Building a Fair Job Recommendation System

Anna Abrahamyan

Laboratoire Hubert Curien

Inasoft*

Supervisors: Antoine Gourru, Christine Largeron, Bissan Audeh

July 1, 2024

Abstract

The rapid growth of information causes job searching to become a time-consuming task that has brought significant advancements in job recommendations field. However, these systems often face challenges related to fairness, where biases in the data or algorithms can lead to unequal opportunities for job seekers from different demographic groups. This project aims to develop a job recommendation system that will mitigate biases and ensure fair job recommendations across diverse demographics. We propose a framework based on a ranking approach that generates high-quality recommendations. Our goal is to deliver real-time fair recommendations to millions of job seekers, considering hundreds of job offers.

Contents

*Inasoft is a technology company focused on developing advanced software solutions for human resources and talent management.https://www.inasoft.fr/

1 Introduction

1.1 Job recommendation systems

Recommender Systems have, for long, been applied to help users find items (e.g., goods or services) that match their personal interests[3]. According to Mauricio Noris Freire [10], recommender systems are designed to address information overload by filtering relevant information from continuously generated data and capturing users' preferences, interests, or behaviors regarding various items. These systems can predict the degree to which a particular user might prefer one item over others based on their profile. With the continuous increase in online information, recommender systems have become an effective solution for managing information overload.

As digital platforms have evolved, the principles of recommender systems have been adapted to various domains, including job search. Job recommendation systems have emerged as crucial tools in the employment market, leveraging the same foundational concepts to match job seekers with potential employers. These systems utilize machine learning algorithms to analyze user profiles and job postings, providing personalized job recommendations that take into consideration the unique qualifications of each job seeker. By filtering through vast amounts of data, job recommendation systems aim to streamline the job search process, making it more efficient and targeted. Recommandation systems are dual: They can be done from the user point of view or from the item point of view. The task of recommending candidates to job postings differs significantly from traditional recommendation problems like suggesting movies or music to users. Factors such as the large amount of textual data, the reciprocal and temporal nature of vacancies, and the fact that these systems deal with personal data does require a tailored approach [7]. E-Recruitment platforms emerged as a feasible solution to help with the problem of allocating, professionals, decreasing the recruitment time and advertisement costs, but, at the same time, increasing the data volume with which the Human Resource Management professionals must deal with [10].

As is common in the literature job recommendation systems are split into Content-Based Recommender Systems (CBR), Collaborative Filtering (CF), Knowledge-based Recommender Systems (KB), and Hybrid Recommender Systems (HRS). The first approach is Collaborative Filtering (CF), which captures information based on historic logs of users to infer their needs, but suffers from the cold-start problem as the main challenge. The second approach is Content-Based Recommendation (CBR), which is based on the comparison between features of USERs and ITEMs. The third approach is called Knowledge-Based Recommendation (KBR), which is based on heuristics that can be built by domain specialists. The limitation of KB recommendation is related to engineering solutions and its scalability. In addition, there are also Hybrid models, which mix other techniques together in order to overcome the problems of single approaches, delivering better results. And it was shown in different papers, that indeed the combined version work the best. [4]

1.2 Challenges of Job Recommendation Systems

As noted by Fernandez and Gallardo-Gallardo [9], e-recruitment faces specific challenges compared to the traditional recommendations on ecommerce platforms (e.g. movies, songs). Firstly, due to the sensitivity of the data, the domain lacks a comprehensive open benchmark for comparative assessment of algorithms. Secondly, the recruitment domain

heavily relies on textual data of partly personal and sensitive nature, for instance, the resumes, and job posts which can contain sensitive information about candidates and employers, respectively. This complexity makes it challenging to create matches.

Additionally, job postings often aim to hire a limited number of employees, imposing quantity constraints on recommendation conversions. Recommending a job to too many candidates can lead to excessive reviewing and interviewing efforts for companies, as well as increased competition among candidates for the same job. This is known as congestion phenomena [13]. Recommender systems may suffer from congestion, meaning that there is an unequal distribution of the items in how often they are recommended [13]. To address this efficiently, we model the candidate recommendation task as a resource allocation problem, recommending candidates under specific quantity constraints. Lastly, due to the significant impact of e-recruitment systems on people's lives, it is crucial to enforce fairness in recommendations and address potential biases in the data.

Another challenge is rightfully defining a job. Job seekers may have different interpretations of a "job". That is, if the definition is too specific, each job would have too few observations for inference. Whereas if the definition is too broad, the recommendation may be accurate, but not precise, and therefore not useful [7]. We define job is a specific good with special characteristics. According to Wikipedia , the notion of the rival good designates a competitor in which consumption by one person prevents consumption by others (ie an apple can only be eaten once, generally by a single person). On the contrary, the non-rival good is one that can be simultaneously consumed by several people without causing direct loss of the good (ie an MP3 song can be downloaded and listened to by millions of people at the same time). The majority of such non-rival goods are often intangible. When one sells the intangible good, one still possesses it. There is therefore no exchange, but rather a duplication of the good. It was Lawrence Lessig , law professor at Stanford University and creator of the legal tool Creative Commons , who first enlightened us about the difference between the rival good and the non-rival good.

As discussed before, the cold start problem (CF) refers to the difficulty of making accurate recommendations when there is insufficient data. This can occur with new users who have little to no interaction history or new job postings with limited information. Luckily, hybrid models, Collaborative Filtering combined with Content-based solve this problem. Data sparsity issue is also common in recommendation systems where there are many users and items, but only a small number of interaction information available. We can solve this kind of problems using collaborative filtering algorithms, such as item-based or user-based approaches, to recommend items based on similar users or items.

Another challenge in job recommendation systems is appropriately defining a "job." Job seekers may have different interpretations of what constitutes a job. If the definition is too specific, there will be too few observations for inference, leading to potentially unreliable recommendations. Conversely, if the definition is too broad, the recommendations may be accurate but lack precision, thus becoming less useful for job seekers and employers alike [7].

In our context, we define a job as a specific good with unique characteristics. This notion aligns with economic concepts of rival and non-rival goods. According to Wikipedia, a rival good is one where consumption by one person prevents consumption by others. In contrast, a non-rival good can be consumed simultaneously by multiple people without diminishing its value. Most non-rival goods are intangible, and selling such a good doesn't result in its loss but rather its duplication [8]. This differentiation is crucial for our job recommendation system, as it helps in understanding and modeling jobs as rival good

entities with specific characteristics that need careful consideration to ensure precise and useful recommendations.

Moreover, defining appropriate metrics to evaluate the performance of a job recommender system is challenging, especially when considering fairness. Yuying Zhao and Yu Wang, discuss in their report the several research questions that need to be addressed [18]. Firstly, the metric selection. Within various metrics, how to choose the specific ones in practice. Next, model optimization, how to balance different objectives, especially when the objectives conflict with each other. Lastly, model evaluations, to compare and evaluate the model performances when multiple metrics are provided. Rank-based evaluations via the average of the ranks in multiple metrics could avoid the scale issue but cannot be applied in model selection.

Addressing these challenges requires dedicated and extensive research efforts.

1.3 Fariness in Job Recommendation Systems

However, despite their benefits, job recommendation systems can inadvertently amplify biases present in the training data, leading to unfair treatment of certain groups of users. For instance, as presented in paper [11], gender stereotypes for particular professions can further find their way into Job Recommendation Systems in the form of algorithmic bias. Algorithmic bias can cause discrimination in the exposure of the job advertisement or the algorithmic hiring itself. The Amazon hiring algorithm, for example, infamously favoring male over female job applicants[6]. The historical data provided to the algorithm suggested that male applicants were preferred because previously more men than women had been hired. Such algorithmic biases can have legal consequences.

Fairness in job recommendations is essential to ensure that all users, regardless of their demographic characteristics, have equal access to job opportunities. This project focuses on building a job recommendation system that addresses not only accurately matching candidates with jobs but also the fairness concerns. By integrating fairness constraints into the recommendation algorithms and evaluating their performance using fairness metrics, we aim to create a system that provides balanced and unbiased recommendations.

In this report, we propose a framework based on a ranking approach based on similarity matching between jobs and candidates, which generates high-quality recommendations of candidates to jobs.

This paper is organized as follows, in section 2 we present the related work of fair job recommendation systems. In section 3, we talk about the context of the project with problem formulation and purpose. In section 4, we present the Datasets, Ground Truths and Exploratory Data Analysis. In section 5, we demonstrate the methods we use for building our model. In section 6, we present the results. Finally, we conclude this work in section 7 and talk about future work.

2 Survey of the State of the Art

Recommender systems represent user preferences for the purpose of suggesting items to purchase or examine. They have become fundamental applications in electronic commerce and information access, providing suggestions that effectively prune large information spaces so that users are directed toward those items that best meet their needs and preferences. [5]

The concept of Recommendation systems was first introduced in the early 1980s by Salton and McGill. And it has been evolving and continuing to evolve ever since. The recommendation systems are usually formulated as $f : U \times I \rightarrow R$ where U is the space of all users, I is the space of all possible items, and f is the utility function that projects all combinations of user-item pairs to a set of predicted ratings R, which is composed of nonnegative integers. [17]

Shuo Yanga, Mohammed Korayem present a model combining content-based and collaborative filtering for job recommendation system, with reduced rate of inappropriate job recommendations. [17]

Many contributions use one of the data sources made available through data science competitions for training and validating job recommender systems. Most noticeably, these include the RecSys 2016[1] and RecSys 2017 [2] competitions, using a dataset from the job board Xing, and the CareerBuilder 2012 Job Recommendation Challenge, which was hosted by CareerBuilder on Kaggle. All three datasets contain data with respect to candidate profiles, vacancies and online interaction between the two. [7]

As discussed, recommender systems are an essential tool to relieve the information overload challenge and play an important role in people's daily lives. As they are working with human beings, an important issue is whether recommendations are fair.

Unfairness exists in different recommendation scenarios and various resources for both users and items. For users, there are significant differences in the recommendation accuracy between users of different ages and genders. For items, existing research has found that minority items could get worse ranking performance and less exposure opportunity. Both traditional recommendation methods and deep learning models can suffer from unfairness.[16]

Many fairness-related works concern unfairness issues from the item side, and focus on the popularity bias in recommendations. Such problem can be addressed by increasing the number of unpopular items, or otherwise, the overall catalog coverage in the final recommendation list. [12] These biases can arise from various sources, such as historical data reflecting discriminatory practices or algorithmic biases that unintentionally favor certain groups over others. As a result, these systems may inadvertently perpetuate inequality in the job market by providing unequal opportunities to different demographic groups.

3 Context of the Project

.

This project is conducted within the framework of the Laboratoire Hubert Curien in collaboration with Inasoft. The aim is to develop an accurate job recommendation system and to address the critical issue of fairness in such systems. This topic has gained significant attention in recent years due to the growing impact of machine learning algorithms on hiring practices.

Laboratoire Hubert Curien is a leading research institute specializing in computer science, information systems, and data science. The laboratory is committed to advancing the state of the art in these fields and addressing societal challenges through innovative research. Inasoft, a partner in this project, is a technology company focused on developing advanced software solutions for human resources and talent management.

The motivation for this project stems from the recognition that traditional job rec-

ommendation systems often suffer from inherent biases. By developing an accurate and fair model, our aim is to enhance the efficiency of matching between candidates and job opportunities within the system without discrimination. In this internship, we are addressing three critical issues:

Defining Ground Truth: Establishing what should be considered as the ground truth is essential for evaluating the accuracy of the job recommendation system. This involves determining the most relevant and unbiased data sources and metrics that accurately reflect successful job placements and candidate-job matches.

Fairness in Recommendations: We are investigating whether the evaluated system provides fair recommendations. This entails analyzing the system's outputs to ensure that they do not favor or disadvantage any group based on gender, race, age, or other protected characteristics, thus promoting equitable opportunities for all candidates.

Enhancing Recommendations with Additional Information: We aim to improve the recommendation system by incorporating additional information. This could include integrating new data sources, utilizing advanced machine learning techniques, or applying domain-specific knowledge to better understand the context and preferences of both job seekers and employers.

By addressing these issues, we hope to create a more reliable and unbiased job recommendation system that benefits both employers and job seekers.

3.1 Problem Formulation

We have a set of jobs and for each job their corresponding set of candidates.

Let's assume J is a set of jobs:

$$J = \{j_1, j_2, \ldots, j_n\}.$$

where each job is denoted by j_i for $i = 1, 2, \ldots, n$.

Each job j is determined by a vector embedding corresponding to the textual representation of the job description, required profile or job title.

For each $j_1, \ldots, j_n$, we have a set of candidates

$$C_j = \{c_1, \ldots, c_m\}.$$

where each candidate is denoted by c_i for $i = 1, 2, \ldots, m$.

So, for all the jobs we have a total of

$$C = \bigcup_{j=1}^{n} C_j$$

candidates.

Each candidate c is represented by a vector embedding corresponding to the textual representation of the candidate's CV or previous job titles from experiences.

We also define accepted candidates by ac_j and predicted candidates by pc_j.

3.2 Aim

The aim of the project is to design a function Φ, that assigns ranks to candidates applying for a specific job based on similarity scores. The objective is to ensure that the accepted

candidate ac_j for the job j holds a rank within a specified limit, such as 5(e.g. Top 5 best candidates for a specific job), indicating their suitability or preference for the position.

More formally, given a job $j \in J$ and a set of candidates C_j that applied to job j, the aim is to build a function Φ such that:

$$\begin{aligned} \Phi : C_j &\to \{1, \ldots, |C_j|\} \\ \Phi(c) &\to r_c \\ \text{s.t.} \quad r_{acj} &< 5 \end{aligned}$$

Where:

- r_c is the rank of candidates
- r_{acj} is the rank of the accepted candidate.

4 Datasets

Inasoft has provided us with several datasets for developing and evaluating the job recommendation system. These datasets contain various aspects of the recruitment process, providing a comprehensive foundation for our analysis and model development.

- **Accepted Applications:** This dataset contains records of all applications that were successfully accepted. Each entry includes detailed information about the candidate, the job they applied for, and the outcome of the application process, and some other demographic information. Around 2260 accepted applicants, out of which only 734 have their corresponding CVs).
- **Rejected Applications:** Similar to the accepted applications dataset, this file includes records of all applications that were rejected. It provides insights into the reasons for rejection and again demographic information.
- **Information about Vacancies (Jobs):** This dataset contains detailed information about the job vacancies, including job titles, descriptions, requirements, locations, and other relevant attributes. This information is critical for matching candidates to suitable job openings.
- **JSON Files with Extracted Information and Textual Content from CVs:** These JSON files contain structured data extracted from the candidates' CVs, including personal information, skills, work experience, education, and other relevant details. Additionally, the raw textual content of the CVs is included, which can be used for advanced natural language processing (NLP) techniques to better understand candidate profiles. (Around 78000 files corresponding to candidates with extracted textual information from the CVs).

4.1 Data Merging Process

To build a comprehensive dataset for developing our job recommendation system, we needed to combine the already mentioned datasets provided by Inasoft. The merging

process involved the following steps. Firstly, we began by combining the accepted applications dataset with the rejected applications dataset. This combined dataset included all applications for the jobs by the candidates. Next, we integrated the combined applications dataset with the JSON files containing extracted information and textual content from candidate CVs. We then merged the enriched candidate dataset with the job vacancies dataset. During the merging process, we ensured to exclude spontaneous applications. This was necessary to maintain the integrity of the dataset and ensure that each application was relevant to a specific job opening. Finally, we filtered the dataset to include only those job vacancies that had received applications from more than one candidate including the accepted candidate. This step was crucial for ensuring that our analysis and recommendations were based on competitive job openings, providing a more realistic evaluation of the recommendation system's performance. From our combined dataset, we define an observation as follows:

An observation represents a single instance of a job application, characterized by the specific job j, the corresponding candidate c who applied for the job, and the decision made on the application (hired or rejected). In Table 1, the number of observations for each file are presented thoroughly.

Steps	Description	Observations
1	Accepted and Rejected applications combined with CV files	65617
2	Remove applications with empty textual information corresponding to the CV	44230
3	Remove jobs that don't have a CV of an accepted candidate	16544
4	Keep only the jobs that have more than 1 candidate	16502
5	Remove spontaneous applications	14427

Table 1: Dataset Formulation Steps

For each of the ground truth datasets, we have different versions that include various features:

- **Case1: Job Description and Text CV:** This version includes required profile combined with job description from the job postings, along with the full textual information from candidates' CVs.
- **Case2: Full Job Description and Text CV:** This version combines the job title, required profile, and job description from the job postings with the full textual information from candidates' CVs.
- **Case3: Required Profile and Experience Titles:** This version contains the required profile from job postings and specific textual information about previous job titles from candidates' CVs.
- **Case4: Job Titles and Experience Titles:** This version includes job titles from the job postings and specific textual information about previous job titles from candidates' CVs.

4.2 Ground Truths

In the context of job recommendation systems, ground truths are essential for evaluating the accuracy and effectiveness of the recommendations. These ground truths are derived from the item-user interaction matrix, which records the interactions between candidates (users) and job postings (items) at various stages of the recruitment process. We define three types of ground truths based on different stages of the recruitment process:

- **Recruitment Ground Truth (GT1):** This is the most strict one. A candidate is considered positive only if they have been successfully recruited. This ground truth represents the final outcome of the recruitment process and is recorded in the item-user interaction matrix as a successful match between a candidate and a job posting.
- **Manager Ground Truth(GT2):** This criterion is slightly less strict. A candidate is considered positive if they have been recruited or if they have received a negative assessment after an interview with a manager (NAE). This ground truth includes candidates who were strong enough to reach the manager interview stage. In the item-user interaction matrix, this is recorded as interactions that reached the interview stage, providing a broader view of candidate suitability.
- **HR Ground Truth(GT3):** This is the least strict criterion. A candidate is considered positive if they have been recruited, received a negative assessment after a manager interview (NAE), or received a negative assessment after a telephonic interview with a Human Resources employee (NAT). This ground truth encompasses the entire funnel of the recruitment process, including early-stage evaluations. The item-user interaction matrix captures these various interactions, allowing for a comprehensive assessment of the recommendation system's performance across all stages of the recruitment process.

4.3 Exploratory Data Analysis

Ground Truth	Total Job Offers	Total Unique Candidates
Recruitment	491	13,566
Manager Interview	656	17,982
Telephone Interview	1,129	26,361

Table 2: Statistics for Different Ground Truth Datasets

Ground Truth	Avg. Candidates per Job	Avg. Rank of Accepted Candidate
Recruitment	29.3	15.2
Manager Interview	29.8	15.4
Telephone Interview	26.5	13.8

Table 3: Statistics for Different Ground Truth Datasets

In this section, we present some interesting insights derived from our dataset using exploratory data analysis (EDA). The following plots provide visualizations that offer a deeper understanding of the data characteristics and distributions.

Figure 1 shows the distribution of candidates across the top 10 postal codes with the highest number of candidates. The most common departments are Nord(59), Seine-Maritime(76), Rhône (69). In Figure 2, we can see the distribution of candidates categorized by gender. So we can see that male candidates are more than females. Next, figure 3 illustrates the distribution of job offers categorized by different types of contracts, highlighting the prevalence of each contract type in your dataset. The most common one out of 4 types is CDI.

Figure 4, presents the distribution of job offers categorized by job roles, offering an overview of the demand across different departments. Last figure displays the distribution of applications categorized by status categories.

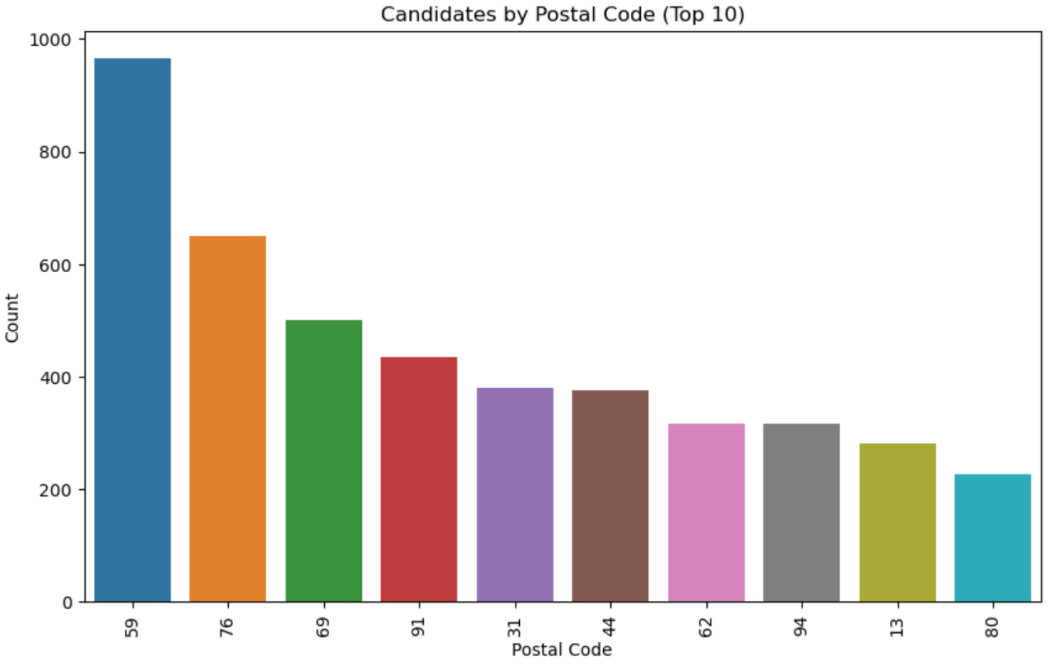

Figure 1: Candidates by Postcode

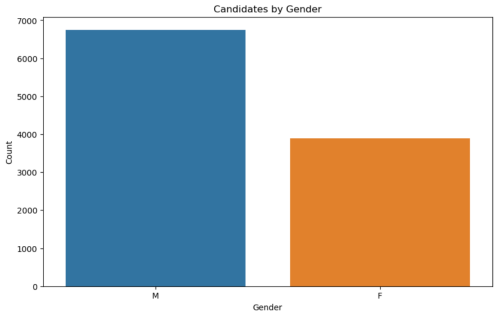

Figure 2: Candidates by Gender

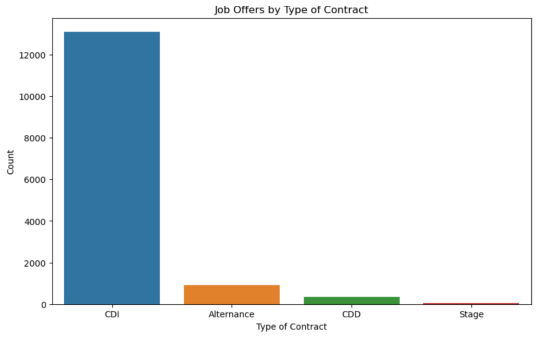

Figure 3: Job offers by type of contract

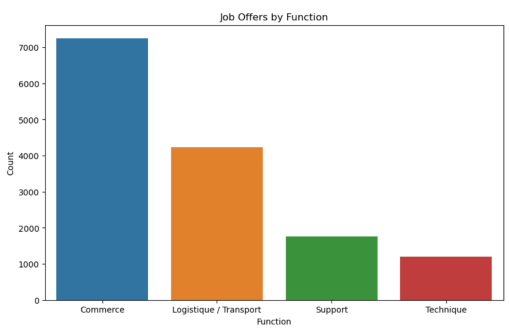

Figure 4: Job offers by function

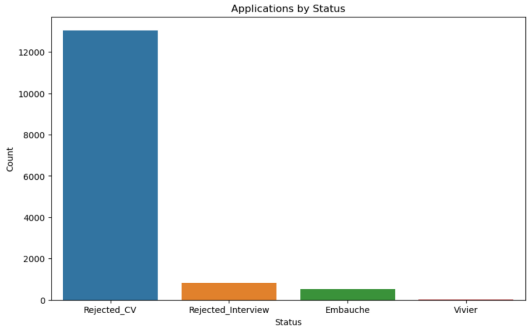

Figure 5: Application by status

5 Model and Methodology

In this section, we outline the model architecture and methodology used for the comparison of candidate rankings based on metric learning and cosine similarity. Initialization of the problem is presented here:

- We organize our dataset by jobs. For each job, we consider both accepted and rejected candidates. And we sort them so the accepted candidate will always be in the first position.
- Example:
  - For job $j1$, we have a list of candidates corresponding to job $j1$ [$c1$, $c2$,$c3$, ... $c15$].
  - For job $j2$, candidate list [$c1$, $c2$,$c3$, ... $c45$].

  Note: For $j1$ and $j2$ the $c1$ accepted candidate id not the same candidate.

5.1 Approach

We want to determine the similarity between the job j and candidate c. To achieve this, we can represent the textual content of both the job and CV as vectors, where each element of the vector corresponds to a term and its frequency in the respective document. To implement this, we use pre-trained Sentence-CamemBERT-Large models. This is the Embedding Model for French language problems.

Let $\hat{j}$ represent the vector representation of the job j, and $\hat{c_j}$ represent the vector representation of the CV of candidate c for the specific job j.

For computing the similarity of jobs and candidates we use two measures:

- Cosine similarity

  Cosine similarity is calculated using the cosine of the angle between the two vectors. The cosine similarity between $\hat{j}$ and $\hat{c_j}$ can be computed using the following formula:

$$(\hat{c_j}, \hat{j}) = \frac{\hat{\mathbf{c}}_\mathbf{j} \cdot \hat{\mathbf{j}}}{\|\hat{\mathbf{c}}_\mathbf{j}\| \|\hat{\mathbf{j}}\|} \tag{1}$$

  Where:

- $\hat{\mathbf{c}}_\mathbf{j} \cdot \hat{\mathbf{j}}$ denotes the dot product of $\hat{\mathbf{c}}_\mathbf{j}$ and $\hat{\mathbf{j}}$.
- $\|\hat{\mathbf{c}}_\mathbf{j}\|$ represents the Euclidean norm (magnitude) of $\hat{\mathbf{c}}_\mathbf{j}$, and similarly for $\hat{\mathbf{j}}$.

This cosine similarity value ranges from -1 to 1, where 1 indicates perfect similarity, 0 indicates no similarity, and -1 indicates perfect dissimilarity.

- Metric Learning

  Metric Learning aims at automatically constructing task-specific distance metrics from weakly supervised data, in a machine learning manner. The algorithm has access to a set of data points with supervision only at the tuple level (For example, the algorithm may receive accepted and rejected pairs of data points).The goal is to learn a distance metric that puts positive pairs close together and negative pairs far away. It is implemented using Mahalanobis distance [15].

$$D(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{Lx} - \mathbf{Lx}')^\top (\mathbf{Lx} - \mathbf{Lx}')} \tag{2}$$

  In other words, a Mahalanobis distance is a Euclidean distance after a linear transformation of the feature space defined by L, where L is a parameter matrix, with a shape (num_dims, n_features). Mahalanobis distance metric learning can thus be seen as learning a new embedding space of dimension num_dims. For each pair we compute scores between pairs of points (the larger the score, the more similar the pair).

Next, we rank r_c the candidates based on these similarity scores, such that the accepted candidate having the highest similarity obtains the first rank and those having the lowest similarity obtain the rank closer to $|cj|$. The rankings generated by the metric learning model and cosine similarity are compared to assess which approach yields more accurate and reliable candidate-job matches.

5.2 Baselines

We use average rank of the accepted candidate with standard deviation, expected baseline,and random baseline to evaluate the model. These metrics provide a comparative benchmark by randomly selecting candidates for each job and recording their ranks based on both cosine similarity and metric learning methods.

The average rank, denoted as $\bar{r}$, tells us on average, where the accepted candidate (first item) in each similarity matrix ranks. A lower average rank indicates that the item of interest is ranked higher (i.e., more similar to the target item) across the candidates. The standard deviation, denoted as σ_r, indicates the variability in the ranks. With a lower standard deviation, the ranks are more consistent and stable.

The expected baseline represents the expected value of rank (for the accepted candidate) if items were ranked randomly (random ordering).

Let's consider an example. For job j, we have a set of candidates $C = \{c_1, c_2, c_3\}$.

Now let's see how many possible ways of ordering we have.

Let's assume the accepted candidate for job j, denoted as ac_j, is the first candidate: $ac_j = c_1$.

For the accepted candidate to be ranked 1st we have $\frac{2}{6} = \frac{1}{3}$ possible ways. We denote the accepted candidate rank as r_{ac_j}.

c1	c2	c3
1	2	3
1	3	2
2	3	1
2	1	3
3	2	1
3	1	2

Table 4: Random Orders of 3 candidates

$$P(r_{ac_j} = 1) = \frac{1}{3}$$

For it to be ranked second we have $\frac{1}{3}$ possible ways.

$$P(r_{ac_j} = 2) = \frac{1}{3}$$

And lastly, for it to be ranked 3rd we have again $\frac{1}{3}$ possible ways.

$$P(r_{ac_j} = 3) = \frac{1}{3}$$

Now the expected value of rank will be

$$E[r_{ac_j}] = \frac{1}{3} \times 1 + \frac{1}{3} \times 2 + \frac{1}{3} \times 3 = \frac{1}{3} \times (1 + 2 + 3)$$

In the general case, when we have n number of candidates for job j, we will have

$$E[r_{ac_j}] = \frac{1}{n} \sum_{i=1}^{n} i = \frac{1}{n} \times \frac{n(n+1)}{2} = \frac{n+1}{2}$$

Finally, the random baseline, works as follows: For each job, candidates are ranked based on cosine similarity and metric learning. From this ranked list, a candidate is randomly selected, and the rank of this randomly selected candidate is recorded. This process is repeated across different jobs, and the average rank of the randomly selected candidates is computed. More formally, we can present it this way:

For job $j \in J$ and candidate set for job j as $C_j = \{c_1, \ldots, c_m\}$, ranked based on cosine similarity:

$$F(c) \to r_c, \text{ where } r_{c1}, \ldots, r_{cm}.$$

We will have a random function, which will take the $|c_1, \ldots, c_m|$, and randomly return an x from the cardinality of the candidate set. Next, we will take the index of the candidate of x and check its rank r_{c_x}. Then, implement this for different jobs and compute the average r_{c_x} across jobs.

5.3 Evaluation Metrics

Evaluating the performance of a job recommendation system is crucial to ensure it meets both accuracy and fairness criteria. Several metrics are employed to assess how well the system matches candidates to job postings and to verify that the recommendations are fair and unbiased. In this section, we discuss the key evaluation metrics used in our project. These metrics help us measure the effectiveness of our recommendation system and ensure it provides high-quality and equitable recommendations to all users.

- **Max Accuracy:** This metric measures the overall correctness of the job recommendation system based on whether the top-ranked candidate matches the ground truth (accepted candidate) for each job. It is computed as the percentage of correctly recommended candidates across all jobs by defining a threshold. Specifically, Max Accuracy is determined by calculating the proportion of cases where the accepted candidate is within the top n recommended candidates, with n being the threshold value.

- **Precision, Recall:** Once the threshold n is defined using Max Accuracy, precision and recall are calculated as follows: Precision (P) measures the proportion of relevant candidates found among the candidates returned while recall(R) measures the proportion of relevant candidates found among the candidates to be found. [14] In other words, precision is computed as the ratio of correctly recommended candidates (true positives) to all recommended candidates. Recall is computed as the ratio of correctly recommended candidates (true positives) to all accepted candidates. Figure 6 graphically illustrates these two criteria.

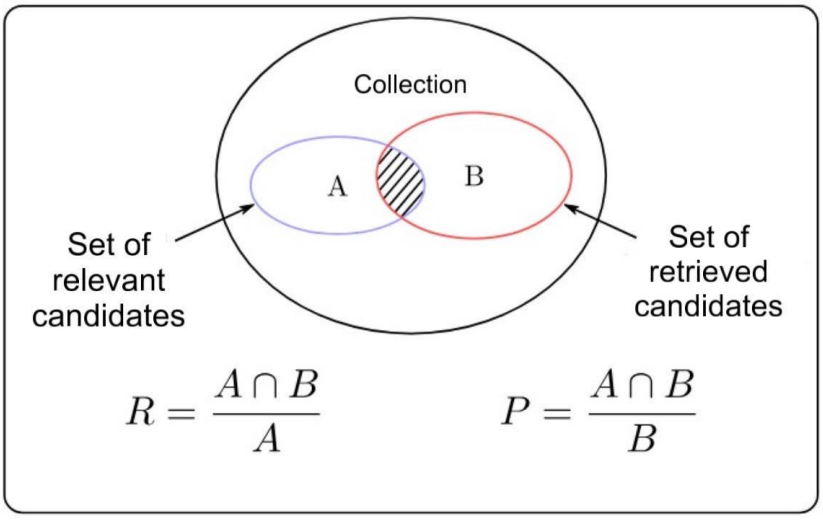

Figure 6: Precision and Recall

We also plan on using more advanced evaluation metrics such as Precision@k, Recall@k, Accuracy@k, and Normalized Discounted cumulative gain(NDCG). Precision@k, Recall@k, Accuracy@k metrics provide a detailed understanding of how well the system performs in identifying the most suitable candidates for a job within the top k recommendations. We consider taking k at 5,10,15 based on the average number of candidates for each job.

- **Precision@k, Recall@k:** For modern recommendation, simple metrics no longer have meaningful results, as any queries have thousands of candidates, and few hiring companies are interested in reading all of them. Precision at k (P@k) is a useful metric (e.g., P@10 or " at 10" corresponds to number of relevant recommended candidates among the top 10 candidates). It is computed by taking the number of relevant items within the top-k recommendations and dividing it by k. Recall@k measures the proportion of accepted candidates that are included in the top-k recommendations. It assesses how well the system identifies all relevant candidates within the top-k recommendations. Recall at k is computed by taking the number of relevant items within the top-k recommendations and dividing it by the total number of relevant items. [14]

$$P_k(N) = \frac{\sum_{r=1}^{k} rel_k(r)}{N} \tag{3}$$

$$R_k(N) = \frac{\sum_{r=1}^{k} rel_k(r)}{|D_k|} \tag{4}$$

where $rel_k(r)$ is a binary relevance function defined as:

$$rel_k(r) = \begin{cases} 1 & \text{if the candidate at rank } r \text{ in list } C_k \text{ is relevant to query } q_k \text{ (belongs to } D_k\text{)}, \\ 0 & \text{otherwise.} \end{cases}$$

- **Accuracy@k:** This metric evaluates the accuracy of the system in recommending the top-k candidates for each job. It measures how often the accepted candidate is found within the top-k recommendations.

6 Results

In this section, we present the results obtained from implementing our job recommendation system on three different ground truths, each incorporating distinct features.

It is important to note that, for the sake of comparison, the ground truths were applied consistently across all 491 job postings. Each ground truth represents a different approach to defining what constitutes a successful job match. By evaluating the system using multiple ground truths, we aim to assess its robustness and accuracy under various conditions. The Tables present a detailed comparison of different models (Metric Learning and Cosine Similarity) across various cases and ground truths. Note, we define Recruitment Ground Truth GT1, Manager Interview Ground Truth GT2, Telephone Interview Ground Truth GT3 and Case1: Description and CV, Case2: Full Description and CV, Case3: Experience and Required Profile, Case4: Experience and Job Titles.

In Table 5, the average rank and standard deviation of these models indicate their performance consistency. Metric Learning consistently outperforms random baselines and cosine similarity across different ground truths and cases, suggesting that the model effectively learns job-candidate relevance based on advanced features beyond basic keyword matching. Cosine Similarity performs worse than Metric Learning but shows consistent results across different GTs with similar average ranks and standard deviations. Random Metric and Random Cosine have higher average ranks and standard deviations, reflecting

		GT1		GT2		GT3	
		Avg Rank	Std Dev	Avg Rank	Std Dev	Avg Rank	Std Dev
Case1	Metric Learning	24.3	34.5	**17.9**	31.1	26.0	49.2
	Cosine Similarity	27.2	50.2	27.2	50.2	26.8	50.2
	Random Metric	37.4	56.7	37.4	71.1	39.0	54.4
	Random Cosine	32.2	43.8	35.3	62.8	33.8	53.5
	Expected	35.2	93.3	35.2	93.3	35.2	93.3
Case2	Metric Learning	23.2	33.1	25.4	40.9	25.1	47.2
	Cosine Similarity	31.0	58.4	31.0	58.4	30.7	58.4
	Random Metric	33.4	46.6	36.5	54.2	38.6	58.1
	Random Cosine	37.2	52.3	35.5	60.0	36.0	65.0
	Expected	35.2	93.3	35.2	93.3	35.2	93.3
Case3	Metric Learning	10.2	19.9	11.6	21.8	6.5	15.3
	Cosine Similarity	31.3	52.1	31.3	52.1	31.6	52.1
	Random Metric	32.0	42.9	33.3	47.3	35.6	71.7
	Random Cosine	32.2	43.8	38.9	76.1	32.2	48.0
	Expected	35.2	93.3	35.2	93.3	35.2	93.3
Case4	Metric Learning	16.3	35.2	8.0	36.0	6.2	12.8
	Cosine Similarity	34.0	73.1	34.0	73.1	34.1	73.0
	Random Metric	34.6	45.0	32.9	51.5	31.1	57.8
	Random Cosine	27.3	41.1	31.4	44.5	30.2	39.5
	Expected	35.2	93.3	35.2	93.3	35.2	93.3

Table 5: Results of the Ground Truths on different cases

less effective and more variable performance. Additionally, the Expected rank had the highest standard deviation, suggesting it is a less reliable measure compared to specific model evaluations. For Ground Truth 2, the results show that again Metric Learning performs well in ranking job candidates, with lower average ranks and standard deviations compared to random baselines and cosine similarity. This indicates that incorporating managerial insights enhances the accuracy of job recommendations, aligning with specific interviewer expectations. For example, in Case1 for GT2, Metric Learning achieved an average rank of 17.9 with a standard deviation of 31.1, whereas Cosine Similarity had an average rank of 27.2 and a standard deviation of 50.2. Analyzing the results for Ground Truth 3, we again see similar results. This suggests that aligning recommendations with HR perspectives improves the system's ability to predict successful job matches. Specifically, Case 4 outperforms other methods for GT2 and GT3, with Metric Learning achieving the lowest average ranks (8.0 and 6.2 respectively) and low standard deviations, whereas for GT1, Case 3 with Metric Learning has the best performance with an average rank of 10.2 and standard deviation of 19.9. These results highlight the robustness and effectiveness of the Metric Learning approach in matching job descriptions with candidate CVs across various scenarios.

The table 6, presents a comparison of Max Accuracy, Precision, and Recall metrics for Metric Learning and Cosine Similarity models across four cases (Case1, Case2, Case3, Case4) and three ground truths (GT1, GT2, GT3). Metric Learning consistently shows competitive performance across all ground truths. In GT1, while Metric Learning achieves slightly lower Max Accuracy than Cosine Similarity in some cases, it demonstrates higher Precision and comparable Recall values. Notably, in GT1 Case4, Metric Learning achieves

GT	Case	Model	Max Accuracy		Precision		Recall	
			Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
GT1	Case1	Metric Learning	92.6	7.6	3.7	3.8	98.0	11.6
		Cosine Similarity	93.6	7.1	3.7	3.8	96.4	10.8
	Case2	Metric Learning	92.6	7.6	3.7	3.8	94.6	9.5
		Cosine Similarity	93.5	7.0	3.7	3.8	95.2	6.7
	Case3	Metric Learning	94.6	7.6	3.7	3.8	97.5	12.6
		Cosine Similarity	93.7	6.1	3.7	3.9	98.6	11.5
	Case4	Metric Learning	95.6	7.6	3.5	3.5	98.6	11.6
		Cosine Similarity	94.0	6.1	3.6	3.9	94.5	22.6
GT2	Case1	Metric Learning	92.6	7.6	3.7	3.8	98.5	15.6
		Cosine Similarity	93.7	7.1	3.7	3.8	96.8	12.8
	Case2	Metric Learning	92.7	7.6	3.7	3.8	97.4	11.7
		Cosine Similarity	93.4	7.0	3.7	3.8	96.7	12.6
	Case3	Metric Learning	93.6	7.6	3.7	3.8	95.4	16.9
		Cosine Similarity	93.7	8.1	3.7	3.9	98.6	11.5
	Case4	Metric Learning	93.4	7.6	3.7	3.8	96.3	16.8
		Cosine Similarity	92.9	6.1	3.7	3.8	94.6	22.6
GT3	Case1	Metric Learning	92.6	7.6	3.7	3.8	95.4	19.4
		Cosine Similarity	93.6	7.1	3.7	3.8	98.5	22.6
	Case2	Metric Learning	92.7	7.7	3.7	3.8	95.8	12.7
		Cosine Similarity	93.5	7.0	3.7	3.8	96.3	10.6
	Case3	Metric Learning	92.6	7.6	3.7	3.8	94.1	18.7
		Cosine Similarity	91.5	6.1	3.7	3.8	98.6	12.5
	Case4	Metric Learning	94.8	7.6	3.7	3.8	92.7	20.7
		Cosine Similarity	93.9	6.1	3.7	3.8	94.5	22.6

Table 6: Comparison of Max Accuracy, Precision, and Recall with Standard Deviation (Percentage, %)

the highest Max Accuracy (95.6%) and Recall (98.6%), indicating its effectiveness in that scenario. Similarly, Metric Learning performs consistently well in GT2 and GT3, showing high Max Accuracy, Precision, and Recall across different cases. For instance, in GT3 Case3, Metric Learning achieves a Max Accuracy of 92.6%, with Precision and Recall values of 3.7 and 94.1 respectively, underscoring its robust performance across various scenarios. The lower Precision observed in both models can be attributed to an imbalance issue in the dataset, where the job recommendation system encounters a high number of rejected candidates compared to accepted, impacting Precision calculations. These results highlight Metric Learning as a reliable choice for achieving consistent and effective performance in matching job descriptions with candidate CVs across diverse datasets.

7 Conclusion and Future Work

In this study, we addressed the critical challenges faced by job recommendation systems, particularly focusing on fairness and effectiveness. We proposed a ranking-based approach that leverages similarity matching between job seekers and job postings to generate high-

quality recommendations. In conclusion, the comparative analysis of Metric Learning and Cosine Similarity models across multiple cases and ground truths demonstrates that Metric Learning consistently achieves competitive performance in terms of Max Accuracy, Precision, and Recall.

Future work will concentrate on enhancing the model using fairness metrics. By integrating fairness metrics into our evaluation criteria, we aim to mitigate biases that can arise from historical data or algorithmic decisions. One of our primary goals is to implement more refined evaluation metrics that are better suited to recommendation systems. Specifically, we plan to incorporate Precision@k, Recall@k and Normalized Discounted cumulative gain(NDCG), as already mentioned. Also we need to address the imbalance problem in job recommendation datasets. Imbalanced data can also lead to biased recommendations. We also consider fine-tuning our model to get better results. By optimizing hyperparameters, improving feature selection, and refining the model's predictive capabilitieswe will improve the accuracy and efficiency of the recommendations, making the system more responsive to the specific needs of job seekers and employers. Lastly, to validate the robustness and generalizability of our model, we plan to test it on publicly available datasets. By addressing these challenges, we aim to enhance the reliability and equity of job recommendation systems, ensuring they serve all users fairly and effectively.

References

[1] Fabian Abel, András Benczúr, Daniel Kohlsdorf, Martha Larson, and Róbert Pálovics. Recsys challenge 2016: Job recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 425–426, 2016.

[2] Fabian Abel, Yashar Deldjoo, Mehdi Elahi, and Daniel Kohlsdorf. Recsys challenge 2017: Offline and online evaluation. In *Proceedings of the eleventh acm conference on recommender systems*, pages 372–373, 2017.

[3] Shaha T Al-Otaibi and Mourad Ykhlef. A survey of job recommender systems. *International Journal of the Physical Sciences*, 7(29):5127–5142, 2012.

[4] Guillaume Bied, Solal Nathan, Elia Perennes, Morgane Hoffmann, Philippe Caillou, Bruno Crépon, Christophe Gaillac, and Michèle Sebag. Towards job recommendation for all. pages 5906–5914, 2023.

[5] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12:331–370, 2002.

[6] Jeffrey Dastin. Amazon scraps secret ai recruiting tool that showed bias against women. In *Ethics of data and analytics*, pages 296–299. Auerbach Publications, 2022.

[7] Corné De Ruijt and Sandjai Bhulai. Job recommender systems: A review. *ResearchGate*, 2021.

[8] Guillaume Deziel. The rival good vs. the non-rival good. `https://guillaumedeziel.com/`

complements/le-bien-rival-versus-le-bien-non-rival/
the-rival-good-vs-the-non-rival-good/, 2024. Accessed: 2024-06-27.

[9] Vicenc Fernandez and Eva Gallardo-Gallardo. Tackling the hr digitalization challenge: key factors and barriers to hr analytics adoption. *Competitiveness Review: An International Business Journal*, 31(1):162–187, 2021.

[10] Mauricio Noris Freire and Leandro Nunes de Castro. e-recruitment recommender systems: a systematic review. *Knowledge and Information Systems*, 63:1–20, 2021.

[11] Deepak Kumar, Tessa Grosz, Navid Rekabsaz, Elisabeth Greif, and Markus Schedl. Fairness of recommender systems in the recruitment domain: an analysis from technical and legal perspectives. *Frontiers in big Data*, 6, 2023.

[12] Yunqi Li, Michiharu Yamashita, Hanxiong Chen, Dongwon Lee, and Yongfeng Zhang. Fairness in job recommendation under quantity constraints. 2023.

[13] Yoosof Mashayekhi, Bo Kang, Jefrey Lijffijt, and Tijl De Bie. Recon: Reducing congestion in job recommendation using optimal transport. page 696–701, 2023.

[14] Christophe Moulin. Modélisation de documents combinant texte et image: application à la catégorisation et à la recherche d'information multimédia. 2011.

[15] Contributors to scikit-learn-contrib/metric learn. Introduction to metric-learn. https://contrib.scikit-learn.org/metric-learn/introduction.html, Year accessed, e.g., 2024. Accessed: 2024-06-27.

[16] Yifan Wang, Weizhi Ma, Min Zhang, Yiqun Liu, and Shaoping Ma. A survey on the fairness of recommender systems. *ACM Transactions on Information Systems*, 41(3):1–43, 2023.

[17] Shuo Yang, Mohammed Korayem, Khalifeh AlJadda, Trey Grainger, and Sriraam Natarajan. Combining content-based and collaborative filtering for job recommendation system: A cost-sensitive statistical relational learning approach. *Knowledge-Based Systems*, 136:37–45, 2017.

[18] Yuying Zhao, Yu Wang, Yunchao Liu, Xueqi Cheng, Charu Aggarwal, and Tyler Derr. Fairness and diversity in recommender systems: a survey. *ResearchGate*, 2023.