Big Data Research 39 (2025) 100509

Contents lists available at ScienceDirect

Big Data Research

journal homepage: www.elsevier.com/locate/bdr

A novel approach for job matching and skill recommendation using transformers and the O*NET database

Rubén Alonso [a,d], Danilo Dessí [b], Antonello Meloni [c], Diego Reforgiato Recupero [a,c,*]

[a] R2M Solution s.r.l., ICT Division, Polo Tecnologico di Pavia, Pavia, Italy
[b] Department of Computer Science, College of Computing and Informatics, University of Sharjah, Sharjah, United Arab Emirates
[c] Department of Mathematics and Computer Science, University of Cagliari, Cagliari, Italy
[d] Programa de Doctorado, Centro de Automática y Robótica, CSIC-Universidad Politécnica de Madrid, Madrid, Spain

ARTICLE INFO

Keywords:
Information extraction
Transformers
Online enrolling process
Natural language processing
Course recommendation

ABSTRACT

Today we have tons of information posted on the web every day regarding job supply and demand which has heavily affected the job market. The online enrolling process has thus become efficient for applicants as it allows them to present their resumes using the Internet and, as such, simultaneously to numerous organizations. Online systems such as Monster.com, OfferZen, and LinkedIn contain millions of job offers and resumes of potential candidates leaving to companies with the hard task to face an enormous amount of data to manage to select the most suitable applicant. The task of assessing the resumes of candidates and providing automatic recommendations on which one suits a particular position best has, therefore, become essential to speed up the hiring process. Similarly, it is important to help applicants to quickly find a job appropriate to their skills and provide recommendations about what they need to master to become eligible for certain jobs. Our approach lies in this context and proposes a new method to identify skills from candidates' resumes and match resumes with job descriptions. We employed the O*NET database entities related to different skills and abilities required by different jobs; moreover, we leveraged deep learning technologies to compute the semantic similarity between O*NET entities and part of text extracted from candidates' resumes. The ultimate goal is to identify the most suitable job for a certain resume according to the information there contained. We have defined two scenarios: i) given a resume, identify the top O*NET occupations with the highest match with the resume, ii) given a candidate's resume and a set of job descriptions, identify which one of the input jobs is the most suitable for the candidate. The evaluation that has been carried out indicates that the proposed approach outperforms the baselines in the two scenarios. Finally, we provide a use case for candidates where it is possible to recommend courses with the goal to fill certain skills and make them qualified for a certain job.

1. Introduction

The job market has been heavily influenced by the tons of information that are posted on the Web every day regarding job supply and demand. The online enrolling process has become efficient for applicants as it allows them to present their resumes using the Internet and, as such, simultaneously to numerous organizations (companies or research centers). One problem that arises from the application to multiple online systems is that there is no universal standard format to adopt when filling resume information (although systems such as Europass[1] or LinkedIn[2] have some methods to automatically generate structured profiles). It follows that resumes are all very different from each other in terms of structure, design, and format, hindering an efficient and fast analysis from people working within human resources divisions of

* Corresponding author at: Department of Mathematics and Computer Science, University of Cagliari, Cagliari, Italy.
E-mail addresses: ruben.alonso@r2msolution.com (R. Alonso), ddessi@sharjah.ac.ae (D. Dessí), antonello.meloni@unica.it (A. Meloni), diego.reforgiato@unica.it (D. Reforgiato Recupero).

[1] https://europa.eu/europass/en.
[2] http://www.linkedin.com.

https://doi.org/10.1016/j.bdr.2025.100509
Received 18 August 2022; Received in revised form 22 January 2025; Accepted 6 February 2025

Available online 7 February 2025

the different organizations. Studies on job-resume matching have highlighted the challenges posed by the lack of standardization in resumes, which might lead to inconsistencies in candidate evaluation [1,2].

The simplicity and speed of sending online applications hide one more problem related to resume analysis. When an organization posts a certain job, there is a number of required job-specific and mandatory skills. However, several applicants who do not satisfy them keep sending their resume "just to give it a try". This happens because there are no risks in sending non-eligible applications (although, in certain situations, they might end up in some black list of low-quality applications[3]).

Online systems such as Monster.com[4], OfferZen[5] and LinkedIn contain millions of job offers and resumes of potential candidates leaving to companies the hard task to face enormous amounts of data with the goal of selecting the most suitable applicant for their needs[6]. Quickly assessing the resumes of candidates and providing automatic recommendations on which one suits best a particular position have, therefore, become essential to speed up the screening and hiring process for companies [3–5]. Similarly, it is important to help applicants to quickly find a job appropriate to their skills and provide recommendations about what they need to master to become eligible for their desired jobs [6].

The first problem to solve has been the text extraction from resumes represented by non-textual documents (e.g., images or PDF files). This problem has already been discussed and faced in other domains as well leading to the creation of systems with very accurate extraction of body text [7–9]. For instance, the study presented in [10] discusses methods for extracting information from documents like resumes, highlighting advancements in this area.

Once the body text has been extracted, the next challenge is to identify words or compound expressions that match the candidate's skills peculiar to the underlying job. One resource that has been created to support this problem is the O*NET database [11][7]. It contains a rich set of variables that describe work and worker characteristics, including skill requirements. It also contains hundreds of descriptors on almost a thousand occupations related to the United States. Thus, the O*NET database represents a valuable resource to identify skills in a candidate's resume and associate them with related jobs. Using O*NET, the challenge to find words indicating job skills is therefore narrowed to the identification of words or expressions within a certain resume which correspond to those described within the O*NET database. For example, a resume might contain the word *programming* which, in O*NET, corresponds to a skill for several jobs. Simple syntactic rules (i.e., string matching, approximate string matching using the Levenshtein distance, etc.) may not be enough; in fact, the presence of word synonyms and the different forms a sentence can be formulated hinder their use. As a further example, in O*NET the description associated with the technology skill *Google Drive*, which, in turn, is one of the skills required for the job *Marketing Managers*, is *Cloud-based data access and sharing software*. Hardly a resume will contain the same formulation; more likely it will include different expressions conveying the same meaning such as *use of cloud-based software*, or *use of sharing applications* to name a few.

With the advancements of Machine Learning technologies, we have today very powerful methods able to compute the semantic similarity of two pieces of text (documents, paragraphs, expressions). Furthermore, the evolution of Deep Learning transformers [12,13], enabled by the huge amount of publicly available annotated datasets and disposal of powerful graphical processing units (GPU) to run neural networks in parallel, have provided cutting-edge solutions for a great number of problems included those revolved around the Semantic domain [14]. Transformers are able to efficiently solve several tasks such as sequence classification, question answering, language modeling, text generation, named entity recognition, and most of all, semantic similarity between two texts. Therefore, thanks to the advancements in Natural Language Processing (NLP), Semantic Web, Artificial Intelligence, Deep Learning, and Information Extraction, we have a plethora of technologies (software and hardware) to efficiently extract and identify valuable and relevant information from candidates' resumes. For instance, recent work has proposed the use of Large Language Models (LLMs) models to produce personalized job descriptions given input CVs [15], leverages semantic-enhanced transformers to parse job descriptions and align them with user skills, improving the accuracy of recommendations [16], handling job recommendation as resource allocation tasks to study the fairness of machine learning-based models [17]. Despite these advancements, LLM- and deep learning-based recommenders face a significant challenge: they frequently introduce biases and fail to provide clear, interpretable justifications for why a particular CV is matched with a given job, raising concerns about fairness and accountability in the recommendation process.

Our paper lies in this context and proposes a new approach to identify skills from candidates' resumes and job postings and match the two of them. We consider the O*NET database information related to technology skills, skills, knowledge, abilities, work activities, task ratings, and tools used. Then, given the resume of a candidate, we run different NLP tools to extract information that is then matched against the O*NET entities and returns the most suitable job for the underlying resume. We consider two scenarios, one useful for the companies needed to screen several resumes and another one useful for candidates when looking for a job among several job postings:

- First scenario: a resume is provided as input. Then the goal is to identify the O*NET occupation with the highest match with the resume;
- Second scenario: a resume and a list of job postings are given in input. The goal is to find which of the input job postings matches the most candidate's skills present in the resume.

The contributions of our paper are therefore the following:

- we employ Deep Learning transformers to identify the pieces of text in a resume with high semantic similarity with the entries of corresponding O*NET entities;
- using ad-hoc metrics we identify the most related O*NET job to the resume of the underlying candidate;
- we provide two complementary scenarios where we applied our approach;
- for the first scenario, we tested our approach on a dataset of 105 resumes and outperformed two baseline methods we have defined; for such a purpose, three independent experts have annotated the dataset which we have publicly released;
- for the second scenario, we tested our approach on a dataset of 100 resumes where each of them was associated with 10 different job postings (the resume was qualified for only one of them); our approach outperformed two more baselines we have defined; again, for such a purpose, three independent experts have annotated the dataset which we have publicly released;
- we provide a use case where we apply our approach to recommend courses and lectures to candidates with the goal of acquiring certain skills and thus resulting qualified for certain jobs they had targeted;
- we release the source code of our approach for both the scenarios and the recommendation task in a public repository[8];

[3] https://www.monster.com/career-advice/article/things-that-will-blacklist-you-from-job.

[4] http://www.monster.com.

[5] https://www.offerzen.com/.

[6] https://economicgraph.linkedin.com/resources/linkedin-workforce-report-april-2022.

[7] https://www.onetcenter.org/database.html.

[8] Scenarios source code: https://gitlab.com/hri_lab1/using-transformers-and-o-net-to-match-jobs-to-applicants-resumes.

- we provide an interactive demo[9] and its source code[10] that implements our proposed approach to match applicants' resumes with the O*NET database jobs to measure the eligibility of a user for a given job.

The remainder of this paper is organized as follows. Section 2 discusses related works on resume extraction and job recommendation. Section 3 details the tools we have used in this paper as well as the O*NET database. The proposed approach is presented in Section 4 whereas Section 5 introduces the scenarios we have considered. Section 6 illustrates the evaluation we have carried out for the two scenarios. A use case we propose in the paper which can leverage the proposed approach to recommend skills to job seekers is shown in Section 7. Finally, Section 8 ends the paper with conclusions, limitations, and future works where we are headed.

2. Related work

The last two decades have seen the development of online recruiting platforms, a topic that has recently acquired increasing attention. Authors in [18,19] presented two surveys of existing recommendation approaches that have been proposed to create recommendation systems for job seekers and recruiters. On the one hand, extracting features from resumes and from job postings has always been challenging. In particular, user profiling deals with acquiring, extracting, and representing the features of users [20]. On the other hand, job profiling is a representation of job descriptions and their requirements. Usually, job descriptions come in unstructured text with no attribute names with well-defined values. It follows that the skill set for a particular job includes skills with a Boolean value: True if the skill is required for that job, False otherwise.

A kind of approach that achieved great success to recommend jobs is collaborative filtering. It is based on the assumption that if users A and B have similar behaviors they will rate other items similarly [21–23]. Authors in [23] developed a job recommendation system using a model-based collaborative algorithm with clustering algorithms. The Latent Semantic Analysis (LSA) and Singular Value Decomposition (SVD) have been adopted to create a lower-rank matrix with information about skills and positions. Moreover, the inverse cosine similarity was employed as distance to perform the agglomerative clustering to create clusters of positions. Authors in [22] created a system for job recommendation by making clusters of users that are based on skills extracted from different websites. The Euclidean distance has been leveraged as a measure of similarity between the skills of users. Then, identified skills with low occurrence were removed from the list and a classification using Naive Bayes was employed to rank the final set of recommendations for a user.

The Labor Market Explorer interactive dashboard for job seekers has been presented by authors in [24]. It was built with a careful user-centered design processing where both job seekers and job mediators were involved so that the matching process between jobs and job seekers could be optimized. The dashboard enables an exploration of the job market in a personalized way based on the skills and competencies of the applicants. Efforts related to career exploration and the detection of training needs have also been and are being carried out. For example, the STAR[11] project has the goal to design new technologies to enable the deployment of standard-based secure, safe, reliable, and trusted human-centric AI systems in manufacturing environment. There, the Workers' Training Platform is being developed. This platform allows workers to self-assess themselves and detect training needs related to skills or knowledge while offering them training recommendations for those skills. All in an anonymous way and based on public occupation databases such as O*NET or ESCO[12].

Other authors in [25] created a system to suggest jobs based on users' profiles. Users and jobs have been treated as text documents, and a model that incorporates job transitions trained on the career progressions of a set of users has been adopted. The authors also showed that combining career transitions with cosine similarity outperforms the system using just career transitions. The evaluation proving the statements above has been carried out on a dataset of 2,400 Linkedin users with the task of predicting users' current positions by looking at their profiles and their jobs history.

The works illustrated above do not leverage a widely recognized taxonomy of skills or other similar entities. Sometimes the set of the considered skills is too high and approaches to reduce the dimensional space need to be adopted with the drawback of losing precision. Differently from the previous approaches, we use the O*NET taxonomy by including not just skills but several other entities of a different kind (i.e., knowledge, abilities, technology skills, etc.). O*NET is one of the main occupational databases, and is almost a reference for occupation analysis and worker requirements. It is the primary source of occupational information in the United States and it includes regularly updated occupational characteristics based on questionnaires made to several hundred workers. In doing so, we inject into our approach cognitive, interpersonal, and physical skill knowledge representing worker and job requirements coming from a large number of workers and companies, thus making the person-job matching more robust. Finally, our work differs from previous studies [21–25] because it does not rely on previous applicants for a specific job position like in collaborative filtering approaches. Therefore, each resume is used as it is and is matched against jobs thus allowing us to avoid biases as well as the cold start problem [26]. Last but not least, to match each of these entities with applicants' resumes and job postings we make use of sentence transformers thus leveraging the semantics of each word, sentence, and context. This differs from the existing studies which often make use of only well-defined attributes mentioned in the applicants' resumes or job postings, thus making it possible for our approach to deal with the complexity of the natural language text contained in applicants' resumes.

3. Task definition and the used material

In this section, we provide a formal definition of the task addressed in this work, alongside details about the transformers model and the O*NET database leveraged in the proposed approach.

3.1. Task definition

The objective of our approach is to develop a system that automatically identifies and matches relevant skills and qualifications between candidates' resumes and job postings. Formally, given:

- **Resume Information (R):** A structured or semi-structured textual representation of a candidate's skills, experiences, education, and qualifications.
- **Job Information (J):** A set of job classes derived from O*NET, comprising multiple entities such as technology skills, general skills, knowledge, abilities, work activities, task ratings, and tools used.

The task is defined as a mapping function $f : R \rightarrow J$, where for each resume $r \in R$, the function f returns a ranked list of job classes $\{j_1, j_2, ..., j_n\} \subset J$, ordered by their relevance to the candidate's profile.

The relevance score is calculated by leveraging semantic similarity to align the extracted elements from the resume with the O*NET entities corresponding to each job. For each matched element, a job-specific

[9] Demo: http://192.167.149.11:8000.

[10] Demo source code: https://gitlab.com/hri_lab1/onet-db26-transformers-demo.

[11] https://www.star-ai.eu.

[12] https://ec.europa.eu/esco/.

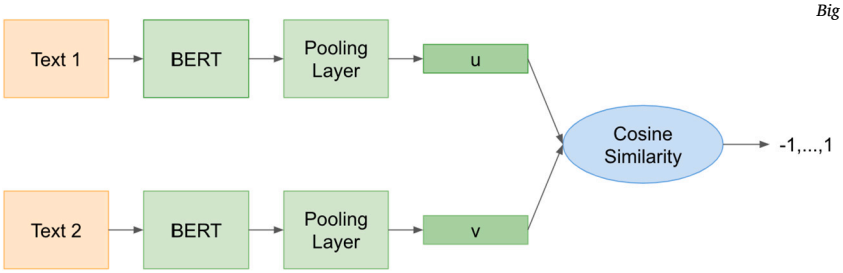


Fig. 1. The figure illustrates the structure of a Siamese network designed for text comparison. Each branch of the network consists of a text input processed through a BERT model that encodes the text into vectorial representations, and a pooling layer that aggregates the token-level features into a fixed-size vector representation capturing the most salient information. The resulting vectors from both branches u and v are then compared using cosine similarity to measure their similarity. .

score derived from O*NET is assigned across various categories, including "skills", "knowledge", "abilities", "work activities", and "tasks". Additionally, conventional scores are applied for "technology skills" and "tools used".

The approach operates in two specific scenarios:

1. **Resume-to-Job Matching (R2J):** Assisting companies in screening multiple resumes to identify the most suitable candidates for a specific job posting.
2. **Job-to-Resume Matching (J2R):** Helping candidates find the most relevant job postings based on their resumes.

By leveraging NLP advancements and semantic similarity computation, the proposed system aims to address challenges such as entity disambiguation, contextual understanding of skills and qualifications, and alignment between heterogeneous textual data from resumes and job descriptions.

3.2. Sentence transformers

SentenceTransformers[13] [27] is the state-of-the-art Python framework for text embeddings generation. The framework leverages models that are built on top of the original BERT model [28] or its further developments such as RoBERTa [29], MPNet [30], and ALBERT [31]. It provides transformer models that use siamese and triplet network structures to transform sentences into embeddings. Then, these can be compared by using metrics (e.g., the cosine similarity) to perform tasks such as information retrieval, clustering, and semantic search. The first models were trained on Natural Language Inference (NLI) datasets [32,33] and successfully became the state of the art for solving Semantic Textual Similarity (STS) tasks.

Today, *SentenceTransformers* pre-trained models are built as extensions of Huggingface Transformers models[14] by applying pooling layers within siamese structures. The reader can observe an example of a common siamese structure with two BERT models to compute the cosine similarity between two texts in Fig. 1. The pool of *SentenceTransformers* models can be found at https://huggingface.co/models.

Within the proposed system, we have chosen to embed the *all-mpnet-base-v2* model, which achieves a $\rho = 0.69$ on encoding sentences over 14 diverse tasks, and a $\rho = 0.57$ on 6 diverse tasks for semantic search such as encoding of queries and questions. According to the official Sentence Transformers documentation[15], this model is specifically noted for achieving higher performance compared to a range of other pre-trained models, including standard BERT-based and RoBERTa-based architectures. The documentation emphasizes that Sentence Transformers models are optimized for sentence-level tasks, such as semantic similarity, making them particularly well-suited for our application, which involves comparing resumes, job descriptions, and O*NET entities. In addition, our use of Sentence Transformers aligns with the nature of our task, where capturing nuanced semantic similarity between textual data is critical. These models are specifically designed to deliver superior results for tasks involving sentence embeddings and pairwise comparisons, which are central to our approach. The reader notices that ρ corresponds to the Spearman rank correlation coefficient [34] between the ranking of sentence pairs using the cosine similarity as score and the gold standard ranking for various semantic textual similarity tasks. The Spearman rank correlation coefficient gives a score in the continuous range $[-1, 1]$ where a value of 1 indicates a perfect correlation between the two rankings, a value of 0 indicates a weak correlation, and a value of -1 means a perfect negative correlation.

The *all-mpnet-base-v2* model has a size of 420 Mbytes, a Max Sequence Length of 384 tokens, and an embedding speed of $2,800$ sentences/sec on a V100 GPU. The generated embeddings are 768-dimensional vectors. The suitable score functions are the dot product, the cosine similarity, and the Euclidean distance.

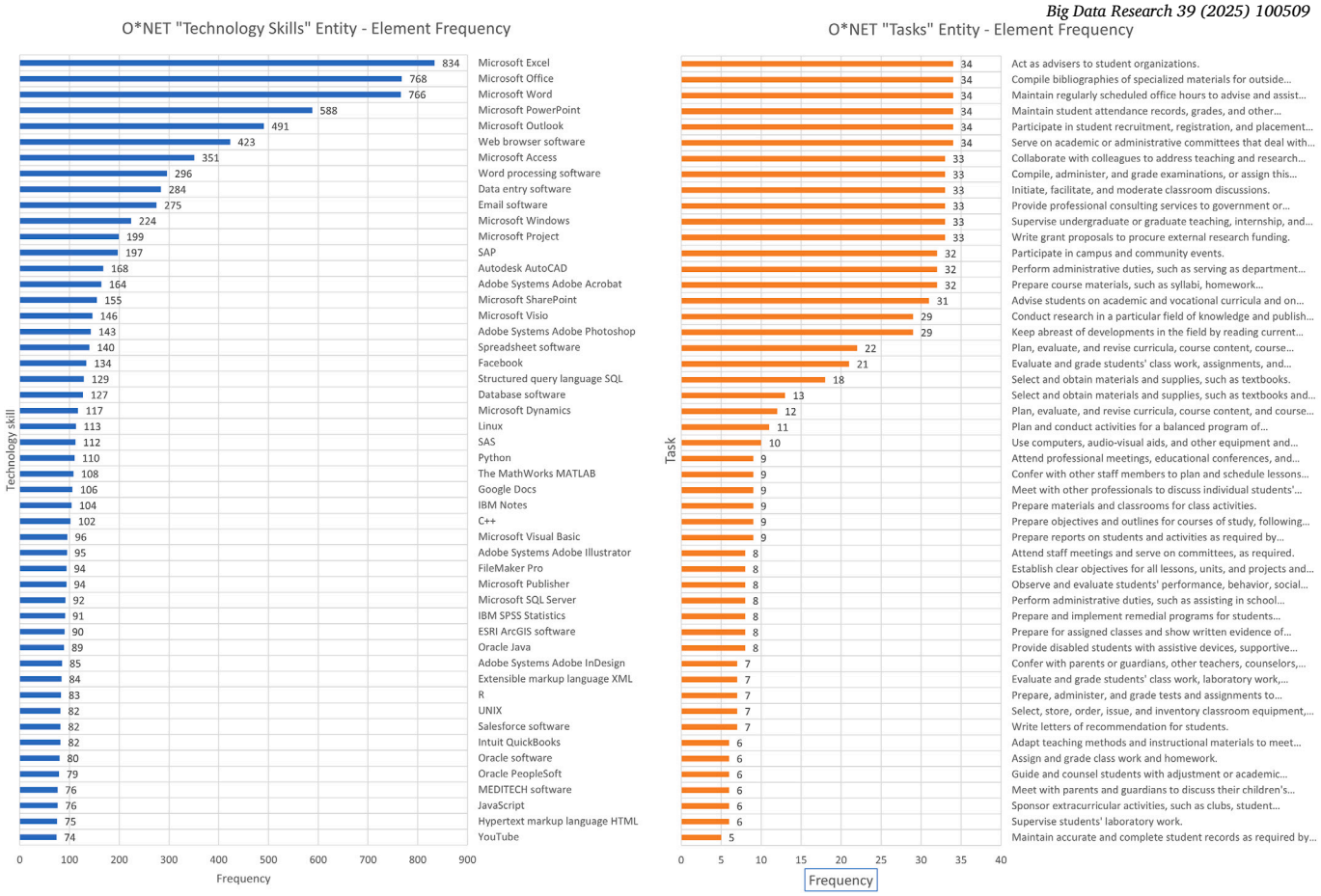
3.3. The O*NET database

The O*NET Program[16] is the primary source of occupational information for the United States. The goal of its creation was to understand the rapidly changing nature of work and how it impacts the workforce and the U.S. economy. One of the main outcomes of the program is the O*NET database, which includes hundreds of standardized and occupation-specific descriptors on almost a thousand occupations covering the entire U.S. economy. The database is freely available, under a Creative Commons license, and is continuously updated on a quarterly basis by different institutions. It has already been used by millions of persons for career exploration and to discover which training is necessary to be eligible for a position, and by employers to find skilled workers to better compete in the marketplace.

Each occupation included in the O*NET database requires a disparate mix of knowledge, skills, and abilities and is performed using a variety of activities and tasks. We have used the O*NET database version 26.2 which includes 1,016 occupations, 52 *Abilities*, 33 *Knowledge*, 35 *Skills*, 41 *Work Activities*, 4,127 *Tools Used*, 17,975 *Task Ratings*, and 8761 *Technology Skills* grouped into 135 categories. In the remainder of the paper, we will refer to *Abilities, Knowledge, Skills, Work Activities, Tools Used, Task Ratings* and *Technology Skills* as O*NET entities. Each occupation is associated with a unique identifier, a title, and a related description. For example, the occupation *Computer and Information Research Scientist* has description *Conduct research into fundamental computer and information science as theorists, designers, or inventors. Develop solutions to problems in the field of computer hardware and software*.

The *Abilities* consist of a set of capacities related to each occupation. One occupation may be associated with many abilities, each with a related score. A given ability (as for *Knowledge*, *Skills* and *Work Activities*) might be present in different occupations with different scores.

[13] https://www.sbert.net/.
[14] https://huggingface.co/docs/transformers/index.
[15] https://www.sbert.net/docs/pretrained_models.html.
[16] https://www.onetcenter.org/.



(a) O*NET "Technology Skills" Entity.

(b) O*NET "Tasks" Entity.

Fig. 2. O*NET "Technology Skills" and "Tasks" Entities - First 50 elements by frequency.

One example is *Oral Comprehension* which, among others, appears in the occupations *Chief Executives* and *Biostatisticians* with a score value of, respectively, 4.5 and 4. The score reflects the importance of that ability with respect to the associated occupation and has a value in the [1-6] continuous range. The higher the score, the more important that ability is for the job it refers to.

Knowledge represents the required area of expertise for the underlying job and has scores in the [1-7] continuous range. An example is provided by *Mathematics* which has a score of 6.83 for the occupation *Mathematicians*. Similarly to the abilities, also Knowledge has many-to-many relations with the occupations and each different pair (occupation, knowledge) has a different score value.

Skills are the competencies required for each occupation and have a score in the [1-6] continuous range. One example is *Programming*, associated with the job *Computer Systems Engineers/Architects* with a score value of 3.38.

For *Work Activities* (values range in the [1-7] continuous interval), an example is given by *Analyzing Data or Information*, with a score of 6.61 with respect to the occupation *Financial Quantitative Analysts*. Both Skills and Work Activities have many-to-many relations with different score values for each different job.

One example of *Task Ratings* is *Direct and coordinate activities involving sales of manufactured products, services, commodities, real estate or other subjects of sale*, related to the job *Sales Manager* with an importance score of 4.22 (scores of Task Ratings are in the interval [1-5]).

An example of *Tools Used* is *Personal Computer*, associated with several jobs (almost all of them).

Finally, an example of *Technology Skills* is *Atlassian JIRA*, related to the occupation *Administrative Services Managers*; Technology Skills have one more field, (*hot technology*), meaning whether the underlying technology skill is hot or not.

The reader notices that the last three described O*NET entities, Task Ratings, Tools Used, and Technology Skills, only occur when they are required for the underlying job. Moreover, Tools Used and Technology Skills do not have a score value associated with an occupation, Task Ratings does. Overall, as previously mentioned, there are 4,127 Tools Used, 17,975 Task Ratings, and 8761 distinct Technology Skills. A certain O*NET job will require a subset of each of the three entities. For example, the O*NET job *Chief Executives* requires 7 Tools Used, 31 Task Ratings, and 49 Technology Skills. Conversely, each O*NET occupation is always associated with a fixed number for each of the first four described entities. Hence, any O*NET occupation will be associated with a vector of 52 Abilities, a vector of 33 Knowledge, a vector of 35 Skills, and a vector of 41 Work Activities.

One more table worth to be mentioned is the content model reference which contains complete descriptions of all the elements included in the entities of the O*NET database previously described. For example, one of the Abilities' elements is *Cognitive Abilities* and its description is *Abilities that influence the acquisition and application of knowledge in problem solving*.

To provide the reader with some statistics, Figs. 2 and 3 show the first 50 elements, in decreasing order by the number of times occurring within the dataset's jobs, of the O*NET entities with a variable number of elements per job, that is "Technology skills", "Tasks", and "Tools Used". For example, *Microsoft Excel* is a technology skill required by 834 O*NET jobs whereas *Personal Computers* is a Tool required by 656 occupations. Similarly, Figs. 4 and 5 show the average values of the scores, calculated over all the O*NET jobs, of the elements of the O*NET entities

Table 1
Dimension of the vector space for the first four O*NET entities.

Name	Dimension	Example elements
Abilities	52	Oral Comprehension, Written Comprehension
Knowledge	33	Sales and Marketing, Computers and Electronics
Skills	35	Science, Active Learning, Service Orientation, Speaking
Work Activities	41	Analyzing Data or Information, Thinking Creatively

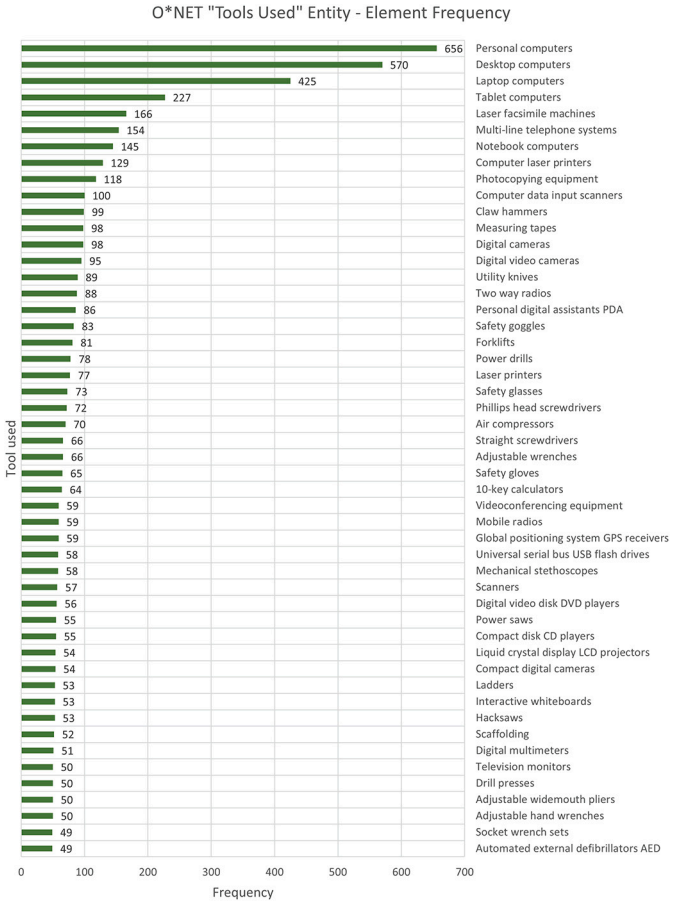


Fig. 3. O*NET "Tools Used" Entity - First 50 elements by frequency.

with a fixed number of elements per job. So, for example, *Active Listening* is the value of the entity Skills with the highest average score (3.60) whereas *English Language* is the value of the Knowledge entity with the highest average score (3.66). Similar considerations can be done for the values of the entities in Fig. 5.

4. The proposed approach

This section describes the proposed approach used to match resumes and jobs.

4.1. Adopted O*NET entities

We used the following O*NET entities:

- "Abilities",
- "Knowledge",
- "Skills",
- "Work Activities",
- "Task Ratings",
- "Technology Skills",
- "Tools Used".

We have mapped the first four entities to n-dimensional vectors in which the components are the scores (already mentioned in Section 3) attributed to the importance of the i^{th} element of the underlying vector (whose size is reported in Table 1) of one of the entities with respect to the 1,016 different jobs present in the O*NET database. For sake of clarity, let us remark that, as discussed in Section 3, there is a difference among the entities. The first four entities have a fixed dimension for each job as indicated in Table 1. The other three entities have dimensions that vary depending on the job. For Task Ratings for example, the job *Sales Manager* includes 17 of them (each with a score or data value, as shown in Fig. 6) whereas the job *Wind Energy Engineers* includes 16 other different tasks.

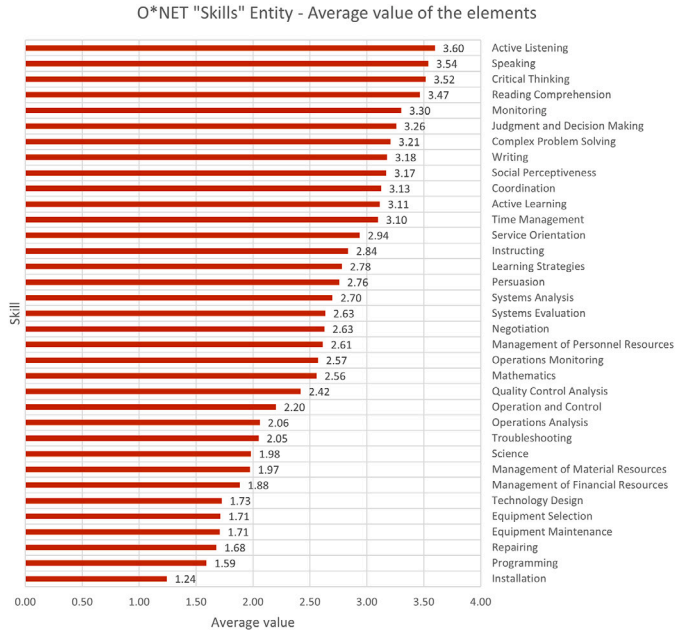
The reader notices that for tasks we have considered the scores associated with the scale importance which lies within the [1-5] interval. As mentioned in Section 3.3, in the O*NET database, the only elements that are not weighted differently based on their relative importance for job matching are technology skills and tools. Our system utilizes all the information provided by the O*NET database as it is, without altering or interpreting the design choices made by its developers, in order to preserve the consistency and structure of the database. Specifically, Technology Skills have no values relative to their importance but are classified by O*NET as "hot technology" or not. Hence, assuming we extract an element from the resume that is matched against a certain Technology Skill, we decided to assign a score of 1 if the recognized item is classified from O*NET as "hot" and 0.75 otherwise. Finally, for a certain *Tool Used* identified in a resume, we simply assign a score of 1.0 if it is identified and 0 if it has not been detected.

4.2. Transformer-based comparison

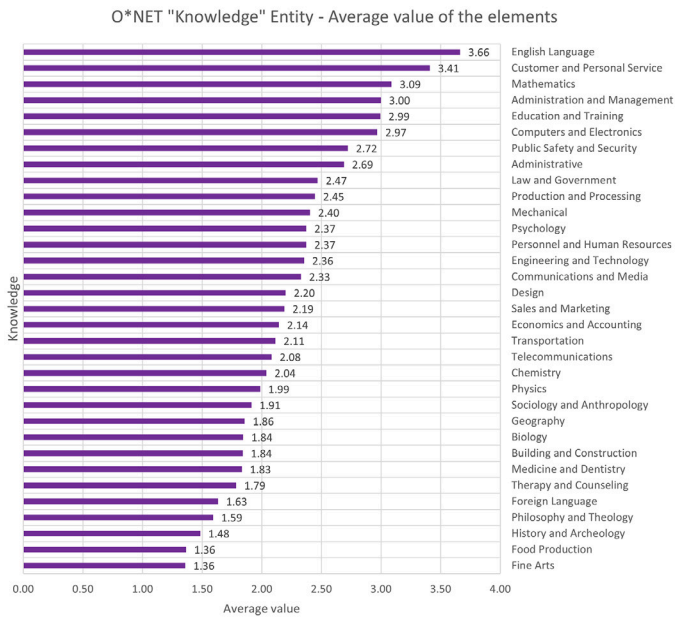
To collect information to compare against the O*NET entities, we extracted sentences, nouns (e.g., *network, mathematics, archaeology*), and noun phrases (e.g., *system administrator, server management, machine learning*) from the text of resumes or jobs descriptions using the *TextBlob* library[17]. It was selected for its simplicity, efficiency, and reliability in extracting nouns and noun phrases, which align well with the structured elements of O*NET entities. This library allowed us to effectively preprocess the textual data, ensuring compatibility with the downstream tasks of semantic similarity and job matching.

The nouns and noun phrases extracted are compared with the values of the elements of the O*NET entities, while the sentences are compared with the descriptions of these elements. The descriptions of elements have been introduced in Section 3 and exist for the entities Abilities, Knowledge, Skills, Work Activities, and Task Ratings. The reason why we chose nouns and noun phrases to be matched with the entities elements is the size of the latter (in general they consist of a few tokens). To not lose the information given by the descriptions of some of the entities we have considered, we have also compared the descriptions with the entire sentences extracted from the resumes or job descriptions. We used a pre-trained neural network model from Sentence Transformers *('all-mpnet-base-v2')* to get text embeddings and leverage the cosine similarity to generate, given a certain resume or job posting, one similarity matrix for each pair (job, entity). Given a similarity matrix for a pair (job, entity), the columns of the matrix are the information extracted from the

[17] https://textblob.readthedocs.io/en/dev/.

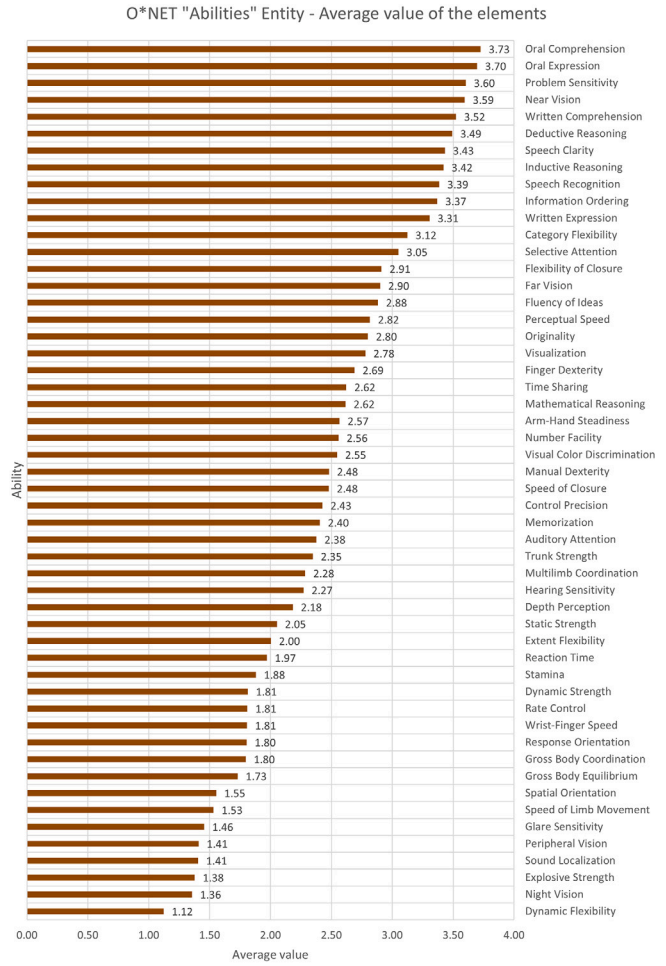


(a) O*NET "Skills" Entity.

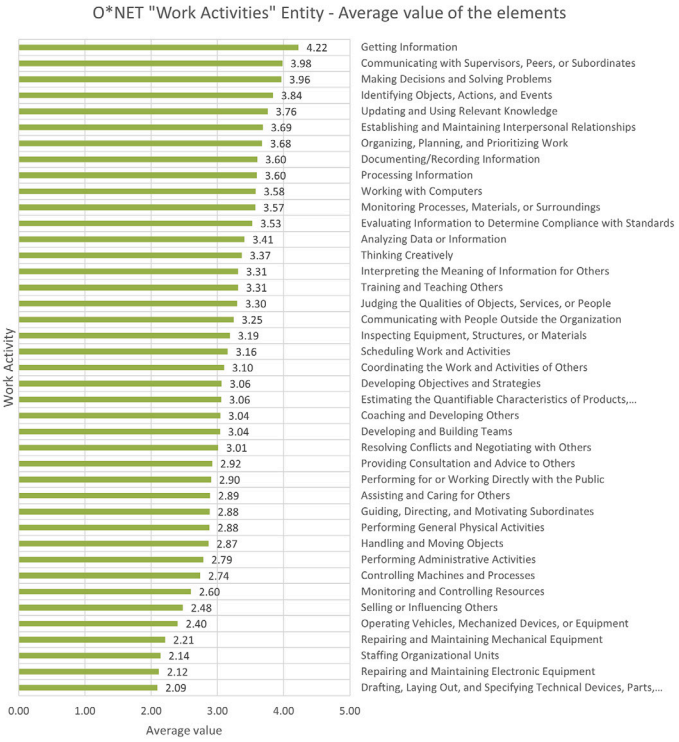


(b) O*NET "Knowledge" Entity.

Fig. 4. O*NET “Skills” and “Knowledge” Entities - Average value of the elements.



(a) O*NET "Abilities" Entity.



(b) O*NET "Work Activities" Entity.

Fig. 5. O*NET “Abilities” and “Work Activities” Entities - Average value of the elements.

O*NET-SOC Code	Title	Task	Task	Scale ID	Data Va
11-2022.00	Sales Managers	1	Resolve customer complaints regarding sales and service.	IM	4,09
11-2022.00	Sales Managers	2	Monitor customer preferences to determine focus of sales efforts.	IM	3,57
11-2022.00	Sales Managers	3	Direct and coordinate activities involving sales of manufactured products, services, commodities, real estate or other subjects of sale.	IM	4,22
11-2022.00	Sales Managers	4	Determine price schedules and discount rates.	IM	3,61
11-2022.00	Sales Managers	5	Review operational records and reports to project sales and determine profitability.	IM	3,95
11-2022.00	Sales Managers	6	Direct, coordinate, and review activities in sales and service accounting and record-keeping, and in receiving and shipping operations.	IM	3,48
11-2022.00	Sales Managers	7	Confer or consult with department heads to plan advertising services and to secure information on equipment and customer specifications.	IM	3,25
11-2022.00	Sales Managers	8	Advise dealers and distributors on policies and operating procedures to ensure functional effectiveness of business.	IM	3,26
11-2022.00	Sales Managers	9	Prepare budgets and approve budget expenditures.	IM	3,59
11-2022.00	Sales Managers	10	Represent company at trade association meetings to promote products.	IM	3,14
11-2022.00	Sales Managers	11	Plan and direct staffing, training, and performance evaluations to develop and control sales and service programs.	IM	3,5
11-2022.00	Sales Managers	12	Visit franchised dealers to stimulate interest in establishment or expansion of leasing programs.	IM	3,44
11-2022.00	Sales Managers	13	Confer with potential customers regarding equipment needs and advise customers on types of equipment to purchase.	IM	4,07
11-2022.00	Sales Managers	14	Oversee regional and local sales managers and their staffs.	IM	3,7
11-2022.00	Sales Managers	15	Direct clerical staff to keep records of export correspondence, bid requests, and credit collections, and to maintain current information on tariffs, licenses, and restrictions.	IM	3,38
11-2022.00	Sales Managers	16	Direct foreign sales and service outlets of an organization.	IM	3,14
11-2022.00	Sales Managers	17	Assess marketing potential of new and existing store locations, considering statistics and expenditures.	IM	3,55

Fig. 6. Task Ratings for the job Sales Manager.

resume (or job posting) and the rows are the elements of the O*NET entity which might occur in the job being considered. For example, when analyzing the entity Abilities for a certain job, the associated similarity matrix will have 52 rows and a number of columns depending on the information (nouns, sentences, noun phrases) extracted from the resume or job posting. We fill each cell of the matrix with the similarity value between the extracted information from the resume/job posting (column c) and the elements of the O*NET entities (row r). From each column, the system extracts the element with the maximum value, which corresponds to the element of the O*NET entity with the highest similarity value with respect to the extracted element of the resume/job posting represented in the underlying column. If the similarity value is greater than an empirically found threshold, the score of the element of the O*NET entity is added to the resume/job posting score for the current job.

This threshold was determined after conducting numerous experiments, and it was adjusted to identify the value that maximized classification accuracy. We tested different values and found that 0.65 provided the best balance between correctly identifying relevant O*NET entities and minimizing misclassifications.

The system then calculates the maximum score obtainable for the current job j and normalizes the score obtained from the resume/job posting by applying the formula:

$$score(j) = \sum_{i=1}^{i=num\_entities} \left(\frac{\sum_{k \in entity_i} score_k(j)}{score\_max_i(j)} + 0.5 * \sum_{k \in entity_i} score_k(j) \right) \quad (1)$$

where $num\_entities$ is equal to seven and corresponds to the entities that have been previously described, $score_k(j)$ is the sum of the scores of each element (identified in the resume/job posting) of the entity i of the job j that is being checked whereas $score\_max_i(j)$ is the maximum score that the job j would obtain if all the related elements of the entity i were detected in a resume/job posting. The term $(0.5 * \sum_{k \in entity_i} score_k(j))$ acts as a corrective factor, empirically determined to prevent the misjudgment of job categories with a large number of entities associated in ONET. Without this correction, the formula would only normalize the scores by dividing by the maximum possible score, which could unfairly disadvantage jobs with many associated entities. For example, if a job category has 25 entities associated in O*NET and only 5 entities are matched in the resume, the score would be normalized by dividing by the sum of the 25 scores, yielding a lower score compared to a job with only 10 associated entities in O*NET. This imbalance could lead to the underrepresentation of jobs with a higher number of associated entities, such as IT-related jobs, which are linked to many tools and technical skills in O*NET, even if a significant portion of their relevant entities is detected in the resume. The corrective factor is necessary to address this issue, ensuring that jobs with a large number of associated entities do not receive disproportionately low scores due to the normalization process. We conducted several experiments adjusting this parameter (e.g., testing values like 0.4, 0.3, etc.), and found that 0.5 provided the best balance for correct classification across diverse job types. Changing this value would influence the classification results, as a smaller corrective factor could lead to the misclassification of jobs with many associated entities, while increasing the factor could cause underrepresentation of jobs with fewer associated entities. Therefore, the value of 0.5 was chosen based on its ability to yield the highest accuracy across various datasets. This

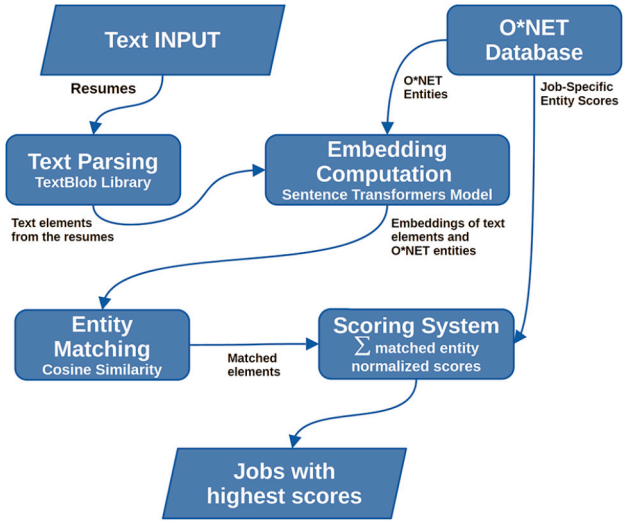


Fig. 7. System diagram.

SIMILARITY MATRIX

	Information from O-NET database job = ['15-1251.00', 'Java Programmer']		Information from the resume						
index	Knowledge element	value	computers	training	maths	english	sales	administrator	engineer
0	Administration and Management	3.26	0.3067	0.2420	0.3296	0.2536	0.4138	0.5764	0.3703
1	Administrative	2.74	0.3654	0.2423	0.2996	0.2819	0.3066	0.8404	0.4794
2	Economics and Accounting	2.47	0.2506	0.1430	0.4473	0.2936	0.4193	0.2083	0.3087
3	Sales and Marketing	2.07	0.3459	0.2934	0.3514	0.2702	0.7480	0.2688	0.3465
4	Customer and Personal Service	3.30	0.2763	0.2987	0.2691	0.1961	0.4606	0.2444	0.3122
5	Personnel and Human Resources	2.00	0.3245	0.3324	0.3236	0.2698	0.3637	0.3159	0.3729
6	Transportation	1.69	0.4229	0.3826	0.3341	0.3014	0.4288	0.1454	0.3436
7	Production and Processing	2.55	0.2568	0.2375	0.2337	0.2191	0.3406	0.1750	0.3123
8	Food Production	1.00	0.1712	0.2081	0.2334	0.1899	0.2773	0.1309	0.2384
9	Computers and Electronics	4.87	0.6389	0.2166	0.3923	0.2473	0.3481	0.2507	0.4653
10	Engineering and Technology	3.47	0.3844	0.2484	0.4444	0.3085	0.2537	0.2114	0.6642
11	Design	2.88	0.4339	0.2534	0.2906	0.2978	0.3643	0.2943	0.4662
12	Building and Construction	1.00	0.3259	0.3185	0.3882	0.2990	0.3212	0.2105	0.5303
13	Mechanical	1.53	0.4239	0.2479	0.3228	0.2206	0.3080	0.2707	0.4703
14	Mathematics	3.53	0.4126	0.2621	0.8136	0.3491	0.3429	0.1322	0.3572
15	Physics	2.11	0.3966	0.2368	0.5643	0.2787	0.2826	0.1021	0.3639
16	Chemistry	1.15	0.3293	0.2326	0.4350	0.3483	0.2403	0.1525	0.3985
17	Biology	1.00	0.2240	0.1572	0.3598	0.3042	0.1845	0.0838	0.2209
18	Psychology	2.04	0.3882	0.3322	0.4652	0.4113	0.3531	0.1604	0.3561
19	Sociology and Anthropology	2.00	0.1975	0.1069	0.3295	0.3473	0.2516	0.1557	0.2554
20	Geography	2.75	0.3247	0.2106	0.3521	0.4288	0.3259	0.1547	0.2702
21	Medicine and Dentistry	1.46	0.1917	0.1796	0.3602	0.2266	0.2339	0.1398	0.3669
22	Therapy and Counseling	1.09	0.1811	0.3672	0.2487	0.2593	0.2463	0.1096	0.1648
23	Education and Training	2.79	0.3007	0.6519	0.3434	0.3106	0.3646	0.2143	0.4727
24	English Language	3.41	0.3422	0.2747	0.4170	0.8104	0.2497	0.2903	0.3680
25	Foreign Language	1.64	0.1954	0.2331	0.2604	0.7266	0.1462	0.1997	0.2232
26	Fine Arts	1.55	0.3638	0.3355	0.4289	0.3827	0.3085	0.2595	0.4454
27	History and Archeology	1.06	0.2172	0.1555	0.3589	0.2983	0.1524	0.1375	0.3388
28	Philosophy and Theology	1.05	0.1351	0.1111	0.2554	0.2408	0.1781	0.0857	0.1532
29	Public Safety and Security	1.45	0.2866	0.2776	0.2625	0.1880	0.2914	0.1930	0.2989
30	Law and Government	1.39	0.2086	0.1231	0.3216	0.3171	0.2190	0.1833	0.2051
31	Telecommunications	2.40	0.5197	0.2846	0.3371	0.3025	0.3396	0.2226	0.3802
32	Communications and Media	2.43	0.4413	0.2283	0.3618	0.3693	0.3582	0.1617	0.2833

Max values		0.6389	0.6519	0.8136	0.8104	0.7480	0.8404	0.6642
Indices of max values		9	23	14	24	3	1	10
Theshold	**0.65**							
Values above threshold		0.0000	0.6519	0.8136	0.8104	0.7480	0.8404	0.6642
Job scores		0.00	2.79	3.53	3.41	2.07	2.74	3.47
Score	**18.01**							
Total for all knowledge elements	71.13							
Normalized and averaged score	**9.26**							

Fig. 8. Example of similarity-matrix for the Knowledge entity.

corrective factor can be considered a hyperparameter, and its optimal value was determined through extensive experimentation.

The described procedure is performed for each job j and returns the first five jobs that have obtained the highest scores. Fig. 7 shows the flow diagram of the algorithm just illustrated.

Fig. 8 shows an example of a similarity matrix for the O*NET entity Knowledge (with 33 rows) with *Java Programmer* as job and *Computers, training, maths, English, sales, administrator, engineer* as information extracted from an input resume. The reader should note that in the similarity matrix, for each piece of information extracted from the resume (column), we have a cell with the maximum value on the row corresponding to an element of the Knowledge entity (as previously mentioned, Knowledge has 33 rows). For example, for the information *Computers*, this maximum value corresponds to the Knowledge element *Computers and Electronics*, but the corresponding similarity value of 0.6389 is below the threshold (empirically fixed for all the entity values at 0.65). This represents a typical “failure case”, where the match is incorrectly excluded despite being highly relevant. The O*NET score of 4.87 (associated with the value *Computers and Electronics* for the job *Java Programmer*) for the entity value *Computers and Electronics* will not, therefore, be added to the total score of the resume. The normalized and averaged score of 9.26 (at the bottom of the figure and corresponding to the computation of the member between the parenthesis of Equation (1) for the Knowledge entity) will be then summed with the other normalized and averaged scores of the other O*NET entities. Their similarity matrices used to calculate the other normalized and averaged scores, work in the same way. What changes is the number of rows and columns, according to the elements of the underlying O*NET entity and the information extracted from the underlying resume or job posting.

5. Scenarios

In this section, we will show the two scenarios that we have considered for job matching. They are related to the identification of the best match between a resume and a certain job description. They are also complementary, in the sense that the first one is company-oriented whereas the second one is candidate-oriented.

5.1. First scenario

The input of the first scenario is a resume. Here the task is to identify the O*NET occupations with the highest match with the given resume. This might be very useful for companies that need to quickly assess different resumes. With an automatic tool like this, companies would be able to categorize plenty of resumes with very high precision in a matter of seconds. In such a case, the application of our approach is straightforward. We would first compute the entities of the resume as described in Section 4. Next, we would compute the score of each job as indicated in Equation (1) and would consider the job whose score is the highest.

5.2. Second scenario

The input of the second scenario we have envisaged consists of a resume and a set of job descriptions. Here the goal is to return the job whose description matches the most with the candidate’s skills present

Education Details
January 2016 B.Sc. Information Technology Mumbai, Maharashtra University of Mumbai
January 2012 HSC Allahabad, Uttar Pradesh Allahabad university
January 2010 SSC dot Net Allahabad, Uttar Pradesh Allahabad university
Web designer and Developer Trainer

Web designer and Developer
Skill Details
Web design- Exprience - 12 months
Php- Exprience - 12 monthsCompany Details
company - NetTech India
description - Working. (salary - 12k)
PERSONAL INTEREST

Listening to Music, Surfing net, Watching Movie, Playing Cricket.
company - EPI Center Academy
description - Working. (Salary Contract based)
company - Aptech Charni Road
description - Salary Contract based)

Fig. 9. A resume within the category Web Designing.

in his/her resume. If the first scenario was thought to help companies in the fast screening of resumes, the second scenario is more applicant-oriented and allows to quickly assess which jobs among a list of many are more suitable to the given resume. To adapt our method for this case we first apply our approach to each job description $jdesc$ present in the input set, after having collected its related entities vectors, it returns the O*NET occupation with the highest match with $jdesc$. Please note that at this stage the returned O*NET occupations OCC are less than or equal to the number of job descriptions present in our set (it might be less if multiple job descriptions are mapped to the same O*NET occupation). Finally, we apply once more our approach to the input resume with the difference that, instead of looking to the best possible match over all the O*NET occupations, we limit the search to the OCC set we identified in the step before. Therefore, one of the O*NET occupations present in OCC will be returned as the one that matches the most with the input resume.

6. Evaluation

In this section, we will illustrate the evaluation assessment we have performed for each scenario. The reader notices that the scoring mechanism that we have introduced for Scenario 1 has been used as well for Scenario 2.

6.1. Evaluating scenario 1

As far as the first scenario is concerned, to check the performances of our approach we used the Resume Dataset version 1[18]. The dataset consists of 963 resumes from 25 different categories. An example of a resume is shown in Fig. 9. Several resumes had duplicates and some categories contained only a few of them. To create the test dataset so that we could analyze the performances on its categories, we fixed as a constraint that at least five resumes should occur for each category. Only 21 categories satisfied such a constraint. Therefore, we selected a set of 105 resumes from 21 different categories (5 resumes per category) and run our method which, for each resume, returned the top five O*NET occupations with the highest match with the underlying resume. The 21 classes were “Data Science”, “Human Resources”, “Advocate”, “Mechanical Engineer”, “Sales”, “Health and fitness”, “Civil Engineer”, “Java Developer”, “Business Analyst”, “SAP Developer”, “Automation Testing”, “Electrical Engineering”, “Python Developer”, “DevOps Engineer”, “Network Security Engineer”, “Database”, “Hadoop”, “ETL Developer”, “DotNet Developer”, “Blockchain”, “Testing”.

To evaluate our method, we performed an extensive manual assessment. The relevance of each suggested job was determined by adopting a similar approach presented by [35]. We rely on a five-point relevance scale. As such, human assessors assign a relevance score to each suggested job. The score can be any of the following values:

- **non-relevant (score 1):** The predicted job is completely different from the original one.
- **ordinary (score 2):** The predicted job has an ambiguous or unfair match with the original one.
- **marginally relevant (score 3):** The predicted job is more generic or somewhat related with the original one or vice-versa.
- **relevant (score 4):** The predicted job fairly matches with the original one.
- **highly relevant (score 5):** The predicted job totally matches with the original one.

Although precision, recall, and F1-scores are widely used metrics for performance evaluation, we deliberately adopted the described methodology to ensure a more nuanced and context-aware assessment of job relevance. This approach allows human annotators to capture qualitative aspects of the predictions, which are not fully accounted for by traditional quantitative metrics.

We used the mentioned ratings for two different annotations ($annotation_1$ and $annotation_2$). $Annotation_1$ assesses the first predicted job. $Annotation_2$ assesses the five predicted jobs for a resume. More in detail, annotators were asked to give a score by picking the most matching job according to their expertise among the 5 proposed by our approach. Therefore, if the resume’s label was Data Science and the five returned O*NET occupations were, respectively, Models, Data Scientists, Logisticians, Computer Programmers, and Mechanical Engineers, one annotator should assign 1 to $annotation_1$ (because Models is not relevant to Data Science) and 5 to $annotation_2$ (because among the first five results, we have Data Scientists which completely matches the original Data Science label). It is straightforward to note that the following statement holds:

$$score(annotation_2) \geq score(annotation_1)$$

Three independent recruiters have annotated the two scores returned by our algorithm for the 105 resumes. To decide which annotation to pick, we applied the majority voting algorithm. If three annotations for a particular resume were all different from each other (so that the majority voting algorithm could not be applied) we computed the average and rounded it to the closest integer score (for example, if three scores were 1, 3, and 4 the final score would be $round(1+3+4)/3 = 3$). The inter-annotator agreement scores computed according to Fleiss’ kappa [36] were 0.37 and 0.27 for both $annotation_1$ and $annotation_2$ respectively as reported in Table 2, indicating a fair/moderate agreement among the annotators, despite the elevated number of classes to pick for each sample. Then, we applied the majority voting and scored the results of the proposed algorithm accordingly. In particular, we obtained an average score of 3.8 for $annotation_1$ and an average score of 4.2 for $annotation_2$, as indicated in Table 3. We compared these results against those obtained using two naive unsupervised approaches which would constitute our baselines.

The two baselines do not take into account any entity of the O*NET database and work as follows. First, each input resume is broken down in nouns, noun phrases, and sentences. For each element, we compute the cosine similarity against each of the 21 class names of jobs ($baseline1$) and against each of the 1,016 O*NET occupations names ($baseline2$).

Table 2
Inter-agreement evaluation of the three experts on our approach and the two baselines.

	$Annotation_1$	$Annotation_2$
Our Approach	0.37	0.27
Baseline1	0.56	0.43
Baseline2	0.39	0.37

[18] https://www.kaggle.com/datasets/gauravduttakiit/resume-dataset.

Table 3
Results of our approach and the two baselines.

	$Annotation_1$	$Annotation_2$
Our Approach	3.8	4.2
Baseline1	3.4	4.9
Baseline2	3.0	3.8

In both cases, we return the five job classes with the highest semantic similarity. Those classes have been annotated by the three experts with the same scoring approach mentioned above and a majority voting strategy has been applied. We computed the inter-annotator agreement scores [36] among the three annotators for the baselines as well. The inter-agreement values for the two baselines are reported in Table 2. The reader might notice a much higher inter-agreement value for the $baseline1$ than the others. The reason is that $baseline1$ returns elements from a set of 21 classes which are much easier to rate for the annotators with respect to 1,016 classes. Finally, as similarly performed for our approach, we consider the majority voting result that we report in Table 3. We can observe that for $baseline1$, for the annotation $annotation_2$, the obtained score is very high. This happens for the same reason as the high inter-annotator agreement value mentioned earlier. Because the set of considered classes for this baseline is 21, it is very likely that out of 21 candidate classes, one of the first five returned jobs is correct (equal to the resume label).

6.2. Evaluating scenario 2

For this second scenario, we employed a dataset of 10k job posts of the CareerBuilder job dataset from the UK in 2019[19]. A subset of 8948 of these 10k job posts are in English and belong to 299 distinct categories. Each category might contain one or more jobs. Each job belongs to one category only. To prepare the dataset to be used with our system, we selected the posts belonging to categories with cardinality greater than or equal to 5 (i.e., each job category contained at least five jobs). We then selected 100 resumes (5 for each category) from the dataset used in scenario 1, belonging to the following 20 categories: *Data Science, Human Resources, Advocate, Mechanical Engineer, Sales, Health and fitness, Pharmacists, Java Developer, Business Analyst, SAP Developer, Automation Testing, Electrical Engineering, Python Developer, DevOps Engineer, Network Security Engineer, Database, Hadoop, ETL Developer, DotNet Developer, Testing.* Then we selected the most semantically similar job post category for each of the 20 resume categories: *Computer and Information Research Scientists, Human Resources Specialists, Lawyers, Industrial Engineers, Sales Managers, Health Educators, Pharmacists, Computer Programmers, Management Analysts, Software Developers, Applications, Software Quality Assurance Engineers and Testers, Electrical Engineers, Computer Programmers, Computer Systems Engineers/Architects, Network and Computer Systems Administrators, Database Administrators, Computer Systems Analysts, Software Developers, Applications, Software Developers, Applications, Software Quality Assurance Engineers and Testers.*[20] Each of the 100 resumes has been associated with 10 different job posts, the last of which belongs to the corresponding resume category. The extraction of job posts was random and each extracted post could occur just once.

Therefore, each sample contained a resume, ten job postings and their related categories included in the CareerBuilder job dataset. For a given sample, the ten job postings included are always of different categories. The last job posting of each sample corresponds to the same category of the resume. The same job posting is never present more than once in the entire collection. Because the related categories of each job posting are not always matched with the occupations found within the O*NET database, we applied our method to each job posting in order to identify the O*NET occupation that matches the most the related job posting. Therefore, for each sample, we also added ten O*NET job categories. Each one is related to each job posting in the sample.

We defined two baselines. The first one works as follows. Given one sample, it starts by extracting noun phrases, nouns, and sentences from the resume. Then for each extracted element it computes the cosine similarity against the 20 job labels (the job posting categories of the CareerBuilder dataset and those from the O*NET database calculated by using our approach) and returns the five job classes whose semantic similarity with any of the extracted element is the highest.

The second baseline uses the same O*NET entities we have defined within our approach but without using the scores. From the resume, it identifies all the values for each O*NET entity and builds one binary vector for each entity. Thus, the entity *ent* will consist of a vector of 0s and 1s: 0 if the i feature of *ent* is not included in the resume, 1 otherwise. To note that we considered all the values of each entity so that vectors of a certain entity will always have the same size. We perform the same operation for the 20 jobs present in each sample (thus we obtain binary entity vectors for each job in the sample) and then compute the Euclidean distance between binary vectors. We take the 5 jobs with the smallest distance with respect to the resume.

Finally, in a given sample, our approach is applied to the resume by computing the entities using the method already illustrated in Section 4 and looking for the best O*NET job label among the twenty we have.

Three human annotators filled two annotations for the two baselines and our approach. The first annotation assesses the first predicted job post whereas the second annotation assesses the top-5 predicted job posts, similarly as defined within the first scenario. To note that a certain prediction is compared against the list of the 10 original job labels and against the list of the derived 10 O*NET job labels. So, for example, if a baseline contains "Architectural and Engineering Managers" as its first prediction, $annotation_1$ gets a 5 if that category is either the tenth among the original job categories or if that is the O*NET category corresponding to the tenth job description. We computed the final annotations by using a majority voting strategy on the three annotators' scores similarly as performed within the first evaluation scenario. The inter-annotator agreement scores according to Fleiss' kappa [36] among the three annotators are reported in Table 4 indicating a substantial agreement.

The reader notices that such values are much higher than those obtained within scenario 1. The reason is because, in this case, each method had to choose among 10 different jobs only. This low number and the fact that they are different from each other helped the role of the annotators that were in agreement more often. Table 5 shows the results we have obtained for the three methods where it

Table 4
Inter-agreement score of the three experts on our approach and the two baselines for the second scenario.

	$Annotation_1$	$Annotation_2$
Our Approach	0.69	0.66
Baseline1	0.81	0.63
Baseline2	0.66	0.65

Table 5
Results of our approach and the two baselines for the second scenario.

	$Annotation_1$	$Annotation_2$
Our Approach	4.17	4.95
Baseline1	3.8	4.87
Baseline2	2.03	3.75

[19] https://data.world/promptcloud/jobs-on-careerbuilder-uk.

[20] The reader notices that such a list is presented in order of similarity with the list of resume categories shown above. That is, the first element, *Computer and Information Research Scientists* is the most semantically similar job category to the resume category *Data Science*, etc.

is possible to note how ours outperforms the two baselines for both $annotation_1$ and $annotation_2$ showing again its efficiency. The data related to the evaluation that has been carried out for both scenarios is freely available at https://gitlab.com/hri_lab1/using-transformers-and-o-net-to-match-jobs-to-applicants-resumes.

7. Use case: recommending skills to job seekers

Our approach is well-suited for a recommendation task for job seekers. In this section, we will show an extension of our method in this direction. It works as follows. The user of the system, that is a person looking for a job, should just provide his/her resume and a list of job names he/she is interested in. First of all, the job names are matched against the O*NET occupations so that a map can be defined between input job names and O*NET occupations. We already performed a similar computation within the evaluation of the proposed second scenario. Then, for each identified O*NET job, the approach would compute a score of the resume against the input O*NET categories according to the Equation (1) introduced in Section 4.2.

7.1. Job seeking and skills recommendation

The reader can find an implementation of this use case at http://192.167.149.11:8000/ under the *first demo* link. Let us assume that the user gives as an input one job name only, and thus only one job from O*NET is selected. Then the user uploads his/her resume. The system elaborates on it. To assess whether the score obtained from the resume for the requested job makes the resume eligible, we compare it with an experimentally established threshold value of 34.62. The value of 34.62 for eligibility represents the average score that resumes from the correct job category obtained during the testing phase. It was chosen as a natural cutoff, distinguishing eligible candidates from ineligible ones based on the distribution of scores across different classes. This value reflects the typical score achieved by resumes correctly classified, ensuring that only resumes with a high likelihood of being relevant to the job are considered eligible. It is calculated by averaging the scores obtained from all elements of the Resume Dataset used in the previous scenarios, under the assumption (confirmed among the release notes of the dataset) that each resume in that dataset is eligible for the O*NET occupation that was selected.

When the user's resume is not eligible with the input job, the system returns the list of the O*NET entity elements not found within the input resume in decreasing order of importance. For the entities with scores (Abilities, Knowledge, Skills, Work Activities, Task Ratings), it means to return their elements in decreasing order of scores. For the other entities (Technical Skills and Tools Used), they are returned in alphabetical order (for the technological skills we first return the entries with hot technologies and then the others). For each element, the system indicates the O*NET score, the name of the element, a brief description (when available), and the value that would be added to the overall CV score if it was found in the resume (that is the value between the parenthesis of the Equation (1) for a fixed i, k and j). If the user notices that he/she has simply failed to include some of his/her skills or knowledge or some previous work activity, he/she can add it to the resume and resubmit it. Otherwise, he/she would need to get more competencies and, consequently, raise the score of his/her resume over the pre-established threshold. In addition to this, the demonstration also allows us to identify the skills, technical abilities, and other qualifications needed for a job and can recommend job opportunities that align with the user's characteristics (*second* and *third demo*).

7.2. Acquiring new skills

To acquire new skills, a system that embeds our approach can cover the different values present in the entities so that the candidate becomes eligible for that specific job.

There are several ways that can be employed for such a purpose. The first one is to leverage existing APIs of well-known massive open online course providers such as Udemy[21] or Coursera[22]. In such a case we would simply leverage the search engine of those providers by feeding them with the entities to be mastered by the applicant.

As an example, we submitted for the job *Mechanical Engineer* a resume from the Resume Dataset belonging to the category *Electrical Engineering*. Our system returns a score of 19.33, which is below the threshold (34.62) by 15.29 points. Hence, the system does not consider the resume eligible for the *Mechanical Engineer* job and lists the elements of the O*NET entities that have not been identified in the CV, including for example: *Reading comprehension*, *Active Listening*, *Mathematics* and *Critical Thinking*. For each item, up to 5 Udemy courses are listed, when available. In our example, the *Become Active Reader & Master Your Reading Comprehension Skills* course is suggested for the first element, *Giving full attention to what other people are saying, taking time to understand the points being made, asking questions as appropriate, and not interrupting at inappropriate times* course for the second, *Math* and *Perfect your Mathematics skills* courses for the third, and *Using logic and reasoning to identify the strengths and weaknesses of alternative solutions, conclusions, or approaches to problems* course for the fourth. Then the candidate should read or attend those lectures and improve his/her resume with the acquired competencies. Afterward, by adding the missing elements listed above to the resume and resubmitting the application to the system, the augmented resume obtains a score of 39.85 and passes the eligibility threshold.

Besides relying on online providers, we can also exploit dumps of resources with courses' information [37,38]. For example, one resource that can be leveraged is the COCO [38] dataset, a large collection of courses that includes lessons, teachers, instructors, and learners' ratings collected with the purpose of developing AI-based e-learning applications on top. For the courses, it provides metadata such as short and long descriptions, the necessary requirements, and the expected skills acquired by successfully attending them. These can be leveraged by our system to find out which courses would support an applicant to master new skills by matching the detected missing skills of a resume. Our system would suggest a list of courses from the COCO dataset similar to what is performed by e-learning providers' search engines. The advantage of using such a dataset lies in the opportunity to develop customized AI-based applications (for example leveraging transformers or other language models) giving freedom to the developers to build their own solutions to find the best matches with the missing skills.

Finally, with the aim to recommend lectures or articles, our approach can also be easily fed with modern state-of-the-art resources such as the "Academia/Industry DynAmic" (AIDA) [39] or "The Computer Science Knowledge Graph" (CS-KG) [40–43]. AIDA is a collection of metadata about $21M$ publications and $8M$ patents categorized within a taxonomy of computer science topics from the Computer Science Ontology (CSO) [44]. CS-KG is an automatically generated knowledge graph that describes the content of $6.7M$ scientific papers with $10M$ research entities and $41M$ relationships among them within the computer science domain. These resources can be exploited to recommend patents and research papers that might be of interest for the applicants to discover where and how state-of-the-art technologies are used to improve their skills and be ready for the job application. Considering the example above, if an applicant has to acquire the skill *Critical Thinking* for the *Mechanical Engineering* job, the system can be easily extended to look for research papers that study it within the CS-KG SparQL endpoint[23], suggesting to the applicant interesting reads to make stronger his/her application. For example, CS-KG suggests "Serious games on environ-

[21] https://www.udemy.com.
[22] https://www.coursera.org.
[23] https://scholkg.kmi.open.ac.uk/sparql/.

```
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX cskg: <http://scholkg.kmi.open.ac.uk/cskg/resource/>
PREFIX cskg-ont: <http://scholkg.kmi.open.ac.uk/cskg/ontology#>
PREFIX provo: <http://www.w3.org/ns/prov#>
PREFIX cso: <http://cso.kmi.open.ac.uk/schema/cso#>
PREFIX dcterm: <http://purl.org/dc/terms/>

SELECT ?sub ?pre ?obj?paperTitle ?doi  FROM
<http://scholkg.kmi.open.ac.uk/cskg>
WHERE { ?t rdf:subject cskg:critical_thinking .
        ?t rdf:subject ?sub .
        ?t rdf:predicate ?pre .
        ?t rdf:object ?obj .
        ?t provo:wasDerivedFrom ?paperID .
        ?paperID dcterm:title ?paperTitle ;
        cskg-ont:hasDOI ?doi .
        ?t cskg-ont:hasSupport ?sup}
ORDER BY desc (?sup)
```

Fig. 10. SparQL query to find papers related to the *Critical Thinking* skill.

mental management"[24] and "Teaching Flooding Attack to the SDN Data Plane with POGIL"[25] as relevant papers to master the *Critical Thinking* skill. Fig. 10 shows the SparQL query to be executed in the CS-KG SparQL endpoint to retrieve the papers related to the *Critical Thinking* skill.

8. Conclusions and future directions

In this paper, we have proposed an approach for job recommendation based on the information extracted from applicants' resumes and job postings. The information is then matched against different O*NET entities using semantic similarity of deep learning transformers in order to identify the most suitable O*NET job to the underlying resume or job posting. Two scenarios have been considered: one useful for companies with the goal of quickly screening several resumes and the other useful for applicants with the goal of quickly identifying the job posting more suitable to their skills.

For the first scenario, an extensive evaluation on the Resume Dataset version 1 has been carried out considering 105 resumes from 21 different categories. We have defined a scoring mechanism with five different values from 1 to 5. Moreover, we came up with two baselines (one returning the output over the 21 categories and the other returning the output over the 1,016 possible jobs) to compare our approach against and manually annotated the results of the three methods with the scoring mechanism we defined. The scoring of the three methods has been applied to two outputs (the job with the highest score and the first five jobs with the highest scores). Our approach outperformed the others in three cases out of four.

For the second scenario, we considered 1000 job postings from the CareerBuilder job dataset and 100 resumes from the Resume Dataset used within the first scenario. With similar scoring metrics as those used within the first scenario, we showed how our method obtained much better results than two more baselines that we defined.

Finally, as a use case of the proposed approach, we discussed a recommending task for job seekers where, given in input a resume and a list of occupations, the system returns, for each occupation, a list of courses that need to be attended to acquire the missing skills and become eligible for that specific job.

As future directions, our aim is to refine our prototype and create a real system that performs all the tasks we have illustrated in this work, including the use case discussed at the end of the manuscript. We will release the APIs so that everyone will be able to play with them and integrate them into existing systems. The API will also be employed to collect users' feedback and create a gold standard to evaluate the accuracy of the system at scale. We believe this system might be either used standalone or incorporated into well-known platforms for job seekers. One feature we would like to add is the possibility to include a generator of resumes starting from a list of skills and abilities: this feature would leverage new text generation models with the goal of preparing a complete resume in natural language. Furthermore, we would like to study the injection of O*NET database knowledge and transformer models into existing recommendation methodologies (e.g., collaborative filtering) with the goal of creating a full-fledged platform for job recommendation that takes benefits from domain knowledge using O*NET, transformers, and past candidate experiences.

One of the areas where the authors are continuing their research focuses on providing recommendations for courses and training materials. These recommendations aim to help job seekers acquire the skills or competencies they lack. Additionally, the focus will shift towards applying the lessons learned in the job-seeking domain to worker reskilling initiatives within organizations, especially under the context of Industry 5.0. In this context, LLMs and retrieval-augmented generation (RAG) systems will be explored for the development of study guides based on the organization's training documentation. Another case study will involve adapting the system for document classification. This adaptation will aim to identify the specific skills and abilities that a job seeker or student could improve through the study of the corresponding document. Finally, we plan to explore fine-tuning strategies and further improvements to the system's performance.

CRediT authorship contribution statement

Rubén Alonso: Conceptualization, Funding acquisition, Visualization, Writing – review & editing. **Danilo Dessí:** Formal analysis, Investigation, Validation, Writing – review & editing. **Antonello Meloni:** Data curation, Investigation, Software, Writing – original draft. **Diego Reforgiato Recupero:** Conceptualization, Funding acquisition, Investigation, Methodology, Supervision, Writing – review & editing.

Declaration of competing interest

No conflict of interest exists.

Acknowledgements

This research was partially funded by the Eupean Un project STAR - Novel AI technology for dynamic and unpredictable manufacturing environments (grant number 956573).

[24] https://doi.org/10.1016/j.scs.2016.11.007.
[25] https://doi.org/10.1145/3368308.3415406.

Data availability

Data will be made available on request.

References

[1] S.D. Risavy, C. Robie, P.A. Fisher, S. Rasheed, Resumes vs. application forms: why the stubborn reliance on resumes?, Frontiers in Psychology 13 (2022) 884205.

[2] S. Rojas-Galeano, J. Posada, E. Ordoñez, A bibliometric perspective on ai research for job-résumé matching, The Scientific World Journal 2022 (1) (2022) 8002363.

[3] U. Goyal, A. Negi, A. Adhikari, S.C. Gupta, T. Choudhury, Resume data extraction using nlp, in: J. Singh, S. Kumar, U. Choudhury (Eds.), Innovations in Cyber Physical Systems, Springer, Singapore, Singapore, 2021, pp. 465–474.

[4] T.M. Harsha, G.S. Moukthika, D.S. Sai, M.N.R. Pravallika, S. Anamalamudi, M. Enduri, Automated resume screener using natural language processing (nlp), in: 2022 6th International Conference on Trends in Electronics and Informatics (ICOEI), 2022, pp. 1772–1777.

[5] A. Deshmukh, A. Raut, Applying bert-based nlp for automated resume screening and candidate ranking, Annals of Data Science (2024) 1–13.

[6] S. Westman, J. Kauttonen, A. Klemetti, N. Korhonen, M. Manninen, A. Mononen, S. Niittymäki, H. Paananen, Artificial intelligence for career guidance–current requirements and prospects for the future, IAFOR Journal of Education 9 (4) (2021) 43–62.

[7] C. Yu, C. Zhang, J. Wang, Extracting body text from academic pdf documents for text mining, in: KDIR, 2020, pp. 229–236.

[8] C. Stahl, S. Young, D. Herrmannova, R. Patton, J. Wells, Deeppdf: a deep learning approach to extracting text from pdfs, in: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), European Language Resources Association (ELRA), Paris, France, 2018.

[9] J. Tiedemann, Improved text extraction from pdf documents for large-scale natural language processing, in: A. Gelbukh (Ed.), Computational Linguistics and Intelligent Text Processing, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 102–112.

[10] Z. Cheng, P. Zhang, C. Li, Q. Liang, Y. Xu, P. Li, S. Pu, Y. Niu, F. Wu, Trie++: towards end-to-end information extraction from visually rich documents, arXiv preprint, arXiv:2207.06744, 2022.

[11] M.J. Handel, The o*net content model: strengths and limitations, Journal for Labour Market Research 49 (2) (2016) 157–176, https://doi.org/10.1007/s12651-016-0199-8.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, 2017, pp. 6000–6010.

[13] C. Helwe, C. Clavel, F.M. Suchanek, Reasoning with transformer-based models: deep learning, but shallow reasoning, in: 3rd Conference on Automated Knowledge Base Construction, 2021, https://openreview.net/forum?id=Ozp1WrgtF5_.

[14] A. Meroño-Pe nuela, D. Spagnuelo, Can a transformer assist in scientific writing? Generating semantic web paper snippets with gpt-2, in: A. Harth, V. Presutti, R. Troncy, M. Acosta, A. Polleres, J.D. Fernández, J. Xavier Parreira, O. Hartig, K. Hose, M. Cochez (Eds.), The Semantic Web: ESWC 2020 Satellite Events, Springer International Publishing, Cham, 2020, pp. 158–163.

[15] Z. Zheng, Z. Qiu, X. Hu, L. Wu, H. Zhu, H. Xiong, Generative job recommendations with large language model, arXiv preprint, arXiv:2307.02157, 2023.

[16] Z. Guan, J.-Q. Yang, Y. Yang, H. Zhu, W. Li, H. Xiong, Jobformer: skill-aware job recommendation with semantic-enhanced transformer, ACM Transactions on Knowledge Discovery from Data (2024).

[17] Y. Li, M. Yamashita, H. Chen, D. Lee, Y. Zhang, Fairness in job recommendation under quantity constraints, in: AAAI-23 Workshop on AI for Web Advertising, 2023.

[18] J. Dhameliya, N. Desai, Job recommender systems: a survey, in: 2019 Innovations in Power and Advanced Computing Technologies (i-PACT), vol. 1, 2019, pp. 1–5.

[19] R. Mishra, S. Rathi, Efficient and scalable job recommender system using collaborative filtering, in: A. Kumar, M. Paprzycki, V.K. Gunjan (Eds.), ICDSMLA 2019, Springer, Singapore, Singapore, 2020, pp. 842–856.

[20] Y. Lu, S. El Helou, D. Gillet, A recommender system for job seeking and recruiting website, in: Proceedings of the 22nd International Conference on World Wide Web, WWW '13 Companion, Association for Computing Machinery, New York, NY, USA, 2013, pp. 963–966.

[21] W. Shalaby, B. AlAila, M. Korayem, L. Pournajaf, K. Aljadda, S. Quinn, W. Zadrozny, Help me find a job: a graph-based approach for job recommendation at scale, in: Proceedings of the International Conference on Big Data (BiG Data), 2017, pp. 1544–1553.

[22] S. Choudhary, S. Koul, S. Mishra, A. Thakur, R. Jain, Collaborative job prediction based on Naïve Bayes classifier using python platform, in: 2016 International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS), 2016, pp. 302–306.

[23] G. Domeniconi, G. Moro, A. Pagliarani, K. Pasini, R. Pasolini, Job recommendation from semantic similarity of linkedln users' skills, in: Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods, ICPRAM 2016, SCITEPRESS - Science and Technology Publications, Lda, Setubal, PRT, 2016, pp. 270–277.

[24] F. Gutiérrez, S. Charleer, R. De Croon, N.N. Htun, G. Goetschalckx, K. Verbert, Explaining and exploring job recommendations: a user-driven approach for interacting with knowledge-based job recommender systems, in: Proceedings of the 13th ACM Conference on Recommender Systems, RecSys '19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 60–68.

[25] B. Heap, A. Krzywicki, W. Wobcke, M. Bain, P. Compton, Combining career progression and profile matching in a job recommender system, in: D.-N. Pham, S.-B. Park (Eds.), PRICAI 2014: Trends in Artificial Intelligence, Springer International Publishing, Cham, 2014, pp. 396–408.

[26] X. Zhao, Cold-start collaborative filtering, SIGIR Forum 50 (1) (2016) 99–100, https://doi.org/10.1145/2964797.2964819.

[27] N. Reimers, I. Gurevych, Sentence-bert: sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2019, https://arxiv.org/abs/1908.10084.

[28] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: pre-training of deep bidirectional transformers for language understanding, arXiv preprint, arXiv:1810.04805, 2018.

[29] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, Roberta: a robustly optimized bert pretraining approach, arXiv preprint, arXiv:1907.11692, 2019.

[30] K. Song, X. Tan, T. Qin, J. Lu, T.-Y. Liu, Mpnet: masked and permuted pre-training for language understanding, Advances in Neural Information Processing Systems 33 (2020) 16857–16867.

[31] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, R. Soricut, Albert: a lite bert for self-supervised learning of language representations, arXiv preprint, arXiv:1909.11942, 2019.

[32] S.R. Bowman, G. Angeli, C. Potts, C.D. Manning, A large annotated corpus for learning natural language inference, in: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 632–642, https://aclanthology.org/D15-1075.

[33] A. Williams, N. Nangia, S. Bowman, A broad-coverage challenge corpus for sentence understanding through inference, in: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1 (Long Papers), Association for Computational Linguistics, New Orleans, Louisiana, 2018, pp. 1112–1122, https://aclanthology.org/N18-1101.

[34] C. Spearman, The proof and measurement of association between two things, The American Journal of Psychology 100 (3/4) (1987) 441–471, http://www.jstor.org/stable/1422689.

[35] A. Konjengbam, N. Kumar, M. Singh, Unsupervised tag recommendation for popular and cold products, Journal of Intelligent Information Systems 54 (2019) 545–566.

[36] J.L. Fleiss, J.C. Nee, J.R. Landis, Large sample variance of kappa in the case of different sets of raters, Psychological Bulletin 86 (5) (1979) 974.

[37] D. Dessì, G. Fenu, M. Marras, D. Reforgiato Recupero, Coco: semantic-enriched collection of online courses at scale with experimental use cases, in: Á. Rocha, H. Adeli, L.P. Reis, S. Costanzo (Eds.), Trends and Advances in Information Systems and Technologies, Springer International Publishing, Cham, 2018, pp. 1386–1396.

[38] D. Dessì, G. Fenu, M. Marras, D. Reforgiato Recupero, Bridging learning analytics and cognitive computing for big data classification in micro-learning video collections, Computers in Human Behavior 92 (2019) 468–477, https://doi.org/10.1016/j.chb.2018.03.004, https://www.sciencedirect.com/science/article/pii/S0747563218301092.

[39] S. Angioni, A.A. Salatino, F. Osborne, D.R. Recupero, E. Motta, AIDA: a knowledge graph about research dynamics in academia and industry, Quantitative Science Studies 2 (4) (2021) 1356–1398, https://doi.org/10.1162/qss_a_00162.

[40] D. Dessì, F. Osborne, D.R. Recupero, D. Buscaldi, E. Motta, H. Sack, AI-KG: an automatically generated knowledge graph of artificial intelligence, in: The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, in: Lecture Notes in Computer Science, vol. 12507, Springer, 2020, pp. 127–143.

[41] D. Dessì, F. Osborne, D.R. Recupero, D. Buscaldi, E. Motta, Generating knowledge graphs by employing natural language processing and machine learning techniques within the scholarly domain, CoRR, arXiv:2011.01103, 2020, https://arxiv.org/abs/2011.01103.

[42] D. Dessí, F. Osborne, D.R. Recupero, D. Buscaldi, E. Motta, Scicero: a deep learning and nlp approach for generating scientific knowledge graphs in the computer science domain, Knowledge-Based Systems 258 (2022) 109945.

[43] D. Dessí, F. Osborne, D. Reforgiato Recupero, D. Buscaldi, E. Motta, Cs-kg: a large-scale knowledge graph of research entities and claims in computer science, in: International Semantic Web Conference, Springer, 2022, pp. 678–696.

[44] A.A. Salatino, T. Thanapalasingam, A. Mannocci, F. Osborne, E. Motta, The computer science ontology: a large-scale taxonomy of research areas, in: International Semantic Web Conference, Springer, 2018, pp. 187–205.