Towards an Integrated Retrieval System to Semantically Match CVs, Job Descriptions and Curricula

Ioannis - Aris Kostis
My Company Projects O.E., Thessaloniki, Greece, giannis.kostis@mycompany.com.gr

Dimitrios Sarafis*
My Company Projects O.E., Thessaloniki, Greece, sarafis@mycompany.com.gr

Konstantinos Karamitsios
My Company Projects O.E., Thessaloniki, Greece, kk@mycompany.com.gr

Konstantinos Kotrotsios
My Company Projects O.E., Thessaloniki, Greece, kotrotsios@mycompany.com.gr

Kalliopi Kravari
Department of Information and Electronic Engineering, International Hellenic University, Greece, kkravari@ihu.gr

Costin Badica
Computer and Information Technology Department, University of Craiova, Romania, costin.badica@edu.ucv.ro

Periklis Chatzimisios
Department of Information and Electronic Engineering, International Hellenic University, Greece, pchatzimisios@ihu.gr

The job market is continuously evolving. The specific occupations, skills, competences and qualifications that people need change over time, as does their description. To deal with this, effective and intelligent communication and information exchange between the job market and the education and training sector is vital. On the other hand, and from the perspective of the individual (job seeker), especially the less privileged there is a need for approaches that combine practical tools with motivation and mentoring support since skill-matching it is not enough, skill-building is also needed. In this context, the current approach follows a bottom-up methodology investigating the problem of formalizing the lifelong learning process in a dynamic and flexible way. On the other hand, this proposal utilizes a parallel top-down approach in applying semantics and standards upon data in order to alleviate the gap among individuals, workplaces and educational contexts for the benefit of all in a transparent way. More specifically, this article reports towards an approach on tackling the complex task of interconnecting job seekers, employers and educational agents in the current European labor market. To perform this task, we implement an end-to-end service to parse resumes, job descriptions and open courses descriptions, retrieve information on the qualifications associated with the aforementioned, and semantically match them. The proposed implementation effectively detects the underlying information associated with those sources, and manages to interlink job seekers' resumes to occupations and job vacancies, while being able to assign skill deficits to courses provided by educational agents. The performance of our implementation on CVs, job descriptions and course descriptions in English, Greek, Romanian and Bulgarian, indicate that our approach yields results on par with the state-of-the-art, however on a much larger scale: to the best of our knowledge, this is the first research work that engages with this task on three stakeholders (job seekers, employers, educational agents) and in four European languages.

CCS CONCEPTS • Information systems • Information retrieval • Users and interactive retrieval

Additional Keywords and Phrases: Information Retrieval, Linked Data, Semantic Web, ESCO

ACM Reference Format:

Ioannis - Aris Kostis, Dimitrios Sarafis, Konstantinos Karamitsios, Konstantinos Kotrotsios, Kalliopi Kravari, Costin Badica, Periklis Chatzimisios. 2022. Towards an Integrated Retrieval System to Semantically Match CVs, Job Descriptions and Curricula. In PCI '22: Pan-Hellenic Conference on Informatics 2022, 25 - 27 November, Athens, Greece. ACM, New York, NY, USA, 10 pages.

1 Introduction

Job seekers currently face an unprecedented period of accelerated digitization and economic uncertainty due to the COVID-19 pandemic. Moreover, the skills, competencies, and qualifications that people need change over time, as does their description. Actually, in today's perpetually evolving labor market, the skills and competencies required continuously change as job vacancies become more specialized and, consequently, demanding. It is thus crucial to interconnect the triangle of the job market, education, and Life-Long training. From the job seekers perspective, skill building and life-long learning constitutes a necessity in order for them to be able to adapt in those circumstances. This created skill gap is proven to be even greater for under-privileged job seekers, such as graduands, inexperienced workers, long-time unemployed ones, socially excluded, etc. Educational agents, primarily universities, are slowly becoming unable to provide adequate, specialized knowledge in order for younger people to be able to keep up with the requirements of the industry, especially in more rapidly advancing ones, such as IT. This gap is being filled by an increasing amount of private educational agents (e.g. Coursera, EdX, Udemy, etc.), with a plethora of Massive Open Online Courses, some of which are taught by trained professionals and are continuously upgraded to match the standards set by the labor market. On the other hand, recruiters and hiring manages frequently find themselves in front of a deluge of CVs. Glassdoor, in 2015, stated that each job offer attracts 250 resumes [1]. Naturally, the amount of work for one recruiter multiplies accordingly when having to serve more than one job openings.

In order to address this educational, skill and talent gap, in our work we propose a novel framework addressed towards all three of the aforementioned stakeholders. Through the use of modern technologies, we manage to design and implement an application to semantically match resumes, job descriptions and units of learning, therefore giving the potential to job candidate's to apply for a job most suitable to them, or further educate themselves towards their desired career path, companies to adjust and optimize their hiring process in context of the candidate availability, and educational agents to adjust and upgrade their content according to the state of the labor market and the skill gap between it and the candidate pool. Hence, this proposal follows a bottom-up methodology investigating the problem of formalizing the lifelong learning process in a dynamic and flexible way. On the other hand, it utilizes a parallel top-down approach in applying semantics and standards upon data in order to alleviate the gap among individuals, workplaces and educational contexts for the benefit of all in a transparent way.

The rest of the paper is organized as follows: In Section 2 we review recent research works that revolve around Information Retrieval from CVs, job descriptions and online courses, as well as systems that interlink the aforementioned in pairs. In Section 3 we present our methodology followed in developing our prototype application. The data involved in our approach, their method of acquisition, the experiments related to our work and the final results of it are presented in Section 4. Ultimately, we summarize our conclusions on Section 5.

2 Related Work

Over the past few years, several research works have been presented revolving around automatic CV annotation, Information Retrieval (IR) from CVs, expediting CV screening, connecting job candidate CVs to job vacancies, and even attempts on linking the labor market with the educational one.

In their 2016 work, Zimmerman et al. [2] present their CV analysis prototype, targeted towards recruiters or hiring managers. Through the utilization of Natural Entity Recognition (NER), they were able to parse each segment of the candidates' CVs. Machine Learning (ML) and Natural Language Processing (NLP) were used to identify past work experience, education and specific hard or soft skills. Given a certain job vacancy, the candidates are then filtered based on employment criteria defined by the employer themselves. In 2017, Van-Duyet et al. [3] approach the subject from the angle of the recruiter by proposing a novel NLP approach, called Skill2vec, in order to determine the skills required for a job by its description. Basing their approach on the functional methodology of Word2Vec [4], they manage to convert extracted (hard) skills from job descriptions into word vectors and associate them with other skills, based on their occurrence in the same job description.

In 2018, Maheshwary and Misra present a more complex approach on semantically matching semi-structured job candidates' CVs to job descriptions [5]. Their approach is based upon Deep Learning (DL) methods, namely Siamese Convolutional Neural Networks (CNNs). By converting the CVs and job descriptions into vectorized word embeddings, they reduce the problem at hand at a classification task, trying to identify whether a CV is a match to a job or not based on an accordingly labeled dataset. In a similar manner, Fernandez-Reyes and Shinde in their work of the same year [6], they propose and approach based also on Word embeddings, both trained and pre-trained. The matching is conducted through a cosine similarity function. In 2019, Bhatia et al. [7] utilized the BERT [8] language model in order to classify retrieved, structured information from CVs into suitable or not for a certain job based on the latter's description. Finally, in 2022, Vo et al. [9] presented a more complex, DL-based hybrid course recommendation system. NER and NLP is utilized in order to extract skills contained in job postings and course descriptions. The extracted data were into an Long Short-Term Memory (LSTM) - based Neural Network, amplified with BERT and FastText [10] in order to perform text classification. Within the context of a certain domain (IT), the proposed model yielded promising results.

All of the aforementioned approaches focus on specific vocational domains, if not on specific job vacancies. In our work, we propose a singular, end-to-end system, to retrieve information and semantically match candidate resumes, job descriptions and (open) courses, across the European labor market and in four languages providing added-value to the less privileged.

3 Methodology

3.1 Use Cases

During the implementation and evaluation phase of the Semantic Web based Skill-Matching system a variety of use cases are taken into account. This subsection presents part of these uses cases for purposes of better understanding. Mention that JV stands for Job Vacancy and CV stands for Curriculum Vitae.

USE CASE	No. 1
Title:	Recommend JVs to a job seeker
Actors	Job seeker
Precondition:	A number of JVs uploaded in the database
Story:	An individual enters the web-app in order to search for some jobs. He uploads his CV which passes through the Annotator module. The module extracts his skills and previous work experience which are then matched to ESCO skills and occupations. Then, based on the percentage the extracted CV skills match the required JV skills, the system proposes him the top job vacancies.
USE CASE	No. 2
Title:	Recommend a few occupations to a job seeker
Actors	Job seeker
Precondition:	None extra
Story:	An individual enters the web-app in order to search for some jobs. He uploads his CV which passes through the Annotator module. The module extracts his skills and previous work experience which are then matched to ESCO skills and occupations. Based on the ESCO info extracted from the CV, the system proposes a few occupations to him.
USE CASE	No. 3
Title:	Recommend a few courses to a job seeker
Actors	Job seeker
Precondition:	A number of courses uploaded in the database
Story:	An individual enters the web-app in order to search for jobs and he clicks on a specific job vacancy. He has already uploaded his CV, meaning it has been processed by the Annotator module which has extract his ESCO matching skills and occupations. The system finds which skills required by that specific JV are missing from the individual's CV. It then proposes him a number of courses that provide those skills.
USE CASE	No. 4
Title:	Help a company improve a JV's description
Actors	Company/Employer
Precondition:	None extra
Story:	A company enters the web-app and uploads a few JVs. These JVs go through the Annotator module and the ESCO skills and occupations are extracted. Then the system, based on the ESCO info extracted proposes some changes the company can make to the job description in order to make it more compatible with the ESCO format/ontology.
USE CASE	No. 5
Title:	Help a company improve a course's description
Actors	Educator
Precondition:	None extra
Story:	A course-provider enters the web-app and uploads a new course. The course goes through the Annotator module and the ESCO skills and occupations are extracted. Then the system, based on the ESCO info extracted, proposes some changes the course-provider can make to the course's description in order to make it more compatible with the ESCO format/ontology.

As revealed the chosen use case set includes different types of cases, such as job offers/recommendations, course recommendations, company support on description improvement, etc, in order to cover the whole range of needs.

3.2 Data Acquisition

In our approach, we utilize two types of data in order to implement our proposed concept. As reference data - or as a "common vocabulary" - we utilize the European Skills, Competencies and Occupations classification "ESCO" [11]. ESCO constitutes a 27-language semantic framework that describes Skills, Occupations and their relations. Every Occupation is connected to a set of Essential Skills to it, and a set of Optional Skills to it. Similarly, a Skill has a set of Occupations that it is Essential to, and a set that it is Optional to. As a data model, it is published as Linked Open Data and it is provided by the European Commission in RDF format [12].

The second type of data utilized in our work, we call "user-data". User data comprise of candidate CVs, job vacancy descriptions (JVs) and open online courses descriptions (UoLs). CVs were crowd-sourced through past open job vacancies available in our company. JVs and UoLs were web-crawled by job searching, open courses or online courses websites.

In order to guarantee the functionality of our approach, data were acquired in all four languages of the project: English, Romanian, Bulgarian and Greek. The final dataset is described in Subsection 4.1.

3.3 Skill Extraction

The common axis between the three stakeholders constituting our target audience for the project is "Skills": job candidates (CVs) have Skills, jobs (JVs) *require* Skills and courses (UoLs) *provide* Skills. In that context, we aim to extract from the three aforementioned data sources both the Skills and Occupations that describe them.
The extraction process is common for all three data sources, and operate as follows. Initially, each entity is parsed to acquire its content in text form. We proceed into tokenizing the collected text. Stop words, punctuation, accents, newline characters and tabs are then removed, and the tokens are transformed into lower case characters. The then filtered tokens are parsed into creating lexical bi-grams (e.g. "artificial intelligence") and tri-grams (e.g., "computer vision engineer"). Each filtered token, bi-gram and tri-gram is looked up in the "preferred" and "alternative" labels of Skills and Occupations, as they are defined and provided by ESCO. By the end of this matching process, we have acquired sets $S^e = \{s_1^e, s_2^e, \ldots, s_m^e\}$ of extracted Skills and $O^e = \{o_1^e, o_2^e, \ldots, o_n^e\}$ of extracted Occupations accordingly.
In the next step of the process, we aim to "verify" each extracted Skill and Occupation, by matching every extracted value to an ESCO "preferred" label, thus acquiring sets $S^v = \{s_1^v, s_2^v, \ldots, s_i^v\}$ and $O^v = \{o_1^v, o_2^v, \ldots, o_j^v\}$, containing the "preferred" labels of the extracted Skills and Occupations respectively. As a final step, we retrieve the Skill set associated with each verified Occupation and we incorporate it to the S^v, therefore acquiring $S_o^v = \{s_1^v, s_2^v, \ldots, s_i^v, s_1^{ov}, s_2^{ov}, \ldots,, s_k^{ov}\}$, post omitting any duplicates.
In the context of using solely Skills as the common vocabulary between stakeholder data, we replace every attributed Occupation to them with its associated Skills in order to be able to match previous experience in a certain occupational role between a candidate and a job, while simplifying the matching process. Aligning with our methodology, both Skills and Occupations are extracted from CVs and JVs, while Occupations located into UoLs are not taken into account.

3.4 Data Interconnection

In order to handle the data provided by the ESCO RDF model, we load the RDF Turtle (TTL) graph on a Triples Database (TDB). Being able to query it, we implement a set of wrapper functions including queries involving the retrieval of Occupations given the Skills, and vice versa. Through 18 queries we manage to cover all available "one-to-many" relationships between these entities, along with every "many-to-many" relationship, therefore acquiring access to the full range of information provided. This set of wrapper functions serves as the infrastructure of our overall implementation.
Our main focus in our work is to link all job candidates to available jobs, or even Occupations, while covering skills they are missing through relevant courses. Since, as stated above, the link between the three stakeholders is Skills, we proceed into storing every available CV, JV and UoL into the TDB. Each data entity is described by a name, an ID, a set of skills, a URI, and, in the case of JVs and UoLs, the publisher's name.
By calculating the overlap percentage between each candidate's skill set and the set of skills required by a job or occupation, we are able to define an indicator of the candidate's suitability to the job/occupation at hand. To achieve that, we implemented a second set of queries to the TDB, involving the calculation of these scores through the intersections.

Let S_o^v denote the finalized set of verified skills extracted from a CV, $\tilde{S}_o^v$ be the finalized set of verified skills extracted from a JV, and the symbol # denote multitude. The suitability score between a candidate (CV) and a job is calculated as follows:

$$score_{job} = 1 - \frac{\#(S_o^v - \tilde{S}_o^v)}{\# \tilde{S}_o^v}$$

Let E_s denote the set of the Essential Skills attributed to an Occupation, and O_s the set of the Optional ones. The suitability score between a candidate (CV) and an Occupation is calculated as follows:

$$score_{occ} = 0.75 * \frac{\#(S_o^v \cap E_s)}{\# E_s} + 0.25 * \frac{\#(S_o^v \cap o_s)}{\# o_s}$$

The results of each of the aforementioned implemented queries are rendered available for further processing through an API.

4 IMPLEMENTATION

4.1 Operational Setup

Out implementation has been developed in its entirety in Python 3.8. Data pre-processing and processing has been implemented using the NumPy [13] and Pandas [14] frameworks. All wrapper functions have been implemented through the SPARQLWrapper framework. Text tokenisation has been conducted through NLTK [15]. The API has been implemented using Flask [16], the WSGI application server is implemented through Gunicorn and the web server, operating as a proxy to the application server, is implemented through Nginx. The TDB is hosted on an Apache Jena Fuseki SPARQL server, and all queries are implemented in SPARQL.

4.2 Dataset & Data Integration

The annotators involving the extraction of Skills from CVs, JVs and UoLs accordingly, output a skillset related to the type of input/stakeholder. Apart from Skills, a defined set of features are assigned to each entity based on its nature. These features include the following data:

CVs	
name:	the name of the candidate
skills:	the set of Skills, as exported from the annotator
occupations:	the set of Occupations, as exported from the annotator
uri:	The URI/URL indicating to the CV
md5_checksum:	a checksum exported from an MD5 hashing function, generated from the content of the initial input. Due to its nature, it serves as a unique identifier to the entry
JVs	
job_title:	the title provided to the vacancy by the employer
publisher_name:	name of the employer / company that posted the vacancy
skills_required:	the set of skills required by the position, as exported from the annotator
esco_title:	the ESCO defined title for the Occupation the position is about. Exported by the annotator
uri:	The URI/URL indicating to the JV
md5_checksum:	a checksum exported from an MD5 hashing function, generated from the content of the initial input. Due to its nature, it serves as a unique identifier to the entry
UoLs	
uol_title:	title of the UoL/course, as provided by its publisher
publisher_name:	name of the educational agent providing the UoL/course
skills_provided:	set of skills provided by the UoL upon completion. Exported by the annotator
uri:	The URI/URL indicating to the UoL
md5_checksum:	md5_checksum: a checksum exported from an MD5 hashing function, gen-erated from the content of the initial input. Due to its nature, it serves as a unique identifier to the entry

The final feature assignment is streamlined along with the annotation process through Python functions, specifically designed for each type of stakeholder, resulting in a Python dictionary for its entry. Upon completion of feature assignment, these dictionaries are converted into JSON arrays and stored with the appropriate queries into the RDF TTL Graph. The queries involving storage are described in the upcoming paragraph. We further decided to implement and include a set of functions conducting this operation in batch, given the directory path where the accumulated data of each stakeholder are stored.

Regarding the relations of information included into CVs, JVs and UoLs and how it is interconnected to the Skills and Occupations included in the graph, the connection was conducted with the assistance of Concepts provided by the RDF, RDFS, FOAF and SARO ontologies. The relations are described via SPARQL syntax and through a relevant ontology vocabulary. Part of this ontology, referred to Cvs, is depicted below:

```
<cv_uri> rdf:type saro:User.

<cv_uri> foaf:name <name>

<cv_uri> rdfs:label <md5_checksum>

<cv_uri> saro:hasSkill <skill_uri>
```

Upon the incorporation of the data to the graph, the graph can serve queries regarding both Skills and Occupations from ESCO, as well as queries involving the data of the stakeholders.

4.3 SPARQL Queries: Integration & Interconnection

For the purpose of inserting, accessing, modifying and deleting stakeholder's data from the graph, 12 SPARQL queries were implemented, 4 for each stakeholder. The queries are wrapped within Python functions that streamline the process and handle the responses of the requests to the Apache TDB. The functionality of the wrapper functions for the queries, categorized by stakeholder, is as follows:

CVs	
Retrieve CVs	depending on whether a specific md5 checksum was provided as input to the function when called, the function returns either one specific CV based on the md5 hash, or all of the CVs
Store CV	Provided the JSON array including the features assigned to the CV and describing it, the CV is stored accordingly to the Apache Jena TDB
Delete CV	Provided either the URI or the md5 hash assigned to a CV, the corresponding CV is deleted from the Apache Jena TDB
Update CV	Provided either the URI or the md5 hash assigned to an al-ready existing CV, along with a JSON array containing the new and updated information regarding this particular CV, its information and features are up-dated accordingly
JVs	
Retrieve JVs	depending on whether a specific md5 checksum was provided as input to the function when called, the function returns either one specific JV based on the md5 hash, or all of the JVs available
Store JV	Provided the JSON array including the features assigned to the JV and describing it, the JV is stored accordingly to the Apache Jena TDB
Delete JV	Provided either the URI or the md5 hash assigned to a JV, the corresponding JV is deleted from the Apache Jena TDB
Update JV	Provided either the URI or the md5 hash assigned to an already existing JV, along with a JSON array containing the new and updated information regarding this particular JV, its information and features are updated accordingly
UoLs	
Retrieve UoLs	depending on whether a specific md5 checksum was pro-vided as input to the function when called, the function returns either one specific UoL based on the md5 hash, or all of the UoLs available
Store UoL	Provided the JSON array including the features assigned to the UoL and describing it, the UoL is stored accordingly to the Apache Jena TDB
Delete UoL	Provided either the URI or the md5 hash assigned to a UoL, the corresponding UoL is deleted from the Apache Jena TDB
Update UoL	Provided either the URI or the md5 hash assigned to an al-ready existing UoL, along with a JSON array containing the new and updated information regarding this particular UoL, its information and features are up-dated accordingly

Finally, having all the stakeholders' data stored into the TDB, we are able to implement methods to conduct the interconnections between these and the data provided by ES-CO involving the Skills and Occupations. These methods operate in a hybrid manner, by initially acquiring data from the graph and performing tasks and mathematical operations involving them. Two query wrappers were implemented as utility functions in order to assist these methods.

Example: Candidate to Occupation Score

Provided the URI of a candidate (CV) and the preferred label of an Occupation, this method calculates a score between 0 and 1 regarding the percentage of matching based on the Skills owned by them and the Skills required by the Occupation. For performance reason, the opera-tion is conducted solely through a SPARQL query. Within the query, we retrieve the Skills in common, splitting them in Essential and Optional. After that, we di-vide their count with the count of the Essential and Optional Skills respectively required by the Occupation. We factor their contribution in a ratio of 3:1, thus we multiply the fractions by 0.75 and 0.25 respectively. Finally, in order to produce the overall score, we add the two individual scores, and we return the result

4.4 API: Endpoints

The endpoints provide URLs that expose the aforementioned resources to requests. All of the endpoints implement either POST or GET type of requests. The server responds to the requests either by code 200 on success, along with any data requested, and with 4XX codes in case of bad requests, timed-out requests, or failure to execute the operation requested. In case of internal server errors, the sender receives a 501 error code. Examples of these endpoints are depicted below:

Title	Method	Syntax	Response
Essential Skills for Occupation	GET	curl --location --request GET '<host_ipv4>:<port>/query/essential_skills_for_occupation/<occupation_pref_label>'	{ "essential_skills": ["skill_1", "skill_2", "...", "skill_N"] }
Retrieve CV	GET	curl --location --request GET '<host_ipv4>:<port>/query/retrieve_cv/<md5_hash>'	{"md5_checksum": <md5_hash>, "skills": ["skill_1", "skill_2", "skill_N"], "uri":<cv_uri> }
Store CV	POST	curl --location --request POST '<host_ipv4>:<port>/store/cv?name=<candidate_name>&cv_uri=<cv_uri>'	"CV stored successfully. Your ID is: <md5_hash>}"
Store CV batch	POST	curl --location --request POST '<host_ipv4>:<port>/store/cv/batch'	"Started Storing CVs."

Title	Method	Syntax	Response
Update JV URI	POST	curl --location --request POST '<host_ipv4>:<port>/update/jv/uri?jv_uri=<jv_uri>' \ --header 'Content-Type: application/json' \ --data-raw '{ "job_title": <job_title>, "publisher_name" : <publisher_name>,"skills_required": ["skill_1", "skill_2", "skill_N"], "md5_checksum":<md5_hash> }'	"JV <jv_uri> updated successfully. Your ID is: <md5_hash>."

5 Conclusion

The proposed approach revolves around the implementation of an infrastructure that facilitates the communication between the three stakeholders: the EU citizens (Job Candidates), the EU Companies (Employers) and the EU Educators (Educational Agents). In order to tackle the issue of interconnecting CVs, JVs and UoLs, we based our approach upon concepts of Linked (Open) Data and the Semantic Web. Interlinking these three entities through the Skills they incorporate and via a common vocabulary, constitutes our integral design axis. More specifically, this article reports on the first goal of our project which is to scaffold candidates to match their professional profiles with digital learning resources and receive access to available content regarding their skills and external relevant knowledge from multiple courseware providers, investing on the linked data approach. Towards dealing with the issue, as reported in this paper, such matching typically imposes the key challenge of having software systems talking to each other in a common language supported by semantics that understood from all sides. Hence, the ambition of the proposed approach is to focus on learning outcomes enhanced and annotated by semantics found in standard specifications to enhance competence-based matching. The European Skills/Competences, Qualifications and Occupations multilingual (ESCO) classification, when combined with IMS educational standards, provide a promising approach towards the solution of bridging gaps amongst individuals, jobs and teachers. On the other hand, we pay attention to skill building, hence, our approach includes a variety of artificial intelligence such as agent technology. In this context, the second goal, which is currently under implantation, is to provide intelligent agents as-a-service (web accessed) in order to provide virtual mentoring to each individual jobseeker. This mentoring will use the knowledge bases and reasoning (inference machines) and will associate one (or more) agent to each individual. This agent will store knowledge and preferences even beliefs and desires of the individual and then it will provide guidelines for job opportunities or life-long training using the rest tools of the approach. To achieve the above objectives, we intend to further extend the proposed multi-step and multi-layered research approach.

ACKNOWLEDGMENTS

This research was co-funded by the Erasmus+ programme of the European Union, under the “2021 Round 1 KA2” call; project title: “Bridging the Gap – An AI-enabled versatile skill matching tool to assist the less privileged”, project reference: 2021-1-EL02-KA220-YOU-000028780.

REFERENCES

<bib id="bib1"><number>[1]</number>“50 hr recruiting stats that make you think - glassdoor”. Retrieved from: https://www.glassdoor.com/employers/blog/50-hr-recruiting-stats-make-think/</bib>
<bib id="bib2"><number>[2]</number>T. Zimmermann, L. Kotschenreuther, and K. Schmidt. 06 2016, Data-driven hr - resume analysis based on natural language processing and machine learning. Retrieved from: https://arxiv.org/abs/1606.05611</bib>
<bib id="bib3"><number>[3]</number>V.-D. Le, M.-Q. Vo, and D. Quang-An. 07 2017. Skill2vec: Machine learning approaches for determining the relevant skill from job description. Retrieved from: https://arxiv.org/abs/1707.09751</bib>
<bib id="bib4"><number>[4]</number>T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. 09 2013. Efficient estimation of word representations in vector space. Retrieved from: http://arxiv.org/abs/1301.3781</bib>
<bib id="bib5"><number>[5]</number>S. Maheshwary and H. Misra. 2018. Matching resumes to jobs via deep Siamese network. In Companion Proceedings of the The Web Conference 2018, ser. WWW ’18. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2018, p. 87–88. Retrieved from: https://doi.org/10.1145/3184558.3186942</bib>
<bib id="bib6"><number>[6]</number>F. C. Ferńandez-Reyes and S. Shinde. 2019. Cv retrieval system based on job description matching using hybrid word embeddings. Computer Speech & Language, vol. 56, 73–79, 2019. Retrieved from: https://www.sciencedirect.com/science/article/pii/S0885230817302851</bib>
<bib id="bib7"><number>[7]</number>V. Bhatia, P. Rawat, A. Kumar, and R. R. Shah. 2019. End-to-end resume parsing and finding candidates for a job description using bert,” ArXiv, vol. abs/1910.03089.</bib>
<bib id="bib8"><number>[8]</number>J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. Retrieved from ArXiv, vol. abs/1810.04805, 2019.</bib>
<bib id="bib9"><number>[9]</number>N. N. Vo, Q. T. Vu, N. H. Vu, T. A. Vu, B. D. Mach, and G. Xu. 2022 .Domain-specific nlp system to support learning path and curriculum design at tech universities,” Computers and Education: Artificial Intelligence, vol. 3, 100042. Retrieved from: https://www.sciencedirect.com/science/article/pii/S2666920X21000369</bib>
<bib id="bib10"><number>[10]</number>A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. 2016. Bag of tricks for efficient text classification,” arXiv preprint arXiv:1607.01759.Integrated Retrieval System to Match CVs, Jobs and Curricula 7</bib>
<bib id="bib11"><number>[11]</number>M. le Vrang, A. Papantoniou, E. Pauwels, P. Fannes, D. Vandensteen, and J. De Smedt. 2014. Esco: Boosting job matching in europe with semantic interoper-ability,” Computer, vol. 47, no. 10, 57–64.</bib>
<bib id="bib12"><number>[12]</number>J. D. Smedt, M. le Vrang, and A. Papantoniou. 2015. Esco: Towards a semantic web for the european labor market,” in LDOW@WWW.</bib>
<bib id="bib13"><number>[13]</number>C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. Fern´andez del Rıo, M. Wiebe, P. Peterson, P. G´erard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, “Array programming with NumPy. 2020. Nature, vol. 585, p. 357–362.</bib>

<bib id="bib14"><number>[14]</number>W. McKinney et al. 2010. “Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference, vol. 445. Austin, TX, 2010, 51–56.</bib>
<bib id="bib15"><number>[15]</number>S. Bird, E. Klein, and E. Loper. 2009. Natural language processing with Python: analyzing text with the natural language toolkit. ” O’Reilly Media, Inc.</bib>
<bib id="bib16"><number>[16]</number>M. Grinberg. 20018. Flask web development: developing web applications with python. O’Reilly Media, Inc.</bib>