Annals of the Academy of Romanian Scientists
Series on Science and Technology and Information
Online ISSN **2066 - 8562** Volume **19**, Number 1/**2026**

HYBRID RESUME–JOB MATCHING WITH ADAPTIVE SCORING

Victor-Valentin ANGHEL[1], Theodor BORANGIU[2], Cătălin NEGULESCU[3]

Abstract. *This paper describes a software system that supports recruitment in the IT domain, where the number of applications is usually high and resumes come in many different formats. Because of this, going through them manually becomes difficult and, in many cases, inefficient. In practice, recruiters often need to process large amounts of information in a short time, so some level of automation becomes necessary.The proposed solution has two main steps. First, resumes are processed and assigned to common IT roles such as Data Scientist, Machine Learning Engineer, DevOps Engineer, Software Developer, QA Engineer, Project Manager or Cloud Architect. For this, an XGBoost model is used that considers several types of information extracted from resumes, like education, skills, programming languages, certifications, foreign languages, and previous experience. These information are combined to get an estimate of how well a candidate might fit a certain role. A scoring step is then added where the recruiter can adjust the importance of each feature. This is done because it was noticed that, in real hiring scenarios, not all criteria matter equally and their relevance can change from one position to another. The service system, currently under development, operates in a containerized environment and is orchestrated through Kubernetes technology to ensure scalability and component isolation.*

Keywords: Kubernetes, Natural Language Processing (NLP), Named Entity Recognition (NER), Automated Recruitment, AI Model Fine-Tuning.

DOI [10.56082/annalsarsciinfo.2026.1.50](https://doi.org/10.56082/annalsarsciinfo.2026.1.50)

1. Introduction

Automating recruitment in the IT sector remains challenging due to the large number of applicants and the diversity of CV formats. To address this issue, this paper proposes a modular architecture based on three autonomous microservices deployed through Docker and Kubernetes, ensuring scalability, flexibility, and high availability [1,2]. The solution processes unstructured CVs and converts them into a standardized JSON representation, enabling efficient candidate evaluation and

[1] Ph.D. student, POLITEHNICA Bucharest, victor.anghel@stud.acs.upb.ro
[2] University Professor, Ph.D., POLITEHNICA Bucharest, Academy of Romanian Scientists, theodor.borangiu@upb.ro
[3] Ph.D. student, POLITEHNICA Bucharest, catalin.negulescu@stud.electro.upb.ro

transparent CV-job matching by highlighting the contribution of individual semantic features to the final recommendation.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 introduces the proposed microservice-based architecture for CV processing and candidate selection. Section 4 details the classification model, the datasets used for training and evaluation and the semantic features considered in the matching process. Section 5 describes the results obtained from the evaluation process and discusses their significance. The last section provides a summary of the work and points to possible extensions of the proposed approach.

2. Related work

Automated CV analysis relies on NLP techniques such as TF-IDF, NER, and transformer models to extract and compare information from resumes and job descriptions. However, experience and educational background remain more difficult to represent accurately than explicitly stated skills [1].

Tokenization methods such as WordPiece, together with BERT-based models, support the identification of information related to personal details, education, experience, and skills. Nevertheless, content embedded in tables or images is still difficult to process reliably [3].

YOLOv5 and OCR-based methods have been used to process visual content in resumes, although their performance is strongly influenced by image quality [4].

The combination of BERT, BiLSTM, and CRF helps improve entity recognition by capturing both contextual and sequential information [5].

The performance of resume classification models improves with larger datasets and the use of pre-trained models such as BERT. Adapting these models to the target domain can further reduce the impact of document variability [6].

Commercial platforms such as Eightfold AI and SmartRecruiters provide automated support for candidate profiling and CV-job matching. Their functionality generally includes the extraction of relevant information from resumes and its comparison against job requirements in order to assist recruiters during candidate selection. Nevertheless, the decision logic behind these platforms remains only partially visible. Although they may provide matching scores or candidate recommendations, they rarely explain how each criterion contributes to the final outcome. As a result, the CV-job association process remains difficult to interpret and evaluate in a transparent manner.

3. Overall system architecture

The proposed platform follows a microservice-based architecture designed to support scalability and modularity. As shown in Figure 1, a frontend application

communicates through an API Gateway, which manages authentication and forwards requests to specialized microservices.

- *The Ingestion and CV Pre-processing service* receives unstructured data and extracts raw text from them.
- *The Document Structuring service* transforms the extracted data into semantically validated JSON formats using an advanced, fine-tuned BERT model. This approach eliminates the need for separate intermediate steps, increasing processing efficiency and ensuring semantic consistency of the data. The model extracts relevant entities while ontological validation guarantees interoperability and uniformity of the generated data [7].
- *The Classification service* is designed to assess the compatibility of candidates with available positions (Data Scientist, Machine Learning Engineer, DevOps Engineer, Backend/Frontend Developer, Project Manager, QA Engineer and Cloud Architect) using the XGBoost algorithm trained on a JSON-formatted dataset (120000 size) that encodes specific elements for each of the six semantic features (education, soft skills, programming languages, certifications, foreign languages and professional experience).
- Additional services include adaptive interview test generation, detailed evaluation and candidate recommendation, employee allocation to projects, as well as a dedicated logging and monitoring service for traceability and optimization.

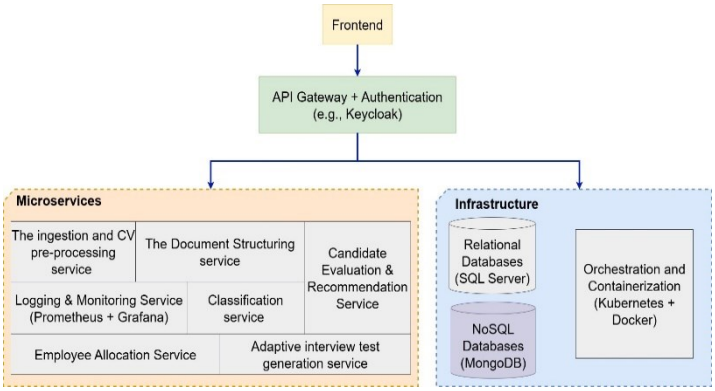

Figure 1. Overview of the proposed microservice-based architecture for automated recruitment

The platform uses Keycloak at the gateway entry point. When a user sends a request, OAuth2 and OpenID Connect are used to confirm the associated token and role before the request is sent to the requested microservice.

Figure 2 follows a CV from upload to matching. The ingestion service receives the document, extracts the text and removes unnecessary content. The processed text is sent to the structuring service, where a fine-tuned BERT model

identifies the required fields. The extracted data is checked, converted to JSON, and stored in MongoDB. In the last stage, the classification service reads the CV information together with the job records available in SQL Server. An XGBoost model then assigns a match score to the CV and the selected job.

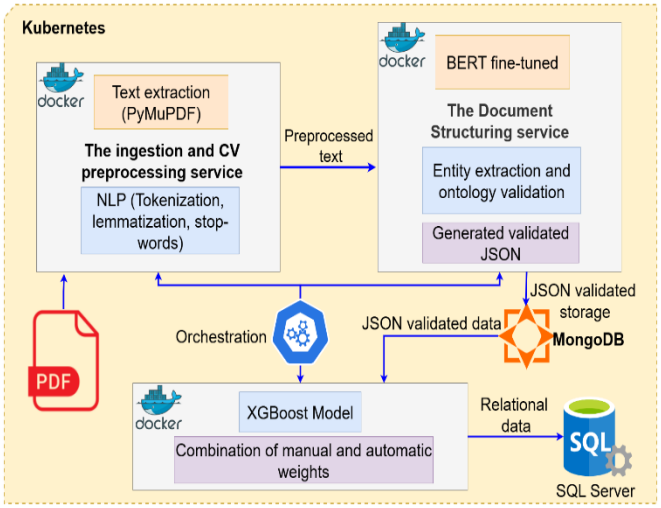

Figure 2. Workflow from CV ingestion to XGBoost-based candidate-job matching

4. Automated resume-to-job description classification

A. Data structure and collection method

To examine how individual feature influence CV-job matching, both CVs and job descriptions were represented in JSON format. The CV dataset included 4,817 records obtained from Hugging Face [4] and 7,183 additional synthetic records created from a predefined set of IT skills. Each record covered programming languages, soft skills, education, foreign languages, certifications, and experience. Proficiency levels ranging from beginner to expert were assigned to programming and foreign languages.

The job description set was created separately, as an appropriate JSON source could not be identified. It contains 200 generated examples covering eight IT positions: Data Scientist, Machine Learning Engineer, DevOps Engineer, Backend Developer, Frontend Developer, Project Manager, QA Engineer, and Cloud Architect.

In future development, these datasets will be provided by the microservice responsible for converting text into JSON format using a Named Entity Recognition (NER) model for semantic feature extraction.

[4] https://huggingface.co/datasets/datasetmaster/resumes

B. Preparing the train dataset

To build the training set, each CV was randomly paired with 10 of the 200 job records. The resulting pairs were labelled as accepted or rejected by applying cosine similarity together with a tolerance margin for borderline cases. A separate score was then calculated for each semantic feature.

For soft skills and certifications, the score indicates whether the required item occurs in the CV. Programming and foreign language matches were evaluated according to the level required for each item. For programming and foreign languages, the encoded levels were beginner = 0.25, intermediate = 0.50, advanced = 0.75 and expert = 1.00. Accordingly, the Java score for intermediate in the CV and advanced in the job record is 0.67.

This procedure generated 120000 CV-job combinations. As reported in Figure 3, 70000 were classified as rejected matches, while the remaining 50000 were classified as accepted matches. Since the two classes are not equally represented, the imbalance was considered during model training, as described in the following section.

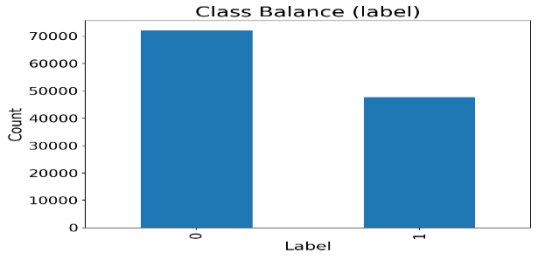

Figure 3. Class distribution of CV-job pairs in the matching dataset

Figure 4 highlights a focus on technical roles with the model having contextual information to classify CVs for the most in-demand IT positions.

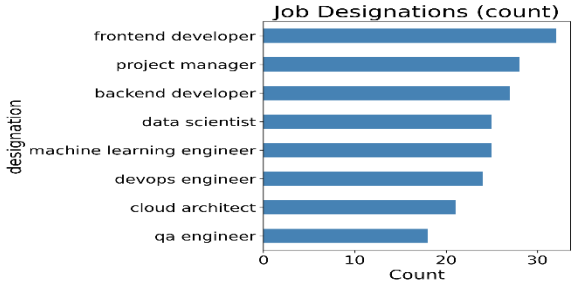

Figure 4. Job role distribution across the dataset

C. Training model

The input data contain both element-level values and aggregated scores for the semantic features. XGBoost was selected because it can handle this type of mixed input and capture relationships that are not necessarily linear [8]. The model performs binary classification while also allowing the contribution of each feature

to the CV-job match to be examined. We separated the data so that the same CV or job description could not be found in both training and testing subsets.

An 80:20 double disjoint split was applied, excluding from testing any CV or job record already used for training. XGBoost hyperparameters were selected with RandomizedSearchCV over 600 configurations and evaluated by five-fold StratifiedKFold, yielding the results presented in Table 1.

Table 1. XGBoost hyperparameter tuning with randomized search and five-fold stratified cross-validation

Hyperparameter	Search space	Best selected value
max_depth	[3,4,5,6,8]	5
learning_rate	np.logspace(-2.5, -0.5, 10)	0.04
reg_alpha	np.logspace(-4, 0, 7)	0.001
gamma	np.linspace(0, 2, 9)	2
min_child_weight	[1,2,3,5,7,10]	2

After hyperparameter selection, 88% of the training data were used to retrain the XGBoost classifier, while 12% were kept aside for probability calibration. Platt scaling was applied on this second subset through CalibratedClassifierCV, converting the scores returned by the model into probabilities that are easier to interpret in the CV-job matching task.

The model uses both scores computed for individual items and aggregated values for each semantic feature. Although these inputs may be correlated, XGBoost can reduce their effect through regularization and subsampling. Retaining both representations was useful in this study, especially for sparse features such as certifications [9].

5. Evaluation and experimental results

The performance of the trained model was evaluated using several representative metrics for a classification problem, namely the ROC-AUC score, Precision score.

As shown in Figure 5 the model demonstrates a strong ability to separate CV-job pairs into positive and negative classes, achieving a ROC-AUC score of 0.9.

Complementary, the precision-recall curve in Figure 6 confirms a robust average precision of 0.86.

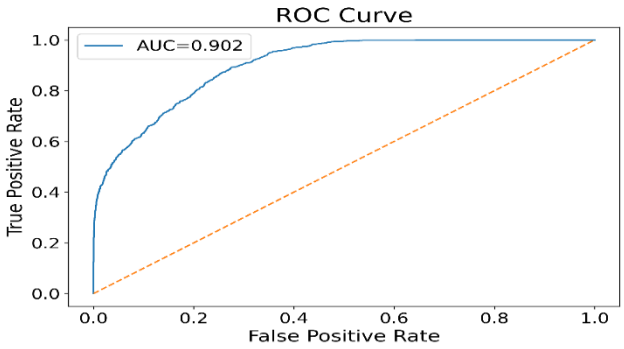

Figure 5. ROC curve for CV-job pair classification

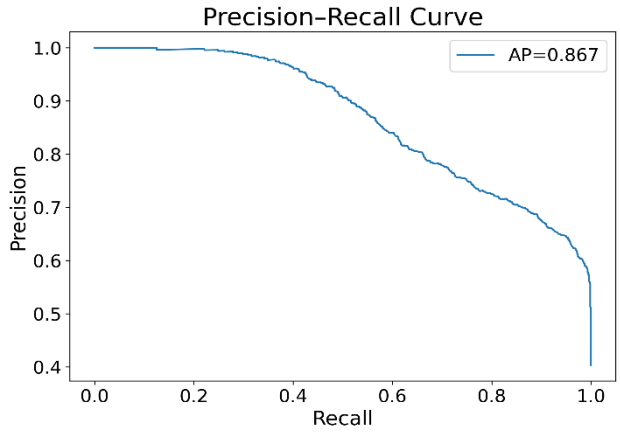

Figure 6. Precision-Recall curve for CV-job pair classification

6. Future work and conclusions

This work proposed a hybrid method for CV-job matching based on structured JSON representations of candidates and job profiles. This format allows the model to use semantic information more explicitly than raw text and supports a clearer interpretation of the classification result. SHAP values were used to identify the contribution of each semantic feature and of individual elements, while recruiter-defined weights were added to reflect domain expertise in the final matching score. XGBoost was chosen as the classifier because it works well with mixed feature types and uneven class distributions.

The evaluation used a double disjoint split, so that the same CV or job record was not shared between training and testing. During training, stratified cross-validation and RandomizedSearchCV were used to tune the model parameters. Platt scaling was applied after training to adjust the probability outputs. Thus, the

following results were obtained: a precision-recall score of 0.86 and an RMSE of 0.12.

The system could later keep recruiter feedback from completed selection rounds and use it when ordering new CV-job matches. This would help the model follow changes in hiring preferences. At the same time, the system would need safeguards against concept drift, such as job-level feedback history, time decay for older decisions, recruiter-specific preferences, and manual correction of rules. These extensions could make the framework more adaptive while preserving the interpretability provided by SHAP.

R E F E R E N C E S

[1] J. Devlin, M. Chang, K. Lee and K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, Proc. of NAACL-HLT, pp. 4171–4186, ACL, 2019, doi:10.18653/v1/n19-1423.

[2] B. Burns, B. Grant, D. Oppenheimer, E. Brewer and J. Wilkes, Borg, Omega, and Kubernetes, Communications of the ACM, vol. 59, nr. 5, pp. 50–57, 2016, doi:10.1145/2890784.

[3] Z. Yang et al., XLNet: Generalized Autoregressive Pretraining for Language Understanding, NeurIPS 2019, doi:10.48550/arXiv.1906.08237.

[4] J. Redmon et al., YOLOv3: An Incremental Improvement, arXiv 2018, doi:10.48550/arXiv.1804.02767.

[5] Lample et al., Neural Architectures for Named Entity Recognition, NAACL 2016, doi:10.18653/v1/n16-1030.

[6] T. Wolf et al., Transformers: State-of-the-Art Natural Language Processing, EMNLP 2020, doi:10.18653/v1/2020.emnlp-demos.6.

[7] T. M. Truong et al., The Microservice Based Architecture for IoT Systems, ICICM '24, 2025, doi:10.1145/3711609.3711613.

[8] Imani M., Beikmohammadi A and Arabnia H.R, Comprehensive Analysis of Random Forest and XGBoost Performance with SMOTE, ADASYN and GNUS Under Varying Imbalance Levels, doi: 10.3390/technologies13030088.

[9] Cao N. et al. A deceptive review detection framework: Combination of coarse and fine-grained features, 2021, doi:10.1016/j.eswa.2020.113465.