Technical report. Semantic reasoning of skills

Classification: 2 Internal Document

Table 1. Document classes

Class	**Description**
1 public	This is a public document. It is also available for third parties, such as customers and partners. Document can be passed freely without restrictions.
2 internal	An internal document that can be shared with selected partners. The information may not be published.
3 private	An internal document that is available within the organization. This document is not shared with third parties.
4 confidential	An internal document that is only passed on to specific recipients. Passing on is strictly prohibited.

1. Data

A dataset constructed from 10782 opened jobs on 18.03.2021 is used. It contains the following meta-data:

- title - title of an offer
- description - description of an offer as raw html. Sometimes contains general text about a company, sometimes is empty
- identifier - a list of dictionaries with a job id (used in the backend and in an offer's URL) and apiId (when a job is imported from an ATS)
- datePosted - date and time when an offer is posted, time is mostly 0
- responsibilities - first part of the description as raw html
- experienceRequirements - second part of the description as raw html
- hiringOrganization - a dictionary containing information about a company, such as name, jobShop url and contact information
- jobLocation - location of an offer, usually as a city name
- url - url of an offer
- itemListElement - possible actions that can be applied to an offer (such as 'save' or 'apply')
- incentiveCompensation - a list of texts, that describe sometimes financial information about an offer, sometimes general company information (e.g. work-life balance). Almost half are null
- employmentType - type of an offer (full-time, part-time), 70% are null
- disambiguatingDescription - general text about a company, 70% are null

For further processing only 'title' and 'description' fields are selected, which are cleaned (html characters are removed) and merged. The baseline method should be trained on the documents in one language, therefore, spacy language detector is used ("de_core_news_sm") in order to label the language of each job description. 91% of texts are in German, therefore, it is chosen as the language of the corpus. The following processing steps are applied:

1. Replace umlauts and special characters
2. Replace punctuation with spaces
3. Remove digits
4. Remove german stopwords, according to the German nltk stopwords corpus
5. Lemmatize and tokenize words based on the German spacy model (same training data as for thr language detector)

Further in the text 'job descriptions' is used to describe these tokenized texts.

---

2. Baseline model

In order to evaluate the performance of our skill-encoding method on the task of similar jobs a baseline model needs to be introduced. This model should be a state-of-the-art document comparison model trained on a large corpus of German texts or on the cleaned corpus of job descriptions directly. The following 2 approaches have been implemented so far:

- Word2Vec word embeddings and cosine similarity matrix
- Topic modelling (LDA) and hellinger distance matrix

Each method is designed once, but is implemented for subsets of data separately, due to the fact that there is no large pool of job offers. Each company has an individual career platform, therefore, the goal of the task is to provide similar jobs from that company only.

2.1. Word2vec word embeddings

Since training a Word2Vec requires a large dataset, the German language model in open access trained on the German Wikipedia and Shuffled German news is used. Every token is mapped to a 300-dimensional word embedding and multiplied by Smooth Inverse Frequency (SIF) parameter. SIF is a weighting scheme designed to improve document embeddings. It uses word frequency information to identify significant ones and weight them more. Weights of a word w are computed by a/(a + p(w)) where a is a parameter and p(w) is the word frequency of w. Token frequency is obtained from the full corpus of job descriptions, a is set to 0.001.

Each document is represented as an average of word embeddings contained in it. Finally, a cosine similarity matrix is calculated.

2.2. Topic modelling (LDA)

LDA is an example of topic modelling, aimed at discovering topics (set of words) in a collection of documents, and then automatically classifying any individual document within the collection in terms of how "relevant" it is to each of the discovered topics. In contrast with word2vec, there is no need to obtain a pre-trained LDA model, since inferred topics should be representative for the collected job descriptions.

Grid search is implemented to choose the following parameters for the gensim LDA model:

- K - Numbers of requested latent topics to be extracted from the training corpus
- dir_priors - Default prior selecting strategies for topics and words, can be ’symmetric’,’asymmetric’ or ’auto’
- random_states - List of seeds to generate random state
- num_passes - Number of passes through the corpus during training
- iterations - Maximum number of iterations through the corpus when inferring the topic distribution of a corpus

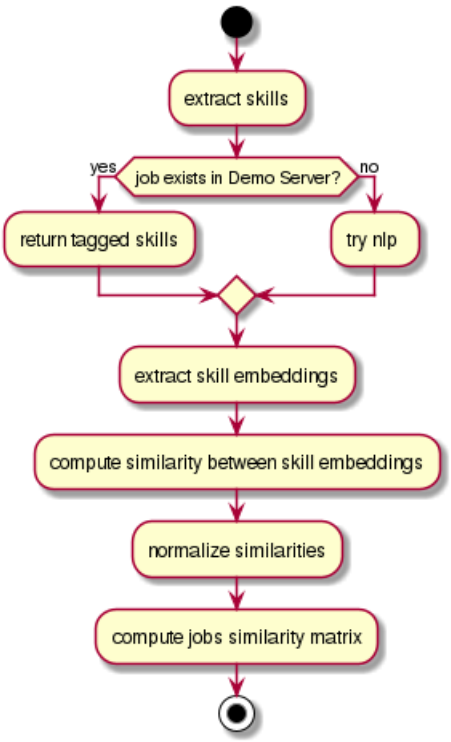
To choose the best parameters coherence scores are calculated for each LDA model. Typically, coherence scores are used to evaluate topic models, since these scores rate topics regarding to their understandability. Topic coherence measures the degree of semantic similarity between high scoring words in the topic. In this work 'c_v' is used, which combines the boolean sliding window with the indirect cosine measure and the normalized pointwise mutual information (NPMI).

Once the model with the best parameters is chosen, the topic propability distributions of each job description are inferred. Finally, to retrieve a similarity matrix pairwise Hellinger distance is computed for all job descriptions in a subset. The Hellinger distance metric gives an output in the range [0,1] for two probability distributions, with values closer to 0 meaning they are more similar.

---

3. Mixed Pipeline (current approach)

The core idea of this approach is to utilize the information about skills as the essense of a job. Mixed pipeline is a method that infers the skills from textual descriptions, transforms them and computes their similarity matrix. Step-by-step description of this approach is represented in a graph below:



1. Extraction of skills is implemented by querying the Demo Server. In case a job has not been pushed to it, the skills are directly detected using an NLP algorithm.
2. Tagged skills are converted to the existing word2vec embeddings.
3. Skill similarity is a simple scalar product of all vector combinations. A result is a matrix containing all skills (around 14000) and their pairwise similarity scores.
4. Similarities are normalized, according to the following scheme:

```
'threshold': 0.8, 'score': 1.0,
'threshold': 0.75, 'score': 0.75,
'threshold': 0.65, 'score': 0.5,
'threshold': 0.6, 'score': 0.25
```

This way we disregard the dissimilar skills and significantly reduce the number of stored similarities. 5. Each job is represented by a bag of equally weighted skills that have been detected in it. Each of those skills is compared to the list of

skills for another job and the highest similarity is taken. Finally, an average of those scores is calculated and added to the matrix as a similarity measure between 2 jobs. A more fine-tuned approach would assign different weights for the different directions of matching and have a different function for averaging the skills similarities (e.g. hard-skills may be more important soft-skills).

---

4. Ground truth dataset

There has been 3 approaches implemented for constructing a Ground Truth dataset for 3 different companies - Telekom, Vodafone and Fressnapf. Each of those has a moderate variety of offers (in third case of jobs or groups of offers) that enabled creating a fairly diverse dataset. Each subsection below contains a description of methods themselves and a link to the dataset in Excel (also attached with the current file).

4.1. Telekom (manual approach)

Telekom contains 382 offers and 30 unique jobs (unique job descriptions) which makes it feasible to manually annotate a dataset. Moreover, there is not enough visitors on the career website to unitilize traffic data (more about it in the subsection about collaborative filtering).

For each given job (here named query) a set of related jobs are selected and ranked. Full descriprions are taken in consideration when comparing the texts. Type of an offer (e.g. Duales Studium or Ausbildung), skills and department are the main subjects of manual data labelling.

The minimum number of jobs for a query is 1 and the maximum is 10.Link to the dataset.

4.2. Vodafone (semi-automatic approach)

Vodafone contains 522 unique job descriptions, which makes manual annotation too tedious and time-consuming. Therefore, a filtering algorighm has been introduced to limit the pool of jobs from which to select the similar ones for a given query. First of all, the dataset has been reduced by the location and only the jobs in Düsseldorf/Köln/Bonn are selected, as this is the most popular in terms of offers region. Then, 26 jobs have been randomly selected and processed, such that there are no repetitions of the same of very closely-related titles.

For each of those sampled query jobs a set of keywords from a title is selected, which contain the key information about a job. For instance, for an offer 'Discover Trainee (m/w/d) Big Data / Data Science' these would be 'big data' and 'data science'. Afterwards a filtering algorithm is applied that returns jobs from the dataset that contain these keywords in the title.

Finally, an annotator ranks these filtered samples on the scale from 1 - a job is not similar at all to the given query - to 10 - a job is the same as the query. Is it implemented using a Google Form to normalize the rankings afterwards and calculate intra-annotator agreement coefficient.

Currently the filtered samples have been ranked by one annotator and the results can be foundin this file or in the attached Excel files.

4.3. Fressnapf (collaborative filtering)

Collaborative filtering is a method applied in recommender systems by utilizing the data about users' actions and choices to produce automatic predictions. We apply this technique to create a ground truth dataset of similar jobs without any manual annotation. The core idea in the context of the task is the following: if a user saves or applies to more than one offer, these offers are assumed to be similar. This rule is applied only in cases where more than one user has chosen the same jobs in order to eliminate random choices or users that are switching the careers. Collaborative filtering has the main advantage of ommiting the formal definition of jobs similarity as it is based on the users' opinion - in the end the goal of providing a job

recommendation should be targeted at the users of this system.

However, the only limitation to this method is the sufficient amount of data to provide a variety of jobs confirmed by multiple users. Therefore, Fressnapf career portal was selected as it has more than 10000 visitors monthly and 118 unique jobs.

One significant decision of this approach was grouping the same-titled offers into **jobs** and performing collaborative filtering on the job groups instead of unique offers. This way there will not be duplicates of the same titles in the dataset.

Data about users' actions was collected using the Google Analytics API for the time range 01.03.2021 - 31.05.2021. The offers on which events occured have been selected if they appear in the dataset with jobs descriptions scraped on 18.03.2021. There has been 1832 unique users that performed actions on the 105 unique **jobs**. The resulting dataset contains 40 queries that have 1 to 12 jobs matched to them. The dataset is available in this file or in the attached files.

---

5. Results

The evaluation metrics consist of ROC-AUC score as well as mean average precision (MAP) and recall. They are computed based on the similarity threshold, which is a distance cut-off for the predicted result. There is a minimum and maximum limit - 2 and 10 for the returned results, even if retrieved distance does not fall within theshold (e.g. if the threshold is 1 there will be 2 jobs returned for each query with the maximum similarity score). The table below shows the highest ROC-AUC scores and their respective similarity thresholds in brackets for each method and ground truth dataset described in the previous sections:

Table 2. ROC-AUC scores

Dataset	**Word2Vec**	**LDA**	**Mixed pipeline**
Telekom	0.51 (0.48)	0.73 (0.61)	0.75 (0.61)
Vodafone	0.68 (0.97)	0.73 (0.81)	0.68 (0.75)
Fressnapf	0.86 (0.94)	0.49 (0.98)	0.64 (0.8)

The most consistent results are on the Vodafone dataset. There is no method that performs better on all 3 datasets (or even on 2), in every case the highest score is for a different approach.

Another part of the evaluation included MAP and recall metrics and their comparison. The following table includes the highest scores and their respective similarity thresholds in brackets for each method and dataset:

Table 3. Mean average precision and recall

Dataset	**Word2Vec**		**LDA**		**Mixed pipeline**	
	MAP	**Recall**	**MAP**	**Recall**	**MAP**	**Recall**
Telekom	0.36 (1.0)	0.68 (0.0)	0.32 (0.5)	0.56 (0.5)	0.52 (0.9)	0.61 (0.0)
Vodafone	0.12 (1.0)	0.11 (0.0)	0.08 (0.0)	0.13 (0.0)	0.15 (0.7)	0.16 (0.0)
Fressnapf	0.19 (0.8)	0.19 (0.0)	0.12 (0.0)	0.10 (0.0)	0.22 (0.8)	0.37 (0.0)

Among the datasets Telekom has shown the best-performing results in all 3 methods. If compared across the methods, mixed pipeline outperforms baseline models on all datasets.

Last updated 2021-06-24 18:33:05 +0200