Taylor & Francis
Taylor & Francis Group

Applied Artificial Intelligence
An International Journal

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/uaai20

HR-Specific NLP for the Homogeneous Classification of Declared and Inferred Skills

Lorenzo Ricciardi Celsi, Jesus Fernando Cevallos Moreno, Federico Kieffer & Valerio Paduano

To cite this article: Lorenzo Ricciardi Celsi, Jesus Fernando Cevallos Moreno, Federico Kieffer & Valerio Paduano (2022) HR-Specific NLP for the Homogeneous Classification of Declared and Inferred Skills, Applied Artificial Intelligence, 36:1, 2145639, DOI: 10.1080/08839514.2022.2145639

To link to this article: https://doi.org/10.1080/08839514.2022.2145639

© 2022 The Author(s). Published with license by Taylor & Francis Group, LLC.

Published online: 02 Dec 2022.

Submit your article to this journal

Article views: 558

View related articles

View Crossmark data

Full Terms & Conditions of access and use can be found at
https://www.tandfonline.com/action/journalInformation?journalCode=uaai20

APPLIED ARTIFICIAL INTELLIGENCE
2022, VOL. 36, NO. 01, e2145639 (3963 pages)
https://doi.org/10.1080/08839514.2022.2145639

Taylor & Francis
Taylor & Francis Group

OPEN ACCESS Check for updates

HR-Specific NLP for the Homogeneous Classification of Declared and Inferred Skills

Lorenzo Ricciardi Celsi[a], Jesus Fernando Cevallos Moreno[b], Federico Kieffer[a], and Valerio Paduano[a]

[a]ELIS Innovation Hub, Roma, Italy; [b]Department of Computer Control and Management Engineering Antonio Ruberti, Sapienza Università di Roma, Roma, Italy

ABSTRACT

The use of natural language processing in human resource management has become of paramount importance in order to provide support for recruiting and corporate population management. This paper proposes a heuristic algorithm to solve two problems: (i) semantic matching among heterogeneous datasets storing the hard skills possessed by the company's employees to obtain a homogeneous catalog, according to the O*NET and ESCO competence dictionaries, and (ii) inferring the employee's soft skills with respect to his/her own declaration of interests, work experience, certifications, etc., given his/her curriculum vitae. Empirical results demonstrate that the proposed approach yields improved performance results by comparison with baseline methods available in the literature.

ARTICLE HISTORY
Received 19 July 2022
Revised 3 November 2022
Accepted 4 November 2022

Introduction

Recent technological evolution has mandated the analytical use of big data analytics and artificial intelligence in business management. Still, there have been few success stories concerning the application of such methods and techniques to effective Human Resource (HR) management, mainly to provide support for recruiting and corporate population management in large organizations (Bizer et al., 2005).

An early need for many companies is to hire people who can fill multiple roles (especially to cope with ongoing readjustment needs). In addition, an ever-present, but now more critical, need is to carry out the task of dynamically cataloging the skills (both of current employees and future hires) that the market shows to be of relevance to its core business: such an analysis, led by data from the corporate population cross-referenced with public data such as the European Skills, Competencies, Qualifications, and Occupations (ESCO) classification, would enable more effective management of the recruiting process, as well as of the company's own internal professional growth paths,

CONTACT Lorenzo Ricciardi Celsi ricciardicelsi@diag.uniroma1.it ELIS Innovation Hub, via Sandro Sandri 81, Roma 00159, Italy

© 2022 The Author(s). Published with license by Taylor & Francis Group, LLC.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

allowing it to improve the reputation that the company builds with its workers, and most importantly to enhance employer branding, keeping the company aligned with current labor market characteristics.

Therefore, differently from model-based solutions such as (Ricciardi Celsi et al. 2015; Battilotti et al. 2019), the path toward a data-driven vision forces companies to conceive data as the strategic focus of their investments, aiming at a significant change of step that is now pervading every business function, including HR. The implementation of this transformation cannot be separated from the construction of a data-driven architecture capable of:

- selecting data sources;
- collecting and transforming different data formats;
- creating the knowledge base typical of the considered business context;
- data democratization, that is, creating and displaying reports useful as decision support.

Data source selection can take place in an initial data gathering stage, which collects the necessary information from a set of appropriately chosen data sources.

Next, there is a stage of data engineering, which carries out the so-called ETL process, that is, extracting, transferring and loading the data onto the data layer of the cloud platform hosting the IT system of the company.

In this way, it is possible to create the desired knowledge base (in particular, a homogeneous catalog of the skills of the company's population) and an AI engine that acts as a decision support system for efficient resource reallocation and job rotation – which not only takes into account hard skills that are usually easier to read because they are stated, but also takes into account soft skills thanks to inference systems that can extract these from other, less declarative data sources. In this regard, the use of text mining algorithms, possibly based on machine learning and deep learning techniques, is essential.

Finally, the stage of data democratization makes available, to those in charge of personnel management, a set of advanced report visualization tools useful for optimal management of recruiting as well as of the current company population.

Relying on such data-driven architecture in order to tackle the above-mentioned use cases yields the following benefits.

Avoiding the manual and very often subjective collection of employees' training needs and minimizing the manual and subjective calculation of the return on investment of training, ultimately offering a choice of training courses that is necessarily in line with the employee's job profile;

Optimization and adaptation of resource staffing on activities: by systematizing the entire professional and extra-professional career and aspirations of

each employee, there is the possibility of more dynamic, efficient and rigorous construction of work teams, saving time and making the staffing process more effective.

Related Works

We have divided the related works into two broad categories.

- Skill extraction or taxonomy generation systems, explored in Section 2.1.
- Skill matching and job recommendation systems, explored in Section 2.2.

Skill Extraction

In (Colucci et al. 2003) the concept of implicit skills is introduced. Indeed, in this work the authors propose a method to mine implicit skills using word and document embeddings. In (Bastian et al. 2014) the LinkedIn team builds a large-scale topic extraction pipeline that includes constructing a folksonomy of skills and expertise and implementing an inference and recommender system for skills.

A framework for skill extraction and normalization is proposed also in (Zhao et al. 2015). Kivimaki et al. (2015) propose a system for skill extraction from documents primarily targeting hiring and capacity management in an organization. The system first computes similarities between an input document and the texts of Wikipedia pages and then uses a biased, hub-avoiding version of the spreading activation algorithm on the Wikipedia graph to associate the input document with skills.

Skill Matching

There is a wide variety of approaches in the literature regarding the skill matching problem, ranging from description logic (Cali et al. 2004), through machine learning (Faliagka et al. 2012), to semantic and ontology-based approaches (Mohaghegh and Mohammad 2004; Hassan et al. 2012; Bizer et al. 2005).

It is worth noting that, whatever the approach used, the matchmaking process turns out to be as much successful as effectively the participants' profile is modeled (Joshi et al. 2010).

There is an emerging trend in terms of relying on a tree or graph structure to model the employee's profile (Hassan et al. 2012): this structure is usually modeled with the help of an ontology, on which an algorithm operates to perform matching (Lv and Zhu 2006): such ontology is modeled as a graph where nodes are skills and weighted edges are relationships among skills. The

authors' approach toward matching a job offer with a candidate is an exhaustive search which compares all the skills from the offer with all the skills of a candidate and computes the shortest path in order to determine the best match. The advantage of such an approach is that it is precise and computationally inexpensive with the right implementation choice; however, the main drawback is that the ontology weights need to be manually assigned, possibly by domain experts, which proves to be quite laborious. An alternative is to replace the ontology with a taxonomy for describing and classifying skills. In this respect, Bizer et al. (2005)) rely on a similarity function that is evaluated to be inversely proportional to the distance between two concepts in the taxonomy. Furthermore, in order to have a more customizable matching process, one could allow the user to make a distinction between nice-to-have and must-have requirements (Fazel-Zarandi and Fox, 2010). In this way, not all matches contribute with the same amount to the final matching result.

Effective skill matching is typically necessary in job recommendation systems. According to the definition given by Al-Otaibi and Ykhlef, (2012), a job recommendation system generally returns a set of job recommendations in response to the user's current profile. In this respect, job seekers, on the one hand, can upload their skills or resume or the job search criterion, whereas the employers, on the other hand, can upload job descriptions or skill sets needed along with any additional information such as location, position, and job-specific details. A candidate acquires skills through formal education, internships, and/or previous job experience. At some point in time the candidate starts identifying new relevant jobs based on the acquired skills. The key function of a job search engine is to help the candidate by recommending those jobs which represent the closest match to the candidate's existing skill set. This recommendation can be provided by matching skills of the candidate with the skills mentioned in the available job description.

The easiest approach to perform skill matching is to use standard keyword matching or a standard information retrieval framework as proposed by Salton and Buckley (1988). More recently, Malinowski et al. (2006) propose bilateral matching via an expectation maximization algorithm, whereas Golec and Kahya (2007) introduce a fuzzy model for competence-based employee evaluation and selection. Paparrizos et al. (2011), instead, use a naïve Bayes classifier.

However, all the above-mentioned recommendation systems are only effective within a single organization where there are standardized job roles. More in detail, a few challenges that thereby emerge are the following: (a) the skill may be mentioned in different forms or through synonyms in curricula vitae and job descriptions; (b) there may exist skills which might not be specified in a candidate's profile or a job description, but can be easily inferred by means of business knowledge – for example, experience in Java, being it an object-oriented programming language, implies experience in object-oriented

programming –; (c) a skill could be an out-of-dictionary skill, that is, it could be missing from the dictionary or it could belong to a new unseen domain for which the system does not currently exhibit any skills.

For this reason, in many industry sectors where the job description is not easily standardized, especially relative to advanced/executive career paths and to job descriptions for senior profiles, job recommendation is still an unsolved challenge. Some attempts to tackle this problem have already been made in the literature. Liu et al. (2016) propose to perform job recommendation by combining a time-based ranking model applied to historical observations and a recurrent neural network to map sequence properties. Gugnani and Misra (2020) rely on generated a temporal skill graph that is used to recommend optimal career path, namely suggesting the next set of skills to acquire. Instead, Lin et al. (2016) propose a convolutional neural network to match resumes with relevant job descriptions.

Motivation

This work is aimed at solving the following tasks:

- *Task 1*: performing semantic matching among heterogeneous datasets storing the hard skills possessed by the company's employees – namely between, on the one hand, a data source containing various texts related to the professional profile of the employee, and, on the other hand, the O*NET and ESCO competence dictionaries – to obtain a homogeneous catalog.
- *Task 2*: relative to the company's full scale, inferring the employee's soft skills with respect to his/her own declaration of interests, work experience, certifications, etc., given his/her curriculum vitae and/or LinkedIn profile.

Methodological Framework

Task 1: Semantic Skill Matching (Or NER)

We rely on Named-Entity Recognition (NER) – as introduced and discussed in Li et al., 2021a, (Li et al. (2021) and Li et al. (2022) – as the relevant methodology for tackling the problem of *semantic skill matching for HR data homogenization*. More in detail, this problem consists in matching the hard skills possessed by the company's employees with the records of the O*NET and ESCO competence dictionaries in order to obtain a homogeneous catalog of the competences of the company's employees.

Semantic skill matching requires text mining on the curricula vitae (CV) of the company's employees. Thus, the text in the CV must be organized in such

a way as to be compatible with data analytics tools: this implies transforming text into numerical tables that can be processed by text mining algorithms such as those adopted hereinafter.

With *raw data* we define the data input, consisting of the CV of the company's employees either in Europass format or in LinkedIn profile format. We will refer to a candidate's skill set with the term CV (as in the curriculum vitae of the candidate). Let $\{CV_i\}_{i=1,\ldots,e}$ be the set of all the CVs in the search space, considering a total number of e employees. We refer to each sentence appearing in a CV as a *document*, which, in turn, consists of a sequence of words, called *terms*, out of which the hard skills of the employee are extracted and fed to the skill matching engine at a generic time instant. Each term can be enriched with a *tag*, denoting the role the term plays in terms of grammatical analysis of the natural language expressed by the document. In this work, we rely on tags that explain the lexical class (namely, *Part of Speech* or PoS) of each term in the document (e.g., whether the term is a noun, an adjective, a verb, an adverb).

Let us represent our algorithm as the series combination of two functions, $F(\cdot)$ and $G(\cdot)$. We now explain in detail how these two functions are designed.

Function $F(\cdot)$. The first function takes in input the CV represented as a matrix U. The rows of U are the input documents denoted by the word vectors U. Function F outputs a skill matrix X, whose rows are the vectors containing the extracted skills for each input document U, namely $X = F(U)$.

More in detail, let us assume that function $F(\cdot)$ processes a different document in the CV at each time instant t. Let then

$$X(t) = \begin{bmatrix} x_1 \\ \vdots \\ x_t \end{bmatrix} = \begin{bmatrix} x_{1,1} & \ldots & x_{1,j} & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ x_{t,1} & \ldots & x_{t,j} & x_{t,j+1} & \ldots & x_{t,n} \end{bmatrix}$$

denote the generic CV of an employee at time t, consisting of a sequence of t documents, each corresponding to a different row of the X matrix, for $t \in |t = 1, \ldots, z$. Namely, z is the total number of documents in the input CV, j is the number of detected hard skills in the declaration made by the employee in the considered input document i, for $i = 1, \ldots, t$, and n is the maximum number of hard skills detected among all documents in the input CV considered up to time t. For the sake of clarity, we denote the generic element of matrix $X(t)$ with $x_{i,j}$ $(i = 1, \ldots, t; j = 1, \ldots, n)$, that is, a string accounting for a generic skill phrase. A skill phrase may be either a monogram (e.g., Python) or a bigram (e.g., Java Script) or even a trigram (e.g., Project Management Office), so long as it denotes the possession of a single skill.

Function $G(\cdot)$. Function $G(\cdot)$ instead takes in input the matrix X and outputs a homogeneous skill matrix R, forced by stacking vectors R. Each R vector

is a corrected version of the corresponding X vector in X, such that the extracted skills are homogenized with respect to a reference skill dictionary.

Hence, we assume to have a reference skill dictionary denoted with the time-varying set $\mathcal{D}(t) = \left\{x_o^{ref}\right\}, o = 1, \ldots, m(t), m(t) \in |m(t) \gg n$ and we denote the homogenized CV with matrix $R(t)$, whose entries are cherry-picked among the elements of $\mathcal{D}(t)$, i.e.,

$$R(t) = \begin{bmatrix} r_1 \\ \vdots \\ r_t \end{bmatrix} = \begin{bmatrix} r_{1,1} & \ldots & r_{1,j} & 0 & \ldots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ r_{t,1} & \ldots & r_{t,j} & r_{t,j+1} & \ldots & r_{t,n} \end{bmatrix}$$

$$|r_{i,j} \in \mathcal{D}(t), for i = 1, \ldots, tand j = 1, \ldots, n. \quad (1)$$

Note that the reference skill dictionary $\mathcal{D}(t)$ is not static but, over time, it is enriched with newly detected skills extracted from the data input: hence, its dimension $m(t) \in$ is time-varying and eventually larger than $\bar{m}$ (i.e., the cardinality of the reference dictionary at time $t = 0$). More in detail, the skill dictionary is a set of standardized skill phrases denoted with x^{ref}, that is, groups of single or multiple words that represent a specific skill, as extracted from the following sources: O*NET (2022) and ESCO (2022).

The current version of the reference skill dictionary $\mathcal{D}(t)$ in this work contains approximately $m = 23,000$ entries and can be arranged to be regularly updated by downloading the latest version of O*NET and ESCO from the web.

The O*NET database is a publicly available specialized database containing skill phrases and field terminologies. It contains a rich set of variables that describe work and worker characteristics, including skill requirements.

The ESCO database is a multilingual classification of the European Skills, Competences, Qualifications and Occupations, which are relevant for the EU labor market, education and training.

Given this framework and given the notation introduced above, we propose a heuristic NER algorithm that performs skill extraction and matching as the series combination of the two functions $F(\cdot)$ and $G(\cdot)$, so that $R = G(F(U))$, according to the functional flow shown in Figure 1.

In this respect, the results of three parallel processing streams are conveyed into a unique output skill set, which eventually returns the hard skills declared by the generic employee translated according to the terminology of the reference skill dictionary.

Stream a (PoS)

With *Stream A (PoS)* we denote the first processing stream, which leverages a foundation model for natural language processing, namely the Google-trained BERT (Bidirectional Encoder Representation from Transformers)

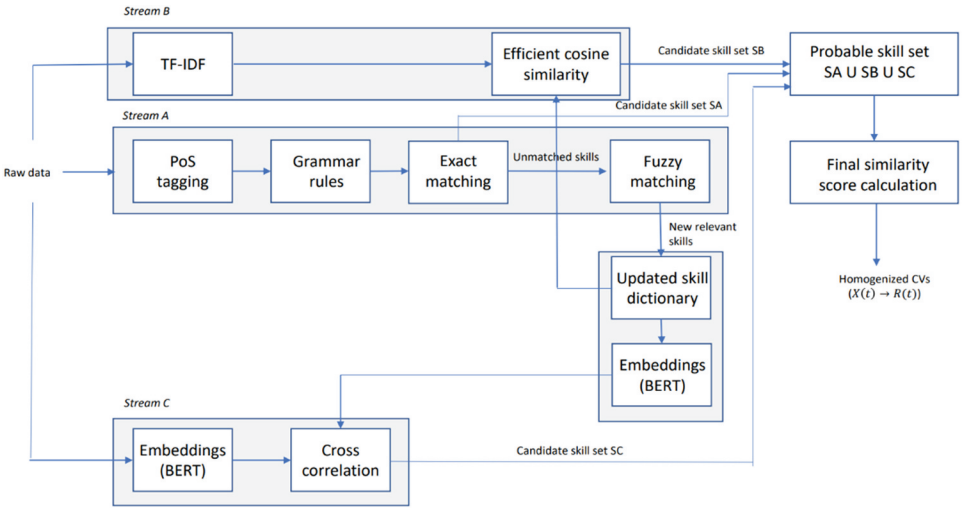

Figure 1. Functional flow of the algorithmic framework.

algorithm (Devlin et al. 2019) implemented using the Hugging Face Transformers APIs (), with a set of classification layers devoted to PoS tagging. Starting from this pre-trained algorithm, the eventual classification layers have been subject to fine-tuning via a transfer learning approach based on the Treebank of Learner English dataset (Berzak et al., 2016). In this respect, we did a manual annotation exercise to identify text patterns of skill-terms occurrence. We asked five HR professionals to manually label and annotate, in a few hundred CVs collected by ELIS Innovation Hub, the terms/keywords that they recognize as skill phrases. We analyzed these CVs and manually annotated how the skills occurred in terms of PoS tags. Based on such data we developed a generalized set of rules to identify potential occurrence of skills in a sentence. As a result, this processing stream, consistently with the rationale of PoS tagging, embeds three different *grammar rules* that are applied, for skill extraction purposes, to the input sentence, or document, coming from the raw data (CV or LinkedIn profile). Namely,

(1) the "comma" rule: any nouns separated by commas in a document are likely to be skills;
(2) the "and" rule: any nouns preceded by the conjunction "and" are likely to be skills;
(3) the single-skill rule: any isolated nouns are likely to be skills.

For example, in the document "I have the ability to code in Python, Java, and Octave," according to the three rules all three programming languages will be considered as possible skills for the subsequent matching stage with respect to the O*NET and ESCO competence dictionaries. All the aforementioned rules

also ensure robustness with respect to the detection of multi-word skills (bigrams and trigrams): this means that Stream A (PoS) is even successful at detecting a bigram (e.g., "Java Script") or a trigram as a single skill.

Algorithm 1. Stream A (PoS) pseudocode for time instant t.

1: Map input document $u(t) = [u_1, \ldots, u_w]$, where $w \in |w\rangle j$ is the cardinality of the input document, to PoS tag vector $y(t) := [PoS_1, \ldots, PoS_w]$ using the PoS tagging fine-tuned BERT model.

2: Apply grammar rules 1), 2) and 3) to PoS tag vector $y(t)$ for skill extraction

3: Return candidate skill vector x_t associated with input document $u(t)$

4: Apply exact matching on extracted skills in x_t

5: Apply fuzzy matching on extracted skills in x_t

6: Return homogenized skill vector r_t (associated with x_t) using only entries from $\mathcal{D}(t)$

7: Update $\mathcal{D}(t)$ with feedback from newly discovered skills

The first output of stream A (PoS) is the so-called *candidate skill set SA*, which lists the union of the skills detected by the three above-mentioned rules.

In addition, the second output of stream A (PoS) contains a list of new relevant skills x_{new}^{ref} that are fed back, in a dynamic fashion, to the skill dictionary, thus being added to the already existing ones so that $\mathcal{D}(t') = \left\{x_1^{ref}, \ldots, x_m^{ref}, x_{new}^{ref}\right\}$ with $t' > t$. These new skills are selected based on a combined frequency-dissimilarity criterion: the new skill under investigation has to appear a sufficiently high number of times T_O in the input and has to be sufficiently different (i.e., above a given threshold T_D) from the skills that have already been matched with the ones in the previous version of the reference dictionary (i.e., $\mathcal{D}(t)$).

In general, the role of Stream A (PoS) is to perform *grammatical analysis* of the input document, that is, (i) to decompose the sentence into its atomic elements, or terms, and (ii) to supervisedly mark a term with the corresponding PoS tag so that subsequently a direct matching algorithm with a standardized dataset (that is, $\mathcal{D}(t)$) can be run. Algorithm 1 contains the pseudocode of Stream A (PoS).

Exact matching – as in ln. 4 of Algorithm 1 – matches each input skill resulting from skill extraction at ln. 2 to all possible skills in the reference skill dictionary and selects only those that score 100% similarity. Exact matching can be considered as a naïve baseline for every other robust skill text mining algorithm. We then use fuzzy matching at ln. 5 (by resorting to the FuzzyWuzzy Python library (2022)) to compute a correlation score between substrings from the input sentences in the CV and the reference dictionary. Fuzzy matching is useful to make sure that the new candidate skills to be added to the skill dictionary are sufficiently different from the

ones that are already present in it, thus avoiding duplication. The scoring thresholds can be manipulated to achieve different levels of matching accuracy. In this respect, we performed extensive parameter optimization for achieving the highest similarity scores. The main disadvantage of fuzzy matching is the quadratic time complexity of such an approach with respect to the dimension of the reference skill dictionary. This is the main reason why we chose to add Stream B and C too, thus allowing an extensive reduction of execution times.

Stream B (TF-IDF)

Stream B (TF-IDF), in turn, performs the task of *approximate* skill matching by exploiting the classical TF-IDF (Term Frequency-Inverse Document Frequency) algorithm according to the definition in Rajamaran and Ullman (2011). It returns the *candidate skill set SB*, listing the skills belonging to the reference skill dictionary $\mathcal{D}(t)$ that score the highest cosine similarity degree with respect to the skills declared by the employee. For instance, if an employee declares "I have programming abilities in C, Piton and Micosoft Excel" in her or his CV, Stream B, as a result of processing this document, returns the candidate skill set (C, Python, Microsoft Excel), uniformly with the current reference skill dictionary $\mathcal{D}(t)$with the corresponding cosine similarity score (hereinafter referred to as $\hat{s}_B$) expressed on a scale from 0 (i.e., no similarity) to 1 (highest similarity degree). For the purpose of such calculation, we relied upon efficient cosine similarity according to the approach proposed by the dedicated Python package sparse_dot_topn (Singh and Sathi, 2018,), exploited for optimizing the computation of cosine similarity between big sparse matrices.

Algorithm 2. Stream B (TF-IDF) pseudocode for time instant *t*.

1: Encode the reference skills appearing in the reference skill dictionary $\mathcal{D}(t)$ via TF-IDF based n-gram vectorization according to the procedure described in Singh and Sathi,).

2: Encode the generic skill phrase x_{ij} in document x_t via TF-IDF based n-gram vectorization according to the same procedure as in ln. 1.

3: Compute efficient cosine similarity between computed n-grams according to Python package sparse_dot_topn (2022) and return only the skill phrases that are assigned the highest similarity score (above a given threshold T_B).

In general, Stream B offers a second level of text mining, by performing an *importance analysis* of the skill phrases appearing in the input document followed by direct matching with the reference dictionary. This layer makes the overall algorithmic framework robust with respect to any typos appearing in the input data (that is, the raw information related to the skills declared in the CV).

Stream C (W2V)

Stream C (W2V), instead, computes standard cross-correlation between two inputs: the word embedding of the skill phrase appearing in the input sentence, on the one hand, and the word embeddings of the skills available in the dictionary. Both these embeddings are generated using a pre-trained version of the BERT encoder (https://colab.research.google.com/drive/1ZQvuAVwA3IjybezQOXnrXMGAnMyZRuPU). In this case, the cross-correlation function returns a similarity score (hereinafter referred to as $\hat{s}_C$), between the two input embeddings, falling in the range [0,1]. The output of Stream C is the so-called *candidate skill set SC*, which consists of those skills belonging to the reference skill dictionary $\mathcal{D}(t)$ that score the highest cross-correlation degree with respect to the skills declared the employee.

For example, if in the input CV the following skill phrase appears – "I have experience programming in cplusplus" –, the cross-correlation between the "cplusplus" string and the "c++" skill in the reference dictionary will turn out to be above average, causing the skill "c++" to be added to the list of potential skills of the candidate. Such a semantic method can be used to improve the robustness of the skill extraction algorithm. The semantic awareness property of the Stream C (W2V) is inherited from the pre-trained embedding machinery used: both the list of embeddings associated to the input CV and the reference skill dictionary are calculated using a pre-trained BERT encoder. As a result, the skills in the reference dictionary that are assigned the highest similarity scores (above the threshold T_C) are selected as *candidate skill set SC*.

Algorithm 3. Stream C (W2V) pseudocode for time instant *t*.

1: Compute the word embedding reference skills appearing in the reference skill dictionary $\mathcal{D}(t)$ using the pre-trained version of the BERT encoder in (https://colab.research.google.com/drive/1ZQvuAVwA3IjybezQOXnrXMGAnMyZRuPU, BERT, 2022.

2: Compute the word embedding of the generic skill phrase x_{ij} in document x_t using the same pre-trained version of the BERT encoder as in ln. 1.

3: Compute standard cross-correlation between the word embeddings computed at ln. 1–2.

4: Return as candidate skill set *SC* only the skills in the reference dictionary that score the highest similarity (namely, above the threshold T_C).

The role of Stream C (W2V) is to perform *semantic analysis* of the input document, thus relating the syntactic structure of the input document to its language-independent meaning. More in detail, the usage of pre-trained BERT encoders allow to compute semantic-aware word embeddings. As a consequence, the cross-correlation between any two word embeddings is a similarity kernel that resembles semantic proximity. For this reason, the reference skills' embeddings can be semantically compared with the word embeddings of the skill phrases of each document through cross-correlation: thus, this layer makes the performance of the

overall algorithm robust with respect to the semantic variety that may characterize the input document.

The outputs of the three streams are then conveyed into a unique final similarity score $s_{p,q}$ of the skill phrase x_p with skill phrase x_q^{ref} from the reference dictionary $\mathcal{D}(t)$, computed according to a weighted sum rule applied to the related partial scores (namely, $\hat{s}_A$, $\hat{s}_B$, and $\hat{s}_C$). See the Appendix for the formula of the final similarity score $s_{p,q}$.

Furthermore, a rule for the generation of an accurate final skill set has been established. The rule makes use of an adaptive cutoff threshold μ that is applied to the final similarity score of each skill. In particular, the threshold is computed by considering a suitable percentage μ of the maximum final score that has been assigned (i.e., $s := \max_{p,q} s_{p,q} \quad p \in \{1, \ldots, n\}, \quad q \in \{1, \ldots, n\}$). Hence, if

$$s_{p,q} > \mu \cdot s_{max},$$

then the skill phrase x_p appearing in the original CV is preserved, otherwise it is discarded. In addition to the adaptive cutoff threshold μ, a final threshold is introduced for the purpose of false-positive avoidance. This additional threshold admits a maximum number of $k \cdot N$ top skills as output, where N is the number of words that are present in the input CV and k is an adjustable coefficient. All the other detected skills are automatically discarded. So, among the x_p's that result from the previous cutoff stage, only the first $k \cdot N$ skill phrases are selected and each of them is ultimately replaced with the reference skill phrase x_q^{ref}, thus adding relevant content to the t-th row of the homogenized CV $R(t)$. As a result, the overall flow is the following:

Algorithm 4. Stream C (W2V) pseudocode for time instant t.

Given CV_i of employee i, apply NER control action $a(t)$ so that $X(t) \rightarrow R(t)$.

The NER control action $a(t)$ is a sequence of parallelized instructions (namely, lines 1–3 below) as follows:

1: Stream A pseudocode for time instant t (PoS)

2: Stream B pseudocode for time instant t (TF-IDF)

3: Stream C pseudocode for time instant t (W2 V)

4: Then, compute final similarity score $s_{p,q}$, p, q relevant to document t in CV_i (i.e., associated with the t-th row of $X(t)$)

5: Select only the the first $k \cdot N$ skill phrases x_p such that $s_{p,q} > \mu \cdot s_{max}$ and discard all other skill phrases.

6: For the selected x_p's, replace $x_p \leftarrow x_q^{ref}$.

Task 2: Skill Inference

The problem of *skill inference* consists in inferring the employee's soft skills with respect to his/her own declaration of (a) interests and hobbies, and (b)

work experience and job position, given his/her CV. It is interesting to note that this inference task has an intrinsic degree of complexity due to the absence of a precise mapping between the input (interest and work experience), on the one hand, and the expected output (inferred soft skills). This inference task has been addressed with the methodological approach described in Algorithm 5 below.

Algorithm 5 makes use of a fine-tuned input-to-skill classifier, for inferring soft skills from any declaration of interest and hobbies (hereinafter named *BERToby*), as well as work experience and current and past job positions (hereinafter, named *JoBERT*). In this case, in order to perform a proper classification, adequate classes must be defined.

BERToby

Algorithm 5. Skill inference pseudocode (BERToby for hobbies and JoBERT for work experience).

1: Preprocessing input through BERT embeddings in order to identify and restrict the inference domain.

2: The fine-tuned fully-connected layers appended to the BERT model allow to classify the embedded input.

3: The skills associated to the predicted class are inferred.

In the case of BERToby, we resorted to the hobby categorization introduced by the Discover a Hobby portal (2022). This categorization can be regarded as representative of the main possible hobbies: namely, 12 distinct classes are identified – art and craft, collecting, extreme sports, food and drink, games, individual sports, model and electronics, music, performing arts, pets, spiritual and mental, sports.

More in detail, a specific dataset must be assembled in order to fine-tune the preexisting BERT model for the purpose of hobby classification. This is done in two steps: first of all, a list of 666 hobbies is downloaded from Kaggle (2022) and elementwise associated with each of the aforementioned twelve classes, thus creating an augmented dataset by resorting to the nlpaug Python library (2022).

Once created, the labeled dataset is split into training, validation and test datasets in order to properly fine-tune the model. The pre-trained BERT model in (https://colab.research.google.com/drive/1ZQvuAVwA3IjybezQOXnrXMGAnMyZRuPU, 2022) was fine-tuned for 4 training epochs, thus reaching a validation accuracy of about 95%. Note that the initial validation accuracy is greater than the training accuracy, due to the presence of a dropout layer, used during the training phase with an anti-overfitting purpose. However, we can see from Figure 2 that the validation and training accuracy eventually end

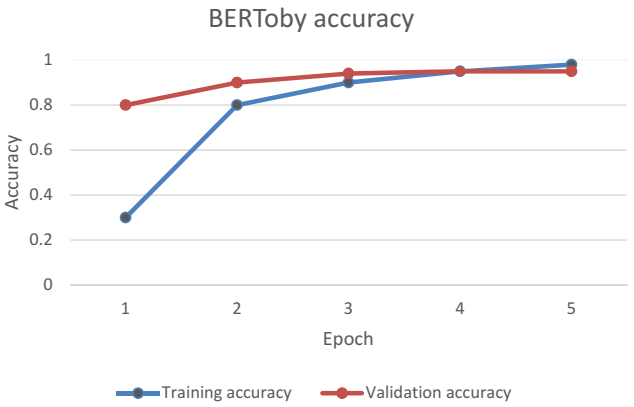

Figure 2. Fine-tuning BERT in order to obtain BERToby.

up at a similar satisfactorily high value at the end of training, implying that we eventually get to the condition of a good fit.

Once BERTOby has been properly trained to correctly classify the input hobbies, then, for each class in a static way we associate a series of soft skills. As a result we can then use BERTOby for inference purposes.

More precisely, the input hobby is first processed by the BERToby classifier, thus detecting one of the 12 classes as the best fitting prediction. Then, the set of skills corresponding to the predicted class is inferred and finally associated to the employee.

JoBERT

Analogously, in the case of jobs, we need to classify jobs into a series of predefined classes. This is done by training a JoBERT-based machine learning model. Afterwards, we perform static association of soft skills to classes.

First of all, we selected a wide range of occupations from the ESCO database, thus identifying 30 occupational classes, namely: administrative and commercial managers; assemblers, building and related trade workers (excluding electricians); business and administration associate professionals; business and administration professionals; chief executive, senior officials and legislators; customer services clerks, drivers and mobile plant operators; electrical and electronics trade workers, food processing, woodworking, garment and other craft and related trade workers; handicraft and printing workers; health associate professionals; health professionals; hospitality, retail and other service managers; information and communications technicians, information and communications technology professionals; laborers in mining, construction, manufacturing and transport; legal, social and cultural professionals; legal, social, cultural and related associate professionals; market-oriented skilled agricultural workers; metal, machinery and related trade workers; numerical and material recording clerks; personal service workers; production

and specialized services managers; protective service workers; sales workers; science and engineering associate professionals; science and engineering professionals; stationary plant and machine operators; teaching professionals.

As already did in the case of BERTOby, proper fine tuning of the pre-trained BERT model in for the job classification task needs a specific dataset to be built. This is done by automatically downloading the ESCO occupation-skill mapping for all the above mentioned occupational classes by climbing back the ISCO categorization according to the ISCO taxonomy (2022). In this respect, some classes were dropped because of low cardinality, absence of relevant job positions in those categories (e.g., street and related sales and service workers) within the considered company. Once created, the labeled dataset is split into training, validation and test dataset in order to properly fine-tune the model. The model was fine-tuned for five training epochs reaching a validation accuracy of about 88%. Note that, even in this case, the initial validation accuracy is greater than the training accuracy. Due to the presence of a dropout layer, used during the training phase with an anti-overfitting purpose. However, we can see from Figure 3 that the validation and training accuracy eventually reach a similar high value at the end of training, meaning that we eventually get to the condition of a good fit.

Once JoBERT has been properly trained to correctly classify the input job positions, the final step is the one that associates each class to a set of skills. As anticipated, a ESCO standard association between occupations and skills is used in this respect. More in detail, the category-skill association has been refined by picking only the most representative skills for each category according to the number of skill occurrences in the group.

More precisely, the input job position is first processed by the JoBERT classifier which identifies one of the 30 classes as the best fitting prediction. Then, the set of skills corresponding to the predicted class is inferred and finally associated to the employee.

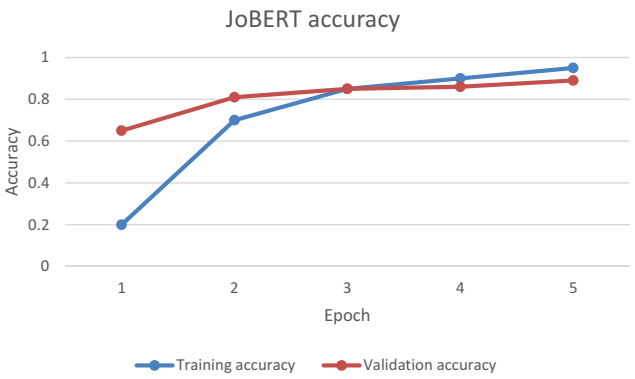

Figure 3. Fine-tuning BERT in order to obtain JoBERT.

Numerical Results

The authors used a Cloudera node from Cloudera Data Platform (CDP) as simulation environment for testing the semantic skill matching (NER) algorithm as well as the skill inference algorithm. The Cloudera node was hosted by a virtual machine equipped with 32-core CPU, no GPU, and 256-GB RAM.

Semantic Skill Matching (NER) Results

In the absence of any standard large open-source dataset for job recommendation task, this paper uses a dataset consisting of 200 resumes in Europass format collected by ELIS Innovation Hub and enriched with 200 CVs created from as many LinkedIn profiles through the Linkedin Resume Builder. The objective of the authors was to recreate as faithfully as possible a dataset similar to the dataset used in (Maheshwary and Misra, 2018), so that results could be compared.

Table 1 shows the accuracy performance – at first, third and fifth recommendation, respectively – of the overall semantic skill matching (NER), composed of the three parallel streams (PoS, TF-IDF, W2 V), by comparison with the accuracy performance of the JD (Job Description) matching algorithm proposed by Gugnani and Misra (2020).

Also, Figure 4, by plotting the algorithm runtime versus the dimensionality of the reference skill dictionary (namely, the number of skill phrases appearing in $\mathcal{D}(t)$), shows that both JD matching (Gugnani and Misra 2020) and semantic skill matching (NER) outperform a simple exhaustive search algorithm as the one proposed by Petrican et al. (2017). The blue line denotes the behavior of the exhaustive search algorithm, the green line the

Table 1. Accuracy performance for 400 candidate profiles at 1st, 3rd and 5th recommendation, respectively.

Algorithm	*@1*	*@3*	*@ 5*
Semantic skill matching (NER)	0.89	0.98	1
JD matching (Gugnani and Misra (2020))	0.84	0.95	0.98

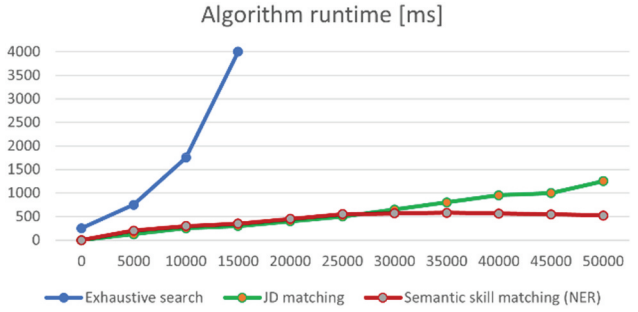

Figure 4. Algorithm runtime [ms] vs. dimensionality of $\mathcal{D}(t)$.

JD matching one, and the red line the semantic skill matching one proposed in this work.

Moreover, semantic skill matching (NER) exhibits a stabilizing behavior by contrast with JD matching, scoring a settling time of approximately $550ms$ which corresponds to a dimensionality of $\mathcal{D}(t)$ of approximately $2,300$ entries. In other words, the system dynamics is such that at the above-mentioned settling time the stabilizing behavior implying $X(t) \rightarrow R(t)$ has finally kicked in.

Skill Inference Results

For BERToby, we report in Table 2 the performance metrics achieved with the above mentioned experimental setting. We can see that, when it comes to using the trained model for inference purposes, in the worst case (that is, the case of the "extreme sports" class) there is an amount of misclassified samples up to 12%. Consequently, the F1-score measure in the worst case is 88%.

For JoBERT, we report in Table 3 the performance metrics achieved. Perfect precision classes like "Chief Executives" and "Sales Workers" exhibit perfect recall, meaning an overall perfect classification accuracy (i.e., neither false positives nor false negatives). Consequently, we cannot say that our classification model is biased toward these classes. The worst-case harmonic mean of precision and recall, that is, F1-score is 77% and corresponds to the class "Legal, Social and Cultural Professional and Food Processing, Woodworking, Garment Workers:" in this case misclassification is due to the inner imbalance of the original ESCO-ISCO mappings.

Table 2. Bertoby performance metrics.

Class	Precision	Recall	F1-score
Art & craft	1	0.92	0.96
Collecting	1	0.84	0.91
Extreme sports	0.88	0.88	0.88
Food & drinks	0.89	1	0.94
Games	0.92	0.96	0.94
Individual sports	0.88	0.92	0.90
Model & electronics	0.96	1	0.98
Music	1	1	1
Performing arts	0.96	0.96	0.96
Pets	1	1	1
Spiritual & mental	0.96	0.92	0.94
Team sports	0.96	1	0.98
Accuracy			**0.95**
Macro average	**0.95**	**0.95**	**0.95**
Weighted average	**0.95**	**0.95**	**0.95**

Table 3. JoBERT performance metrics.

Class	Precision	Recall	F1-score
Administrative and commercial managers	0.74	0.96	0.83
Assemblers	0.87	1	0.93
Building and related trade workers	0.96	0.92	0.94
Business and administration associate professionals	0.89	0.68	0.78
Business and administration professionals	0.79	0.78	0.78
Chief executives, senior officials and legislators	1	1	1
Customer service clerks	0.96	1	0.98
Drivers and mobile plant operators	0.92	0.89	0.91
Electrical and electronics trade workers	0.86	0.96	0.91
Food processing, woodworking, garment and other craft and related trade workers	0.80	0.74	0.77
Handicraft and printing workers	0.96	0.89	0.92
Health associate professionals	0.88	0.78	0.82
Hospitality, retail and other service managers	0.93	0.96	0.94
Information and communication technicians	0.96	1	0.98
Labourers in mining, construction, manufacturing and transport	0.93	1	0.96
Legal, social and cultural prfessionals	0.77	0.77	0.77
Market-oriented skilled agricultural workers	0.96	1	0.98
Numerical and material recording clerks	0.96	0.96	0.96
Personal service workers	0.92	0.85	0.88
Production and specialized service managers	0.79	0.88	0.84
Protective service workers	0.89	0.96	0.93
Sales workers	1	1	1
Science and engineering associate professionals	0.84	0.74	0.78
Science and engineering professionals	0.88	0.81	0.79
Stationary plant and machine operators	0.78	0.81	0.79
Teaching professionals	0.96	0.96	0.96
Accuracy			**0.88**
Macro average	**0.88**	**0.88**	**0.88**
Weighted average	**0.88**	**0.88**	**0.88**

Conclusions

This paper proposes a heuristic algorithm aimed at solving two tasks: namely, (i) performing semantic matching between, on the one hand, a data source containing various texts related to the professional profile of the employee, and, on the other hand, the O*NET and ESCO competence dictionaries, to obtain a homogeneous catalog, and (ii), relative to the company's full scale, inferring the employee's soft skills with respect to his/her own declaration of interests, work experience, certifications, etc., given his/her curriculum vitae and/or LinkedIn profile.

The algorithmic rules we used to construct our proposed NER algorithm were empirically helpful in reducing the training complexity of the deep learning models. We note, however, that these fixed rules may restrict the variance of our skill association machinery. Future enhancements of the proposed solutions may explore learning the fixed parameters with gradient-based methods taking into account the specific downstream tasks we faced.

Due to the non-availability of standard large open-source dataset for the considered tasks, we evaluated the proposed algorithm on a dataset arranged with the same setting as in (Maheshwary and Misra, 2018). The empirical

results obtained on such a dataset demonstrate that the proposed approach yields improved performance results by comparison with baseline methods available in the literature (Petrican et al. 2017; Gugnani and Misra 2020).

Future work will be aimed at validating the proposed approach on an even larger dataset, at using skill graphs to infer skill-gap in candidate profiles for job recommendation purposes and at improving the efficiency of the proposed algorithmic framework while keeping the same computational resources.

Acknowledgements

The authors thank Ing. A Valli for the fruitful discussions.

Disclosure statement

The work presented in this paper was carried out while Ing. F. Kieffer and Ing. V. Paduano were with ELIS Innovation Hub and does not reflect the results of any activity carried out at Generali Assicurazioni and Storm Reply – CNA, to which the two above-mentioned authors are currently affiliated, respectively. No potential conflict of interest was reported by the author(s).

Funding

This research was supported by 'Consel–Consorzio ELIS per la formazione professionale superiore' within the framework of the 2020 Joint Research Project initiative funded by Enel S.p.A. This research was also partially supported by the European Commission in the framework of the Erasmus+ I4EU project (Key competences for a European model of Industry 4.0) under Grant Agreement no. 2019-1-FR01-KA202-062965.

References

Al-Otaibi, S. T., and M. Ykhlef. 2012. A survey of job recommender systems. *International Journal of the Physical Sciences* 7 (29):5127–42. doi:10.5897/IJPS12.482.

Bastian, M., M. Hayes, W. Vaughan, S. Shah, P. Skomoroch, H. Kim, S. Uryasev, and C. Lloyd. Linkedin skills: Large-scale topic extraction and inference. In *Proceedings of the 8th ACM Conference on Recommender Systems* Foster City, Silicon Valley, CA, USA, pp. 1–8, ACM, 2014.

Battilotti S and d'Angelo M. (2019). Stochastic output delay identification of discrete-time Gaussian systems. Automatica, 109 108499 10.1016/j.automatica.2019.108499

BERT encoder (pre-trained), https://colab.research.google.com/drive/1ZQvuAVwA3IjybezQOXnrXMGAnMyZRuPU, last accessed on June 8th , 2022.

Berzak, Y., J. Kenney, C. Spadine, J. X. Wang, L. Lam, K. S. Mori, S. Garza, and B. Katz. Universal dependencies for learner English. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 737–46, Berlin, Germany. Association for Computational Linguistics, 2016.

Bizer, C., R. Heese, M. Mochol, R. Oldakowski, R. Tolksdorf, and R. Eckstein. 2005. *The impact of semantic web technologies on job recruitment processes*, 1367–81. Heidelberg: Physica-Verlag HD.

Cali, A., D. Calvanese, S. Colucci, T. D. Noia, T. Di, N. Francesco, F. M. Donini, and U. D. Tuscia. A description logic based approach for matching user profiles. In *Proc. of the 8th Int. Conf. on Knowledge Based Intelligent Information & Engineering Systems (KES 2004)* Whistler, British Columbia, Canada, volume 3215 of Lecture Notes in Artificial Intelligence, 2004, pp. 187–95.

Cloudera data platform (CDP) datasheet, https://it.cloudera.com/content/dam/www/marketing/resources/datasheets/cloudera-data-platform-datasheet.pdf?daqp=true, last accessed on June 8th, 2022.

Colucci, S., T. Di Noia, E. Di Sciascio, F. Donini, M. Mongiello, and M. Mottola. 2003. A formal approach to ontology-based semantic match of skills descriptions. *Journal UCS* 9 (12):1437–54.

Devlin, J., M. W. Chang, K. Lee, and K. Toutanova. June 2-7, 2019. *BERT: Pre-training of deep bidirectional transformers for language understanding*. Minneapolis (MN):*NAACL-HLT 2019*.

Discover a Hobby portal, https://www.discoverahobby.com/, last accessed on June 8th, 2022.

ESCO dictionary, https://esco.ec.europa.eu/en, last accessed on June 8th , 2022.

Faliagka, E., K. Ramantas, A. Tsakalidis, and G. Tzimas. Application of machine learning algorithms to an online recruitment system. In *Proc. of the Seventh International Conference on Internet and Web Applications and Services* Stuttgart, Germany, 2012.

Fazel-Zarandi, M., and M. S. Fox. 2010. *Reasoning about skills and competencies*, 372–79. Berlin Heidelberg: Springer.

FuzzyWuzzy Python library, https://pypi.org/project/fuzzywuzzy/, last accessed on June 8th, 2022.

Golec, A., and E. Kahya. 2007. A fuzzy model for competency-based employee evaluation and selection. *Computers & Industrial Engineering* 52 (1):143–61. doi:10.1016/j.cie.2006.11.004.

Gugnani, A., and H. Misra, Implicit skills extraction using document embedding and its use in job recommendation. In *Proceedings of the 32nd Innovative Applications of Artificial Intelligence Conference (IAAI-20)* New York Hilton Midtown, New York, New York, USA, 2020, 34(08), 13286–93.

Hassan, F. M., I. Ghani, M. Faheem, and A. A. Hajji. August. 2012. Ontology matching approaches for e-recruitment. *International Journal of Computer Applications* 51(2):39–45. doi:10.5120/8018-0917.

Hugging Face Transformers APIs, https://huggingface.co/, last accessed on June 8th , 2022.

ISCO (International Standard Classification of Occupations) taxonomy, https://www.ilo.org/public/english/bureau/stat/isco/index.htm, last accessed on June 8th, 2022.

Joshi, M., V. C. Bhavsar, and H. Boley. 2010. Knowledge representation in matchmaking applications. *Advanced Knowledge Based Systems Model Applications & Research*.

Kaggle–, list of hobbies from, https://www.kaggle.com/datasets/muhadel/hobbies, last accessed on June 8th , 2022.

Kivimaki, I., A. Panchenko, A. Dessy, D. Verdegem, P. Francq, H. Bersini, and M. Saerens. A graph-based approach to skill extraction from text. In *Proceedings of TextGraphs-8 Graph-based Methods for Natural Language Processing* Seattle, Washington, USA, pp. 79–87, 2015.

Li, J., B. Chiu, S. Feng, and H. Wang. Few-shot named entity recognition via meta-learning. *IEEE Transactions on Knowledge and Data Engineering* 34(9):1 Sept. 2022 4245–56. doi:10.1109/TKDE.2020.3038670.

Lin, Y., H. Lei, P. Clement Addo, and X. Li. Machine learned resume-job matching solution. *ArXiv e-prints*, 2016.

Li, J., S. Shang, and L. Chen. Sept. 2021. Domain generalization for named entity boundary detection via metalearning. *IEEE Transactions on Neural Networks and Learning Systems* 32 (9):3819–30. doi:10.1109/TNNLS.2020.3015912.

Liu, K., X. Shi, A. Kumar, L. Zhu, and P. Natarajan. Temporal learning and sequence modeling for a job recommender system. In *Proceedings of the Recommender Systems Challenge*, 7. ACM, 2016.

Lv, H., and B. Zhu. Skill ontology-based semantic model and its matching algorithm. In *Proc. of the 2006 7th International Conference on Computer-Aided Industrial Design and Conceptual Design*, Nov. 2006, pp. 1–4.

Maheshwary, S., and H. Misra. 2018. Matching resumes to jobs via deep siamese network. In *Companion of the The Web Conference 2018 on The Web Conference 2018* Lyon, France, 87–88. International World Wide Web Conferences Steering Committee.

Malinowski, J., T. Keim, O. Wendt, and T. Weitzel. 2006. Matching people and jobs: A bilateral recommendation approach. *HICSS*.

Mohaghegh, S., and R. R. Mohammad. An ontology driven matchmaking process. *TSI PRESS SERIES vol. 16*, pp. 248–53, 2004.nlpaug Python library, https://nlpaug.readthedocs.io/en/latest/, last accessed on June 8th, 2022.

O*NET dictionary, https://www.onetonline.org/, last accessed on June 8th , 2022.

Paparrizos, I. K., B. B. Cambazoglu, and A. Gionis. 2011. Machine learned job recommendation. *RecSys*.

Petrican, T., C. Stan, M. Antal, I. Salomie, T. Cioara, and I. Anghel. Ontology-based skill matching algorithms. In *Proceedings of the 2017 13th IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)* Cluj-Napoca, Romania, 2017.

Python package to accelerate the sparse matrix multiplication and top-n similarity selection, https://github.com/ing-bank/sparse_dot_topn, last accessed on June 8th , 2022.

Rajamaran, A., and J. D. Ullman. 2011. *Data mining*, mining of massive datasets, 1–17. Cambridge University Press.

Ricciardi Celsi L, Bonghi R, Monaco S and Normand-Cyrot D. (2015). On the Exact Steering of Finite Sampled Nonlinear Dynamics with Input Delays. IFAC-PapersOnLine, 48(11), 674–679. 10.1016/j.ifacol.2015.09.265

Salton, G., and C. Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information Processing & Management* 24 (5):513–23. doi:10.1016/0306-4573(88)90021-0.

Singh, S., and C. Sathi. *Advanced data analytics. machine learning for text (TFIDF and NGrams)*. Lecture notes, UCI, 2018

Zhao, M., F. Javed, F. Jacob, and M. McNair. Skill: A system for skill identification and normalization. In *AAAI*, pp. 4012–18, 2015.

Appendix

The similarity score introduced in Section 3.1 is defined below as follows.

$$s := \max_{p,q} s_{p,q} \quad p \in \{1, \ldots, n\}, \quad q \in \{1, \ldots, n\})$$

where $\hat{s}_A, \hat{s}_B$ and $\hat{s}_C$ are used to denote the partial scores of Streams A, B, and C, respectively; $w_A, w_B,$ and w_C are instead the weights associated to each of the three partial scores, and all fall within the range $[0, 1]$, while p_B and p_C are two prefixed parametric penalties associated to a missing detection from the Stream B and Stream C, respectively. For both Stream B and Stream C a confidence threshold τ_C is established to determine a missing detection (i.e., a zero score). More precisely, if the score reported by either Stream B or Stream C is below the associated confidence threshold τ_C, the skill association is discarded, that is, a zero score for the considered stream is assigned.