Acta Scientiarum
http://periodicos.uem.br/ojs/acta
ISSN on-line: 1807-8664
Doi: 10.4025/actascitechnol.v47i1.72998



# Semantic Interpretation in Job Recommendations: Ontology-Driven Resume Parsing for Personalized Job Recommendation Systems

**Duygu Çelik Ertuğrul*** **and Selin Bitirim**

Department of Computer Engineering, Faculty of Engineering, Eastern Mediterranean University, 99628, Famagusta, North Cyprus, Mersin 10, Turkey.
*Author for correspondence. E-mail: duygu.celik@emu.edu.tr

**ABSTRACT**. In this study, an ontology-based Hybrid Job Recommendation System (HJRS) has been proposed to facilitate job seekers in finding the right positions and employers in identifying the most suitable candidates in a complex and dynamic job market. This system has been developed to better meet the needs inadequately addressed by traditional job recommendation systems (JRS), by combining both syntactic and semantic approaches. HJRS consists of three main components: (1) Resume Recommender System Ontology—RRSO, (2) Ontology-Based Resume Text Parsing Module (OntRTPM), and (3) System Database. While RRSO is used to semantically interpret job postings and candidate information, OntRTPM extracts the necessary personal information, educational background, work experiences, and skills from candidates' resumes. The candidate extracted data is saved in the appropriate field of the system database, and an OWL file is created for the candidate. The system operates through four main steps: (1) preprocessing, (2) feature processing, (3) ontology matching and labelling, and (4) saving structured resume data. In the preprocessing stage, data is cleaned and normalized; in the feature processing stage, concept extraction and semantic matching operations are performed. As a result of these processes, candidates' resumes are stored in both OWL file and the relevant tables in the system database. The dataset used consists of 100 anonymized Turkish resumes randomly selected from the database of Kariyer.Net, a large career company in Turkey. OntRTPM has been subjected to accuracy and reliability tests by extracting information from these resumes. The system aims to significantly improve job search and recruitment processes by providing more accurate and effective recommendations for both job seekers and employers in the job market. Future work will include expanding the resume dataset and further optimizing system performance.

**Keywords**: Job recommendation systems; ontology; text parsing; information extraction; semantic web.

Received on July 27, 2024.
Accepted on October 23, 2024.

## Introduction

The increase in the number of dissatisfied employees as well as the increase in the desire to change jobs in many sectors causes an increase in the number of new job searches and employee changes. Besides these, the increase in unemployment rates causes companies to advertise millions of job opportunities every day. This situation causes HR managers to have difficulties in finding the right personnel from the existing employee pools, matching and assigning them to the most suitable position in a short time, and causes a lot of time loss on the HR side. Failure to identify and assign the most suitable personnel to the target job position in a short time may cause loss of performance, loss of time, loss of productivity and service, as the workload increases.

Resumes are important documents that define individuals and reveal their career life, talent, ability, business power and experience of people. Resumes are generally used in job applications and are prepared by people to introduce their potential to employers. It contains a lot of information such as demographic information, skills, past work experience, past education status, competency certificates, etc. Resumes are usually created by the people themselves using an editor and can have various extensions such as ‘doc, docx, pdf, txt’. The formats of these documents may vary from individual to individual (e.g., different fonts, different font sizes) and may contain different structural forms (e.g., use of tables, graphics, etc.). Therefore, resumes are mostly uploaded in a non-standard way and contain unstructured data. However, individuals looking for a job can upload their resumes through the web portals of career providers that provide job/worker recruitment services or through the software products of the HR units of sector-based corporate companies.

Resumes uploaded through such portals are often uploaded in a semi-standard manner and may contain semi-structured data.

The accurate extraction of relevant information plays a critical role in recommendation systems. The effectiveness of search engines like Google in both text and image search has been the subject of numerous research studies aimed at examining the strengths and weaknesses of advanced search technologies in areas such as e-commerce (Bitirim et al., 2020; Celik & Elci, 2008; Celik & Elçi, 2005a; Celik & Elçi, 2005b). The analysis of textual and/or visual content present in resumes can enable candidates to present their competencies and experiences more effectively. Therefore, effective search approaches can significantly contribute to the extraction and interpretation of the key elements within resumes.

The general purpose of this study is to extract essential data from resumes, such as personal profile information, past work experiences, education details, and competencies, from unstructured resumes stored in the HR resume pools of companies and to store this information in a more structured format.

While numerous Job Recommender Systems (JRS) have been developed in English (e.g., CASPER, ResuMatcher, eRecruiter, FES, etc.), business JRS solutions in Turkish are underrepresented in the literature. There are a few popular JRS solutions in Turkish (e.g., kariyer.net, secretcv.com) that assist job candidates and companies in quickly finding job opportunities and accurately matching candidates. Therefore, this study develops a Turkish-supported Ontology-Based Resume Text Parsing Module, designed to integrate into a Hybrid Job Recommendation System. Its goal is to effectively match suitable candidates with current job openings in HR departments. A dataset consisting of 100 Turkish resumes is used to assess the system performance. A resume ontology knowledgebase is developed to support, specify and labelling the general concepts of crucial information in resumes such as profile, educational information, work experiences, skills and qualifications in resumes written in Turkish language.

The remainder of this article is structured as follows: Section 2 reviews relevant literature, focusing on studies that classify and compare different methodologies and techniques used in JRSs. Section 3 introduces the proposed system, detailing its components and operational mechanisms. Additionally, it provides insights into the resume dataset utilized and outlines the structure of the collected resumes. Section 4 elaborates on the system components, methodologies employed, techniques applied, and materials utilized. This is followed by Section 5, which presents a case study illustrating the application of the proposed system. Section 5 presents the findings and evaluates the outcomes of this research. Section 8 concludes with future avenues for research.

## Literature review

Pawar et al. (2021) designed a model to extract career and education details from resumes. This model was produced to overcome the difficulty of extracting information from unstructured/differently written resumes. An approach that divides an unstructured document into meaningful parts is proposed. The authors implemented their work in a common neural model consisting of two sequential labeling layers: a horizontal Bidirectional Long Short-Term Memory-Conditional Random Field (BiLSTM-CRF) layer (for words in resume) and a vertical BiLSTM-CRF layer (for sentences in resume). BiLSTM with Global Vectors for Word Representation (GloVe) embedding layer was used for faster inference and better management of resource constraints. As a pre-processing step, regular expressions (RE) were applied. They used BIO (Inside–Outside–Beginning) encoding for entity type labels for words such as, B-Employer, I-Degree, O for extracting features. In classification step, horizontal BiLSTM-CRF used over words and vertical BiLSTM-CRF used over sentences in resume. While measuring the performance of the system, F1 Score results were obtained to calculate precision and recall evaluation metrics. According to these results, When the proposed techniques were compared with the basic method, it was found that the proposed techniques showed 10% higher performance than the basic method. While the sequential model obtained the highest F1 score for EDU, the joint model obtained the highest F1 score for CAREER.

Çetindağ et al. (2023) employed both machine learning- and deep learning-based approaches for named entity recognition in Turkish legal texts. Specifically, they utilized Conditional Random Fields (CRF) alongside deep learning architectures such as BiLSTM, character-level CNN, and character-level BiLSTM. In addition, they experimented with word embedding techniques, including GloVe and Morph2Vec, and achieved the highest performance by combining GloVe, Morph2Vec, and a character-based BiLSTM within a hybrid architecture.

Batbaatar and Ryu (2019) designed a system which predicts health-related named entities such as diseases, symptoms, and pharmacologic substances from noisy Twitter messages. They applied pre-processing methods which involves lower case conversion, removing URLs, unwanted characters and non-English characters from the texts. Researchers developed a Health Name Entity Recognition (HNER) application in the field of health to identify/categorize diseases, medications and symptoms, which can recognize/analyze health-related entities mentioned in Twitter messages. A Reversal Neural Network (RNN) based architecture has been implemented. BiLSTM model and Convolutional Neural Network (CNN); Conditional Random Field (CRF) model was used for labeling and prediction. The BiLSTM-CRF model and POS (Parts-Of-Speech) tagging are used to improve performance. They applied feature extraction techniques such as Word2Vec, POS tagging and GloVe for word and character embeddings. CNN and BiLSTM-CRF are applied to extract related entities and identify the relationship between them from Twitter messages for classification step.

Mittal et al. (2020) have developed a resume parser, by applying Name Entity Recognition (NER) techniques, RE and pattern matching methods were applied to identify names, phone numbers, emails, education, skills, languages, city, etc. on 100 resumes collected from students and job seekers belonging to different universities. Pre-processing, feature extraction and classification techniques were applied while parsing the resumes. As the pre-processing step in parsing the resumes, the text was cleaned by applying lower case conversion, RE, tokenizing the text and remove stop words. The other step is the feature extraction step. To extract the technical skills and soft skills from a candidate's resume feature extraction was performed by applying Term Frequency and Inverse Document Frequency (TF-IDF) and cosine similarity technique. The candidate's characteristics are estimated using the logistic regression classification technique.

JRSs are typically handled using filter-based methods or recommendation systems based on categorical job and candidate characteristics. Nigam et al. (2019) developed a new machine learning model that focuses on job preferences. With this JRS, in addition to providing recommendations, it also aims to overcome challenges such as generating unexpected recommendations and solving the cold start problem for new jobs and candidates. As a feature extraction part, Word2Vec and Continuous Bag of Words (CBOW) were applied on a dataset (Dataset contains 4208 unique candidates and 2334 unique jobs) of chatbot queries to represent job and candidate skills. Multiple machine learning models were used in classification parts such as BiLSTM-A, Natural Language Processing (NLP), linear regression, Random Forest (RF), eXtreme Gradient Boosting (XGBoost), Artificial Neural Network (ANN). The system was evaluated, and F1-Score of BiLSTM-A is the best compared to other models are concluded by the researchers. Pre-processing step was not applied on their JRS.

Xu et al. (2018) proposed an RS based on XGBoost classifier. This model has created a consumer behavior system based on consumer behavior records/product recommendations on the shopping website (Alibaba), emphasizing the importance of online shopping. However, in response to challenges posed by traditional collaborative filtering-based recommendation systems, they proposed an RS model using the XGBoost algorithm. They applied XGBoost algorithm for feature extraction step. The system had good performance (F1-Score=%7,97) with XGBoost. In addition to these algorithms, NLP, RF, Support Vector Machine (SVM) were applied on their RS system as a classification step. Pre-processing step was not applied on their system.

Qin et al. (2019) focused on question RS for intelligent job interview. The purpose of this study is to find the right candidates for the open job position. In this study, a system developed to accurately evaluate the skills and experience of candidates with DuerQuiz, a question-answer system in job interviews. Skill-Graph was created as a job skills knowledge graph to comprehensively model relevant competencies that should be evaluated in a job interview. The Skill-Graph was used to increase the efficiency of job interview evaluations. A new capability entity extraction approach based on LSTM and CRF layer (LSTM-CRF) neural network is proposed. Cosine similarity, Linear Discriminant Analysis (LDA), POS Tagging and Word2Vec are used for feature extraction. BiLSTM-CRF, SVM and RF are applied for classification. Pre-processing step was not applied on their system.

Fernández-Reyes and Shinde (2019) have designed a system to obtain relevant resumes based on job description. The authors used the Average Word Embedding (AWE) model to infer relevant resumes based on the job description. Word embedding model is proposed and Word2Vec, Skip-Gram which is word prediction algorithm and CBOW word embedding techniques were applied on resumes. PCA algorithm was applied for feature extraction. At the evaluation stage, firstly, a balanced corpus of resume summaries obtained from ever is Mexico's recruitment department was produced and 4 resumes were identified as Java, Tester, SAP HCM and SAP DSD-SD. Then, a test set (585 resumes) was created using resumes uploaded from job requests of

different projects. Precision, recall, Mean Reciprocal Rank (MRR) metrics were used for evaluation. Pre-processing and classification steps were not applied on their system.

Gaur et al. (2020) have presented a model for determining the names and degrees of educational institutions from the education section of the resume. They applied pre-processing methods which involves remove irregular spaces and unwanted characters, conversion document format, resolving unbalanced parenthesis and replacing & with and. They applied feature extraction techniques such as POS tagging, Levenshtein Distance and BIOES/IOBES (2024). BILSTM-CRF and CNN are applied for classification.

Jiechieu and Tsopze (2021) have created a model for identifying and extracting skills expressed in documents such as resumes or job postings. They applied Word2Vec as a feature extraction technique. Logistic regression, multi text classification, the Fully Connected Neural Network (FCNN) and CNN are applied for classification. Pre-processing step was not applied on their model. A CNN-based multi-label architecture was used to predict high-level competencies from resumes. As a result of the evaluation of the system, they obtained recall (98.79%) and accuracy (91.34%) rates. They have also been observed to achieve greater than 99% accuracy for certain competencies.

Gugnani and Misra (2020) proposed a JRS to match resumes with job descriptions in non-standard and unstructured formats. Word2Vec, POS tagging, Cosine similarity and Term Frequency - Inverse Document Frequency (TF-IDF) methods are applied in feature extraction step. In the classification part, NLP tools were used. Performance evaluation was made by calculating precision/recall/F1 score metrics on 100 and 275 job descriptions. The results were precision (0.80), recall (0.93), accuracy (0.75) and F1-score (0.86).

A system designed for the field of job postings related to Software Engineering and Information Technologies has been presented by Sandanayake et al., (2018). This system can sort resumes based on various fields included in their detail, thus saving recruiters a huge amount of time and effort required for manual screening. They applied pre-processing methods which involve lower case conversion, remove irregular spaces and unwanted characters, stemming, conversion document format, lemmatization and tokenizing the text. TF-IDF methods are applied in feature extraction step. In classification part, SVM and Naïve Bayes (NB) were used. In the study, SVM classifier for the job category model was chosen for evaluation. NB classified the scores with 0.8695 accuracy, while the SVM classifier achieved 0.9778 accuracy and approximately 0.7557 accuracy for the Maximum Entropy Gain job category model.

Bafna et al. (2019) have worked to expand the term-document matrix and obtain clusters that will produce recommendations. This study focuses on providing a solution to scalability problems and ensuring consistency in cluster quality. The first step is pre-processing which involves removing stop words, stemming and resolving unbalanced parenthesis. In feature extraction part TF-IDF and cosine similarity are applied to extract information from unstructured data. Classification step was not applied in the system.

Celik (2016) presented an ontology-based recommendation system that aims to extract information from unstructured resumes. Resumes are written in Turkish and English were parsed with the proposed ontology-based information extraction system called Ontology-Based Resume Parser (ORP). During the pre-processing phase, conversion document format (.doc,.txt, etc.) and abbreviations were applied. Jaro Winkler distance algorithm is applied for feature extraction. Classification step was not applied in the system.

Das et al. (2018) have studied the text analysis process and how to extract entities with big data tools. NLP was used to parse a text into paragraphs and sentences. They applied pre-processing which involves text cleaning (removal of unwanted/inconsistent data), tokenization step. In feature extraction part, n-gram creation, POS tagging, BoW and Vector Space, semantic technologies and counting frequent words. NLP methods were applied on the study as classification part.

Deepak and Santhanavijayan (2020) applied NLP techniques to achieve the ability of HR to provide positive results in terms of the quality and speed of the entire recruitment process. Res were used for pre-processing. NLP methods were applied on the study as classification part. Feature extraction step was not applied in the system.

Mughaid et al. (2019) proposed the job search system and presented a system that makes job recommendation based on the location where a job candidate lives. This study examined the job candidate's posts history on social media networks such as Twitter and Facebook in order to match job seekers with the best job opportunities. The first step is pre-processing which involves REs. Text mining techniques and NLP classifiers were applied such as SVM, NB, and RF were applied for extracting information from textual analysis to enrich the description within a predictive algorithm. Feature extraction step was not mentioned.

Tobing et al. (2019, June) presented resume extraction system to extract content by segmenting job candidates based on the headers (e.g., Personal Information, Education, and Job Experience) in their

unstructured resumes in Indonesia. In pre-processing step Lower case conversion, conversion document format (.doc, .txt.), remove irregular spacing between words and bullet and numbering. Hidden Markov Model (HMM), Euclidean Distance and REs were used as feature extraction methods. SVM and NLP techniques were applied as a classification part.

Lin et al. (2016) presented a solution for the Resume-Job Matching problem using deep learning methods and machine learning. Researchers trained the Chinese Word2Vec model using the texts of all resumes. In this way, the average word embeddings of an expression could be represented by its semantic meaning. Classification was made by applying the K-means method, LSTM, CNN, RF, XGBoost and NLP techniques and LDA. Pre-processing step was not mentioned. To measure the performance of the system, CNN, LSTM, RF and XGB techniques were compared and while XGB showed the best performance, the CNN model was found to have acceptable accuracy.

The work by Mashayekhi et al. (2024) focuses on the unique challenges faced by e-recruitment systems, differentiating them from traditional recommendation systems. The authors conducted a comprehensive literature review, identifying 123 relevant papers to analyze the nuances of e-recruitment recommendations. They categorize the challenges and propose solutions based on recent studies, emphasizing the multi-faceted nature of data involved, the short interaction histories of job seekers, and the necessity for fairness and trustworthiness in recommendations. The paper is structured to first outline these challenges, followed by an exploration of the proposed solutions in the literature, concluding with discussions on the implications for future research in this field. The survey aims to enhance understanding and improve the effectiveness of e-recruitment recommendation systems by addressing their unique context and requirements.

In addition, understanding employee turnover and predicting potential job changes is vital for employers. Min et al. (2024) utilized machine learning techniques to compare job satisfaction with alternative job opportunities, demonstrating how recommender systems can effectively predict turnover behavior. They emphasize the correlation between the quality and quantity of desirable job alternatives and future turnover.

Zou et al. (2024) provide a comprehensive analysis of job recommendation systems, meticulously evaluating the various methodologies employed, such as collaborative filtering and content-based approaches. They highlight the strengths and weaknesses of these systems, focusing on factors like data sparsity and the importance of personalization for enhancing user experience. This critical review not only sheds light on current practices but also informs future research directions aimed at optimizing job recommendation strategies in a competitive employment landscape.

Chandak et al. (2024) proposed a resume parsing and job recommendation system that leverages machine learning and natural language processing to extract skills from resumes, highlighting the importance of semantic extraction in job recommendation processes. Together, these studies underscore the significance of effectively utilizing extracted information from resumes to enhance the accuracy and relevance of job recommendations.

Detailed comparison tables of similar studies discussed above for the Preprocessing, Feature Processing and Classification steps are presented in Tables 1, 2 and 3, respectively.

**Table 1.** Preprocessing Techniques Reported in the Literature.

| | PREPROCESSING METHODOLOGIES USED | Mittal et al. | Gaur et al. | Mughaid et al. | Tobing et al. | Bafna et al. | Batbaatar and Ryu | Pawar et al. | Sandanayake et al. | Celik et al. | Das et al. | Deepak & Santhanavijayan | Ours |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PREPROCESSING | Lower Case Conversion | * | | | * | | * | | * | | | | * |
| | Replacement Of Emojis and Emotions | | | | | | | | | | | | |
| | Remove Common Colloquial Terms | | | | | | | | | | | | |
| | Remove Stop Words | * | | | | * | | | | | | | * |
| | Replacement Letters | | | | | | | | | | | | * |
| | Remove One-Character Words | | | | | | | | | | | | * |
| | Conversion Document Format (.doc, .txt, etc.) | | * | | * | | | | * | * | | | * |
| | Remove Duplicate Files | | | | | | | | | | | | |
| | Remove All URLs and Hashtags | | | | | | * | | | | | | * |
| | Irregular Spacing Between Words | | * | | * | | | | | | | | * |
| | Replacing &Amp With &, And | | * | | | | | | | | | | |

| | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Removing Unwanted Characters, Punctuation | | | * | | | | | | * | | | * | | | | | * |
| | Resolving Unbalanced Parenthesis | | | * | | | | | | | | | | | | | | |
| | Stemming | | | | | | | | * | | | | * | | | | | * |
| | Correct Misspelled Words | | | | | | | | * | | | | | | | | | * |
| | Remove Non-English Characters | | | | | | | | | * | | | | | | | | |
| | Regular Expressions | | | * | | * | * | | | | | * | | | | | * | * |
| | Lemmatization | | | | | | | | | | | | * | | | | | * |
| | Abbreviations | | | | | | | | | | | | | * | | | | * |
| | Tokenizing The Text | | | * | | | | | | | | | * | | * | | | * |
| | Removal Bullet and Numbering | | | | | | * | | | | | | | | | | | * |

**Table 2.** Feature Processing Techniques Reported in the Literature.

| | FEATURE EXTRACTION METHODOLOGIES USED | Mittal et al. | Gaur e et al. | Xu e et al. | Fernandez-Reyes e et al. | Lin et al. | Nigam et al. | Jiechieu et al. | Gugnani et al. | Bafna e et al. | Qin et al. | Batbaatar and Ryu | Pawar et al. | Sandanayake et al. | Celik et al. | Das et al. | Tobing et al. | Ours |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FEATURE EXTRACTION | Cosine Similarity | * | | | | | | * | * | * | * | | | | | | | |
| | TF-IDF | * | | | | | | | * | * | | | | * | | | | * |
| | XGBoost | | | * | | | | | | | | | | | | | | |
| | LDA | | | | | * | | | | | * | | | | | | | |
| | PCA | | | | * | | | | | | | | | | | | | |
| | Explicit-Implicit Skills Extraction | | | | | | | | * | | | | | | | | | |
| | Quick UML Stool | | | | | | | | | | | * | | | | | | |
| | FastText | | | | | | | | | | | | | | | | | |
| | GloVe | | | | | | | | | | | * | * | | | | | |
| | Word2Vec | | | | * | * | * | * | * | | * | * | | | | | | |
| | Skip-Gram(SG) | | | | | | | | | | | | | | | | | |
| | Bag-of-Words (BOW) | | | | * | | | | | | | | | | | | | |
| | CBOW | | | | * | | * | | | | | | | | | | | |
| | Part-of-Speech (POS) Tagging | | * | | | | | | * | | * | * | * | | | * | | * |
| | Levenshtein Distance | | * | | | | | | | | | | | | | | | |
| | BIOES-IOBES | | | | | | | | | | | | * | | | | | * |
| | Character/Word Embedding | | * | | | | | * | | | | * | | | | | | |
| | Jaro Winkler Distance | | | | | | | | | | | | | | * | | | * |
| | Semantic Matching | | | | | * | | | | | | | | | * | * | | * |
| | N-Gram | | | | | | | | | | | | | | | * | | |
| | Counting Frequent Words | | | | | | | | | | | | | | | * | | |
| | Hidden Markov Model (HMM). | | | | | | | | | | | | | | | | * | |
| | Euclidean Distance | | | | | | | | | | | | | | | | * | |
| | Regular Expressions | | | | | | | | | | | | | | | | * | * |

**Table 3.** Classification Techniques Reported in the Literature.

| | CLASSIFICATION METHODOLOGIES USED | Mittal et al. | Gaur et al. | Xu et al. | Tobing e et al. | Deepak and Santhanavijayan | Lin et al. | Nigam et al. | Jiechieu et al. | Gugnani et al. | Qin et al. | Batbaatar and Ryu | Pawar et al. | Sandanayake et al. | Das e et al. | Mughaid et al. | Ours |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CLASSIFICATION | BiLSTM/BiLSTM-A/LSTM/BiLSTM+ CRF | | * | | | | * | * | | | * | * | * | | | | |
| | Logistic Regression | * | | | | | | | * | | | | | | | | |
| | K-Mean Method | | | | | | * | | | | | | | | | | |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Decision Trees | | | | | | | | | | | | | |
| Conditional Random Fields (CRF) | | | | | | | | | | | | | |
| AdaBoost | | | | | | | | | | | | | |
| Rectified Linear Unit (ReLU) | | | | | | | * | | | | | | |
| SoftMax | | | | | | | | | | | | | |
| N fold Cross-Validation | | | | | | | | | | | | | |
| KNN | | | | | | | | | | | | | |
| Naive Bayes (NB) | | | | | | | | | | | * | * | |
| CNN | | * | | | * | | * | | * | | | | |
| Multi Text Classification Model | | | | | | | * | | | | | | |
| FCNN | | | | | | | * | | | | | | |
| Linear Regression | * | | | | | * | | | | | | | |
| Random Forest | | | * | | | * | * | | * | | | * | |
| XGBoost | | | * | | | * | * | | | | | | |
| SVM | | | * | * | | | | | * | | | * | |
| ANN | | | | | | * | | | | | | | |
| Ontology Is_a Matching | | | | | | | | | | | | | * |
| NLP | | | * | * | * | * | * | | * | | | * | * |

## General architectural model of the proposed system

### Hybrid Job Recommendation System Proposed

This study focuses on the Hybrid Job Recommendation System (HJRS), which combines content-based filtering and ontology-based term matching methods (Çelik & Elçi, 2011; Ertuğrul & Bitirim, 2025) to recommend suitable job positions to job-seeking candidates. Ontology matching labels important text fragments from resumes by matching them with conceptual connections in the ontology, while content-based filtering uses the more structured resume information to generate a list of suitable job positions as recommendations.

The key elements of resumes include concepts such as person/institution, place, time, degree, and qualifications (e.g., personal profile information, education details, past work experiences, skills, certificates, etc.). Therefore, instead of merely focusing on textual content in resumes, identifying and semantically labelling common patterns of named entities that refer to ontology concepts can provide a different perspective to the literature. Additionally, an Ontology-Based Resume Text Parsing Module (OntRTPM) can be developed as a solution that can be integrated into existing candidate search and matching systems used in HR departments.

In this study, the HJRS (as shown in Figure 1) is divided into 3 main components: (1) a Parsing Module (OntRTPM), (2) a Resume Recommender System Ontology — RRSO (Bitirim and Ertuğrul, 2024), and (3) a Resume Database. In our previous study (Bitirim & Ertuğrul, 2024), the development of RRSO, which aims to comprehensively represent the information required for job position-candidate matching within JRSs, was mentioned (Bitirim and Ertuğrul, 2024). Through the RRSO, it becomes possible to automatically extract and semantically match critical concepts and attributes in resumes (e.g., personal profile, education details, work experiences, certificates, awards, skills, etc.) when matching a resume and a job position in JRS solutions. RRSO has been developed using Stanford University's Protégé 5.0 ontology development tool (O'connor et al., 2005) and the Web Ontology Language (OWL), a W3C standard (Cardoso & Pinto, 2004).

Latest version of RRSO is publicly shared for other researchers under the web address (http://ontologydesignpatterns.org/wiki/Main_Page/RRSO) to allow access for other researchers under the NeOn project's Ontology Design Patterns (ODP) site. The latest version of RRSO includes 28 classes, 23 object property types, 7 data property types, 25,314 logical axioms, 24,982 annotation axioms, 24,924 class assertions, totaling 75,356 axiom definitions. RRSO provides a conceptual framework for modeling received resume data in syntactic and unstructured forms. Since our study primarily focuses on OntRTPM, detailed information about RRSO is not included in this article.

In addition, OntRTPM includes four main processes: (1) Preprocessing, (2) Feature Extraction, (3) Ontology Matching and Labelling, and (4) Saving Structured Resume Data (Figure 1).

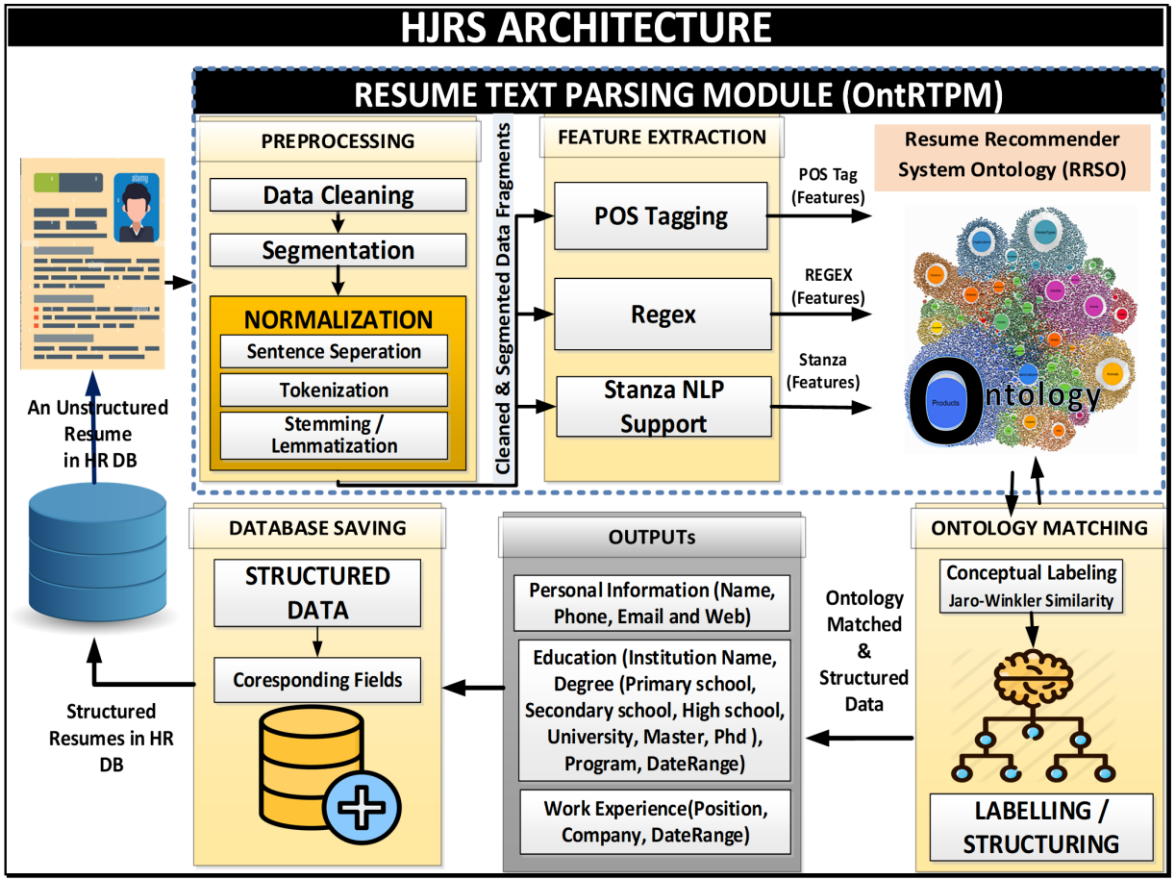



**Figure 1**. Architectural model of the HJRS proposed.

1) **Preprocessing Step.** This step involves several processes aimed at optimizing data for subsequent analysis: (a) Data Cleaning— removes redundant parts, terms, and symbols to enhance parsing accuracy; (b) Segmentation— divides the resume text into meaningful segments or sections; and (c) Normalization— standardizes the text through sentence separation, tokenization, stemming, and lemmatization, simplifying it for effective extraction of essential conceptual terms (e.g. standardizing abbreviations). These integrated/sequential steps facilitate the extraction of relevant information from resumes and ensure the necessary preparation before mapping to the conceptual framework in the ontology.

2) **Feature Processing.** It involves three main tasks for feature extraction: (a) POS Tagging, (b) Regex Operations, and (c) Support from the Stanza NLP Tool (Qi et al., 2020). This step enables the extraction of essential text fragments from a resume to handle both syntactic and semantic dependencies, and to analyze morpheme structures based on four segments (profile, education, work experiences, skills) within resumes.

3) **Ontology Matching**. In this step, OntRTPM identifies and tags common structural representations found in resumes by aligning them with concepts defined in RRSO through its ontology matching operations.

4) **Saving Structured Resume Data**: In this final step, OntRTPM stores the structured and tagged data from resumes into a database, enabling efficient retrieval and utilization within the JRS. These resume data structured in data pools can be used to list suitable job offers for candidates.

## Resume dataset used

In data set collection step, 100 resumes as MS Word documents were selected randomly from the resume database of the Kariyer.Net[1] (https://www.kariyer.net/) recruitment company in Turkey. Large recruitment companies with millions of users like Kariyer.Net have unstructured/semi-structured or free-style resumes in many different languages in their resume pools. Additionally, job candidates' resumes can be uploaded in various formats such as ‘.doc’, ‘.docx’, ‘.pdf’, ‘.txt’. The database of Kariyer.Net company contains more than 6,000,000 unstructured, free-style resumes written in both English and Turkish. In this study, 100 resumes with ‘.doc’ and ‘.docx’ extensions written in Turkish were selected for our data set.

[1] *www.kariyer.net company.*

## Operations of the ontology-based resume text parsing module (Ontrtpm)

### Preprocessing stage

Preprocessing step involves 3 main operations (1) Data Cleaning, (2) Segmentation, and (3) Normalization. In this study, the following preprocessing operations are applied on the 100 Turkish resumes in our dataset.

**(a) Data Cleaning:** The data cleaning step of the OntRTPM is developed using Pyhton Spyder (Spyder-IDE, 2023). This step involves several data cleaning and replacement tasks, which are detailed below.

*Irregular spacing between terms.* Extra spaces between terms in the 100 resumes have been removed. This step is easily done using strip() and replace() methods of pandas of Python, as seen Table 4.

*Remove Stop Words.* One of the important preprocessing techniques involves stop-word removal. Stop-words are commonly used words in a language that contribute little to the overall meaning of a text. Removing stop-words is a technique typically employed to enhance performance in NLP applications.

In our study, stop words were identified and removed from existing resumes using the 'nltk.corpus' (Natural Language Toolkit, 2024) library in Python (Python Spyder, 2024). Examples of common stop words in English include "a, the, and, or, of, on, this, we, were, is, anymore, but, however, some, even", and so on.

Additionally, the 'nltk.corpus' library is widely recognized for detecting and removing stop words in Turkish as well, such as 'gibi' (EN: like), 've' (EN: and), 'için' (EN: for), 'veya' (EN: or), 'ama' (EN: but), among others. Please refer to Table 5 for the code snippet used in this process.

*Removing punctuation characters:* To remove punctuation characters, including non-ASCII characters, commonly used in resumes such as ':', '!', '?', '%', '&', '/', etc., a 'removePunct' function was developed. This function ensures that documents are formatted uniformly by eliminating these punctuation characters. To remove these punctuation characters, 'removePunct' function is developed as seen Table 6.

**Table 4.** Removing unnecessary spaces in the resume dataset.

```
1  def removeSpaces(_dataFrame):
2      __dataFrame=_dataFrame.copy()
3  return __dataFrame.applymap(str.strip).replace('\s+',' ',regex=True)
```

**Table 5.** Removing stop words in Turkish Language.

```
1  from nltk.corpus import stopwords
2  def trStopwords(_language):
3      return list(stopwords.words(_language))
4
5  def removeStopwordsFromDF(_dataFrame, _stopwordList):
6      _dataFrame=_dataFrame.copy()
7      for _column in _dataFrame.columns:
8          _columnContent=[removeStopwords(_content, _stopwordList) for _content in _dataFrame[_column]]
9          _dataFrame[_column]=Series(_columnContent)
10     return _dataFrame
11
12 def removeStopwords(_content,_stopwordList):
13     _content=_content.split()
14     return ''.join([word for word in _content if not word in _stopwordList])
15 stopwordList=h.trStopwords('Turkish')
```

*Turkish Stop Words List: https://github.com/sgsinclair/trombone/blob/master/src/main/resources/org/voyanttools/trombone/keywords/stop.tr.turkish-lucene.txt

**Table 6.** Removing punctuation characters in the resume dataset.

```
1  def removePunct(_content):
2      _content=_content.split('\n')
3      _content = [_c for _c in _content if not _c.startswith(' < Punc')]
4      return '\n'.join(_content)
5  'call the removePunct function for the profile segment of a job candidate
6  profile_info=[h.removePunct(kb) for kb in profile]
```

**(b) Segmentation:** The resumes involved a set of sections such as the user profile, educational background, work experiences, certificates, competences/skills, hobbies, awards, references, etc. In order to divide an unstructured resume written in free format into meaningful sections, the location of these sections

must be determined. The formats of the resumes in our data set are usually MS Word documents with the extension '.doc/.docx'. Before sending a resume as input to OntRTPM and starting the parsing process, it is segmented, as it is shown in Figure 2, which are 'Profile Information', 'Educational Background', 'Work Experience', 'Certifications', 'Skills'. Currently, in this study, the segmentation task for 100 Turkish resumes has been performed manually, resulting in a comprehensive dataset divided into segments. This manual segmentation process has enabled the separation of specific sections of each resume, facilitating a more efficient analysis and processing. Each resume segment (Profile, Education, Work Experiences, Skills/Competencies) is sent separately to OntRTPM to initiate sentence separation and normalization, and feature processing and ontology matching tasks are carried out sequentially.

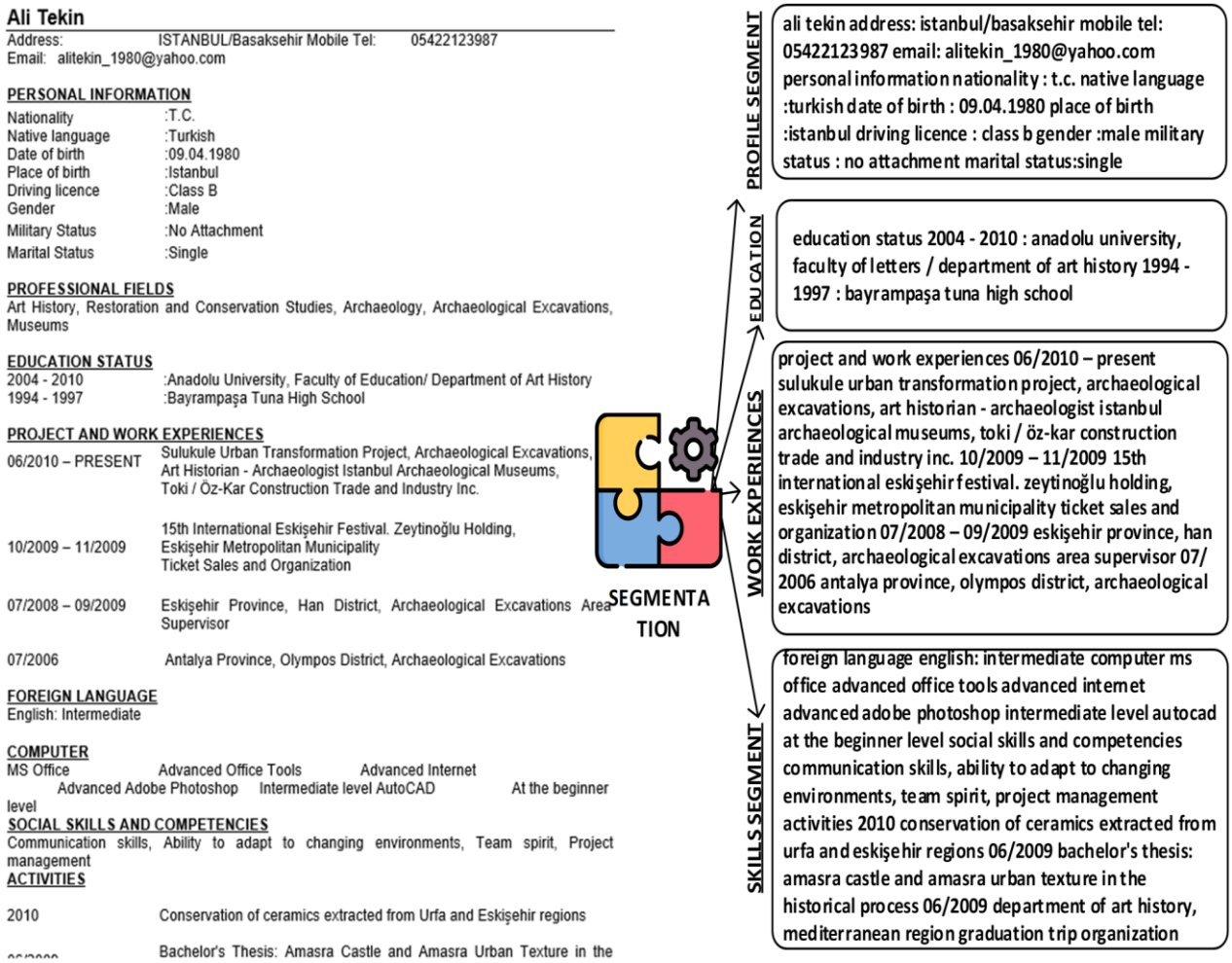



**Figure 2**. Segmentation of a resume.

**(c) Normalization:** The next step of preprocessing is Normalization. The process is the putting of different forms of a text into the same form. In other words, text normalization is the process of organizing terms in the text into a single canonical form. By normalizing term snippets extracted from resumes before analysis, it ensures consistency between inputs received prior to similarity matching task. This step also informs the system about which types of text snippets (e.g. LTD is a shortened term for the word 'Limited') to normalize and how to process them later.

*Tokenization, Stemming and Lemmatizing.* There are many agglutinative languages in the world, such as Turkish, Hungarian, Finnish, and Mongolian. In agglutinative languages, new words are typically derived by adding suffixes to the end of a base word. For example, in Turkish, the word 'tr:kapı' means 'en:door', whereas 'tr:kapı-cı' means 'en:doorman'. The former refers to an object, while the latter denotes a profession. Therefore, before applying similarity matching to terms extracted from resumes, these terms need to be tokenized and lemmatized to prevent any loss of meaning during similarity comparisons.

Normalization varies from system to system and generally involves the execution of four different processes: (a) *tokenization*, (b) *stemming*, (c) *lemmatization*, and (d) *sentence separation.* Tokenization divides

a text into smaller units, such as words, sentences, or n-grams. Through stemming and lemmatization techniques, it is possible to reduce a word to its root form by removing suffixes. For example, the term 'runs' can be reduced to its root form 'run.' However, there are some differences between stemming and lemmatization. Stemming is the process of reaching the root word by separating the suffixes, while lemmatization involves transforming the word into its morphological base form. Stemming is simpler and faster, whereas lemmatization takes longer and uses its own set of rules and algorithms. Stemming can sometimes produce invalid results, leading to ambiguity and performance loss. However, lemmatization, with its complex algorithms involving vocabulary and morphological analysis, can more accurately detect the root term. For instance, 'ladies' becomes 'ladi' through stemming but 'lady' through lemmatization. There are some well-known Python NLP libraries (e.g., split(), nltk.tokenize, or spacy.tokenizer) to apply these steps in English. Additionally, to lemmatize an English word, libraries like nltk.stem, spacy.lemmatizer, or gensim.lemmatizer can be effectively used. Another option is WordNet (Miller, 1995), which is one of the most popular lemmatization tools. Since WordNet is a word association database initially developed for the English language and is generally preferred for lemmatization, we found it unsuitable for stemming Turkish resume terms. Therefore, in this study, Zemberek (Akın & Akın, 2007), a lemmatization tool suited to the Turkish language, was identified and adopted to find the root term of an input term extracted from resumes.

## Feature Extraction

After data cleaning, text fragments extracted from each segment of a resume are sent to OntRTPM for feature processing. The purpose of this step is to capture related term groups that indicate general concepts in resumes (e.g., driving license, age, primary school name, skills information etc.). To do this, 3 methods are used (1) POS tagging, (1) Regex Operations, and (1) Stanza NLP Operations.

**(a) POS Tagging:** Part-of-Speech Tagging (POS Tagging) is an important NLP application applied by machines and used in processes such as disambiguating word meaning, question and answer parsing, etc. It is tagging a word in a context as 'nouns', 'verbs', 'adjectives', 'adverbs', etc. POS tagging is used to provide additional capabilities for feature processing and improve accuracy. POS tags are used to determine the person names, organization names, location names, etc. in a given input context. In this study, POS tagging is used to find 'nouns', 'verbs', 'adjectives', and 'adverbs'.

For Turkish resumes, Turkish Zemberek (Akın & Akın, 2007) is used in POS tagging step ('jpype' library used). For English resumes, BIOES ('from nltk.chunk import conlltags2tree' library used) can be used in this stage. In addition to these tagging, Named Entity Recognition (NER) tagging which is BIOES (Sang & Veenstra, 1999) are used for sequence labelling. BIOES are used for tagging the 'person', 'location', and 'organization' concepts.

**(b) Regex Operations:** At this stage, other tagging processes for concepts like 'date', 'e-mail', 'phone', and abbreviations are applied using regular expressions. After extensive development and testing, regular expressions proved effective in accurately predicting crucial concepts such as dates, phone numbers, email addresses, and common abbreviations from resume segments. The 'RE' library ('import re') was used to develop the RE module of the OntRTPM in Python.

*Replacement of special characters and abbreviations*. In various languages, the removal of special characters and standardization steps must be conducted under specific conditions. An input resume document may not only contain resume-focused terms such as school, driver's license, age, work experience, etc., but also a wide range of other elements such as numbers (e.g., 23), dates (e.g., 12/5/2001), abbreviations (e.g., USA, USD, LTD), or symbols (e.g., $, ©). At this stage, date, email, and phone patterns are extracted using the Regex library with the code snippets provided in Table 7.

**Table 7.** Specifying dates, phone numbers, and e-mails in the resume dataset.

| 'Specifiyg Date Data Using RE |
|---|
| date = re.search(r'(? < =tarihi)(\s)?(\=\|\-\|\:\ \|\: \|\= \|\- )?(([0-9]{2}(\/\|\.\|\-\|\s\| / \|, )[0-9]{1}(\/\|\.\|\-\|\s\| / \|, )[0-9]{4})\|([0-9]{2}(\/\|\.\|\-\| / \|, )[0-9]{1}(\/\|\.\|\-\| / \|, )[0-9]{4})\|([0-9]{2}(\/\|\.\|\-\| / \|, )[0-9]{2}(\/\|\.\|\-\| / \|, )[0-9]{4})\|([0-9]{2}(\/\|\.\|\-\| / \|, )[0-9]{2}(\/\|\.\|\-\| / \|, )[0-9]{2})\|((?:ocak\|şubat\|mart\|nisan\|mayıs\|haziran\|temmuz\|ağustos\|eylül\|ekim\|kasım\|aralık)(\/\|\.\|\. \|\-\|\s\| / \|, )[0-9]{4})\|([0-9]{4})(\/\|\.\|\. \|\-\|\s\| / \|, )(?:ocak\|şubat\|mart\|nisan\|mayıs\|haziran\|temmuz\|ağustos\|eylül\|ekim\|kasım\|aralık)\|([0-9]{2})(\/\|\.\|\. \|\-\|\ \|\/\| / \|, )(?:ocak\|şubat\|mart\|nisan\|mayıs\|haziran\|temmuz\|ağustos\|eylül\|ekim\|kasım\|aralık)(\/\|\.\|\. \|\-\|\s\| / \|, )([0-9]{4}))',_date) |
| 'Specifiyg E-Mail Data Using RE |
| mail=findall(r\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z\|a-z]{2,}\b',_content) |

```
ˈSpecifiyg Phone Data Using RE
repatterns=[r'\+090\s\d{3}\s\d{3}\s\d{2}\s\d{2}',r'[(05)|(02)]\d{10}',r'\+90\s[5|2]\d{2}\s\d{3}\s\d{2}\s\d{2}',r'[(05)|(02)]\d{3}\s\d{3}\s\d{2}\
s\d{2}',r'0\s\(\d{3}\)\s\d{7}',\

r'[(05|02|03|04)]\d{3}\s\d{3}\s\d{4}',r'[(05)|(02)]\d{3}\s\d{7}',r'0\s\d{3}\s\d{3}\s\d{4}',r'0\s\d{3}\s\d{3}\s\d{2}\s\d{2}',r'\(\d{3}\)\s\d{3}\s\d
{2}\s\d{2}',r'\(\d{3}\)\s\d{7}',r'\(\s\d{4}\s\)\s\d{3}\s\d{2}\s\d{2}']
```

Additionally, some words in the candidate's resume may contain different characters (e.g., é, á, í, ì, Ĭ, ĭ) such as in ˈdoğum tarĭhĭˈ (EN: ˈbĭrth dateˈ), which can be replaced with the standard canonical form ˈdogum tarihiˈ (EN: ˈbirth dateˈ). Furthermore, some words in resumes may include accented characters like ˈé, à, ç, ñˈ commonly used in many languages such as French, Spanish, Portuguese, and Turkish. To address this issue, a Python function was developed using the ˈunicodedataˈ library, which works across different languages and replaces accented letters. To solve this problem, a simple code segment *ˈunicodedata.normalize('NFKD', text).encode('ASCII', 'ignore').decode('utf-8', 'ignore')ˈ* was applied. In Turkish texts, identifying and labelling text fragments such as dates, abbreviations, symbolic explanations, phone numbers, emails, and accented characters before removing punctuation marks is crucial to prevent potential data loss in subsequent steps.

**(c) Stanza NLP Tool Used:** The aim of this step is to identify institution names critical in resumes such as preschools, primary schools, high schools, universities, faculties, and departments. In the previous step, the Zemberek tool (Akın & Akın, 2007) which includes a Turkish language-supported Named Entity Recognition (NER) module and proved effective in identifying these critical institution names, was utilized.

However, due to instances where Zemberek fell short in detecting institution or department names during NER operations, Stanza .run (Stanza Tool, 2024) is utilized alternatively, which offers extensive language support, as a supplementary NLP tool. Following the development of this system and extensive testing, most of the school and company names in Turkish resumes were determined using the NER modules of the Zemberek and Stanza tools. However, not only the common information/concepts belong to the institutions (e.g. departments, locations, addresses) but also other concepts such as professions, skills, degrees and titles belong to candidate must be identified and labeled in a resume. At this stage, terms need to be examined semantically as well as syntactic matching. This part is discussed in the next step.

## Ontology matching and conceptual labelling

Tools like Zemberek and Stanza can be effectively utilized for syntactic matching of Turkish phrases, while more meaningful comparisons can be achieved through *ontologies*. Particularly, a more efficient solution is provided during the matching process between terms extracted from resumes and criteria defined for a job position. Therefore, in the RRSO ontology, similarity matching is conducted between concepts related to resume fields and terms extracted from resumes, enabling semantic contextual definition and labeling of the terms. Additionally, user-generated typos in resume terms can be detected and replaced with a validated ontology element under the appropriate class in the ontology. Additionally, during the matching process, unidentified resume terms are not ignored and are directed to user verification. This situation is explained in more detail in the next section.

**(a) Correcting Misspelled Words:** Misspelled words are frequently encountered in the resumes of job candidates, and these need to be corrected during the ontology matching. For example, when the word ˈsuper high schoolˈ is written as ˈsupre high schoolˈ, the system can automatically detect this spelling error and change it to ˈsuperˈ. To achieve this, the Jaro-Winkler distance (Wang & Wang, 2017), a sequence similarity metric that measures the regularization distance between two terms, was used in this study. The lower the Jaro-Winkler distance between two phrases (term(s) from ontology versus term(s) from resume), the more similar the words are. The score is normalized, so that 0 indicates an exact match and 1 indicates no similarity.

$$d_j = \begin{Bmatrix} 0 & if\ m = 0 \\ \frac{1}{3}\left(\frac{m}{|n_1|} + \frac{m}{|n_2|} + \frac{m-t}{m}\right) & otherwise \end{Bmatrix}, n_1 = size\ (term_1); n_2 = size\ (term_2); \quad (4)$$

where, $m$ is the number of matching characters, $t$ is the number of transpositions, and $dj$ is the Jaro distance between the two terms. For example, let's consider ˈchicagoˈ and ˈcihagcoˈ terms. The values of $n_1$, $n_2$, number of matched characters, half of unmatched characters are obtained as $|n_1| = 7$, $|n_2| = 7$, $m$=7, and $t$=2.5, respectively. The result of $d_j$ is obateined as $d_j =$ 88.10. As the next step, the Winkler distance $dw$ are defined through:

$$d_w = d_j + (l.p.(1 - d_j)); \quad (5)$$

where, $l$ gives the length of common prefix at the start of the string up to a maximum of 4 characters, $p$ is the constant scaling factor denoting how much the score has been adjusted upwards for common concepts. Standard value for this constant in Winkler's work is $p$ = 0.1.

The values of $l$, $p$, and $d_j$ are defined as $l$ =5/2=2.5, $p$ =0.1, and $d_j$= 88.10, respectively. The result of $d_w$ is obateined as $d_w$ = 89.30. as a result, it is possible to say the 'match is considered true' due to the Jaro-Winkler similarity score $d_w$ is above 80%.

In this step, after importing *jaro* library, *jaro.jaro_winkler_metric(u'STRING1', u'STRING2')* function is easily can be used to get a similarity score between two terms. In this study, if the matching score between two terms is over 85.00%, the match is considered correct, and the misspelled term is replaced with the correctly spelled one. On the other hand, unidentified resume terms during the matching process are not ignored but directed to user verification. If a school name is misspelled in the user's resume and the system does not detect it, it will be displayed as it is in the user interface, allowing the user to make corrections. The methodology applied for ontology matching is presented below.

**(b) Conceptual Matching and Labelling:** It is anticipated that key terms will be identified and labeled from resumes in the semantic context extraction process for unstructured resume data uploaded to HR repositories. Capturing and labeling connections such as 'İstanbul → is_a → City' (İstanbul is a City) and 'Eastern Mediterranean University → is_a → University → is_a → Organization' (Eastern Mediterranean University is an Organization) are crucial steps for extracting important information and semantic labelling in resumes. In this manner, the system semantically understands and labels primary school names, middle school names, high school names, university names, and other educational institutions within the education section of a resume using standardized and verified ontology elements. Additionally, it has been used to identify and standardize abbreviations found within a resume. For instance, (1) institution abbreviations (e.g., Eastern Mediterranean University —EMU), (2) degree abbreviations (e.g., PhD for Doctor of Philosophy, MS for Master of Science, BS for Bachelor of Science, etc.), (3) city name abbreviations (e.g., Istanbul —IST, etc.), (4) country name abbreviations (e.g., Canada —CA, Turkey —TR, Kazakhstan —KZ, etc.), and (5) corporate term abbreviations (e.g., Limited —LTD, company —Co., etc.) have been identified and labeled.

Additionally, each resume, after completing conceptual descriptions and labeling with the RRSO resume ontology, is modeled into OWL format for storage. This ensures that in future 'Job Posting - Resume' matching processes, the calculated similarity matching score will be more effective.

Table 8 presents the algorithm used for conceptual description, matching, and labeling terms.

**Table 8.** Terms Similarity Matching Functions for Semantic Matching Algorithm using RRSO Resume Ontology.

| | |
|---|---|
| 1 | CLASS: RRSO_Terms_Matching |
| 2 | VARIABLES: |
| 3 | File OntologyFile // Physical address of the RRSO ontology file |
| 4 | IRI Onto_Base_IRI // Base IRI of the RRSO ontology |
| 5 | OWLOntologyManager manager // OWLOntologyManager object |
| 6 | OWLDataFactory factory // OWLDataFactory object |
| 7 | OWLOntology ontology // Reference to the loaded RRSO ontology |
| 8 | METHOD: loadOntology() |
| 9 | TRY |
| 10 | Load the ontology from the OntologyFile |
| 11 | Set the loaded ontology to the ontology variable |
| 12 | CATCH OWLOntologyCreationException |
| 13 | Print error message |
| 14 | METHOD: getAllIndividualsOfClass(className) |
| 15 | Create an empty list individualsList |
| 16 | Call loadOntology() |
| 17 | IF ontology is not null THEN |
| 18 | TRY |
| 19 | Create IRI for the specified class using className |
| 20 | Get the OWLClass object for the classIRI |
| 21 | Create a reasoner |
| 22 | Query for instances of the class |
| 23 | PRINT 'Individuals of class className:' |
| 24 | FOR each individual in the instances |

| | |
|---|---|
| 25 | Get the individual's name |
| 26 | Add the individual's name to individualsList |
| 27 | PRINT the individual's name |
| 28 | CATCH Exception |
| 29 | Print error message |
| 30 | RETURN individualsList |
| 31 | METHOD: replaceTermIfSimilarityHigh(termFromResume) |
| 32 | Call loadOntology() |
| 33 | IF ontology is not null THEN |
| 34 | Create Jaro-Winkler similarity object |
| 35 | FOR each OWLClass in the ontology |
| 36 | Get the termFromOntology from the class IRI fragment |
| 37 | Calculate the similarity score between termFromResume and termFromOntology |
| 38 | IF similarity score is higher than 0.85 THEN |
| 39 | PRINT 'Similarity Score: similarityScore' |
| 40 | PRINT 'Term from Resume: termFromResume' |
| 41 | PRINT 'Term from Ontology: termFromOntology' |
| 42 | Perform replacement operation if needed |
| 43 | RETURN |
| 44 | METHOD: calculateJaroWinklerSimilarity(term1, term2) |
| 45 | Create a Jaro-Winkler similarity object |
| 46 | Calculate the similarity score between term1 and term2 using Jaro-Winkler similarity |
| 47 | RETURN similarity score |
| 48 | METHOD: extractTermsFromResume(resumeText) // Create an empty list terms |
| 49 | Split the resumeText by spaces into words |
| 50 | FOR each word in words |
| 51 | Convert word to lowercase and add to terms |
| 52 | RETURN terms |
| 53 | METHOD: main() // Create an instance of RRSO_Terms_Matching |
| 54 | CALL getAllIndividualsOfClass('specified class') |
| 55 | PRINT 'Individuals of specified class:' |
| 56 | FOR each individual in the returned list |
| 57 | PRINT individual |
| 58 | CALL replaceTermIfSimilarityHigh('specified term from resume') |
| 59 | CALL extractTermsFromResume('specified resume text') |
| 60 | PRINT 'Extracted Terms from Resume:' |
| 61 | FOR each term in the returned list |
| 62 | PRINT term |

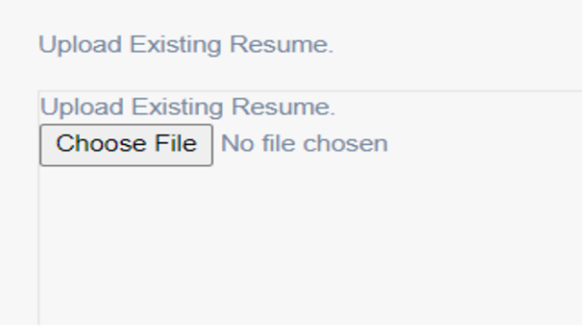
## Case study

In today's competitive job market, finding the right job position is crucial for job seekers to achieve their career goals. The following case study begins with job seeker Mr. Ali Tekin uploading his resume to our ontology-based HJRS system and includes the steps of structuring the resume with both syntactic and semantic analysis. It shows step by step the process of storing the structured resume in a more structured form, either in OWL format or in a database.

### Resume uploading and information extraction

Mr. Ali Tekin first logs into our ontology-based HJRS system via the web application user interface and uploads his resume to the system by selecting the 'Upload Resume' option, as seen Figure 3 (a). As next, the resume uploaded by Mr. Ali Tekin is automatically processed by OntRTPM parsing module and extracts the candidate's personal information, education background, work experience and skills segments from the resume. Figure 3(b) shows a section from the profile information section written by the candidate in his resume. Additionally, personal information parsed from the resume via the OntRTPM parsing module is shown in Figure 3(c). Finally, the screen for validating the profile data extracted during development and unit validation tests is presented in Figure 3(d).

Figure 4(a) and Figure 4(b) show the original Turkish and English education status information written by Mr. Ali Tekin, respectively. Additionally, Figure 4(c) showcases education information parsed from the resume via the OntRTPM parsing module. Finally, Figure 4(d) presents the screen used to validate the extracted education details during development and unit validation tests.




(a) Uploading the resume of Mr. Ali Tekin.

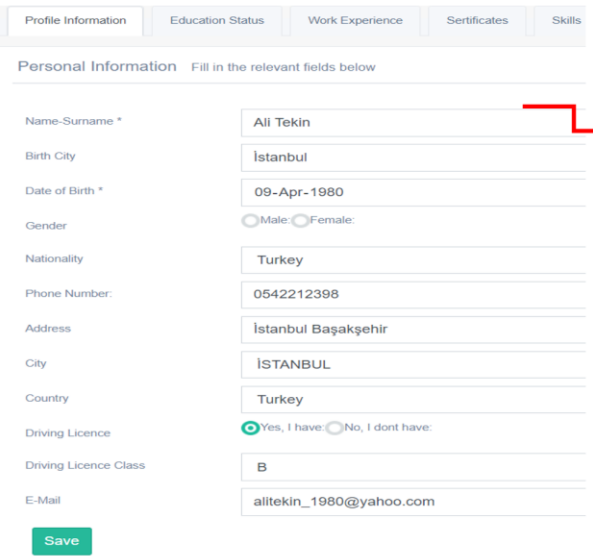
**Ali Tekin**

Address: ISTANBUL/Basaksehir Mobile Tel: 05422123987
Email: alitekin_1980@yahoo.com

**PERSONAL INFORMATION**

| | |
|---|---|
| Nationality | : T.C. |
| Native language | :Turkish |
| Date of birth | : 09.04.1980 |
| Place of birth | :Istanbul |
| Driving licence | : Class B |
| Gender | :Male |
| Military Status | : No Attachment |
| Marital Status | :Single |

(b) Profile data on the original resume in English.




(c) Information extracted from Profile Segment.


(d) Validation of Profile Information on Python.

**Figure 3**. Information extraction from the Profile Segment of the resume of Mr. Ali Tekin.

**EĞİTİM DURUMU**

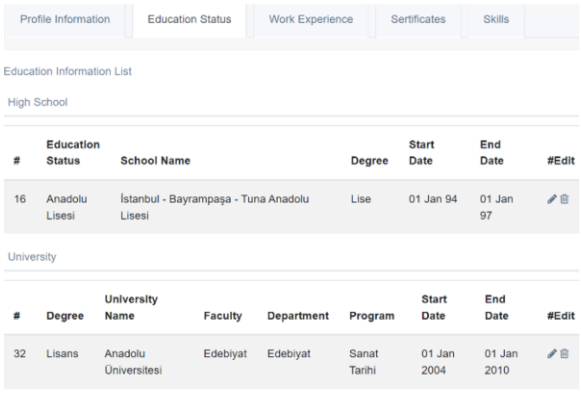
| | |
|---|---|
| 2004 - 2010 | : Anadolu Üniversitesi, Edebiyat Fakültesi / Sanat Tarihi Bölümü |
| 1994 - 1997 | : Bayrampaşa Tuna Lisesi |

(a) Education section of the original resume in Turkish.

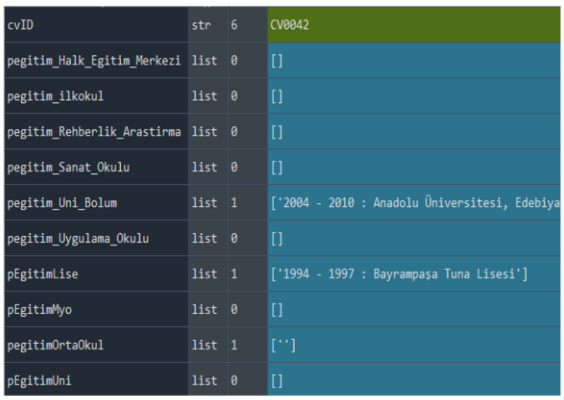
**EDUCATION STATUS**

| | |
|---|---|
| 2004 - 2010 | : Anadolu University, Faculty of Education / Department of Art History |
| 1994 - 1997 | : Bayrampaşa Tuna High School |

(b) Education section of the original resume in English.



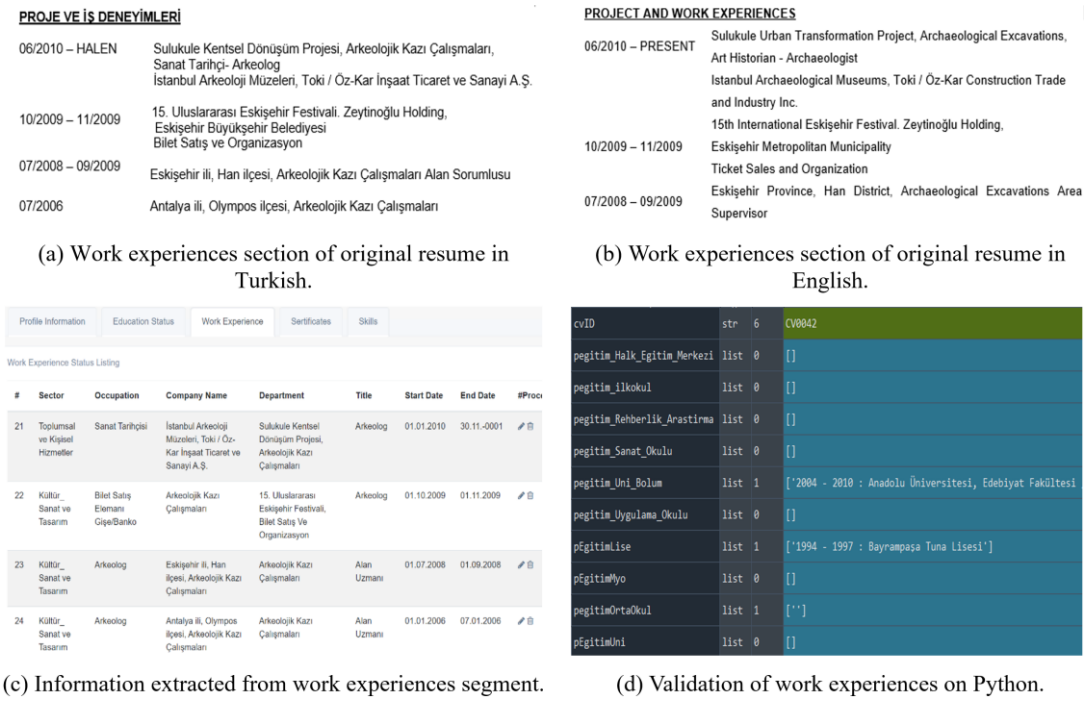

(c) Information extracted from education segment.




(d) Validation of education information on Python.

**Figure 4**. Information extraction from the Education Segment of the resume of Mr. Ali Tekin.

Figure 5(a) and Figure 5(b) show the original Turkish and English work experiences information written by Mr. Ali Tekin, respectively. Furthermore, Figure 5(c) showcases work experience information parsed from the resume via the OntRTPM parsing module. Lastly, Figure 5(d) presents the screen used to validate the extracted work experience details during development and unit validation tests.



**PROJE VE İŞ DENEYİMLERİ**

| | |
|---|---|
| 06/2010 – HALEN | Sulukule Kentsel Dönüşüm Projesi, Arkeolojik Kazı Çalışmaları, Sanat Tarihçi- Arkeolog İstanbul Arkeoloji Müzeleri, Toki / Öz-Kar İnşaat Ticaret ve Sanayi A.Ş. |
| 10/2009 – 11/2009 | 15. Uluslararası Eskişehir Festivali. Zeytinoğlu Holding, Eskişehir Büyükşehir Belediyesi Bilet Satış ve Organizasyon |
| 07/2008 – 09/2009 | Eskişehir ili, Han ilçesi, Arkeolojik Kazı Çalışmaları Alan Sorumlusu |
| 07/2006 | Antalya ili, Olympos ilçesi, Arkeolojik Kazı Çalışmaları |

(a) Work experiences section of original resume in Turkish.

**PROJECT AND WORK EXPERIENCES**

| | |
|---|---|
| 06/2010 – PRESENT | Sulukule Urban Transformation Project, Archaeological Excavations, Art Historian - Archaeologist Istanbul Archaeological Museums, Toki / Öz-Kar Construction Trade and Industry Inc. |
| 10/2009 – 11/2009 | 15th International Eskişehir Festival. Zeytinoğlu Holding, Eskişehir Metropolitan Municipality Ticket Sales and Organization |
| 07/2008 – 09/2009 | Eskişehir Province, Han District, Archaeological Excavations Area Supervisor |

(b) Work experiences section of original resume in English.

Profile Information | Education Status | Work Experience | Sertificates | Skills

Work Experience Status Listing

| # | Sector | Occupation | Company Name | Department | Title | Start Date | End Date | #Proc |
|---|---|---|---|---|---|---|---|---|
| 21 | Toplumsal ve Kişisel Hizmetler | Sanat Tarihçisi | İstanbul Arkeoloji Müzeleri, Toki / Öz-Kar İnşaat Ticaret ve Sanayi A.Ş. | Sulukule Kentsel Dönüşüm Projesi, Arkeolojik Kazı Çalışmaları | Arkeolog | 01.01.2010 | 30.11.-0001 | |
| 22 | Kültür_ Sanat ve Tasarım | Bilet Satış Elemanı Gişe/Banko | Arkeolojik Kazı Çalışmaları | 15. Uluslararası Eskişehir Festivali, Bilet Satış Ve Organizasyon | Arkeolog | 01.10.2009 | 01.11.2009 | |
| 23 | Kültür_ Sanat ve Tasarım | Arkeolog | Eskişehir ili, Han ilçesi, Arkeolojik Kazı Çalışmaları | Arkeolojik Kazı Çalışmaları | Alan Uzmanı | 01.07.2008 | 01.09.2008 | |
| 24 | Kültür_ Sanat ve Tasarım | Arkeolog | Antalya ili, Olympos ilçesi, Arkeolojik Kazı Çalışmaları | Arkeolojik Kazı Çalışmaları | Alan Uzmanı | 01.01.2006 | 07.01.2006 | |

(c) Information extracted from work experiences segment.

| | | | |
|---|---|---|---|
| cvID | str | 6 | CV0042 |
| pegitim_Halk_Egitim_Merkezi | list | 0 | [] |
| pegitim_ilkokul | list | 0 | [] |
| pegitim_Rehberlik_Arastirma | list | 0 | [] |
| pegitim_Sanat_Okulu | list | 0 | [] |
| pegitim_Uni_Bolum | list | 1 | ['2004 - 2010 : Anadolu Üniversitesi, Edebiyat Fakültesi |
| pegitim_Uygulama_Okulu | list | 0 | [] |
| pEgitimLise | list | 1 | ['1994 - 1997 : Bayrampaşa Tuna Lisesi'] |
| pEgitimMyo | list | 0 | [] |
| pegitimOrtaOkul | list | 1 | [''] |
| pEgitimUni | list | 0 | [] |

(d) Validation of work experiences on Python.

**Figure 5**. Information extraction from the Work Experiences of the resume of Mr. Ali Tekin.

## Saving structured the resume data

The information in the candidate's resume is matched and analyzed using the system's RRSO ontology. For example, term series such as 'Anadolu University', 'Faculty of Education', 'Department of Art and History' mentioned in the resume refer to the certain concepts in the RRSO ontology of the system, and then the concept(s) it belongs to are sent as input to the system's parsing module. For instance, the 'Department of Art and History' department name mentioned in the candidate's resume is matched with the departments as OWL Individual elements under the 'Departments' class of the RRSO, and then the module identified that the term snippets is an educational department information. Then, the semantically defined term snippets are conceptually labeled and displayed on the user screens. Matching semantically defined term series in all segments is applied following the same processes. Finally, through the OntRTPM parsing module, certificate and skill information were extracted from the resume by applying the same processes.

Resumes, structured through both syntactic and semantic analyses, are stored in OWL format and sent to relevant database tables and columns for storage. When candidates log back into the system, they can view their resumes, make edits to any incomplete or incorrect information. The revised details are automatically updated in both the database and individual's OWL file.

*(a) Saving Data to the Database*. It refers to the process where structured resumes, after being processed through syntactic and semantic analyses, are saved into the relevant tables and columns of the system's database. This step ensures that all parsed information from resumes, including personal details, work experiences, education backgrounds, and skills, are stored in a semantically structured format that can be easily accessed and managed by the system. When candidates interact with the system, they can view their resumes and update any inaccuracies or missing information, which are then automatically reflected in both their individual database records and OWL files. This ensures consistency and accuracy in resume data management within the system.

*(b) Saving as OWL Format*. For the resume domain, while creating the RRSO ontology class hierarchy, common concepts related to the field were initially identified (Education segment concepts; primary school, high school, university, vocational school, faculty, etc.). Then, the hierarchical structure was completed by adding more specific concepts. The latest version of RRSO includes 28 classes, 23 object property types, 7 data property types, 25,314 logical axioms, 24,982 annotation axioms, 24,924 class assertions, totaling 75,356 axiom definitions. The main classes (concepts) are grouped under owl, and each main class has subclasses (concepts) created under it: e.g., 'Work_Experience → Sector', 'Work_Experience → Occupation', 'Education_Status → Education Degree', 'Education_Status →Program', 'Candidate → Name Surname', 'Candidate → Gender' and many other sub-concepts have been created. Additionally, in RRSO, besides

creating concepts, many OWL Object Properties or OWL Data Properties such as ‘has Name Surname’, ‘has Gender’, ‘has Marital Status’, ‘has Driver License’, ‘has Date of Birth’, ‘has Nationality’, ‘has Place of Birth’, ‘has Phone’ have also been created. These properties are linked to OWL Individual elements on RRSO to represent the attributes of an individual in a resume document. In Table 9, all these properties created in RRSO are presented, grouped under (1) profile information-based, (2) education information-based, (3) work experience information-based, and (4) skills-based properties.

**Table 9.** All Object Type Properties (23) and Data Type Properties (7) created on RRSO are given below. Additionally, the classes between which a defined relationship is established are given in parentheses.

| (1) Profile Information Based Properties | (2) Education Status Information Based Properties | (3) Work Experience Based Properties |
|---|---|---|
| has Name Surname (Candidate, xsd:String) | has Education Status (Candidate, Education Status) | has Company Name (Candidate, xsd:String) |
| has Gender (Candidate, Gender) | has Degree of Study (Candidate, Education Degree) | has Title (Candidate, Title) |
| has Marital Status (Candidate, Marital Status) | has Kindergarten (Candidate, Kindergarten) | has Sector (Candidate, Sector) |
| has Date of Birth (Candidate, xsd:Date) | has Primary School (Candidate, Primary School) | has Occupation (Candidate, Occupation) |
| has Place of Birth (Candidate, Location) | has Secondary School (Candidate, Secondary School) | |
| has Nationality (Candidate, Nationality) | has High School (Candidate, High School) | (4) Skills Based Properties |
| has Driving License (Candidate, xsd:Boolean) | has Vocational High School (Candidate, Vocational High School) | has Skill (Candidate, Skill) |
| | has Public Education Center (Candidate, Public Education Center) | |
| has Driving License Class (Candidate, Driver License) | has University (Candidate, University) | |
| has Phone (Candidate, xsd:String) | has Faculty (Candidate, Faculty) | |
| has Is Address (Candidate, xsd:String) | has Program (Candidate, Program) | |
| has Home Address (Candidate, xsd:String) | has Degree (Candidate, Degree) | |
| has E_mail (Candidate, xsd:String) | has Academcy (Candidate, Academcy) | |

Integration of Individual Resumes into RRSO in OWL Form. Once the necessary data is extracted from the millions of unstructured resumes in HR pools, it is possible to match these resumes with job position descriptions on a semantic basis. To achieve this, individuals and open job positions need to be described using general concepts from an upper ontology structure like RRSO and stored structurally in HR pools. Therefore, with semantic-based JRS solutions and an upper ontology like RRSO in HR recruitment processes, an OWL file can be instantly created for an unstructured resume uploaded by a candidate. This way, the embedded conceptual definitions in that resume can be matched with the conceptual definitions in an open job posting on a semantic plane. For example, the personal information segment in a resume might include definitions such as ‘has_Marital_Status’, ‘has Date of Birth’, ‘has Place of Birth’, ‘has Nationality’, ‘has Occupation’, ‘has Driver License’, etc. Similarly, the job experience segment might include some important property definitions like ‘has Company Name’, ‘has Title’, ‘has Sector’, ‘has Profession’, etc.

In the case study section, the features inferred from Mr. Ali Tekin's resume, including profile, education, work experiences, and skills information, are converted into OWL form and stored as an OWL file based on the feature definitions in the RRSO upper ontology. As seen in the example processed in the case study section, the candidate's ID number is ‘SSID000042’, marital status is ‘Single’, has a driver's license (B1), work address is ‘Toki / Öz-Kar Construction Trade and Industry Inc’, phone number is ‘05422123987’, gender is ‘Male’, university information is ‘Faculty of Education‘, education degree is ‘Bachelor’, department is ‘Art History’, and faculty name is ‘Faculty of Education’, among other OWL features. A Java function has been developed to generate and process the OWL/XML files of candidates' resumes in OWL form and has been integrated into OntRTPM.

## Experimental studies and evaluations

This section presents the experimental studies and evaluation of the OntRTPM resume parser, which was specifically designed to extract crucial elements from each segment of candidate resumes. To provide a comprehensive analysis of the parser’s performance, each segment was evaluated separately. In this section, we focus on the parser's effectiveness in classifying educational segments from 100 resumes to understand

the methodology of the experimental studies conducted. Other segments were also evaluated using the same logic for performance assessment. The parser's performance in classifying educational segments is assessed using standard metrics, including accuracy, precision, and recall, as detailed below. To ensure thorough validation, the study employs the following metrics to evaluate both the syntactic and semantic aspects of resume parsing:

— True Positive (TP): Instances where the OntRTPM parser correctly extracts and accurately identifies relevant educational information from a resume. For example, correctly identifying a university name or degree program from a resume qualifies as a TP.

— True Negative (TN): Instances where the parser correctly identifies the absence of a specific educational feature in a resume. For instance, correctly recognizing that no public education center information is present in a resume is considered a TN.

— False Positive (FP): Instances where the parser incorrectly identifies or extracts information that is not present in the resume. For example, erroneously labelling a non-existent degree or skill as present is classified as FP.

— False Negative (FN): Instances where the parser fails to extract or misclassifies information that is actually present in the resume. For example, overlooking or mislabeling an actual degree or qualification is considered an FN.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP+FP+TN+FN}} \times 100 = 80{,}85 \quad (1)$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} = 79{,}59\% \quad (2)$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} = 78{,}39\% \quad (3)$$

Table 10 presents the performance metrics obtained for various educational information features extracted from 100 resumes by the OntRTPM resume parser:

The evaluation results indicate that the OntRTPM resume parser achieved a performance level with an overall accuracy of 80.85%. Precision was measured at 79.59%, reflecting the parser's strong capability to minimize false positives. Recall was recorded at 78.39%, demonstrating the parser's effectiveness in identifying relevant educational information.

Performance varies across different educational categories. Features such as University and Program show high precision and recall, indicating effective classification in these areas. This may be attributed to the fact that university and program/department information in the dataset is typically well-filled, accurately provided, and not left blank by individuals.

**Table 10.** Performance metrics on the features of education status segment.

| No | Features Extracted from Education Information Segment of 100 Resumes | TP | TN | FP | FN |
|---|---|---|---|---|---|
| 1 | Education Status (Candidate, Education Status) | 46 | 33 | 9 | 12 |
| 2 | Degree of Study (Candidate, Education Degree) | 48 | 32 | 8 | 12 |
| 3 | Kindergarten (Candidate, Kindergarten) | 3 | 71 | 20 | 6 |
| 4 | Primary School (Candidate, Primary School) | 48 | 41 | 5 | 6 |
| 5 | Secondary School (Candidate, Secondary School) | 50 | 32 | 10 | 8 |
| 6 | High School (Candidate, High School) | 46 | 42 | 5 | 7 |
| 7 | Vocational High School (Candidate, Vocational High School) | 19 | 53 | 16 | 12 |
| 8 | Public Education Center (Candidate, Public Education Center) | 5 | 68 | 15 | 12 |
| 9 | University (Candidate, University) | 54 | 34 | 4 | 8 |
| 10 | Faculty (Candidate, Faculty) | 29 | 51 | 7 | 13 |
| 11 | Program (Candidate, Program) | 53 | 36 | 4 | 7 |
| 12 | Degree (Candidate, Degree) | 46 | 36 | 5 | 13 |
| 13 | Academy (Candidate, Academy) | 21 | 54 | 12 | 13 |
| | Totals | 468 | 583 | 120 | 129 |

In contrast, categories like Kindergarten and Public Education Center (a type of High School) exhibit lower precision and recall, highlighting opportunities for further refinement. This could be due to the tendency of individuals to underreport kindergarten information on resumes and the incomplete coverage of kindergarten data in the ontology. Additionally, the relatively low number of resumes containing Public Education Center information in our dataset may contribute to performance issues.

The performance metrics of OntRTPM across 4 key segments have been evaluated (see Table 11). The profile segment shows the highest performance with 88.40% accuracy, 84.80% precision, and 87.50% recall. The education segment demonstrates 80.85% accuracy, 79.59% precision, and 78.39% recall, while the work experiences segment has the lowest performance with 78.30% accuracy, 75.85% precision, and 76.10% recall. The skills segment presents a balanced performance with 81.40% accuracy, 81.10% precision, and 84.30% recall.

**Table 11.** Performance metrics obtained of the Profile, Education Status, Work Experiences, and Skills segments.

| Focused Segments in Resumes | Accuracy (%) | Precision (%) | Recall (%) |
|---|---|---|---|
| Profile | 88,40 | 84,80 | 87,50 |
| Education Status | 80,85 | 79,59 | 78,39 |
| Work Experiences | 78,30 | 75,85 | 76,10 |
| Skills | 81,40 | 81,10 | 84,30 |
| Average Performance of OntRTPM | 82,24 | 80,36 | 81,57 |

Overall, the average metrics for OntRTPM are 82.24% for accuracy, 80.36% for precision, and 81.57% for recall. These results indicate that OntRTPM achieves a high level of accuracy in extracting information from the four segments we focused on and generally performs satisfactorily. Although OntRTPM performs well overall, there are still some segments in resumes that we have not yet focused on (e.g., certifications, projects, references, etc.). The extraction of such resume segments is one of the areas that need improvement in the future. Additionally, addressing missing data in resumes, incorporating additional supporting data into the extraction module, optimizing the extraction algorithms, or utilizing deep learning architectures are potential areas for future work.

*Limitations*. In this study, we analyzed 100 resumes randomly selected from a real-world pool of resumes from a well-known career company (www.kariyer.net) in Turkey. These resumes were used for system performance studies. However, to conduct this study more effectively, a larger collection of resumes is needed. Therefore, we plan to develop an automated tool that will randomly gather resumes from online job search portals like LinkedIn to expand our dataset. Additionally, we aim to develop a segmentation module that automatically separates segments from Turkish resumes in future work.

## Conclusion

In today's complex and dynamic job market, finding the right positions by job seekers and identifying the most suitable candidates by employers is increasingly challenging. These needs can often be inadequately met by traditional recommendation systems. Therefore, an ontology-based Hybrid Job Recommendation System (HJRS) that combines syntactic and semantic matching approaches, equipped with sophisticated data processing capabilities, has been developed. This system is designed to better meet the needs of both job seekers and employers. HJRS consists of three main components: (1) Resume Recommender System Ontology—RRSO, (2) Ontology-Based Resume Text Parsing Module (OntRTPM), and (3) System Database. The system's OntRTPM uses the system's own RRSO resume ontology and it provides the necessary semantic inferences based on terms extracted from resumes by executing pre-processing, feature processing, and both syntactic and semantic-based similarity matching techniques and processes. Moreover, the OntRTPM focuses on important sections of a resume, such as the candidate's profile information, past work experiences, educational history and skills, and extracts the necessary data from these 4 main sections through semantic and syntactic analysis using RRSO. The preprocessing stage involves data cleaning and normalization, while feature processing utilizes ontology for concept inference, semantic matching, and labelling. This approach enhances the accuracy of resume content categorization and analysis, thereby improving recommendation accuracy. One hundred (100) randomly selected resumes from the pool of a well-known career company (www.kariyer.net) in Turkey were used for system performance studies, but we hope to expand our resume dataset and further optimize system performance in future studies. Additionally, we aim to extend our methodology to extract complex semantic relations in domains beyond resumes, thereby broadening the applicability and utility of our approach across various fields.

## Acknowledgements

This project is collaborated with the Kariyer.Net company (https://www.kariyer.net/) that is titled as ‘Document - based Semantic Information Extraction System from Turkish Résumés through Ontology’,

Project No: 3110289, Supported by: The Scientific and Technological Research Council of Turkey (TÜBİTAK), Grant Program: TÜBİTAK 1501 Industry Research & Development Support Program, Ankara, Turkey.

## References

Akın, A. A., & Akın, M. D. (2007). Zemberek, an open-source NLP framework for Turkic languages. *Structure*, *10*(2007), 1-5. https://doi.org/10.47769/izufbed.880143

Bafna, P., Shirwaikar, S., & Pramod, D. (2019). Task recommender system using semantic clustering to identify the right personnel. *VINE Journal of Information and Knowledge Management Systems*, *49*(2), 181-199. https://doi.org/10.1108/VJIKMS-08-2018-0068

Batbaatar, E., & Ryu, K. H. (2019). Ontology-based healthcare named entity recognition from Twitter messages using a recurrent neural network approach. *International Journal of Environmental Research and Public Health*, *16*(19), 3628. https://doi.org/10.3390/ijerph16193628

BIOES. (2024). *GeeksforGeeks*. from https://www.geeksforgeeks.org/nlp-iob-tags/

Bitirim, S., & Ertuğrul, D. Ç. (2024). İnsan Kaynaklarinda Etkili İşe Alim Süreci İçin Türkçe Bir Ontoloji Geliştirilmesi (Development of a Turkish Ontology for Effective Recruitment Process in Human Resources—RRSO: Resume Recommender System Ontology). *Kahramanmaraş Sütçü İmam Üniversitesi Mühendislik Bilimleri Dergisi*, *27*(2), 401-414. https://doi.org/10.17780/ksujes.1390172

Bitirim, Y., Bitirim, S., Ertugrul, D. C., & Toygar, O. (2020, July). An evaluation of reverse image search performance of Google. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)* (pp. 1368-1372). IEEE.

Celik, D. (2016). Towards a semantic-based information extraction system for matching résumés to job openings. *Turkish Journal of Electrical Engineering and Computer Sciences*, *24*(1), 141-159. https://doi.org/10.3906/elk-1304-130

Celik, D., & Elçi, A. (2005a). A semantic search agent approach: finding appropriate semantic Web services based on user request term (s). In *2005 International Conference on Information and Communication Technology* (pp. 675-687). IEEE.

Çelik, D., & Elçi, A. (2005b). Searching semantic Web services: An intelligent agent approach using semantic enhancement of client input term (s) and matchmaking step. In *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)* (Vol. 2, pp. 916-922). IEEE.

Celik, D., & Elci, A. (2008). Provision of semantic Web services through an intelligent semantic Web service finder. *Multiagent and Grid Systems*, *4*(3), 315-334.

Çelik, D., & Elçi, A. (2011). Ontology-based matchmaking and composition of business processes. In *Semantic Agent Systems: Foundations and Applications* (pp. 133-157).

Çetindağ, C., Yazıcıoğlu, B., & Koç, A. (2023). Named-entity recognition in Turkish legal texts. *Natural Language Engineering*, *29*(3), 615-642. https://doi.org/10.1017/S1351324922000304

Chandak, A. V., Pandey, H., Rushiya, G., & Sharma, H. (2024). Resume parser and job recommendation system using machine learning. In *2024 International Conference on Emerging Systems and Intelligent Computing (ESIC)* (pp. 157-162). IEEE.

Das, P., Pandey, M., & Rautaray, S. S. (2018). A CV parser model using entity extraction process and big data tools. *IJ Information Technology and Computer Science*, *9*, 21-31. https://doi.org/10.5815/ijitcs.2018.09.03

Deepak, G., Teja, V., & Santhanavijayan, A. (2020). A novel firefly driven scheme for resume parsing and matching based on entity linking paradigm. *Journal of Discrete Mathematical Sciences and Cryptography*, *23*(1), 157-165. https://doi.org/10.1080/09720529.2020.1721879

Ertuğrul, D. Ç., & Bitirim, S. (2025). Job recommender systems: A systematic literature review, applications, open issues, and challenges. *Journal of Big Data, 12*. https://doi.org/10.1186/s40537-025-01173-y

Fernández-Reyes, F. C., & Shinde, S. (2019). CV retrieval system based on job description matching using hybrid word embeddings. *Computer Speech & Language*, *56*, 73-79. https://doi.org/10.1016/j.csl.2019.01.003

Gaur, B., Saluja, G. S., Sivakumar, H. B., & Singh, S. (2021). Semi-supervised deep learning based named entity recognition model to parse education section of resumes. *Neural Computing and Applications*, *33*, 5705-5718. https://doi.org/10.1007/s00521-020-05351-2

Gugnani, A., & Misra, H. (2020). Implicit skills extraction using document embedding and its use in job recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, No. 08, pp. 13286-13293). https://doi.org/10.1609/aaai.v34i08.7038

Jiechieu, K. F. F., & Tsopze, N. (2021). Skills prediction based on multi-label resume classification using CNN with model predictions explanation. *Neural Computing and Applications*, *33*(10), 5069-5087. https://doi.org/10.1007/s00521-020-05302-x

Lin, Y., Lei, H., Addo, P. C., & Li, X. (2016). *Machine learned resume-job matching solution*. arXiv preprint arXiv:1607.07657. https://doi.org/10.48550/arXiv.1607.07657

Mashayekhi, Y., Li, N., Kang, B., Lijffijt, J., & De Bie, T. (2024). A challenge-based survey of e-recruitment recommendation systems. *ACM Computing Surveys*, *56*(10), 1-33.

Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, *38*(11), 39-41.

Min, H., Yang, B., Allen, D. G., Grandey, A. A., & Liu, M. (2024). Wisdom from the crowd: Can recommender systems predict employee turnover and its destinations? *Personnel Psychology*, *77*(2), 475-496.

Mittal, V., Mehta, P., Relan, D., & Gabrani, G. (2020). Methodology for resume parsing and job domain prediction. *Journal of Statistics and Management Systems*, *23*(7), 1265-1274. https://doi.org/10.1080/09720510.2020.1799583

Mughaid, A., Obeidat, I., Hawashin, B., AlZu'bi, S., & Aqel, D. (2019). A smart geo-location job recommender system based on social media posts. In *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)* (pp. 505-510). IEEE. https://doi.org/10.1109/SNAMS.2019.8931854

Natural Language Toolkit. (2024, July 19). https://www.nltk.org/

Nigam, A., Roy, A., Singh, H., & Waila, H. (2019). Job recommendation through progression of job selection. In *2019 IEEE 6th International Conference on Cloud Computing and Intelligence Systems (CCIS)* (pp. 212-216). IEEE. https://doi.org/10.1109/CCIS48116.2019.9073723

O'Connor, M., Knublauch, H., Tu, S., Grosof, B., Dean, M., Grosso, W., & Musen, M. (2005). Supporting rule system interoperability on the semantic web with SWRL. In *The Semantic Web–ISWC 2005: 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005. Proceedings 4* (pp. 974-986). Springer Berlin Heidelberg.

Pawar, S., Thosar, D., Ramrakhiyani, N., Palshikar, G. K., Sinha, A., & Srivastava, R. (2021). Extraction of complex semantic relations from resumes. In *ASEA workshop@ IJCAI*.

Python Spyder. (2024, July 19). https://www.spyder-ide.org/

Qi, P., Zhang, Y., Zhang, Y., Bolton, J., & Manning, C. D. (2020). *Stanza: A Python natural language processing toolkit for many human languages*. arXiv preprint arXiv:2003.07082. https://doi.org/10.48550/arXiv.2003.07082

Qin, C., Zhu, H., Zhu, C., Xu, T., Zhuang, F., Ma, C., ... & Xiong, H. (2019). DuerQuiz: A personalized question recommender system for intelligent job interview. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 2165-2173). https://doi.org/10.1145/3292500.3330706

Sandanayake, T. C., Limesha, G. A. I., Madhumali, T. S. S., Mihirani, W. P. I., & Peiris, M. S. A. (2018). Automated CV analyzing and ranking tool to select candidates for job positions. In *Proceedings of the 6th International Conference on Information Technology: IoT and Smart City* (pp. 13-18). https://doi.org/10.1145/3301551.3301579

Sang, E. F., & Veenstra, J. (1999). *Representing text chunks*. arXiv preprint cs/9907006.

Stanza Tool. (2024, July 19). https://stanfordnlp.github.io/stanza/

Tobing, B. C. L., Suhendra, I. R., & Halim, C. (2019). Catapa resume parser: End to end Indonesian resume extraction. In *Proceedings of the 2019 3rd International Conference on Natural Language Processing and Information Retrieval* (pp. 68-74). https://doi.org/10.1145/3342827.3342832

Wang, Y., Qin, J., & Wang, W. (2017). Efficient approximate entity matching using jaro-winkler distance. In *International conference on web information systems engineering* (pp. 231-239). Cham: Springer International Publishing.

Xu, L., Liu, J., & Gu, Y. (2018, July). A recommendation system based on extreme gradient boosting classifier. In *2018 10th International Conference on Modelling, Identification and Control (ICMIC)* (pp. 1-5). IEEE. https://doi.org/10.1109/ICMIC.2018.8529885

Zou, Z., Huspi, S. H., & Nuar, A. N. A. (2024). A review on job recommendation system. *Journal of Advanced Research in Applied Sciences and Engineering Technology*, *41*(2), 113-124.