specific cleaning will also take place when needed. This includes stop words removal, lemmatisation etc.

3.3 Translating Job Postings

In order to encourage variety in our dataset, our approach allows for job postings of different languages to be included. This requires all the job postings to be translated to English, which is currently achieved using a free translation service. In order to minimise the computation time we first detect the language on part of each job posting, and then translate if the job posting is not in english. If a more precise translation is needed, any paid API (such as Google translator) can be easily integrated within our code.

4. Information extraction

After the data has been appropriately pre-processed it will be comprised of standardised strings ready for information to be extracted. This section will describe in detail the methods used in order to extract relevant information from the job postings text.

4.1 Classification and Feature Extraction

After the data has been cleaned and split into HTML chunks, each individual chunk is classified in order to determine what type of information it contains. This is particularly useful when extracting educational requirements and base salary, as seen in the relevant chapters.

In order to chose the model to use for classification we performed an extensive evaluation of five separate classification algorithms: Decision Trees, Multinomial Naive Bays, Logistic Regression, Linear Support Vector Machines and an Ensemble Vote Classifier of the other four algorithms. All the classifiers were implemented using the scikit-learn machine learning library. Our evaluation highlighted that the best performing classifier was the Logistic Regression, which is the one we use in our project. More complex classifiers, such as recurrent neural networks could not be used as the amount of training data available was insufficient.

The features used as input for our classifier included 2-gram features. We have found that the number of features could be reduced by deducting those present in less than two instances. This in turn has the effect of reducing over-fitting, and resulted in the final amount of 2-gram features to be just over 10000. We have also used the Spacy parts of speech tag feature detection model to extract Parts of Speech tag features from the text chunks. After discarding the features that are present in a limited amount of training samples, we used 41 Parts of Speech features. Moreover we used 20 of Spacy's Named Entity Recognition features after discarding rarely occurring features. Finally we used the number of words in each chunk as a feature and whether the chunk conatins a numerical value.

4.2 Job Title

The job title extraction happens on the whole cleaned job posting, and starts with identifying possible titles within the text. This is achieved by matching the whole posting against

a list of more than 70000 job titles. This results in a list of job titles that are mentioned in the posting which need to be filtered in order to extrapolate one correct job title associated with the job posting.

In order to filter the list of job titles the semantic meaning of each was analysed using Word2Vec (Mikolov et al., 2013). Word2Vec is an embedding that translates a word to a vector of 300 numbers such that the cosine similarity of two vectors represents their similarity in meaning. In order to measure the similarity of different job titles to each other, the embedding vectors relative to each word were summed, and the cosine similarity of each was measured. After defining a programmatical way to measure the similarity of two job titles, the algorithm for filtering the set of job titles can be easily defined: for each job title in the set calculate the sum of its distance from all the other job titles. The job title with the highest overall distance should be the furthest in meaning from the rest of the set and can therefore be excluded. This should be repeated until the set of job titles reaches two, at which point it is impossible to use this method to select the correct one. If both job titles are the same, then the search is completed; otherwise the job title that occurs first is selected. This is to account for the fact that in a certain job posting it is more likely that the correct job title is mentioned first (such as in the title), as any other related positions are likely to be mentioned later in the job posting.

4.3 Skills

As previously mentioned, the DISKOW project not only aims to gather information from the web, but also to standardise it. This effort is evident in the skills extraction process, where for each job posting different skills are detected and then standardised according to the ESCO standard.

The first step of this process, similarly to the job title extraction, consists of keyword matching in order to detect what skills are potentially required for a specific job. The set of keyword used for this purpose are the ones provided in ESCO through the alternative labels. For example the“financial management” skill has 5 different alternative labels, which include “monetary administration” and “management of financial tasks”. If any of these labels are found in the text, the skill “financial management” will be added to the list of potential relevant skills.

Matching against the whole set of skills also has its drawbacks: in fact there are words such as “energy” which are often found in job postings unrelated to the skill itself. This is why filtering is necessary in order to provide meaningful information. The filtering of the skills used in this project is based on the job title and the ESCO database. In fact, for any given job, the ESCO database contains a list of essential and optional skills. This knowledge is leveraged by filtering out any skills that are not associated with a job that has at least one word in common (excluding stopwords) with the detected job title. In order for this technique to be more reliable both the main and alternative names of jobs in the ESCO database are considered.

4.4 Job Sector

One of the more challenging aspects of the information extraction process resides in the job sector extraction. Our standardisation efforts are also extended to this category, where

the NACE standard was chosen. This was found to be more informative than the ISCO standard used by the ESCO database, which is more representative of the level of a certain position rather than the sector itself, at least at the first level. As an example, all managers, regardless of the sector, are grouped together in the ISCO classification, whereas the NACE standard includes some well defined sectors where all types of profession can be placed. The intended method to be used to estimate the job sector is to determine which sector is more likely to require the skills matched to each job posting. This approach is possible thanks to the association between jobs and skills provided by the ESCO database. This is also the reason why choosing NACE presented an initial challenge, as the ESCO database used the ISCO structure to categorise occupations. This challenge was overcome with the help of our partners by creating a mapping between the two systems, where the ISCO tree was divided into subtrees each associated to a specific NACE sector. This allowed us to build a tree with the NACE sectors as nodes, and all the occupations in the ESCO database as their children.

This newly built tree allowed us to calculate the percentage of jobs in each sector where a given skill is required or optional. This information was then used to determine which sector is most likely to require a given set of skills. When making this estimate one factor needs to be considered: some skills provide us with more information than others. The skill "cardiology" for example can only be found in jobs in the "human health and social work activities" sector, whereas the skill "manage staff" has a much wider spread across all sectors. In order to account for this, for any given skill (or ability) a and sector s we first calculated the percentage of jobs in s where the skill is relevant:

$$r_s^a = \frac{\sum_{j \in s} p(a|j)}{|s|} \tag{1}$$

where j is a job and $p(a|j)$ is the probability of skill a being relevant to job j (either 0 - not relevant or 1 - relevant). This was then turned into a probability distribution over job sectors by normalising r^a using the following equation:

$$P_s^a = \frac{r_s^a}{\sum_{s' \in S} r_{s'}^a} \tag{2}$$

where S is the set containing all NACE sectors. The purpose of creating probability distribution P_s^a is ultimately to calculate its entropy in order to ascertain how much information can be derived from the presence of each skill. From the entropy we finally derive a metric we call "information" by normalising the opposite of the entropy between 0 and 1.

$$I_a = 1 - \frac{H(P^a)}{max(H(P^{a'}) \; \forall a' \in A)} \tag{3}$$

where $H(x)$ is the entropy of probability distribution x. In the example mentioned above, "cardiology" would carry an information of 1, since it is present in only one job sector. On the other hand "manage staff" would carry an information close to zero, as it is more evenly spread between sectors. A skill with an information of 0 would need to be evenly distributed between sectors, with $P_s^a = P_{s'}^a \; \forall s, s' \in S$. Knowing how much information is carried by a certain skill is then used in the sector estimation, where given a set D containing the skills detected in a certain job posting, the following scoring system is applied for each sector s:

$$score_s = \sum_{a \in D} I_a \cdot r_s^a \tag{4}$$

The sector that receives the maximum score according to the scoring function is finally assigned to the specific job posting.

4.5 Salary and Currency Information Extraction

As mentioned previously, the salary is extracted from previously labelled text chunks; this allows us to ignore numbers present in the job posting which are not related to salary information. The process for extracting salary from text chunks can be summed up in the steps below:

- Then text chunk is checked for any number above the value 450. If one is found, the text chunk goes to the next step.
- A matching between the chunk and a salary-word dictionary is performed. This dictionary includes words or phrases which are usually present in a sentence regarding salary. The text chunks that are successfully matched go to the next step.
- Using Spacy's Named Entity Relation model, the NER tags are detected for each text chunk. If the text chunk contains any one of the 'MONEY', 'CARDINAL', or 'DATE' NER tags, then we target the text of these tag as our next processing inputs.
- A checking is performed to verify the text's currency amount received from the previous step.
- If more than one number is detected as potential salary, then the number associated with a currency is selected. The currency is found using a keyword matching approach on a list of currencies. If there is no currency information in the chunk and there are more than two numbers, the two closest numbers are selected. If on the other hand only two numbers are present both numbers are saved as salary.
- The base salary is considered to be the lowest of the two numbers matched in the previous step; very large numbers are deducted and monthly salaries are converted to yearly salaries.

4.6 Job location Information Extraction

In order to match the Job location, we prepared a list of European countries, as this project is concerned with the job market within the EU. This list was used to select the country names from a text chunk using a rule-based process. We have also used the Spacy Named Entity Feature detection model within our rule-based process. This process follows the steps outlined below:

- Using Spacy Named Entity Relation model, the NER tags are detected for the text chunk. If the text chunk contains a "GPE" tag, we target the tagged text with our next processing inputs. The "GPE" NER tag represents countries, cities or states.

- Then the location information is used to get the country name of that specific location. For this purpose, we have used the python geography library.
- If multiple countries are matched for a given job posting, we only keep the European country names. Finally, if multiple countries remain after the previous filtering, the country that is mentioned the most is chosen.

4.7 Employment Type Extraction

In order to extract the employment type of a given job posting we prepared a list of keywords related to this category. This list comprises the words that can usually be found in a sentence that describes employment type. We have also prepared a list of keywords that are related to the type of employment itself, e.g. full-time, part-time etc. The detected employment type is divided into 5 main categories: full-time, part-time, internship, freelancing and zero-hour. The matching process follows the steps outlined below:

- Keyword matching is used to match a text chunk with the corresponding employment-type. If a match is found, the matched value from the dictionary is stored.
- If no match is available, on the other hand, the employment type is assumed to be full-time.

4.8 Educational Requirements Extraction

A dictionary of 508 unique degree names for bachelor, masters and PhD were prepared using a dataset collected from Kaggle[1]. This dictionary maps a degree name to its corresponding group. 4 main groups were identified: diploma, bachelor, masters and PhD. The matching for this category can be described by the steps below:

- The dictionary of degree names is converted into a 2-gram vector; the chunk from which we need to collect information is also turned into a 2-gram vector. The two vectors are then compared in order to find any n-grams in common. The common n-grams are saved as initially extracted information.
- This primary information is mapped in terms of degree-name-group. (e.g. MBA $\rightarrow$ masters, BSC $\rightarrow$ bachelor)
- It is very common that in one job advertisement, multiple degree names can exist. If this happens in the job posting we are examining, the lowest educational requirement is selected.

5. Data publishing

Once the relevant information has been extracted from the job postings, the next process involves pushing it to the remote SQL database for it to be displayed. The first step in this process involves creating a backup of any existing data in order to be able to always revert the changes if needed. After the backup has been successfully executed, our process pushes

1. https://www.kaggle.com/anasfullstack/mastersportal-programs

the extracted data to a table in the database. This table is not used for display purposes, but only to store the data itself. In order to transfer the data from this secondary table to the display table we built a service that runs 24/7 on the Pascal cluster, that checks whether each job posting is still available and transfers all the available job postings to the display table. Each job posting has a field that represents the date and time of when its validity was last checked. If a job posting has not been checked before (has just been added) or the last time it was checked was more than 12 hours ago, the service checks its validity and updates the display table accordingly.

6. Improvements after first Evaluation

After a manual evaluation on 100 data samples which is described in IO5, we assessed that out of the categories we predict, Job Sector and Educational Requirements are the weakest.

Whenever we are able to predict educational requirements it is estimated that their accuracy is 57%, however only 30% of the total educational requirements were correctly identified over the testing sample. This implies that the main issue to solve with educational requirements is to increase the amount of samples that have a prediction, therefore reducing the null values. As our method for extracting educational requirements is based on a keyword matching approach, we added keywords for primary and secondary education in our database. We also changed the predicted labels for educational requirements based on feedback from the stakeholders, suggesting that our target audience would benefit more from adding primary and secondary education requirements, rather than distinguishing between different types of higher education. We therefore grouped bachelor, masters and PhD in the higher education label. Moreover we also implemented an algorithm that estimates educational requirements whenever a keyword was not found. This algorithm first estimates the first digit of the ISCO code of the job posting, to then suggest a level of education needed for the position based on this digit. This information is available as part of the ISCO standard. Our estimation algorithm resulted in our accuracy to drop by 3% whenever we are able to predict an educational requirement, however, as the number of predictions was increased, our overall accuracy was improved from 30% to 44%.

Regarding the job sector, the issue is more complex, especially because it is estimated based on the skills, as previously discussed. In order to evaluate our approach on a larger data sample, we utilised the data mentioned at the end of Section 3.1. We developed an evaluation method that checks the accuracy of our estimated job sectors based on the ground truth data, hand labelled by the poster. We then developed a new algorithm to filter skills based on the DBSCAN clustering algorithm (Birant and Kut, 2007). For each skill a, in fact, we calculated its relevance for each NACE and ISCO sector s using Equation 1, and derived a feature vector $\vec{R_a} = (r^a_{s1}, ..., r^a_{s29})$. As this vector is in a 29 dimensional space, the concept of our new filtering system is that noise would be scattered along the space, whereas groups of skills that are relevant are going to form clusters. We used the DBSCAN clustering algorithm to cluster skills together and returned only the skills assigned to a cluster as valid, assuming that noisy samples would be isolated. After this first filtering method we also use the old skills filtering method in order to further filter skills based on the job title. We also modified our job sector evaluation to account a required skill more than a preferred skill. When re-evaluating on our manual testing sample the results highlighted

no major difference in the overall accuracy of the new method, however there was a 3% increase in the accuracy whenever a prediction was made. Our automatic evaluation over more than 15000 samples on the other hand highlights how the old method had an accuracy of 40% whenever a prediction was made, whereas the new method results in an accuracy of 47%.

7. Conclusion and Outlooks

In this report we outlined all of the techniques used to extract relevant information from raw unstructured data. We also extensively utilised the ESCO database in order to standardise the skills needed for each job, and used this standardisation in order to filter irrelevant skills and predict the job sector for each posting. Furthermore we used the NACE standard for job sectors in order for our visualisation and data analysis to be more informative. After an initial evaluation we addressed the weaknesses of our approach, in particular concerning the extraction of educational requirements and the estimation of the job sector.

Future work on this project could focus on finding a suitable standard for job titles, as we found the ESCO data base to be lacking in this respect. Moreover if more labelled data was available, Machine Learning models could be trained in order to accurately predict the job sector based on a given set of skills.

References

Derya Birant and Alp Kut. St-dbscan: An algorithm for clustering spatial–temporal data. *Data & knowledge engineering*, 60(1):208–221, 2007.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.