Grenze International Journal of Engineering and Technology, June Issue

Design and Implementation of an AI-Powered Intelligent Job Portal using Machine Learning, WebRTC, and Real-Time AI Integration

Prabhat Shukla[1], Pranay Sharma[2], Prakhar Chaurasiya[3], Rakesh Ranjan[4] and Vivek Srivastava[5]
[1-5]Department of Computer Science and Engineering ABES Engineering College Ghaziabad, India
Email: shuklaprabhatprabhat@gmail.com, pranaysharmaps53@gmail.com, prakharchaurasia400@gmail.com, iiirakeshranjan@gmail.com, viveksrivastava@abes.ac.in

***Abstract*— At this time, the majority of job portals have been reduced to simply a web site where recruited post job description and where applicants post resumes. In practice, the process of hiring is much more complicated: it involves screening, testing, interviewing, and a significant amount of back and forth exchanges. The market however does not provide any good matching jobs and the recruiters have to put up with numerous irrelevant applications, and all this slows down to a crawl and makes the entire process to appear sloppy.**

These issues are addressed through the creation of an AI based job portal as part of our final year capstone. The concept was triggered by observing the disjointed nature of the hiring process. As a rule, a site will simply allow companies to post jobs and the candidate to upload CVs. The hard work, scanning resumes, and/or conducting entry-level tests, or arranging interviews are done in different platforms. That piece meal fashion creates longer deadline and creates chaos. And thus we were aiming to integrate all these processes in a single and unified artificial intelligence-driven system. Our portal is operated not by a simple job board but by machine-learning and basic NLP to process resumes to achieve better matching. It also has the inbuilt messaging and video interviews to enable the recruiters and the candidates to chat without the need to go out of the site. Having all communication and evaluation operations within a single platform allows recruiters to stop moving through dissimilar tools during hiring processes. Having everything under one roof makes workflow very simplified and everything in order.As part of the development of this project, we developed the UI using React.js, and the back-end was developed using Node.js and Express.js. MongoDB is the database which stores all the data of users and jobs. Python was introduced with the express purpose of the machine-learning and data processing components. The entire system was organized in a way that it can be modified or enlarged in the future. In this paper, the approach to design, the implementation process are more detailed and how such a system could be used in real life situations in recruitment is also evaluated.

***Index Terms*— artificial intelligence, WebSocket, NLP, Recommendation System.**

I. INTRODUCTION

It is an AI-based recruitment process that we have created in our last year. Majority of the available job portals simply allow individuals to post advertisements and make their applications. Our system instead attempts to bring together the entire hiring pipe.

Grenze ID: 01.GIJET.12.2.211_24
© Grenze Scientific Society, 2026

Our products are resume parsing to extract candidate information, and AI-based job recommendation pipeline, online tests controlled by recruiters, and live WebRTC video interviews. We have mended all those functions into a single and clean workflow, rather than managing individual tools to screen, test, and interview.

II. INTRODUCTION

Competitiveness and technological weight have become extremely high in the sphere of recruitment in the past two years [1]. Business organizations are tilting towards data-driven approaches to locate, assess, and connect with the best talents within a short time [2]. These are traditional portals such as Naukri, Indeed and LinkedIn which are admittedly popular but are based on search with keywords, manual filtering, and the information that the user leaves [3]. Those strategies tend to overlook the more underlying semantic connections between the capabilities of a candidate and the real requirement of a job resulting into poor matches and protracted hiring processes [4]. Even standard job locations simply do not do live chats or builtin assessment tools [5]. Today, the whole recruitment process is disseminated on various platforms that are spot-on. A company may attract applications on a generic job board, and may even cross to another application to have interviews or online tests [6]. Consequently, recruiters have to oscillate between these systems and remain in contact with the same candidate. It is a sore to handle and information is left in suspension.

On the part of the applicant, it is even more obscure. Once you have sent a resumé, you are never assured of the location of your file. You are given a text update most of the times that indicates change of status but not the reasoning. But now you are on the guessing whether you are rejected or you will be delayed, which is a drag [7].

This is what we chose to address in our capstone. We have not created a vanilla job board, but rather a portal that is in fact a machine learning/simple NLP resume analysis. It was to pair applicants and job specification in a smarter way than a simple key-word- matching. In that manner the game skills are placed into perspective and the goals scored are more authentic.

We also included in-built messaging and video interviewing tools so that recruiters and jobs applicants can communicate inside the platform. Through combining all those pieces, posting, reviewing, testing, interviewing, reporting, we strive to achieve a one-stop shop.

In addition to this, the platform also attempts to reduce the time spent in manual screening and improve the clarity of communication between both sides. A structured system where analysis, interaction, and evaluation are placed together can make the hiring workflow more transparent and manageable. Such an approach not only helps recruiters in identifying suitable candidates more effectively but also provides applicants with a clearer understanding of their application progress. In this way, the overall recruitment environment can become more organized and efficient for all participants involved.

Presently, recruitment is a spaghetti all over various platforms: job boards, video technology of Zoom, and standalone evaluation applications. This makes them not tied to each other and therefore recruiters get to do a lot of manual juggling that increases the workload and may lead to confusion.

The applicants also find it hard to monitor the status of the application after making applications. The entire situation is unclear and anxiety-inducing with virtually no information on the part of the hiring side.

Our project conception was not too complicated: to unite all the hiring processes in one platform rather than resorting to a plethora of unrelated instruments. We would like to streamline the process of communication and allow the recruiters and candidates to monitor what is going on at each step precisely. The system also tries to match the skills of the candidates to the job requirements more realistically, and therefore, decisions are based on useful data rather than filters.

With the analysis of the resumes, the message, and the evaluation in one place, the entire process becomes simpler to manage. It eliminates unnecessary processes and allows recruiters and job seekers to communicate in a more open and transparent manner.

Essentially this is why we began working on this project in the first place, we could not find one place that does all the hiring in one place. As if, we glanced round and all the platforms out there do one or two things and then you are left to it on your own.

Firstly, we wanted something which can actually read through the resumes correctly and pair people to the right jobs not just some type of key word thing but something that actually knows what the person can bring to the table.

Communication was another great thing. Why can’t recruiters and applicants simply converse using the same platform? Similar to chat or video call without the need to open a zoom or Email account? That was so self-evident to us and no one is doing it well.

We also imagined that the system would be absolutely awesome, and it could look at the profile of a person and say, hey you lack this skill, go get this skill here. Rather than simply ghosting people who do not qualify, actually assist them to become better.
And to recruiters we wanted them to be able to make their own custom tests directly on the site and then examine the results of applicants without some third-party tool.
Frankly speaking when we began to researches existing job-portals we were a bit disappointed. The majority of them are only glorified.
You either place an ad or you submit an application and that is about it. No one has ever really considered the next process, the screening process, the back -and-forth process, determining whether someone can do it or not. All that is all over the place in various tools and emails and spreadsheets and it is a mess certainly. So, the thought that came to us was why not keep all of it in one place? The chat, the resume sifting, the skill, the tests, it is all alive and active and realistic therein. We did not want to make something fancy just because it was. We simply wanted something that makes sense and saves people time as the current state of affairs is a bit of a mess in case you consider it.
So, when we sat down and reflected on what we actually want this project to accomplish we thought of a few things. Nothing outrageous, just things
that we really thought were lacking in what is already out there.

III. LITERATURE REVIEW

During the recent years, there is an increasing trend toward the use of artificial intelligence and automation in hiring [8], [9]. However, the majority of research that has been done is aimed at enhancing a single aspect of the recruitment process such as resume screening, candidate recommendation, or interview management and not creating an integrated system that integrates all these functions into one [10].
In one of the previous publications, Jatoba et al. [11] have created a resume parsing application that was able to automatically retrieve the
information about the candidate through the use of machine learning and natural language processing mechanismS . Their methodology mainly relied on the extraction of the key words and filtering was carried out by rules. Though the model was good in the extraction of structured resume information, unstructured or non-standard formats were poor in their interpretation by the model and they never captured important contextual information with regards to the skill and experience of the candidate.
Sivakumar and Rajan (2020) suggested one way to address such deficiencies that involved using natural language processing with Word2Vec and TF-IDF to quantitatively assess semantic similarity between resumes [12].
Jatoba et al. (2019) built a resume parsing tool in one of the prior works that was capable of extracting information about candidates automatically using machine learning and natural language processing. They relied mostly on rulebased filtering and extraction of keywords. Whereas the structured resume information extraction performed well, unstructured or non-standard information was poorly extrapolated through the model and never retrieved important contextual information on candidates as far as their skills and experience are concerned.
To fix such deficiencies, Sivakumar and Rajan (2020) suggested the following model: natural language processing with Word2Vec and TF-IDF and semantic similarity between resumes and job descriptions. These findings indicated that this semantic based method can produce greater valid and meaningful matches than the conventional based systems that rely on keywords [13].
Their system was yet to be able to offer real time interaction and feedback between recruiters and applicants, which compromises the practicality of such systems.
The latter was followed by the work of Johnston and Bhattacharya (2021) who analyzed the opportunity to use WebRTC technology to conduct the low-latency video communication with online interviews by using the browsers [14].
It demonstrated that WebRTC can minimize the reliance on third-party video conferencing applications such as Zoom or Skype and provide smoother interview processes. Once again, this was a system meant to be used only to communicate and no AI-related analytics or automated assessment components. Another study, by Zhou, Li, and Wang (2022) suggested a semi-automated recruitment model, which uses machine learning to assess the performance of the candidates in online tests.
It was proved in the research that the results obtained by applying AI to the testing process were more accurate trustworthy.

TABLE I

Reference	Method Proposed	Accuracy	Advantages	Disadvantages
[15]	Deep learning based resume classification using Convolutional Neural Networks (CNN)	~89% classification accuracy	Automatically extracts features from resumes without heavy manual rules	Requires large labeled datasets for training
[16]	Hybrid recommendation system for job–candidate matching using collaborative filtering and content-based filtering	~85% recommendation accuracy	Improves job matching by considering both candidate profiles and recruiter preferences	Cold start problem when new users or jobs are added
[17]	Named Entity Recognition (NER) based resume information extraction system	~83% entity extraction accuracy	Efficiently extracts structured data such as skills, education, and experience	Performance decreases when resumes have unconventional formatting
[18]	Machine learning based candidate ranking model using Support Vector Machines (SVM)	~86% ranking accuracy	Provides objective ranking of candidates based on multiple attributes	Requires careful feature engineering
[19]	Semantic job matching using ontology-based skill representation	~84% matching accuracy	Captures relationships between related skills and job requirements	Ontology construction and maintenance is complex
[20]	Automated interview analysis using Natural Language Processing and sentiment analysis	~87% evaluation accuracy	Helps analyze candidate responses and communication patterns	May misinterpret tone or context in conversations
[21]	AI-based recruitment decision support system using predictive analytics	~88% prediction accuracy	Assists recruiters in identifying high-potential candidates	Depends heavily on quality and diversity of training data

IV. SYSTEM DESIGN AND METHODOLOGY

When we had initially arranged the process of actually implementing this thing, we made a decision early in that we did not want to put everything in a single massive codebase. That was merely like a nightmare to deal with. We got the entire system and divided it into smaller parts with each part doing its job separately. These components interact with one another using REST APIs and ensure that things are clean and in order. When one bit goes thumsey and it does not take the entire system down all at once which was quite essential to us since it would not have been our desire to debug a giant mess of a system in the first place.

The proposed system relies on a similarity calculation between the resume of a candidate and the job description to put forth the intelligent job matching mechanism. Normalization and tokenization of the resume text and job requirements are carried out to eliminate noise and irrelevant characters in the text. Afterwards, cosine similarity measure is used to determine the semantic overlap between the two representations of the text. The similarity score calculated is the relevance of the skills of the candidate to the requirements of the job, and it allows ranking the applicants automatically. The process of resume job matching in details is given in Algorithm 1.

Algorithm 1: Resume–Job Matching Using Cosine Similarity Input: Resume text R, Job description J Output: Matching score M

- Convert $R$and $J$to lowercase.
- Remove punctuation and special characters.
- Tokenize both texts into word tokens.
- Remove tokens with length < 3.
- Construct vocabulary $V = TR \cup TJ$.
- Compute frequency vectors $FR$and FJ.
- Compute dot product $Dot = \Sigma(FR(i) \times FJ(i))$.
- Compute magnitudes $| R |= \sqrt{\Sigma}(FR(i)2)$and $| J |= \sqrt{\Sigma}(FJ(i)2)$.
- Calculate similarity $Similarity = Dot/(| R |\times| J |)$.
- Compute matching percentage $M = Similarity \times 100$.
- Return M.

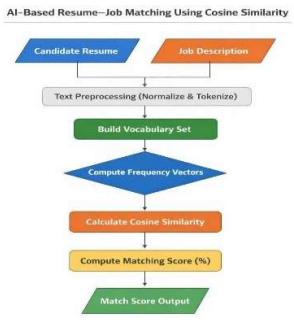

Fig 1: Flowchart of the AI-based Resume–Job Matching process using Cosine Similarity for evaluating the similarity between candidate resumes and job descriptions in the intelligent recruitment system.

A. System Components

Here is the way that we are going to divide everything and the reason behind what we are going to the various parts:

Frontend: Our frontend is React.js, simply because it was the most comfortable to us and it simply fit with what we required. This is essentially what the user views and handles on his screen. The two separate dashboards (recruiter and job seeker) were created due

to the fact that both are quite different in what they require and it would have been highly clogged and confusing to have everything in a single dashboard. We also ensured that it appears to be good on a variety of screen sizes since people do not use laptops only. And as a bonus in the things that require things to appear immediately, such as chat messages and notifications, which do not require page refreshing, we connected websockets to make everything seem alive and responsive.

Backend: In the backend we have used Node.js and Express.js. The back stage logic occurs here, which involves processing of login requests, and user data, and connecting the frontend to the database, as well as redirecting some of the traffic to the AI layer on its behalf. We chose Node due to the fact that it can support a relatively high number of

requests simultaneously and because our platform includes B. Algorithmic Workflow This system operates like a real time applications such as chat and video which was s tepwise pipeline through which it takes you through sign-up rather a necessity to us. to interview recap. The workflow is laid out in the following way:

Database: We chose the MongoDB to store all of it. User profiles, resumes, job posts, chat logs, scores on assessments, all this and more deposits here. The big win? All over place are resumes and profiles. Some of them are glitzy with certification, others sprinkling out project links, some are just basic and simple. MongoDB allows us to dump whatever we want without bringing all the documents to the same appearance which can be used in such a mess.

Smart Layer: It has its own microservice, Smart Layer, which is written in Python using Flask. Its primary backend plays ping - pong with it every time it requires some magic of ML or NLP, such as parsing a resume, finding matches, or suggesting employers that you might be interested in. We did not combine it with Python since Python has an exemplary stack to work with AI and combining Python and JavaScript would have been a nightmare.

Layers: In the case of video/audio we selected WebRTC-peerto-peer as its means low latency and decent quality. Instant text chat, notifications and call alerts we decided to use WebSockets. Polling (frontend continually asking "any new? was the first attempt. a couple seconds each but it seemed like a waste and seemed slow--changing to WebSockets was a massive difference.

User Registration and Data Ingestion: It starts with the registration of a job seeker and uploading of the CV. The platform supports PDFs and plain Is not to mention that you can post anything you desire.

Resume Parsing and Entity Extraction: The Python parser scans the sloppy text. It extracts essential fragments - name, contact, education, work experience, tech stack, and arranges them as neat fields.

Job Description Vectorization: The same happens to the posts of recruiters. NLP models convert the job ads into sentence vectors, then we can do a mathematical comparison of them.

Candidate -Job Matching: With the aid of cosine similarity or other related measures, the engine matches up your vectorized resume with job vectors, determining how similar you are to each job.

ML/NLP Models Used

Hybrid configuration, combining the time-tested machine learning tricks with the up-to-date deep-learning techniques to achieve solid accuracy in parsing and recommending, is used. This direction takes us beyond merely matching of keywords to the text and we actually comprehend the text of what is being said. The exact sequence of the models we have in the pipeline is as follows: TF - IDF and Cosine Similarity: This is our matching baseline tool. TF-IDF allows us to weight the importance of each keyword in a document and the cosine similarity allows us to score the level of overlap between a resume and a job description.

BERT (Bidirectional Encoder Representations with Transformers): TF-IDF will handle the keyword portion of it, but it is BERT that will provide actual semantic meaning. It allows the system to consider that the words coded and programmed can mean one another and improves the accuracy of the matches even in cases where the correct words are not present.

Random Forest: This ensemble learner approximates the qualities of candidate fitness by examining structured features such as years of experience, location and density of skills. The Random Forest generates a score of fit which informs us on the level of suitability of a candidate to a particular position.

V. Implementation Detail

A. Frontend Interface

The frontend was developed using React.js since it is modular and simple to maintain. The UI is divided into two role based dash boards: one is for the recruiters, the other is that of job seekers.

On the side of job seeker, we concentrated on ease of design and fast access of information. Key features include:

Recommendation Feed: A dynamic feed in which the users can browse and apply to the jobs which the backend has ranked to them. Integrated Communication: In-app chat and video call allow job seekers to have a direct chat with recruiters on the dashboard.

Skill Analytics: A special area that displays skill deficiencies and provides recommendation of improvement based on resume analysis. There is the recruiter dashboard that is management and tracking-related. Its core functions are:

Applicant Management: Applicant can be able to post new positions, receive and read the applications as well as access the parsed resume data.

Data Visualization: We incorporated the use of Chart.js to present metrics to ensure that the recruiters have an easy time to detect application trends and hiring statistics.

Real-Time Alerts: With WebSockets, the dashboard provides the new-releases alerts including new applications and messages that appear in real-time- thus no longer do you need to wait and hope that the server will reply

B. Backend API Layer

The back-office is entirely Node.js with Express.js, which is the central point where API requests are fixed and business logic is executed, and communicates with the database. Some of the important functionalities put in place in this layer are: Authentication and Security: To ensure that we leave the session locked, we send JWTs to be stateless (auth) and have passwords churned with Bcrypt to ensure that even pulling the DB, one is left with gibberish.

Input Validation: This is very essential to prevent the standard fraudsters such as SQL injection or sloppy data coming in. Database Management: The information resides in different collections between the users, the job advertisements, the application status, the chat logs- taking the first step out of the strict tables allowing the structure to expand naturally.

File Handling: All resumes, profile pics etc. are handled via multer, and large files get placed in GridFS to make sure the main DB is not clogged.

C. AI Engine Integration

This Python/Flask micro-service is a RESTful-enabled communication with the main server, which allows the main event loop to be blocked by heavy computations which can be performed in the background.

The engine addresses a number of heavy hitters:

Resume Parsing: It uses NLP models such as SpaCy and BERT to extract skills, work experience, and academics of raw resume text.

Match Scoring: This algorithm computes a similarity of vectors between only candidate profiles and job descriptions, which results in a score that is used to operate the recommendation engine.

Individualized Learning Journey: It suggests learning materials (courses, articles) to fill in any gaps on a resume after comparing it to gigs.

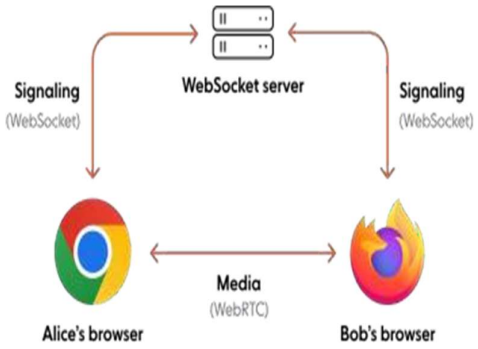

Fig. 2. simple real time chat and call using webrtc and websocket.

VI. System Result Check

The site has two classes of users namely, recruiters and candidates. Every job has other sets of things to do. Recruiters are able to post their job listings, form assessments tests, follow the applicants, and see all of it via their own dashboard.
Applicants have the ability to post resumes, pass the login tests, check the scores, and get in touch with recruiters. We used the same login page of the two however after logging in the system identifies the type of user and redirects them to the right area. The information is stored separately and hence cannot get mixed or be leaked, it is solid and simple in design.
We put a number of safety features on the backend and the front.

A. Experimental Setup

We had initially tried this system on my laptop and also on another network system to determine whether the system continues to perform well when additional people are online. We only wanted to look out to slow downs or accidents.
Setup we used:

- The backend is based on Node.js v18 and Express v4.18.
- MongoDB v6.0 for data storage
- React.js 18.2 for the frontend
- WebChat and WebVideo via WebSocket (Socket.io) and WebRTC.

Has been tested on a regular laptop: Intel i5 (10th gen), 8 32GB RAM, Windows 11.

B. Performance Analysis

We tested the system in terms of the behavior under the load. The recruiters and the job seekers were online at the same time. The backend was fast, approximately 180ms when about 500 users were requesting services. The jobmatching logic was slower, approximately 300ms, though it was okay. WebRTC video was not a problem up to 720p and a 5Mbps connection.
WebSocket chat was nearly real-time with a difference of 50-80ms. In general, there was no crashing or halting.

C. Usability Testing and Feedback of the Users. It was tested by a sample of actual users: some 20 recruiters and 40 applicants. Their feedback:

- 92 percent claimed that the job-matching feature was helpful and precise.
- 85 percent gave likes to links and courses in learning resources.
- 95 percent of them reported chat and video calls to be smooth with no significant lag.
- In brief, the system was popular among people, as it was simple to work with and appeared to be easy to do everyday job.

VII. Scalability and Future Enhancements

At the moment, the system will be able to expand in the future. We can either run services in Docker containers or manage
them in using Kubernetes so that the platform does not crash when traffic soars. We are considering introducing deep-learning models in the future to provide a better evaluation of candidates, based on data to predict the kind of workers companies will require. We can as well test a blockchain layer to check the credibility of the resumes and certificates.

VIII. Discussion

The results of the test were not bad. It has intelligent applications that include live chat and calling that make the hiring process less tedious. The platform also combines job search, applicant screening, and interviews all-in-one as opposed to using 3-4 tools.

A. Recruiter Perspective

This is of great assistance to recruiters. They will be able to screen candidates in a short period of time, perform live interviews and then draw reports. The resume engine is more fairer because it matches the meaning through NLP rather than using keywords. No longer searching to find arbitrary keywords. The charts created using Chart.js display the number of applicants, recruitment schedules, etc. Not very complicated, yet helpful.

B. Candidate Perspective

The site is extremely user-friendly to the job seekers. It suggests roles that suit you and shows areas of skill deficiency as well as learning materials. The educational element aids in upskilling. You can even chat with recruiters in real-time using WebRTC – it is much more realistic than the outdated job boards. There are no more email delays, simple call or chat done. All in all, it is quick, easy and actually superior.

C. Architectural Significance.

It is a modular architecture, and this implies that it is easy to make adjustments or replace the modules in the future. All feature strands such as AI engine, chat, job module are independent and communicate over either REST APIs or websocket. That lessens the possibility of one point of failure. Moreover, it is relatively easy to add new functionality (such as multi-language support or a resume blockchain validator) to it as a bolt-on. In essence, it is easy, malleable and scaled.

IX. Limitations and Future Enhancements

Although it is solid, still there are some hiccups to address in future releases.

A. Current Limitations

Data quality: With a malfunctionally structured resume, the parser will not work well - format counts. Language: It is largely English only at the moment, so nonEnglish speakers are not in luck.
Security WebRTC can be fast, though it may be susceptible to data eavesdropping or unauthorized connections. Behavioral intuition: This is the ranking of candidates, which is yet to seriously investigate their behavior.

B. Proposed Enhancements

To facilitate a smooth roll out we may include: Blockchain authentication to ratify credentials; Web cam emotion analytics to detect confidence; Tenure and performance estimation models; Larger language coverage including models such as mBERT or XLM-R; VR/AR interview rooms to be more immersive; Auto-summaries of recorded interviews to allow recruiters to scan rather than re- view.

X. Conclusion

This project demonstrates that it can make the process of hiring much easier when AI, live chat, and video are combined. We do all that, posting jobs, screening resumes, shortlisting and final interviews in one hub. Websockets and WebRTC ensure the user experience is fast, and its design is modular and therefore scavengable.
It accelerates the hiring and enables applicants to focus on the desired skills gaps, by using NLP to match jobs, conduct live interviews, and provide personalized learning tips. It saves time both to the recruiters and seekers and seems fresh. The
system is smooth and user friendly and can be configured or expanded with the increase in traffic.
It also demonstrates how smart technologies can supplement human judgment in order to have fair and quicker hiring. Firms can insert this into the Hr stack and grow to huge teams. It is not only a job board but the entire talent centre as chat, assessments, and learning are all encompassed.
We will later accommodate several languages and different types of jobs and it would really be global. In general, this project is a preview of what hiring will be in the future, easy, intelligent, and connected, all in one location.

Acknowledgements

I would like to express my gratitude to Dr. Rakesh Ranjan of ABES Engineering College, Ghaziabad, who supported my project and guided me in this project at all times. He advised me a lot and he was always the one who motivated me. An enormous congratulations to the Computer Science and Engineering department who had to offer the systems, tools, and space to conduct the tests. This would not have been possible without those resources.

References

[1] J. B. Boudreau and R. S. Ramstad, Beyond HR: The New Science of Human Capital. Boston, MA, USA: Harvard Business School Press, 2007.

[2] T. H. Davenport, J. Harris, and J. Shapiro, “Competing on talent analytics,” Harvard Business Review, vol. 88, no. 10, pp. 52–58, 2010. [3] M. Chamorro-Premuzic, D. Winsborough, R. Sherman, and D. Hogan, “New talent signals: Shiny new objects or a brave new world?” Industrial and Organizational Psychology, vol. 9, no. 3, pp. 621–640, 2016.
[3] A. Malinowski, T. Keim, O. Wendt, and T. Weitzel, “Matching people and jobs: A bilateral recommendation approach,” in Proceedings of the 39th Hawaii International Conference on System Sciences (HICSS), 2006, pp. 137–146.
[4] S. Kapoor, M. Yadav, and A. Sharma, “Automated resume screening using natural language processing and machine learning techniques,” International Journal of Computer Applications, vol. 178, no. 35, pp. 1–6, 2019.
[5] D. G. B. Allen, R. Vardaman, and J. Bryant, “Recruitment and selection: From human resource management to talent acquisition,” Annual Review of Organizational Psychology and Organizational Behavior, vol. 4, pp. 35–61, 2017.
[6] K. Upadhyay and K. Khandelwal, “Applying artificial intelligence: Implications for recruitment,” Strategic HR Review, vol. 17, no. 5, pp. 255– 258, 2018.
[7] T. H. Davenport and J. Kirby, Only Humans Need Apply: Winners and Losers in the Age of Smart Machines. New York, NY, USA: Harper Business, 2016.
[8] M. Chamorro-Premuzic, T. Winsborough, R. Sherman, and D. Hogan, “New talent signals: Shiny new objects or a brave new world?” Industrial and Organizational Psychology, vol. 9, no. 3, pp. 621-640, 2016.
[9] J. B. Boudreau and R. S. Ramstad, Beyond HR: The New Science of Human Capital. Boston, MA, USA: Harvard Business School Press, 2007.
[10] T. Weitzel, O. Wendt, and T. Keim, “A framework for recruiting support systems based on semantic matching,” Decision Support Systems, vol. 42, no. 2, pp. 627-641, 2006.
[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” in Proc. International Conference on Learning Representations (ICLR), 2013.
[12] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in Proc. NAACL-HLT, 2019, pp. 4171-4186.
[13] C. Jennings, T. Hardie, and M. Westerlund, “Real-time communications for the web (WebRTC): Overview and architecture,” IETF RFC 8825, 2021.
[14] Y. Zhang and Q. Yang, “A deep learning approach for resume classification and candidate screening,” IEEE Access, vol. 7, pp. 187964– 187973, 2019.
[15] P. Lops, M. De Gemmis, and G. Semeraro, “Content-based recommender systems: State of the art and trends,” in Recommender Systems Handbook, Springer, 2011.
[16] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, “Neural architectures for named entity recognition,” in Proc. NAACL- HLT, 2016.
[17] C. Cortes and V. Vapnik, “Support-vector networks,” Machine Learning, vol. 20, no. 3, pp. 273–297, 1995.
[18] S. Staab and R. Studer, Handbook on Ontologies. Berlin, Germany: Springer, 2009.
[19] B. Pang and L. Lee, “Opinion mining and sentiment analysis,” Foundations and Trends in Information Retrieval, vol. 2, no. 1–2, pp. 1–135, 2008.
[20] J. Manyika et al., “Artificial intelligence: The next digital frontier?” McKinsey Global Institute Report, 2017.