Skills2Job: A recommender system that encodes job offer embeddings on graph databases

Anna Giabelli a, c, Lorenzo Malandri a, c, *, Fabio Mercorio a, c, Mario Mezzanzanica a, c, Andrea Seveso b, c

a Department of Statistics and Quantitative Methods, University of Milano Bicocca, Milan, Italy
b Department of Informatics, Systems and Communication, University of Milano Bicocca, Milan, Italy
c CRISP Research Centre, University of Milano Bicocca, Milan, Italy

A R T I C L E   I N F O
Article history:
Received 6 August 2020
Received in revised form 10 December 2020
Accepted 15 December 2020
Available online 31 December 2020

Keywords:
Recommender systems
Graph databases
Labor Market Intelligence
Word embeddings

A B S T R A C T
We propose a recommender system that, starting from a set of users’ skills, identifies the most suitable jobs as they emerge from a large dataset of Online Job Vacancies (OJVs). To this aim, we process 2.5M+ OJVs posted in three different countries (United Kingdom, France, and Germany), training several embeddings and performing an intrinsic evaluation of their quality. Besides, we compute a measure of skill importance for each occupation in each country, the Revealed Comparative Advantage (rca). The best vector model, one for each country, together with the rca, is used to feed a graph database, which will serve as the keystone for the recommender system. Results are evaluated through a user study of 10 labor market experts, using P@3 and nDCG as scores. Results show a high precision for the recommendations provided by skills2job, and the high values of nDCG (0.985 and 0.984 in a [0,1] range) indicate a strong correlation between the experts’ scores and the rankings generated by skills2job.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

Over the past several years, the rapid growth of web services has been making available a massive amount of structured and semi-structured data in different domains. One example is the web labor market with a huge number of Online Job Vacancies (OJVs),1 which are available through web portals and online applications. According to the LinkedIn Pressroom,2 only on LinkedIn, there are 706M+ Users and 20M+ Job Vacancies, and a total of about 36 K skills listed. In this scenario, the problem of processing and extracting insights from OJVs is gaining the researchers’ interest, as it allows modeling and understanding complex labor market phenomena (see, e.g. [1–4]). Given the high number of job positions and applicants, the problem of the person-job fit [5,6] has become increasingly important in recent literature, both as a skill measuring system [7] and as a job recommendation system [8,9]. However, previous literature is not suitable for our tasks, since recommender systems in the LMI domain rely strongly on handcrafed features and expert knowledge that make existing methods costly, difficult to update, and error-prone. Other contributions in this domain use learning-based approaches, but none of them neither uses them in conjunction with co-occurrence statistics nor uses graph databases to perform recommendation queries (see Section 2 for the related literature).

In this article, we propose skills2job, a recommendation system that, given a set of skills provided by the user, recommends the most suitable occupations based on the information automatically extracted from a large corpus of OJVs and organized as a graph database. To this aim, we employ both co-occurrence statistics, using a count-based measure of skill-relevance named Revealed Comparative Advantage (rca), and distributional semantics, through word embeddings. To evaluate the capability of the different embeddings to represent the labor market domain, we select the one for each country that better captures the similarity between occupations and skills based on the European Skills, Competences, Qualifications, and Occupations Taxonomy (ESCO). To do this, we perform an intrinsic evaluation, proposing a measure of semantic similarity in taxonomies named Hierarchical Semantic Similarity (HSS) as a gold standard. Then we store the information extracted through word embeddings and rca in a graph database that is used as a keystone for three recommendation tasks. The main task aims to recommend three occupations...
in a target labor market based on the user's skills. The generated recommendations are validated by an expert study. The approach of skil1s2job is knowledge poor and data-driven, hence it can be adapted to different countries/industries and easily updated over time.

**Motivation.** This work is inspired by the research activities that are part of a European Tender granted by the EU Cedefop agency, an initiative that aims at realizing a system able to collect and classify Online Job Vacancies (OJVs) for the whole EU, including 28 EU country members and all of the 32 languages of the Union through machine learning.

**Contribution.** The contribution of skil1s2job is three-fold:

(i) We exploit web labor market data using distributional semantics (embeddings), a knowledge-based representation (ESCO), and a count-based measure of skill relevance (rca);

(ii) We organize the above-mentioned resources as a graph database for performing graph-traversal queries;

(iii) We present skil1s2job, a recommendation system that exploits the resources developed in (i) and (ii) to suggest the most suitable occupations starting from the user's skills in a certain context.

The workflow followed by our system is represented in Fig. 1. The remainder of the paper is organized as follows: Section 2 presents the related work, and in Section 3 and 4 we explain the methodologies followed. The system is explained in Section 5, while the user evaluation is presented and analyzed in Section 6 and in Section 7. Finally, Section 8 concludes the paper and describes future work.

2. Related work

This section aims to give some background notions about the main topics of our work, namely Word Embeddings, Recommender Systems, Graph Databases, and Labor Market Intelligence (LMI), discussing some related work in the field of LMI and the use of recommender systems in real-life applications.

2.1. Word embeddings

To extract linguistic patterns from OJVs, we resorted to distributional semantics, a branch of linguistics relying on the assumption that words that occur in the same contexts tend to have a similar meaning. Words are represented by semantic vectors, which are usually derived from a large corpus using co-occurrence statistics or neural network training, and their use improves learning algorithms in many NLP tasks. It was empirically shown that word vectors preserve linguistic regularities, plus they are semi-supervised and knowledge-poor, thus suitable for large corpora and evolving scenarios.

To this end we employed FastText, which is an extension of word2vec for scalable word representation and classification. One of its major contributions is considering sub-word information by representing each word as the sum of its n-gram vectors. Formally, given a word and a dictionary of size , the vector representation of the n-gram is , where is the vector representing the context. This simple representation allows one to share information between words and makes it convenient to represent rare words, typos, and words with the same root.

2.2. Recommender systems

Recommender systems aim to provide a user with tailored recommendations about items that may fit his interests. These systems model the user's previous interests or behavior, creating a personalized profile that is then matched with other possibly interesting items.

There are various methodologies for creating a good match. Collaborative-based content filtering finds personal recommendations based on other users' interests who historically had similar interests with the target. Content-based systems aim to model the items and to find a way of comparing them to users' profiles. Hybrid methods combine the previous approaches. Recommender systems may be improved using different techniques, such as sentiment analysis, cognitive models, or word embeddings.

In [22], the authors investigated the effectiveness of Word Embedding techniques in content-based recommender systems and claimed that their performance is comparable to well-established recommendation algorithms. Word embedding techniques can be employed in many different scenarios, using textual as well as non-textual data. In [23] the authors used the locations of check-ins to propose new venues to visit, while in [24] they leveraged the user's purchase history derived from e-mail receipts for delivering new product's suggestions to customers. Finally, in [25] the authors used users' click-through and purchase history data in an e-commerce scenario.

Other works proposed employing Word2Vec embeddings as part of their recommendation approach in the context of academic papers' research, using easily accessible and unlabeled textual data.

2.3. Recommendations for labor market intelligence

Labor Market Intelligence (LMI) is a term that is emerging in the whole EU community. There is a growing interest in designing and implementing real LMI applications to web labor market data for supporting policy design and evaluation activities through evidence-based decision-making (see, e.g., [27,28]).

Many works have attempted to recommend a job to the applicants starting from their skills. Early research on this topic can be attributed to [8], who used handcrafted features: skills, demographic and educational data, job experience, etc. to recommend a good match between persons and jobs.

The recommendation process may be formulated as a supervised machine learning problem: [29] explored this hypothesis using a large dataset of job transitions, while in [9] the authors proposed an extension of General Linear Mixed Models that effectively enables them to be used on large datasets thanks to parallel computing.

In [7] the authors modeled the popularity of jobs' skills using an approach based on the analysis of co-occurrence statistics in a large corpus of job postings.

Deep Neural Networks (DNN) were also used for enhancing Person-Job fit: [6] proposed a Convolutional Neural Network for matching a talent's qualification to the requirements of a job by measuring the distances between corresponding latent representations, while [5] leveraged a Recurrent Neural Network to project words into latent representations, and exploited the hierarchical structure of the skills using attention mechanisms.
Most of the previous work is directed towards proposing to a candidate a specific job vacancy to apply to, in contrast to our system skills2job, which suggests job titles. Recent works use DNNs for creating latent representations, while we explore the idea of using word embeddings for the same purpose.

### 2.4. Graph databases

In recent years the NoSQL movement gained in importance, proposing new data model paradigms that differ significantly from the classical relational model (e.g., key–values, document databases, column-oriented, and graph-databases, see [30,31]). All these new paradigms share a flexible schema that can evolve to always fit the data, the ability to horizontally scale, and the native support for sharing (see, e.g.[32,33]). However, most NoSQL databases store sets of disconnected documents and values (aka aggregate models), as in the case of key–value and document databases. Differently, graph databases (see [34]) present three distinct characteristics useful for our purposes: (i) simple representation of relationships, (ii) advantages in performances thanks to optimized graph algorithms, and (iii) flexibility due to their schema-free nature.

In [35] the authors propose a system that organizes labor market information as a graph; similarly to our research, they analyze OJVs and query the resulting knowledge as a graph. However, their work differs from ours since they do not consider distributional semantics in the construction of the graph and they do not exploit the graph to produce recommendations.

### 3. Word embeddings selection

In this paragraph, we focus on the selection of word embeddings based on their intrinsic quality. We train several embedding models using different algorithms and hyperparameters, and we perform a selection method which aims at choosing the one that better represents the similarity relations in the taxonomy.

The system was created as a research activity within the Cedefop project (see [10]), which has collected OJVs from all 28 EU countries. We focus on OJVs collected from April 2018 to December 2018 in the United Kingdom, Germany, and France for ICT related occupations, that span over 16 ESCO categories (see Tables 1 and 2). We processed about 1+ million OJVs for the United Kingdom, 1+ million for Germany, and almost 500,000 for France.

#### 3.1. Embeddings selection

To select the best word embedding, we perform an intrinsic evaluation. According to [36], we maximize the correlation between a measure of similarity between two terms used as the gold standard and the cosine similarity between their corresponding word vectors. In [36] the authors use expert user similarity values as a gold standard. In this section, we develop a measure, namely Hierarchical Semantic Similarity (HSS), for measuring semantic similarity in a taxonomy based on the similarity values that are intrinsic to the hierarchy itself. Since we want to encode semantic information from a semantic hierarchy built by human experts into our vector model, we adopt those values as a proxy of human judgments. Therefore, similarly to [37] we compute \( \hat{p}(c) \) using an intrinsic measure, exploiting the structure of the taxonomy instead of an external corpus. However, differently from [37], in which the authors use only the number of taxonomic concepts, we consider also the entities of the taxonomy:

\[
\hat{p}(c) = \frac{N_c}{N} \tag{2}
\]

where \( N \) is the cardinality, i.e. the number of entities (words), of the taxonomy, and \( N_c \) the sum of the cardinality of the concept \( c \) with the cardinality of all its hyponyms. Note that \( \hat{p}(c) \) is monotonic and increases with granularity, thus respects our definition of \( p \).
Now, given two words $w_1$ and $w_2$, Resnik defines $c_1 \in s(w_1)$ and $c_2 \in s(w_2)$ all the concepts containing $w_1$ and $w_2$ respectively, i.e. the senses of $w_1$ and $w_2$. Therefore, there are $S_{w_1} \times S_{w_2}$ possible combinations of their word senses, where $S_{w_1}$ and $S_{w_2}$ are the cardinality of $s(w_1)$ and $s(w_2)$ respectively. We can now define $\mathcal{C}$ as the set of all the lowest common ancestor for all the combinations of $c_1 \in s(w_1)$, $c_2 \in s(w_2)$, with the lowest common ancestor of $c_1$ and $c_2$ being the more specific (lowest) concept that has $c_1$ and $c_2$ as hyponyms. The hierarchical semantic similarity between the words $w_1$ and $w_2$ can be defined as:

$$\text{sim}_{HSS}(w_1, w_2) = \sum_{\ell \in \mathcal{C}} \hat{p}(\ell = \text{LCA} \mid w_1, w_2) \times I(\text{LCA})$$

(3)

where $I(\cdot)$ is the self-information of the concept $\ell$ and $\hat{p}(\ell = \text{LCA} \mid w_1, w_2)$ is the probability of LCA being the lowest common ancestor of $w_1, w_2$, and can be computed as follows applying the Bayes theorem:

$$\hat{p}(\ell = \text{LCA} \mid w_1, w_2) = \frac{\hat{p}(w_1, w_2 \mid \ell = \text{LCA}) \hat{p}(\text{LCA})}{\hat{p}(w_1, w_2)}$$

(4)

We define $N_k$ as the cardinality of $\ell$ and all its descendants. Now we can rewrite the numerator of Eq. (4) as:

$$\hat{p}(w_1, w_2 \mid \ell = \text{LCA}) \hat{p}(\text{LCA}) = \frac{S_{[w_1,w_2]}(\ell)}{|\text{descendants}(\ell)|^2} \times N_k$$

(5)

where the first leg of the rhs is the class conditional probability of the pair $(w_1, w_2)$ and the second one is the marginal probability of class $\ell$. The term $|\text{descendants}(\ell)|^2$ represents the number of sub-concepts of $\ell$. Since we could have at most one word sense $w_i$ for each concept $c$, $|\text{descendants}(\ell)|^2$ represents the maximum number of combinations of word senses $(w_1, w_2)$ which have $\ell$ as lowest common ancestor. $S_{[w_1,w_2]}(\ell)$ is the number of pairs of senses of word $w_1$ and $w_2$ which have LCA as lower common ancestor. The denominator can be written accordingly:

$$\hat{p}(w_1, w_2) = \sum_{k \in \mathcal{C}} \frac{S_{[w_1,w_2]}(k)}{|\text{descendants}(k)|^2} \times \frac{N_k}{N}$$

(6)

3.2. Embeddings training

The corpus used to train the word vector models is composed of the OJVs’ descriptions for UK, FR, and DE only regarding ICT occupations. These OJVs’ descriptions were preprocessed applying the following pipeline: (1) punctuation removal, (2) lower case reduction, (3) tokenization, (4) typical OJV expressions and geographical indications removal, and (5) n-grams computation. The data was preprocessed separately for OJVs collected in the three different countries and also the models were trained separately.

We trained the models using the FastText architecture [17], training 72 models (24 for each country dataset). These models where trained using the following parameters’ sets: algorithm $\in \{SG, CBOW\} \times$ embedding size $\in \{50, 100, 150\} \times$ number of epochs $\in \{5, 10\} \times$ learning rate $\in \{0.05, 0.1\}$.

An intrinsic evaluation – as detailed in Section 3.1 – has been performed to select the embeddings that better preserve taxonomic relations in the three countries, by computing the Pearson correlation of the cosine similarity between each couple of skills and their corresponding HSS. The model with highest correlation for United Kingdom’s OJVs, with $\rho = 0.15$ and $p_{\text{value}} \leq 0.001$, has the following parameters: algorithm $\equiv CBOW$, size $= 50$, epochs $= 10$, learning rate $= 0.05$. For France we have $\rho=0.11$ and $p_{\text{value}} \leq 0.001$ and parameters: algorithm $\equiv CBOW$, size $= 50$, epochs $= 10$, learning rate $= 0.1$. For Germany $\rho = 0.13$, $p_{\text{value}} \leq 0.001$, algorithm $\equiv CBOW$, size $= 50$, epochs $= 5$, learning rate $= 0.05$.

4. Graph Database: S2JGraph

In this section, we present the methodology followed to encode into a graph database the information derived from both the descriptions of the OJVs, associating a word vector to each occupation and skill for each labor market considered and information based on the co-occurrences of occupations and skills in the OJVs. Our system, sk111a2job, uses this graph database as a convenient way of storing the labor market data in order to use them for recommending jobs, starting from a set of skills provided by the user. As a consequence, sk111a2job can be seen as an example of one of the many systems that can be realized on top of a graph database that encodes the information of different labor markets derived from web data.

Our graph database, called S2JGraph, is formalized as a directed labeled multi-graph and the formalization is inspired by [35]. To define the S2JGraph we need to formalize the ESCO Taxonomy, the Online Job Vacancies, and their classifier as in [35].

Definition 4.1 (ESCO Taxonomy). The ESCO Taxonomy is a triple $\varepsilon = (O, R, S)$ where $O = \{o_1, \ldots, o_n\}$ is a set of job occupations, $S$ is a set of skills $\{s_1, \ldots, s_k\}$, and $R : O \times S \to \emptyset$ is a relation that associates a job occupation $o$ to a skill $s$, namely $r(o, s) = 1$ iff the skill $s$ is associated to the occupation $o$ in ESCO and 0 otherwise.

Definition 4.2 (Online Job Vacancy (OJV)). An Online Job Vacancy is a 4-tuple $j = (i, s, t, d)$ where $i \in N$ is a unique document vacancy identifier, $s$ is an identifier of the online source from which the job vacancy was retrieved, $t$ is the text describing the title, while $d$ is the full description of the job demanded.

By abuse of notation, we denote with $d(j)$ and $t(j)$ the full description and the title of the OJV $j$ respectively.

Definition 4.3 (Job Vacancies’ Classifier). Let $J = \{j_1, \ldots, j_b\}$ be a set of OJVs as in Definition 4.2. The classification of $J$ under the set of occupation codes $O$ consists of $|O|$ independent problems of classifying each job vacancy $j \in J$ under a given ESCO occupation code $o_i$ for $i = 1, \ldots, |O|$. Then, a classifier for $o_i$ is a function $\psi : J \times O \to \{0, 1\}$ that approximates an unknown target function $\psi : J \times O \to \{0, 1\}$. Clearly, as we deal with a single-label classifier (i.e., each OJV is assigned to one and only one code), $\forall j \in J$ the following constraint must hold: $\sum_{o \in O} \psi(j, o) = 1$.

We can formalize S2JGraph as follows.

Definition 4.4 (S2JGraph). Let $\varepsilon = (O, R, S)$ be the ESCO classification system as in Definition 4.1, and let $\psi : J \times O \to \{0, 1\}$ be a classification function as in [2]. S2JGraph is a tuple $\psi = (\psi, O, R, S, w, w_e)$, where:

- $O = O \cap Q_0$, the set of occupations, with $Q_0$ the set of occupations to which at least one OJV was classified, namely $\exists o \in Q_0, j \in J$ s.t. $\psi(j, o) = 1$;
- $S = S \cup S_0$, the set of skills, where $S_0$ is the set of skills found in the text of at least one OJV;
- $R$ is the relation function that assigns a skill $s \in S$ to the occupation $o \in O$ if and only if exists a job vacancy $j$ such that $\psi(j, o) = 1$;
- $w$ is a weight function $w : R \to R^+$ assigning to each occupation–skill relation $r(o, s) \in R$ a real value that represent the relevance of $s$ within the occupation $o$;
- $w_e$ is a weight function $w_e : S \times S \to R^+$ that assigns to each pair of skills a real number that represents the similarity between skills.

5 https://tinyurl.com/sv4qur.
4.1. Defining the relevance of skills within occupations

As one might easily imagine, the simple use of the skill frequency to compute the relevance of one skill for one occupation within a given set of OJVs might be highly inaccurate. To deal with this issue, we considered the Revealed Comparative Advantage (rca) method, which was used in 2018 to assess the relevance of skills concerning occupations in the US context [38]. This measure of relevance enables one to focus on skills that are over-expressed in occupations.

In [38] authors used a measure, namely onet, provided by the O*NET taxonomy [38]. Unfortunately, the European taxonomy ESCO does not provide such a measure, hence we decided to employ the skill frequency, which for occupation $o_k \in O$ in relation to skill $s_i \in S$ is defined as:

$$sf(o_i, s_i) = \frac{\sum_{k=1}^{m} I(o_k = o_i) \cdot I(s_i = s_j)}{\sum_{k=1}^{m} I(o_k = o_i)}$$ (7)

where $I$ denotes the indicator function and $\sum_{k=1}^{m} I(o_k = o_i) \cdot I(s_i = s_j)$ is the number of times in which the skill $s_j$ is requested within the occupation $o_k$; the term $\sum_{k=1}^{m} I(o_k = o_i)$ refers to the number of job vacancies classified over the occupation $o_k$.

Given the values of $sf$ for each pair of occupation and skill, rca for $o_i$ and $s_i$ is defined as:

$$rca(o_i, s_i) = \frac{sf(o_i, s_i) / \sum_{j=1}^{p} sf(o_i, s_j)}{\sum_{k=1}^{m} sf(o_k, s_i) / \sum_{k=1}^{m} \sum_{j=1}^{p} sf(o_k, s_j)}$$ (8)

which ranges between $[0, +\infty)$. Indeed, skills requested for an occupation classified in the same ICT occupations in the data collected foreach country at one time. This approach allows us to catch the differences between ICT occupations in the United Kingdom, Germany, and France focusing on their skill set, hence discovering the characteristics of local labor markets. Indeed, skills requested for an occupation classified in the same way by a standard classification system like ESCO in the UK can differ significantly from the same profession in other countries, and this is mainly due to different levels of maturity of national labor markets.

4.2. Defining a measure of similarity between skills

To compute the skill-similarity, we considered the cosine similarity between the vectors that represent these skills in the best word embedding model, identified as it is explained in 3.1.

4.3. Defining the graph data model

The S2JGraph data model is represented in Fig. 2, in which both Occupation and Skill’s labels are classified following ESCO Classification.

The rca is modeled as a property of those relationships that connect occupations to skills, while skills self-relationships have the cosine similarity as properties.

5. Skill based recommendations

The pipeline followed by skills2job is the following.

Given a set of starting skills $S$, a starting occupation $o_i$, a starting country $c_5$, an arriving country $c_4$ and a target skill $s_j$, all provided by the user, skills2job gives back to him/her:

(i) The relevance of each $s \in S$ for $o_i$ in $c_5$;
(ii) A list of occupations $O$ in $c_4$ and for each $o_i \in O$:

- The indication of the relevance of each $s \in S$ with respect to $o_i$;
- A list of skills that $o_i$ requires and that are relevant for it and different from those in $S$ (gap skills).

(iii) A list of skills recommended to the user because of $S$ and $s_j$.

All the recommendations are generated using the Cypher query language to retrieve data from S2JGraph.

First task. The code used to perform the first task of skills2job is the following:

MATCH (a:E_L4{id:o1})-[r:SKILL_REQUESTED_c1]->(b) WHERE b.name_EN IN skills RETURN a.name_EN, b.name_EN, r.rca_norm ORDER BY r.rca_norm DESC

Cypher Query 1: Cypher query (i).

where skills is the list that contains the elements in $S$, $o_1$ corresponds to $o_i$ and $c_1$ corresponds to $c_5$. This query simply shows to the user the relevance of the starting skills for his/her occupation in his/her labor market.

Second task. The second query – which is the core of skills2job – matches all the occupations in the arriving country $c_4$ that require at least one of the starting skills in $S$. Then the query matches all the skills that are required by the occupation with a normalized rca greater than 0.6 and that have a cosine similarity with all of the starting skills in $S$ lower than 0.7. These skills are the gap skills, which are relevant for the arriving occupation and different enough from the starting skills to be recommended to the user as skills that he/she should learn to do that job in the arriving country.

The distinction between the starting country and the arriving country is crucial because our system was constructed computing the relevance and similarity measures distinctly, considering only the data collected for each country at one time. This approach allows us to catch the differences between ICT occupations in the United Kingdom, Germany, and France focusing on their skill set, hence discovering the characteristics of local labor markets. Indeed, skills requested for an occupation classified in the same way by a standard classification system like ESCO in the UK can differ significantly from the same profession in other countries, and this is mainly due to different levels of maturity of national labor markets.

The second task is performed following two different methods.

5.1. Revealed comparative advantage based (rcaB) method

The first method lets us rank the occupations on the basis of the $rca_{norm}$ with which the occupations require the starting skills:

$$rank_{it} = \frac{\sum_{j=1}^{4} rca_{norm}(o_i, s_j)}{4}$$ (10)

with $rca_{norm}(o_i, s_j)$ being the normalized rca with which the occupation $o_i$ requires the skill $s_j \in S$.

This method takes into account only the importance of the starting skills for the matched occupations, without considering the relationships between the starting skills and the skills required by these occupations. Those relationships are taken into account only to recommend the gap skills.

The first method is performed using the following code:

MATCH (z)-[r:SKILL_REQUESTED_c2]->(b) WHERE b.name_EN IN skills
WITH z, z.name_EN AS occ_name, collect(distinct b.name_EN) AS skills, count(distinct b.name_EN) AS skills_count, collect(distinct r.rca_norm) AS rca_norm, round(100*sum(distinct r.rca_norm)/4)/100 AS rank
MATCH (z)-[za:SKILL_REQUESTED_c2]->(a)-[ab:SKILL_SIMILARITY_c2]->(b)
WHERE b.name_EN IN skills AND za.rca_norm > 0.6 AND ab.cos_sim < 0.7
WITH occ_name, skills, rca_norm, rank, a.name_EN AS skills_gap, za.rca_norm AS rca_gap, count(b.id)/skills_count AS frequency
ORDER BY rca_gap DESC
RETURN rank, occ_name, skills, rca_norm, collect(skills_gap) AS skills_gap, collect(rca_gap) AS rca_gap_gap ORDER BY rank DESC

Cypher Query 2: Cypher query (ii) first method.

5.2. Cosine similarity based (cosB) method

The second method lets us rank the occupations on the basis of the cosine similarity between the starting skills in S and the most required skills for o:

\[
 rank_i = \frac{\sum_{k \in S} rca_{Norm}(o_i, s_k) \cdot \max_{s \in S} \cos_sim(s_k, s_i)}{|S|}
\]

with S, being the set of skills required by the occupation o_i with a normalized rca of at least 0.6.

Unlike the first one, this method takes into account the strength of the relationships between the starting skills and the skills required by these occupations using the property cosine similarity.

The second method is performed using the following code:

MATCH (z)-[r:SKILL_REQUESTED_c2]->(a)-[ab:SKILL_SIMILARITY_c2]->(b)
WHERE b.name_EN IN skills AND za.rca_norm > 0.6 AND ab.cos_sim < 0.7
WITH occ_name, skills, rca_norm, rank, a.name_EN AS skills_gap, za.rca_norm AS rca_gap, count(b.id)/skills_count AS frequency
ORDER BY rca_gap DESC
RETURN rank, occ_name, skills, rca_norm, collect(skills_gap) AS skills_gap, collect(rca_gap) AS rca_gap_gap ORDER BY rank DESC

Cypher Query 3: Cypher query (ii) second method.

Third task. Concerning the third query, we used a built-in Neo4j function to find the shortest paths between the skills in S and s_f (about this topic see e.g. [39]). A path that starts from a skill in S and arrives at s_f is a path that goes from the starting node to the arrival node passing through nodes connected by relationships having an attribute of cosine similarity greater than 0.8. The shortest of these paths is the one that maximizes the weight of these relationships, which is the cosine similarity. All the skills that are on these shortest paths are suggested to the user because they could interest him/her as well as the target skill s_f.

MATCH (start:SKILL_L3), (end:SKILL_L3 {name_EN: skill_target})
WHERE start.name_EN IN skills
CALL gds.alpha.shortestPath.stream({nodeProjection: 'SKILL_L3', relationshipProjection: { SIM_0_8_c2: {type: 'SIM_0_8_c2', properties: 'cos_sim', orientation: 'UNDIRECTED'} }}, startNode: start, endNode: end, relationshipWeightProperty: 'cos_sim') YIELD nodeId, cost
RETURN gds.util.asNode(nodeId).name_EN AS name, cost

Cypher Query 4: Cypher query (iii).

5.3. Examples of recommendations

As an example of the recommendations provided by skills2job let us consider:

- S = [implement front-end website design, CSS, C#, use markup languages];
- s_o = Web and multimedia developers;
- s_4 = UK;
- s_4 = DE;
- s_f = Python.

The first output is shown in Table 3 where each starting skill has associated its importance with the starting occupation in the United Kingdom. The first part of the second and main query is shown in Table 4 in which the first three recommendations in Germany are shown, ranked using the rcaB method (see 5). For each recommendation, the subset of the starting skills that are required by that occupation is shown, with their corresponding normalized rca.

The second part of the main query shows, for each recommendation, the gap skills, which are listed in Table 5.

At last, in Table 6 are shown the suggested skills that could be useful for the user. The data taken into consideration for this query is the skills that he/she already knows and the one he/she wants to learn in the context of the destination country, e.g. Germany.
6. User study

The recommendations provided by skills2job were evaluated through a user study taking inspiration from [26]. We asked ten labor market experts to judge whether the starting skills are relevant for the occupations provided by the system or not. The participants were all confident in their ability to correctly evaluating the recommendations, being active in the Labor Market Intelligence area of study, and well-acquainted in the ICT domain.

The evaluation of skills2job was performed on the British labor market, using ten different starting sets of four skills, which are shown in Table 7. The sets were chosen within the most popular skills in the ICT labor market, and selecting the clusters with high similarity with each other — with a few changes when they were too similar. For each set of skills, skills2job provided three recommendations with the first method and three with the second (as described in Section 5), for a total of 60 recommendations per user and 600 in total (Table 8 shows the 60 recommendations generated for the user study). To avoid bias we did not disclose which item was recommended by which method and items recommended by both were listed only once for each starting set of skills.

We asked the participants to evaluate the recommendations on a Likert scale ranging from 1 to 5, with 1 being not relevant to 5 being completely relevant. We presented three recommendations for each item and method, so we decided to use Precision@3 (P@3) for measuring the accuracy of skills2job’s recommendations. Since in Likert-type items there is not only one way to discriminate between a true positive and a true negative, we calculated P@3 assuming a user score of at least 3 as a true positive in one case (P@3-3) and of at least 4 in the other (P@3-4). Due to the nature of the discrimination, it is clear that in the second case the accuracy is lower, although the degree of certainty that the result is indeed a true positive is higher. We also computed the normalized Discounted Cumulative Gain (nDCG), which measures the usefulness of an item based on its position in the list of recommendations.

7. Results and discussion

All of the ten experts responded to the user evaluation and there were no missing votes, meaning we obtained the desired 600 votes. We show a visualization of the votes in Fig. 3.
Table 7
Sets of starting skills for the user study.

<table>
<thead>
<tr>
<th>Skill 1</th>
<th>Skill 2</th>
<th>Skill 3</th>
<th>Skill 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDE software</td>
<td>Agile project management</td>
<td>UML</td>
<td>ICT debugging tools</td>
</tr>
<tr>
<td>Web programming</td>
<td>Java</td>
<td>C++</td>
<td>Python</td>
</tr>
<tr>
<td>Maintain database</td>
<td>Manage ICT data architecture</td>
<td>SQL</td>
<td>NoSQL</td>
</tr>
<tr>
<td>Digital marketing</td>
<td>Manage website</td>
<td>Graphic design</td>
<td>WordPress</td>
</tr>
<tr>
<td>Analyze big data</td>
<td>Natural Language Processing</td>
<td>Hadoop</td>
<td>Manage data</td>
</tr>
<tr>
<td>Implement front-end</td>
<td>Use markup languages</td>
<td>CSS</td>
<td>C#</td>
</tr>
<tr>
<td>Data ETL tools</td>
<td>Interpret current data</td>
<td>Manage data collection</td>
<td>Data mining</td>
</tr>
<tr>
<td>ICT communications</td>
<td>Web application security threats</td>
<td>Maintain ICT server</td>
<td>Implement a VPN</td>
</tr>
<tr>
<td>Mobile operating systems</td>
<td>Use digital device operating systems</td>
<td>Mobile device management</td>
<td>Cloud technologies</td>
</tr>
<tr>
<td>Use markup languages</td>
<td>Ajax Framework</td>
<td>JavaScript</td>
<td>Sass</td>
</tr>
</tbody>
</table>

Table 8
Top three recommendations for the 10 skill set and for the two methods for the user study.

<table>
<thead>
<tr>
<th>Skills set</th>
<th>First recommendation</th>
<th>Second recommendation</th>
<th>Third recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>rcaB</td>
<td>Software developers</td>
<td>Applications programmers</td>
</tr>
<tr>
<td></td>
<td>cosB</td>
<td>ICT user support technicians</td>
<td>Software developers</td>
</tr>
<tr>
<td>2</td>
<td>rcaB</td>
<td>Software developers</td>
<td>Database and network professionals</td>
</tr>
<tr>
<td></td>
<td>cosB</td>
<td>Software developers</td>
<td>Systems administrators</td>
</tr>
<tr>
<td>3</td>
<td>rcaB</td>
<td>Database designers and administrators</td>
<td>Software developers</td>
</tr>
<tr>
<td></td>
<td>cosB</td>
<td>Software developers</td>
<td>Database designers and administrators</td>
</tr>
<tr>
<td>4</td>
<td>rcaB</td>
<td>Web and multimedia developers</td>
<td>Web technicians</td>
</tr>
<tr>
<td></td>
<td>cosB</td>
<td>Web technicians</td>
<td>ICT service managers</td>
</tr>
<tr>
<td>5</td>
<td>rcaB</td>
<td>Database designers and administrators</td>
<td>Database and network professionals</td>
</tr>
<tr>
<td></td>
<td>cosB</td>
<td>Software developers</td>
<td>Systems analysts</td>
</tr>
<tr>
<td>6</td>
<td>rcaB</td>
<td>Web and multimedia developers</td>
<td>Software developers</td>
</tr>
<tr>
<td></td>
<td>cosB</td>
<td>Web and multimedia developers</td>
<td>Applications programmers</td>
</tr>
<tr>
<td>7</td>
<td>rcaB</td>
<td>Database designers and administrators</td>
<td>Systems analysts</td>
</tr>
<tr>
<td></td>
<td>cosB</td>
<td>ICT user support technicians</td>
<td>Database and network professionals</td>
</tr>
<tr>
<td>8</td>
<td>rcaB</td>
<td>Computer network and systems technicians</td>
<td>Systems administrators</td>
</tr>
<tr>
<td></td>
<td>cosB</td>
<td>Software and applications developers and analysts</td>
<td>ICT user support technicians</td>
</tr>
<tr>
<td>9</td>
<td>rcaB</td>
<td>Systems administrators</td>
<td>Computer network and systems technicians</td>
</tr>
<tr>
<td></td>
<td>cosB</td>
<td>Software developers</td>
<td>Applications programmers</td>
</tr>
<tr>
<td>10</td>
<td>rcaB</td>
<td>Web and multimedia developers</td>
<td>Applications programmers</td>
</tr>
<tr>
<td></td>
<td>cosB</td>
<td>Software developers</td>
<td>Software and applications developers and analysts</td>
</tr>
</tbody>
</table>

Table 9 shows the results for P@3-3, P@3-4 and nDCG. rcaB outperforms cosB in precision, despite both methods obtained good results in P@3 and nDCG (see [26]). The difference between the two algorithms might be because the cosB algorithm uses the cosine similarity between word vectors to define similarity relations between skills, but we do not use it to model the relation between skills and occupations, which is the core of this task. The cosine similarity between skill vectors, which are known to preserve linguistic regularities that are not captured by co-occurrence statistics, becomes crucial when we identify the skill gap between different occupations since skills which are dissimilar from a morphological point of view can be semantically similar.

The nDCG scores for both methods are similar and close to 1. These results suggest that there is a high degree of correlation between the user evaluation and the ordering rank of our recommendations.

8. Conclusion and future outlook

In this paper, we have proposed a recommender system that identifies a suitable job starting from a set of user’s skills. We use a data driven approach, extracting information from a large dataset of 2.5M+ Online Job Vacancies through distributional semantics and co-occurrence statistics. The information extracted is organized in a graph database, which can be queried to enable
several recommendations. Results were evaluated by labor market experts and show a high precision in identifying jobs starting from a set of skills and a high correlation between experts’ judgments and the recommendation’s rank.

As a future expansion of this work, it would be interesting to develop a method to evaluate skill-job fit not only using co-occurrences statistics (rcos) but employing the information derived from the word embeddings models.
CRediT authorship contribution statement

Anna Giabelli: Methodology, Investigation, Formal analysis, Writing - original draft. Lorenzo Malandri: Methodology, Conceptualization, Writing - review & editing, Validation. Fabio Mercurio: Methodology, Conceptualization, Writing - original draft, Supervision. Mario Mezzanzanica: Funding acquisition, Project administration, Investigation. Andrea Seveso: Methodology, Data curation, Conceptualization, Writing - original draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is partially supported within the research activity of an EU Project entitled “Real-time labor market information on Skill Requirements: Setting up the EU system for online vacancy analysis” in which some of the authors are involved as coordinators and researchers.

References

[10] CEDEFOP, Real-time labour market information on skill requirements: Setting up the EU system for online vacancy analysis, 2016, https://goo.gl/5FZ53E.