Explainable Job Recommender Systems for Recruiters and Consulting Companies

François Mentec

COLLEGE MATHS, TELECOMS
DOCTORAL INFORMATIQUE, SIGNAL
BRETAGNE SYSTEMES, ELECTRONIQUE

Université de Rennes

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES

ÉCOLE DOCTORALE N° 601
Mathématiques, Télécommunications, Informatique, Signal, Systèmes, Électronique
Spécialité : *Informatique*

Par
François MENTEC

Explainable Job Recommender Systems for Recruiters and Consulting Companies

Thèse présentée et soutenue à Rennes, le October 2023
Unité de recherche : DRUID

Rapporteurs avant soutenance :

Maria MADLBERGER — Professeure, Webseter University, Autriche
Jean-Francois PRADAT-PEYRE — Professeur, Université Paris Nanterre et Sorbonne Université, LIP6

Composition du Jury :

Président :	Laurent D'ORAZIO	Professeur, Université de Rennes, CNRS, IRISA
Examinateurs :	Maria MADLBERGER	Professeure, Webseter University, Autriche
	Jean-Francois PRADAT-PEYRE	Professeur, Université Paris Nanterre et Sorbonne Université, LIP6
	Benjamin PIWOWARSKI	Chargé de Recherche, CNRS, ISIR, Sorbonne Université
	Thierry ROGER	Directeur de lab, ALTEN Lab Rennes
Dir. de thèse :	Zoltan MIKLOS	Maître de Conférences HDR, Université de Rennes

Invité(s) :

Prénom NOM — Fonction et établissement d'exercice

Acknowledgement

I am profoundly grateful to the many individuals and organizations whose support and encouragement have made this journey possible.

I owe my deepest gratitude to my primary advisors : Zoltan Miklos, who directed the thesis, and Sébastien Hervieu and Thierry Roger, who guided me along the way. Your mentorship has been invaluable, and I am honored to have had the opportunity to learn from you. I do not forget David Gross-Amblard, who briefly acted as director until Zoltan became an associate professor.

I extend my thanks to the members of my CSI committee : Olivier Ridoux, Benjamin Piwowarski, and Simon Malinowski, for their insightful feedback and expertise in shaping the direction of my research.

This research has been financed by ALTEN with the support of the ANRT, I am grateful for their generous funding.

I am indebted to DRUID members who provided advice and guidance in my research and publications, to my ALTEN coworkers who helped me develop tools to test my theories, and to ALTEN project leaders who used my tools and participated as subjects in experiments.

I am thankful for my parents, Marie-Claire and Jean-Luc, who instilled in me the importance of education. I also want to express my gratitude to my partner, Flo, for her unwavering support, and I wish her the best of luck with her own thesis. Lastly, there's my cat Benjamin, whose cuteness apparently gives him the right to destroy my furniture and disrupt my sleep schedule.

Résumé

Cette thèse explore l'application de techniques d'intelligence artificielle et de traitement du langage naturel à la conception d'un système de recommandation explicable pour le recrutement. L'entreprise ALTEN qui finance cette thèse est intéressée par un tel système en raison de la nature de ses activités de consulting qui l'oblige à constamment recruter de nouveaux consultants. Le système se doit d'être explicable et interactif afin de collaborer avec les recruteurs humains et d'améliorer leur travail sans les remplacer. Une part conséquente de cette thèse met l'accent sur l'explication de la recommandation, son impact sur les utilisateurs, et les opportunités qu'elle présente pour améliorer le système de recommandation.

Durant cette thèse, nous tentons de répondre aux questions de recherche suivantes :

- Comment les techniques basées sur les données, en particulier les ontologies et la vectorisation de texte, peuvent-elles être utilisées pour développer un système de recommandation d'emploi efficace qui soutient les recruteurs pendant le processus de sélection des candidats ?
- Quel est l'impact de la présence d'une explication des recommandations sur l'efficacité du processus de sélection ?
- Quel est l'impact de la présence d'explications sur la qualité de l'expérience utilisateur ?
- Comment un système de recommandation d'emplois peut-il utiliser concrètement des explications pour recueillir des retours utilisateurs plus précis, et comment ces retours peuvent-ils être exploités pour améliorer efficacement le système ? Dans quelle mesure les utilisateurs se sont-ils impliqués dans le système de retours, et quelle a été l'importance des retours utilisateurs dans l'amélioration du système de recommandations ?

Après un premier chapitre introductif qui présente le contexte, les enjeux et les questions de recherche que nous souhaitons adresser, le second chapitre fait un état de l'art des techniques d'intelligence artificielle que nous utilisons, et des travaux similaires. Durant cet

état de l'art, nous présentons notamment des techniques d'apprentissage profond telle que l'apprentissage multi tâches avec MT-DNN, ainsi que le calcul de similarité sémantique entre deux textes à l'aide de méthodes de vectorisation. Ces méthodes qui permettent de transformer divers entités lexicales (i.e. mots, phrases, documents) en vecteurs sont très utiles pour comparer le contenu des offres d'emplois et des CV entre eux. Dans nos travaux nous utilisons notamment des modèles Sentence BERT à cette fin. L'état de l'art fait aussi état de travaux majeurs dans le domaine des systèmes de recommandation pour le recrutement, tel que la découverte du « fossé sémantique » entre les offres d'emplois et les CV pour certaines professions, et l'étude des meilleurs types de systèmes de recommandation en fonction des profils d'utilisateur.

Le troisième chapitre est dédié à la présentation d'une méthodologie que conçue pour construire un jeu de données de mise en correspondance d'offres d'emplois et de CV à partir de CV uniquement. Cette problématique émerge de notre besoin d'un tel jeu de données pour l'entrainement et l'évaluation de nos modèles malgré l'absence d'offres d'emplois rattachés aux centaines de milliers de CV d'ALTEN en notre possession. Nous mettons donc à profit la nature structurée de ces CV (nommés Dossier Technique par ALTEN) pour en extraire les expériences professionnelles et considérer la dernière d'entre elle comme notre offre. Le jeu de donnée ainsi produit est ensuite utilisé pour entrainer et évaluer nos modèles.

Le quatrième chapitre présente un système de recommandation de consultants conversationnel basé sur l'ontologie européenne ESCO. Cette ontologie financée par l'Union Européenne a vocation à couvrir l'intégralité des compétences, professions, et qualifications présentes sur le marché du travail européen. Nous commençons par entrainer un réseau de neurones basé sur BERT à classifier les compétences et postes d'ALTEN dans ESCO. À cette fin, nous utilisons les nombreux labels associés aux nœuds de l'ontologie, et mettons à profit sa structure (en arbre pour les postes et en graphe directionnel acyclique pour les compétences), pour que chaque nœud hérite des labels de ses descendants et ainsi augmenter la quantité de données servant à entrainer le modèle à les reconnaitre/prédire. Nous présentons ensuite notre système de recommandation qui se base sur la mise en correspondance des compétences du CV et de l'offre en utilisant la structure de l'ontologie. Nous proposons trois stratégies de mise en correspondance : une strict, une équilibré et une tolérante, que l'utilisateur peut choisir en fonction de s'il préfère maximiser la précision, l'exactitude ou le rappel. Le système permet à l'utilisateur d'interagir avec en

pondérant l'importance des compétences issues de l'offre. L'utilisateur peut ajuster sa pondération durant le processus de recommandation en se basant sur l'explication (visualisation graphique) de la recommandation. Nous comparons notre système à un réseau de neurones de référence en simulant des utilisateurs, et mesurons un gain quand l'utilisateur parvient à correctement pondérer les compétences de l'offre. Ces travaux ont été publiés dans l'atelier KARS de la conférence RecSys 2021.

Le cinquième chapitre présente une méthode pour entrainer des modèles de classification ontologiques tout en valorisant des données issues de versions obsolètes de l'ontologie. Cette méthode qui s'inspire de méthodes d'entrainement multitâches, possède l'avantage de ne requérir aucune transformation des données et d'afficher une nette amélioration des performances du modèle. Cette problématique est issue d'un processus d'ALTEN visant à classifier ses missions clients dans une ontologie interne afin d'obtenir une vue précise de ces secteurs d'activités. Cette classification de dizaines de milliers de missions à lieu tous les ans et est réalisée à la main par des analystes. Ces données peuvent servir à entrainer un modèle supervisé de classification afin d'aider les analystes dans cette tâche. Le problème est que l'ontologie est ajustée presque tous les ans, et que seule la classification sur la dernière version intéresse les analystes. Au lieu de "jeter" les missions classifiées sur une version obsolète de l'ontologie, nous proposons d'entrainer le modèle en utilisant toutes les missions, mais en les classifiant uniquement sur leur version de l'ontologie. Ainsi le modèle s'entraine sur plusieurs versions et en production nous ne proposons que sa prédiction sur la version la plus récente. Avec cette méthodologie nous mesurons une amélioration des performances sur toutes les versions par rapport à un modèle spécifique par version.

Le sixième chapitre présente AI4HR Recruiter, une plateforme de recommandation de consultants que nous avons développé pour servir de terrain d'expérimentation pour nos recherches. Dans le département R&D d'ALTEN nous avons un processus de recrutement interne nommé « mercato » : chaque semaine, les pilotes projet du département vont sélectionner parmi les consultants disponibles (sans mission client ou projet) lesquels les intéressent pour venir travailler sur leurs sujets. Afin de pouvoir conduire des expérimentations et recueillir des données d'utilisation, nous avons développé une plateforme pour recommander ces consultants aux pilotes projet. Deux systèmes de recommandations y sont intégrés, les deux sont basés sur le contenu des CV (content based) mais l'un utilise de simples mots-clés pour les mettre en correspondance avec les intérêts des pilotes, tandis que le second utilise un modèle de vectorisation de texte Sentence BERT. Nos

expérimentations montrent que le premier a une meilleure précision et exactitude, tandis que le second a un meilleur rappel et est préféré par les utilisateurs. Nous avons aussi pu mesurer un gain d'efficacité sur le processus de recrutement quand une explication a à la recommandation est fournie. Enfin, nous avons proposé une méthode pour que les utilisateurs puissent dire au modèle quand il se trompe (scrutability). Ces retours sont pris en compte au moment de la recommandation et sont utilisés à postériori pour évaluer les modèles de vectorisation et améliorer le système. Ces travaux ont été publié dans la conférence KES 2023 et l'outil est maintenant en production à ALTEN depuis deux ans.

Pour conclure, cette thèse s'est penchée sur la conception et l'étude de méthodes de recommandation de consultants dans des processus du recrutement. Nous avons proposé plusieurs méthodologies allant de la création de jeux de données à l'entrainement de modèles de classification ontologiques, et avons mesuré l'impact de l'explication sur l'efficacité du processus de recommandation. De cette thèse industrielle sont issue deux publications internationales et deux outils en production au sein d'ALTEN. Nous espérons que ces travaux pourront servir de catalyseur vers plus d'explication des systèmes de recommandation et nous souhaitons poursuivre nos travaux par une étude systématique des divers formats d'explications de ses systèmes.

Table of Contents

Chapitre 1

Introduction

This thesis is financially supported by and conducted in collaboration with the consulting company ALTEN, with the objective of studying the feasibility and effectiveness of explainable job recommender systems for enhancing the company's recruitment process.

1.1 The importance of recruitment and its challenges

Recruitment, as a fundamental process in human resource management, has played a crucial role throughout history in shaping the success and growth of organizations. An historic example is Leonardo Da Vinci letter to Ludovico Sforza Duke of Milan (Figure 1.1) which is one of the oldest resume known to us. From past civilizations to modern-day corporations, the ability to identify, attract, and select talented individuals has been a persistent challenge and a strategic imperative for businesses across various industries. The importance of recruitment lies in its direct impact on organizational performance, productivity, and ultimately, the achievement of business goals.

In the past, recruitment methods relied heavily on personal networks, word-of-mouth referrals, and localized advertising to attract potential candidates. "The Office of Adresses and Encounters", founded in 1650 [1], is supposedly the first recruitment agency and advertised itself in the newspaper. These approaches often limited the pool of applicants, resulting in narrow talent pipelines and overlooked individuals who were not within immediate reach. As the industrial revolution unfolded, the expansion of industries and the emergence of larger corporations necessitated more systematic recruitment practices.

1. https://quod.lib.umich.edu/e/eebo2/A91885.0001.001

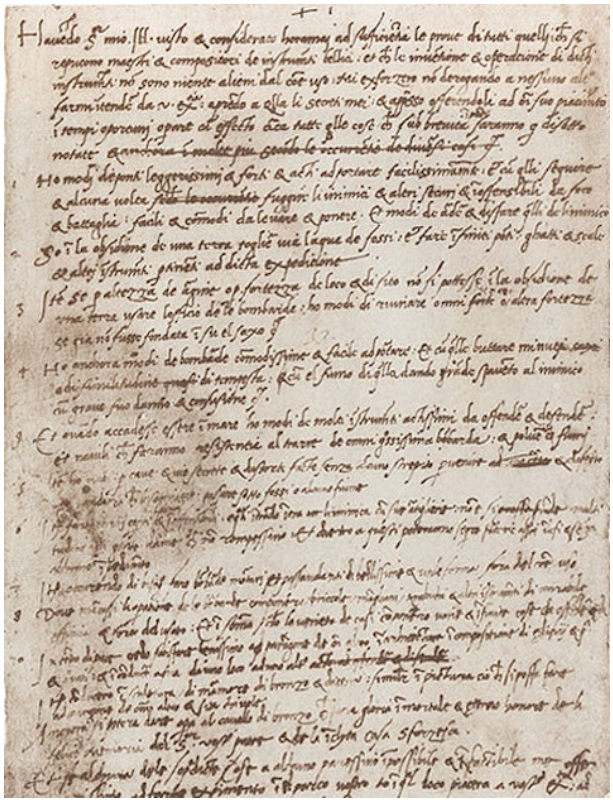


FIGURE 1.1 – Leonardo Da Vinci "resume" letter to Ludovico Sforza Duke of Milan (Source : openculture.com), its translation can be read in Appendix A.1

Another precursor among recruitment agencies is "Gabbitas" founded in 1873 [2] [3] which focused on the field of education. Formalized job advertisements, newspaper listings, and employment agencies became increasingly prevalent during World War II and its aftermath. First, they aimed to replace workers who had departed to serve in the military, and later they focused on meeting the employment needs of war veterans transitioning into civilian life. At the time, recruitment agencies worked mainly for job seekers, but over time switched to working for companies, enabling organizations to cast a wider net and reach a broader range of candidates.

With the advent of the digital era, recruitment practices have undergone a profound transformation, fueled by technological advancements and the widespread adoption of the Internet. Online job boards (for example The Monster Board created in 1994 merged

2. `https://gabbitas.com/heritage/`
3. `https://www.thegazette.co.uk/London/issue/27664/page/2218`

with Online Career Centre in 1999 to become monster.com [4]), social media platforms, and professional networking sites have dramatically expanded the reach and accessibility of job opportunities (for example LinkedIn which launched in 2003 and currently counts more than 850 millions user accounts [5]), breaking down geographical barriers and enabling organizations to tap into a global talent pool. The ease of disseminating job postings and the ability to collect and analyze vast amounts of applicant data have transformed the recruitment landscape.

However, despite these advancements, recruitment remains a complex and challenging endeavor for companies of all sizes. The scale of the problem has intensified in recent times because the advent of digital platforms has led to an unprecedented volume of job applications (for example Google supposedly received fifty thousand applications per week [6] or three million per year [7]). Organizations now receive a staggering number of resumes and applications for each open position. Sorting through this large pool of applicants to identify the most suitable individuals demands efficient and accurate screening mechanisms to avoid missing out on potential talents while filtering out irrelevant or unqualified candidates.

Furthermore, the ever-evolving nature of technology, industry, and society has led to a continuous influx of new skills and competencies. As innovations emerge, market demands shift, and industries transform, a multitude of novel skills and knowledge areas are being introduced at an unprecedented pace. This rapid proliferation has resulted in an overwhelming number of available skills in today's job market and presents a challenge for both job seekers and employers. Job seekers must navigate a dynamic landscape, continuously updating their skill sets to remain competitive and relevant in the job market. Conversely, employers face the daunting task of identifying the specific skills needed for each role and evaluating candidates' proficiency in those skills.

To address these challenges, continuous learning has become a cornerstone of career development, with individuals proactively seeking opportunities to acquire new skills, stay updated with industry trends, and adapt to the changing demands of the job market. Online learning platforms, professional development programs, and industry certifications

4. https://bebusinessed.com/history/history-online-job-search
5. https://about.linkedin.com/
6. https://dazeinfo.com/2022/11/04/google-rejects-resume-job-seekers-mistakes
7. https://www.cnbc.com/2019/04/17/heres-how-many-google-job-interviews-it-takes-to-hire-a-goog
html

have emerged as valuable resources for individuals striving to enhance their skill portfolios.

For employers, there is a pressing need for innovative solutions that can enhance the efficiency, accuracy, and fairness of the recruitment process. Job recommender systems, driven by advanced data analytics and machine learning techniques, have emerged as promising tools to address these complexities. By leveraging the power of algorithms and intelligent decision-making, these systems aim to provide personalized, data-driven recommendations that optimize the candidate selection process.

The significance of addressing the recruitment problem in today's context cannot be overstated. Organizations that effectively leverage modern recruitment technologies and methodologies gain a competitive edge in attracting and retaining top talent. Moreover, by streamlining the recruitment process, reducing biases, and improving the overall candidate experience, organizations can enhance their employer brand and establish themselves as employers of choice.

Through this thesis, we investigate the performance and feasibility of employing data-driven techniques, specifically ontologies and embeddings, to assist employers in conducting recruitment in a real-life context, using the company ALTEN as a case study. Additionally, we explore the impact of providing users with explanations from these intelligent systems on the recruitment process, particularly in terms of efficiency and trust. The objective is not to replace recruiters with automated pipelines, but rather to support them in managing the increasingly overwhelming task of candidate screening, which has become so complex in recent times.

1.2 Thesis industrial context : ALTEN

ALTEN is a consulting company that was established in 1988 and currently operates in 30 countries, employing over 54,100 individuals. With a financial turnover of 3.72 billion euros in 2022, the company specializes in providing engineering professionals to support various client projects. ALTEN manages a diverse pool of highly skilled talents who are contracted for indefinite periods but assigned to projects with finite duration, often requiring specific skill sets.

The company's clients encompass major players in sectors such as Aeronautics & Space, Defence & Naval, Security, Automotive, Rail, Energy, Life Sciences, Finance, Retail, Telecommunications, and Services. This broad client base ensures a consistent demand for consultants, requiring ALTEN to match the right professionals to specific missions. The hiring process involves both external recruitment to attract engineers with profiles that appeal to customers or match existing missions, and internal recruitment that assign new missions to available consultants. This strategic approach to talent acquisition is a fundamental aspect of the company's business model.

The motivation behind initiating this thesis was to explore the potential role of Artificial Intelligence in enhancing human resources management at ALTEN. Specifically, the focus was on developing an explainable AI-based tool to aid in the recruitment of engineers based on project needs. The research centered around one of the company's internal recruitment process called "mercato", which involves assigning available consultants to internal projects. This process served as an ideal setting for in-vivo experimentation, enabling the gathering of valuable data and the production of qualitative and quantitative results.

By delving into the integration of AI into human resources management, this thesis aims to provide insights that can enhance ALTEN's recruitment processes, improve decision-making, and ultimately contribute to the company's overall success.

1.3 Research questions

This thesis seeks to address the challenges faced by recruiters during the screening process by developing an explainable job recommender system that can assist them. The goals are to evaluate the feasibility of such systems and highlight issues faced in a real-life scenario and assess the usefulness of the provided explanation. Research questions that guide this study are as follows :

Q1. How can data-driven techniques, specifically ontologies and text embeddings, be utilized to develop an effective job recommender system that supports recruiters during the screening process ?

This question focuses on the exploration of data-driven techniques, specifically ontolo-

gies and embedding methods, for the development of a job recommender system. By thoroughly examining the potential of these techniques, the objective is to enhance the accuracy and quality of job recommendations, thereby assisting recruiters in efficiently navigating through a vast pool of candidates during the screening process. The goal is to leverage the power of data-driven approaches to deliver reliable and relevant recommendations that optimize the recruitment workflow and enhance decision-making for recruiters.

Ontologies and embeddings represent two distinct approaches in representing knowledge within a job recommender system. Ontologies, being built by humans, offer an easily understandable and interpretable framework. They provide a structured representation of domain knowledge, explicitly defining relationships and hierarchies. This human-readable nature of ontologies allows for intuitive comprehension hence facilitating the production of explanation and collaboration between users and the machine. On the other hand, embeddings, which are learned automatically from data, offer a more nuanced and comprehensive representation of information. These high-dimensional vector representations capture intricate patterns and semantic relationships in a way that facilitates machine learning algorithms to effectively process and analyze data. However, embeddings are not inherently interpretable by humans, as the underlying patterns and relationships are embedded in the vector space. Despite the lack of direct human interpretability, embeddings excel in capturing complex information and facilitates accurate and efficient recommendations generation by the job recommender system.

Q2. What is the impact of providing explanations for the recommendations produced by the job recommender system on the efficiency of the screening process ?

This question focuses on evaluating how the presence of explanations affects the time required for recruiters to assess and make decisions about candidates. Introducing explanations into the screening process has the potential to enhance efficiency by providing recruiters with additional insights and context. These explanations can help recruiters understand the underlying factors, such as skills, qualifications, or experience, that contribute to a candidate's ranking or suitability for a specific position. By having this information readily available, recruiters may be able to expedite their evaluation process, saving valuable time in assessing each candidate individually.

On the other hand, it is important to consider the potential challenges that explanations

may introduce in terms of time efficiency. While explanations provide valuable context, they also require additional cognitive effort from recruiters to review and interpret the provided information. This cognitive load can potentially extend the time needed for recruiters to thoroughly analyze and make decisions based on the recommendations and accompanying explanations. Thus, understanding the potential trade-offs between explanation provision and time efficiency is crucial for optimizing the overall screening process.

Moreover, the impact of explanations on efficiency can be influenced by factors such as the clarity and comprehensibility of the explanations themselves. If the explanations are concise, well-structured, and aligned with the needs of recruiters, they have the potential to streamline the decision-making process and improve efficiency. Conversely, if the explanations are overly complex, vague, or difficult to understand, they may hinder the efficiency of recruiters, leading to confusion and unnecessary delays. Ultimately, the goal is to strike a balance between providing valuable insights to recruiters and ensuring that the screening process remains efficient, enabling recruiters to make informed decisions in a timely manner.

Q3. To what extent does the availability of explanations enhance recruiters' trust in the job recommender system ? How does the provision of explanations influence recruiters' perception of system accuracy and their confidence in the recommended candidates ?

With the exponential growth of intelligent systems and their widespread adoption across various domains, there has been a parallel increase in users' defiance and skepticism towards these systems. As intelligent systems become more prevalent in our daily lives, individuals are increasingly questioning their reliability, transparency, and potential biases. This growing defiance stems from concerns about the black-box nature of these systems, where decision-making processes often lack transparency and explanations, leading to a lack of trust in their outcomes.

Users, including recruiters in the context of job recommender systems, are becoming more discerning and demanding when it comes to understanding the inner workings of these intelligent systems. They seek transparency and explanations to comprehend why a particular recommendation is made or how certain outcomes are reached. Without clear and interpretable explanations, users may harbor suspicions about the system's motives, biases, or potential errors, further fueling their defiance.

Moreover, high-profile cases of algorithmic biases, privacy breaches, and ethical controversies have amplified users' concerns and fueled a sense of distrust towards intelligent systems. The consequences of relying on flawed or biased recommendations can be significant, leading to adverse effects on individuals' opportunities, fairness, and overall well-being. A shocking example of such case is the Dutch childcare benefits scandal [59, 63, 23] which resulted in thousands of low-income families being wrongly accused of fraud due to "foreign sounding names" and "dual nationality".

To address this growing defiance and in order to comply with the laws passed to prevent these issues (for example the General Data Protection Regulation (GDPR) in the European Union, and especially the recital 71 readable in Quote 1 which explicitly mention "e-recruiting practices"), it has become imperative for developers, researchers, and organizations to focus on the development of explainable intelligent systems. The provision of transparent and interpretable explanations helps to bridge the gap between the system's decision-making processes and users' expectations. By offering insights into the underlying factors, inputs, and reasoning behind recommendations, explainable systems can foster trust, alleviate skepticism, and enhance users' confidence in the system's outcomes.

> The data subject should have the right not to be subject to a decision, which may include a measure, evaluating personal aspects relating to him or her which is based solely on automated processing and which produces legal effects concerning him or her or similarly significantly affects him or her, such as automatic refusal of an online credit application or **e-recruiting practices** without any human intervention.

Quote 1 – General Data Protection Regulation (EU Law) first sentence of Recital 71 (link). "e-recruiting practices" are explicitly mentioned.

Q4. How can the explainable job recommender system be further optimized and customized to accommodate the specific needs and preferences of recruiter ? How can the system gather user feedback on the quality of the recommendations it produces, and leverage those feedback to improve its performances ?

This question aims to identify opportunities for optimizing and customizing the job recommender system to align with the specific needs and preferences of recruiters. This can be achieved through user research, surveys, and interviews to gather insights into the desired features, functionalities, and decision-making criteria that recruiters consider

important. By incorporating these insights into the system design, it becomes possible to align the recommendations with the unique context and priorities of each recruiter, enhancing their satisfaction and adoption of the system.

In addition, the system can actively gather user feedback on the quality of the recommendations it produces. This can be accomplished through feedback mechanisms integrated within the user interface, such as rating scales, comments sections, or structured feedback forms. By capturing user impressions, satisfaction levels, and any perceived shortcomings of the recommendations, the system can gather valuable data for assessment and improvement. The explanations presence offers a great opportunity to gather such feedback organically.

Collected feedback can plays a crucial role in developing and improving the system. In the context of machine learning, it can be utilized to select pre-trained models and incorporated into the training and testing datasets, ensuring a tailored model that meets users' specific needs. The feedback can also be integrated into the recommendation algorithm and used at inference time, enabling immediate improvements to users' recommendations when they provide feedback.

1.4 Scientific Contributions

Throughout our research, the following scientific contributions to the field of job recommender systems have been made :

1 - Development of a Method for Resume Job Matching Dataset Creation

A novel method that enables the creation of a resume job matching dataset using a list of resumes as input have been developed and used successfully. This method leverage the resumes structure to fabricate job descriptions and match them together. A comprehensive dataset for training and evaluating job recommender systems has been built using ALTEN dataset which proved particularly helpful to our work. This contribution addresses the challenge of limited annotated data availability for job recommendation tasks, is presented in Chapter 3, and has been published [36] at the KaRS workshop of the renowned RecSys conference as part of our article on ontology-based conversational job recommendation.

2 - Ontological Classifier for the ESCO [8] Ontology using BERT

We proposed and implemented an ontological classifier based on BERT (Bidirectional Encoder Representations from Transformers) for the ESCO (European Skills, Competences, Qualifications, and Occupations) ontology. This multilingual classifier leverage the tree-like structure of ESCO to efficiently use samples from children class to also train parent class and showcase great performances. This classifier is a corner stone of the conversational job recommender system that is the subject of our article [36] published at RecSys (KaRS workshop) and is presented in Chapter 4.

3 - Ontology-based Conversational Job Recommender System

The utilization of the ESCO ontology and the classifier we trained has enabled the development of a job recommender system. This system effectively matches skills extracted from resumes and job offers to ESCO skills, leveraging the hierarchical structure of the ontology to generate accurate and relevant recommendations. By relying on an ontology as a human-readable knowledge base, the system facilitates seamless interactions between humans and machines, resulting in a conversational recommender system. This interactive approach allows to fine-tune recommendations and enhances the overall user experience. The research work, encompassing the development of the system, its various building blocks, and the thorough evaluation conducted, was published [36] in the workshop on conversational job recommender systems at the prestigious RecSys conference. The RecSys conference is widely recognized as one of the most influential conferences in the field of job recommender systems. This research work is the topic of the Chapter 4.

4 - Development and in-vivo experimentation of two Explainable Job Recommender Systems

We have successfully developed and deployed two explainable job recommender systems : one based on keyword matching and the other based on embeddings. These systems have been integrated into a job recommendation platform we built for ALTEN and have been actively utilized in the weekly internal recruitment process of the innovation department for over a year. Through this extensive real-world utilization, valuable usage data was gathered and has been instrumental in evaluating and comparing both models and conducting experiments on the impact of recommendation explanations in terms of time efficiency and user perception. The findings from our research, including the methodo-

8. `https://esco.ec.europa.eu/`

logy employed and the results obtained, have been published [9] in a paper at the KES (Knowledge-Based and Intelligent Information & Engineering Systems) conference. These findings are the topic of Chapter 6.

Overall, this thesis presents significant scientific contributions in the areas of dataset creation, ontological classification, explainability in job recommender systems, and their evaluation in real-world scenarios. These contributions have been published in two prestigious conferences, highlighting their relevance and significance within the scientific community.

1.5 Industrial Contributions

Our research work has led to significant industrial contributions for ALTEN. Firstly, we trained a model specifically designed to classify ALTEN missions in an internal ontology. This model is based on our work on the ESCO classification model, with the addition of a method to use data annotated with an outdated version of the ontology. This model, its training, and the thorough evaluation of its different iterations is the topic of Chapter 5. This classification model has been integrated into an API, which is utilized in an application that assists these experts in accurately classifying thousands of missions on an annual basis. This development has streamlined the mission classification workflow, enabling efficient and accurate categorization. Following the success of this model, ALTEN already asked us to update it to a new version of the ontology once, and we are set to do that again by the end of 2023.

Furthermore, our work has resulted in the creation of a job recommender platform that has been actively used by ALTEN innovation project leaders for over a year and a half. This platform, which is the focal point of Chapter 6 and our publication at the KES conference, has proven its value and practicality. Its regular utilization by project leaders has provided valuable insights and feedback for further development. The platform is currently undergoing active enhancements and is poised to serve as a research tool for ongoing experimentation centered around the usage of AI-based techniques in enhancing recruitment processes. Additionally, plans are underway to expand its implementation to other countries and recruitment processes.

In summary, our research has not only contributed to academic advancements but has

9. Reference coming soon, the conference will occurs early September 2023

also made significant practical contributions to the industrial operations of ALTEN. The development of a mission classification model and API has improved efficiency in mission classification, while the job recommender platform has been successfully integrated to the recruitment process for ALTEN's innovation projects. These industrial contributions highlight the practicality and relevance of our research in addressing real-world challenges and enhancing operational processes within the company.

1.6 Document Structure

In this section presents the thesis structure, and is designed to provide a systematic and comprehensive exploration of our research. The following paragraphs outline the content of each chapter, highlighting the key areas of focus and the contributions they make to the overall thesis.

Chapter 2 explores the existing literature and research relevant to the thesis topic. It begins with an overview of Natural Language Processing (Section 2.1), the sub-section 2.1.1 focuses on the application of Deep Learning techniques to NLP and discusses various approaches and methods employed in the field. Next is Section 2.1.2 which presents the field of Semantic Textual Similarity that is about measuring similarity between texts at a semantic level. The chapter then shifts its focus to Recommender Systems (Section 2.2), with a particular emphasis on Job Recommender Systems (Section 2.2.1), the evaluation methodologies for Recommender Systems (Section 2.2.2), and finally an exploration of Explainable Recommendations (Section 2.2.3).

Chapter 3 centers around the resumes dataset used in the thesis, which is specific to ALTEN. The structured resume format utilized by ALTEN, known as Technical Documents, is presented and analyzed (Section 3.1). Additionally, our method to build a Job Recommendation Benchmark Dataset from those structured resumes alone without job description nor manual annotations is described in Section 3.2.

Chapter 4 introduces the ESCO (European Skills, Competences, Qualifications, and Occupations) ontology in Section 4.1 as the foundation for developing a Conversational Recommender System. The structure of ESCO is explained (Section 4.2), encompassing ESCO Occupations and ESCO Skills/Competences. The chapter then presents our ESCO classifiers (Section 4.3), including occupation and skill classifiers. The ESCO Recommen-

der System is detailed next (Section 4.4), including a Full Deep Learning approach for comparison (Section 4.4.1) followed by a thorough evaluation of the system (Section 4.4.2), and finally the incorporation of conversational/interactive elements into the system are discussed (Section 4.4.3). Additionally, ESCO Explorer, a tool for exploring ESCO data, is presented (Section 4.5). The chapter concludes with a summary of the findings and limitations of the system (Section 4.6).

Chapter 5 focuses on the classification of missions within the context of ALTEN. The data used for mission classification is described (Section 5.1), followed by an initial model and its results (Section 5.2). Then a more advanced multi-year model is presented (Section 5.3), it includes its development and evaluation. Finally, the chapter concludes with a summary of the results and implications (Section 5.4).

Chapter 6 delves into the AI4HR Recruiter, a recommendation platform developed as part of the thesis. First the business process "Mercato", which motivates the creation of the platform and provide an opportunity for conducting a case study, is explained in Section 6.1. Then the various components of the recommendation platform, including the structured resume, preferences acquisition, recommendation/search algorithms, explanation of recommendations, and bookmarks, are discussed in detail in Section 6.2. Followed by the presentation of the experiments conducted with the platform (Section 6.3), including an analysis of users' preferences (Section 6.3.1), an evaluation of the time spent on the platform (Section 6.3.2), an assessment of the business process improvements (Section 6.3.3), and finally the protocol and results of conducted cross-over trials are presented (Section 6.3.4). The chapter concludes with a summary of the findings permitted by the AI4HR Recruiter platform and its impact on the recruitment process (Section 6.4).

Chapter 7 concludes the thesis. Firstly, Section 7.1 review the answers to our research questions and the contributions we made to arrive at those answers. Then, in Section 7.2 we reflect on our research methodology. Next, in Section 7.3 we discuss future research endeavors. Finally, in Section 7.4 we summarize our conclusions.

Chapitre 2

Related Work

This chapter presents and discusses publications that are related to our topic, either because they directly deal with issues similar to ours, or because some aspects they address are of use in the proceedings of our work. The nature of the application area, namely "Hiring", means that most data at our disposal are candidate resumes, and job offers/descriptions. Most of these data come in the form of raw materials written in large part in Natural Language, may it be in a structured format, that is to say, data that come in the form of fields containing natural text or standardized values, or in the form of completely unstructured text in which the data is a corpus of unannotated text. This means that Natural Language Processing is a major constituent of our work.

As not every resume contains all of its information in a textual form. More artistic jobs often rely on creative designs to stand out. An example of such a resume can be seen in figure 2.1. A Natural Language Processing model would not be able to capture all of the appeals of this resume. In that case, a computer vision model could be used in addition to the natural language processing model. But since we work on engineering profiles and the resumes we use follow a semi-structured format without any graphical element, we decided to disregard this issue entirely.

A significant part of the Hiring process is the act of choosing a candidate to fit a job offer. As we will see in the course of the work, the hiring decision often involves a compromise between the choosing of different candidates that each fit different parts of the job requirements. For example, one candidate fits 75% of the job requirements, whereas another candidate satisfies the other 25%. Together they cover all the skills required to conduct the job described in the offer. Hiring is a choice between imperfect options, and the role of the Artificial Intelligence based systems we propose is not to replace the recruiter but to empower him. As such, an interaction between the recruiter and the system must be

FIGURE 2.1 – A creative resume that carry a lot of information about the candidate through its design. It is a good example as to why limiting a resume to a text document can be reductive. Source : behance.net

initiated in order to allow the recruiter to perform the compromise. In the literature, the topic that matches the best this need for human-computer interaction for a task of decision-making is the field of Recommender Systems.

The first section of this chapter is dedicated to Natural Language Processing. It begins with a presentation of some of the most important Deep Learning models for the field and finishes on a specific task of NLP : semantic similarity. What we present in this section is used extensively in Job Recommender Systems and our work.

The second section is about Recommender System. It begins with a presentation of the field before raising the specifics of job recommendation, then recommender systems evaluation, and finally, the explanation of recommendations.

2.1 Natural Language Processing

Natural Language Processing is a field of computer science dedicated to understanding Natural Language, i.e. the language used by humans to communicate. It is defined by Gobinda G. Chowdhury [8] as "an area of research and application that explores how computers can be used to understand and manipulate natural language text or speech to do useful things.".

The first part is dedicated to a brief history on Deep Learning applied to NLP, followed by the current state of the art, with a focus on the recent apparition of Transformers and the significant impact they have of the field. The second part is dedicated to semantic similarity. It consists in determining if two texts have a similar meaning, this is useful in a number of application, including job recommendation.

2.1.1 Deep Learning for NLP

Deep Learning refers to Neural Networks that include multiple layers of non-linear projections [32]. They learn tasks using a variety of techniques, the most popular for supervised learning being Stochastic Gradient Descent through Back Propagation. Supervised Learning require huge datasets of examples to learn from. Deep Learning drastically improved the state of the art of numerous fields, like Computer Vision (e.g. image classification, object recognition, ...) or Natural Language Processing (e.g. text generation, text segmentation, ...).

Neural Networks take tensors as input, and finding appropriate methods to transform data into one can be challenging. A Feed-Forward Neural Network take a vector with a pre-defined length as input, however texts are of variable length (i.e. not all texts are composed of the same number of words/characters), and NLP researchers tried many ways to fit them into tensors. One of the first method is the Bag Of Word which is vector representing a vocabulary, each index of the vector corresponds to a word and takes as value the number of occurrences of the word in the text. The vector can be fed to a feed-forward network, but with a Bag Of Words any information about the position of words in the text is lost.

The state of the art rapidly moved toward Recurrent Neural Network. Words are encoded

in a one hot vector and fed to the network one after another. At each step the network takes as input the word associated to the step along with the output of the previous step. This kind of networks suffer from a vanishing gradient issue that make learning difficult. It is drastically improved upon by Long Short Term Memory Networks (LSTM) [25] which includes multiple gates (update gate, output gate, and a forget gate) and pass a state vector along with the output vector to the next step. Figure 2.2 shows a schematic of an LSTM chain.

At each timestep t (a timestep correspond to one token, a token can be a word, part of a word, or a letter) the LSTM take as input the representation X_t of the current token (it can be an embedding or a one hot vector) along with the output h_{t-1} and the state vector from the previous timestep. h_{t-1} and X_t are concatenated and first used in a forget gate that will pass them through a neural network layer with a sigmoid output function. The output of this layer is a vector of values between 0 and 1 that is the same size as the state vector and is multiplied with it effectively "forgetting" part of it. Next is the update gate, the result of the multiplication between a sigmoid layer and a tanh layer is added to the state vector. And finally, there is the output gate that apply a last transformation to the state vector before outputting it.

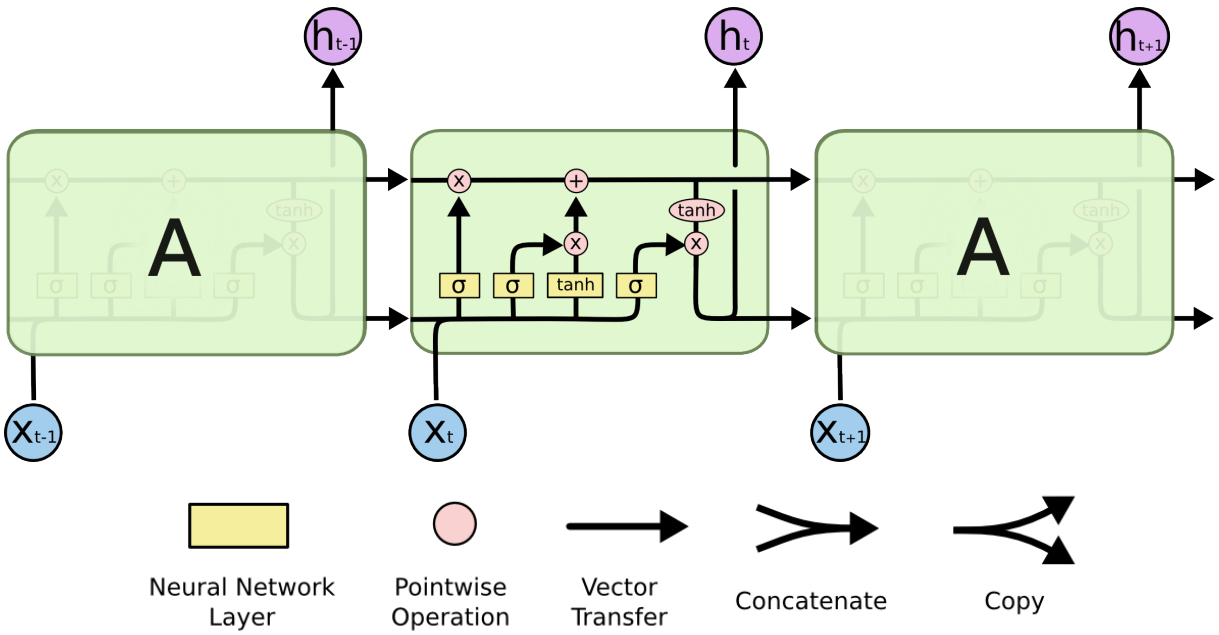


FIGURE 2.2 – An LSTM chain with the detail of an LSTM unit [1].

Cho et al. [6] propose Gated Recurrent Unit (GRU) a recurrent unit similar to LSTM but simpler. They remove the output gate and only the output vector is passed to the next step. There is no additional state vector. GRU have fewer parameters per unit yet exhibit performances similar to those of LSTM [9]. Both LSTM and GRU are significantly better than traditional RNN like tanh units.

Pre-training is a popular way to improve models' performances [12]. The most common pre-training task for NLP is Language Modelling. It consists in predicting the next word in a text given the previous words. It can easily be done with any corpus of text and does not require annotation.

Bidirectional recurrent neural networks are an important improvement over traditional RNN [53]. One RNN process the data forward and a second process it backward. The output of both networks is then combined using a variety of techniques, the most popular of which is a third network. Bidirectional LSTM also are a significant improvement over unidirectional LSTM [19].

Attention mechanism is also a substantial improvement to NLP models [7]. This mechanism attempt to mimic cognitive attention by weighting all the parts of the data hence drawing the focus of the model toward those that are the most important. It is commonly used with RNN to weight the output of each timestep.

Training very Deep Neural Network is a challenge because of gradient vanishing. It is overcome by using residual connections [22]. Those are connections that skip over multiple layers and allow the training of models with hundreds of layers.

Currently, the neural network architecture that achieve the best results in modern NLP is the Transformer [61]. It entirely ditch the recurrence which has a variety of unresolved issues like vanishing gradient, difficulties to learn long term relations and poor parallelization capabilities (each timestep is dependant of the previous one).

Figure 2.3 shows the architecture of the Transformer. The first version published by Vaswani et al. [61] showcases an encoder and a decoder because it was intended to perform translation, but for classification tasks only the encoder is required. It first start with an embedding layer, each token is mapped to an embedding that is usually learned during training. Then we find the positional encoding, it preserve the information of the position of each token in the text by summing tokens embedding with an embedding of the position.

The positional embedding can be computed with cosinus and sinus functions and fixed, or learned during training. Fixed embedding is usually preferred because it allows the model to process texts longer than those encountered during training. This positional embedding is what allows to ditch the recurrence. Next is the Multi-Head attention, it performs multiple Scaled Dot-Product Attention that can be computed is parallel.

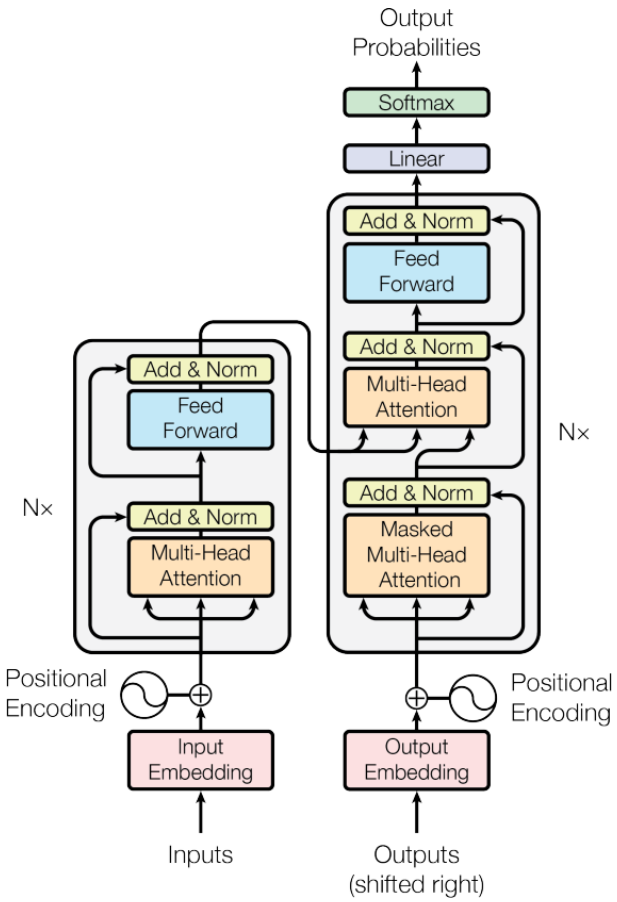


FIGURE 2.3 – The architecture of the Transformer [61]

"Attention is all you need" [61] (the publication that introduced Transformers) features a encoder/decoder structure due to the goal of the model being translation, but most transformer models only use the encoder part of the original model. One of the core feature of the encoder is the bidirectionality (the decoder masks future positions), unfortunately for pre-training, Language Modelling aims to predict the next word hence cannot be bidirectional. Devlin et al. [10] propose to solve this with masked language modelling. Instead of predicting the next word, some words are masked (i.e. hidden) and the model

must predict them. This allow to leverage the model bidirectionality during pre-training which improve the model accuracy. Not every word tagged to be hidden are masked, most of them are, but some of them are instead changed by a random word, or are kept the same. This is done to bias the model toward the word at this position. During pre-training, BERT models also train to predict if two sentences are following each others. The input start with a special classification token (the prediction will be made at this position) followed by both sentences, each of them ended by a special token.

With the transformers architecture and BERT pre-training, we have a robust model that is ready to be trained which leads us to the following innovation we want to present : MT-DNN [34]. This model has the particularity to be trained on multiple tasks at the same time. Each mini-batch contains elements from different tasks. The model has a specific output layer for each task, and use an appropriate loss function for each of them the result of which is then aggregated. This particular training allowed MT-DNN to become the first model to beat the human baseline on the GLUE benchmark [62] and showcase the potential of multitask models.

Multitasking is not the only way to improve models performances, another way is just to aim bigger. This is demonstrated by openai and their model GPT-2 [44] with its 1.5 billions parameters. It, not only achieve state of the art performances in language modelling, but it also manage to perform other tasks it was not trained for, like translation or question answering. This is done by providing the goal of the task as an initial input and letting the model generate text afterwards. For example, a user can provide the input "The cat ate the mouse, which translate in french by", and the model will most likely propose a french translation of "Le chat a mangé la sourie". It is possible because the dataset it was trained on contains such sentences. This dataset is named WebText and contains 8 millions document for a total of 80 GB of text. The even more impressive GPT-3 [3] with its 175 billions parameters confirmed those findings.

In transformers, tokens are embedded using an embedding layer that is learned along the model. They do not use Word2Vec or any other techniques that yield fixed embeddings. This is the current trend for word embedding [28]. Model learned embedding are also where the state of the art of text embedding is, the most popular embedding model being Sentence BERT [48]. In the following section we present various embedding techniques that are used for Semantic Textual Similarity.

2.1.2 Semantic Textual Similarity

In this section, we present a history of Semantic Textual Similarity, and how the Transformers we introduced in the previous section showcase state-of-the-art performances in text embedding and are used for STS.

Semantic Textual Similarity (STS) is a growing area of research that aims to measure the relatedness between two pieces of text (pieces of text refer to any written material, such as paragraphs, sentences, phrases, words, and even individual characters). It has become increasingly important in the context of natural language processing and information retrieval, due to its potential in providing meaningful insights into the relationships between different documents and topics. Given two pieces of text a STS model will assesses the degree to which they are semantically equivalent to each other [4]. This assessment usually takes the form of a numeric value.

The SemEval workshop [4], that focus on Semantic Evaluation, often features semantic similarity tasks. A popular technique is text embedding. It consists in representing text under the form of a real-valued vector. Vectors produced by the same model are always of the same size even not matter the length of the text they represent. This allows to use mathematics on texts, and to input this text to a variety of models. Text embedding can be word level embedding, sentence level, or embed an entire document.

We can determine the Semantic Similarity of two texts by computing the cosine similarity of their respective embeddings. This is possible because embeddings are produced in ways that capture semantic information. Semantic Similarity models are usually evaluated by computing their correlation with a test set. Such test sets take the form of a list of pairs of text associated with their semantic similarity as perceived by humans annotators. Pearson and Spearman are the most common correlation measures for this purpose.

However, Reimers et al. [45] found out that Person and Spearman correlations measures are poor indicators of embedding models performance in STS related tasks. Actually, there is a negative correlation between the results of the intrinsic evaluation of models with those measures compared to their performance on the tasks.

Word2Vec by Mikolov et al. [39] is the first word embedding model to gain widespread popularity. It relies on 2 log linear models :

1. CBOW (Continuous Bag Of Words) : attempts to predict a word from its neighbors.

2. Skip-Gram : attempts to predict the neighbors of a word.

Continuous Bag Of Words (Figure 2.4) represents the word's neighbors as one-hot vectors the size of the vocabulary. Those vectors are projected into vectors the size of the embedding using the same projection matrix. The projections are averaged and the model attempts to predict the word from this average. The projection matrix and the prediction layer are trained as a regular Neural Network. It is named Bag Of Words, because the words' position do not influence the projection. Skip-gram does the opposite, and attempts to predict the neighbors of the word given as input.

Once the models are trained, the projection layer can be used to produce the embedding for each words. We can use only CBOW or Skip-gram, or concatenate the embeddings from both.

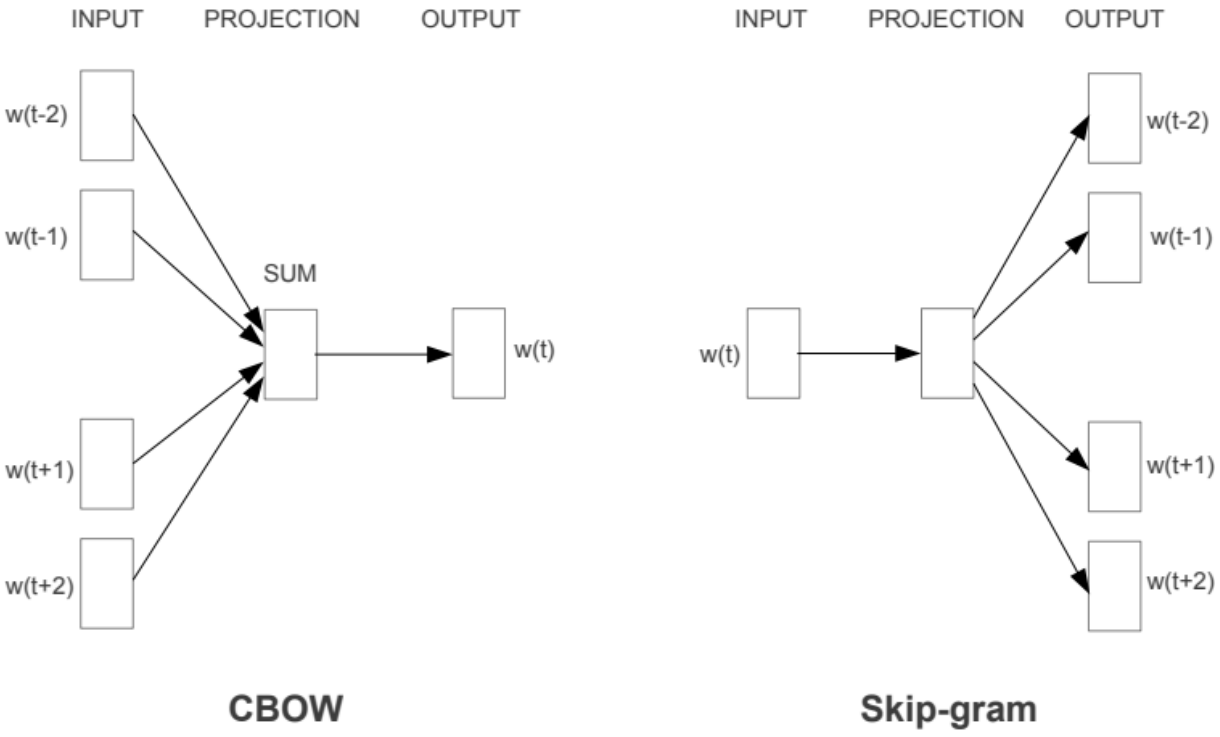


FIGURE 2.4 – Show both word2vec models structure[39].

GloVe [42] is another model that also became very popular shortly after Word2Vec and competed directly with it. It is a log-bilinear model trained on words co-occurrence probability.

This trend of representing/learning the meaning of a word based on its neighbors comes

from the field of Linguistic Theory, and was summarized by J.R. Firth [14] as "you shall know a word by the company it keeps".

Beyond word embedding is text embedding. Being able to compare words is important, but the meaning of a word depends mostly on its context. For example there is Paragraph Vector [31] which improve upon Bag of Words model. More recently, embeddings are learned by Deep Neural Networks[28], the most popular models being Sentence Transformers which are derived from Sentence BERT [47, 46]. At the time of SBERT publication state of the art performances in Semantic Textual Similarity was achieved by BERT models which take both text as input. It is computationally expensive as finding the most similar pairs in a dataset of 10000 sentences would take 65h on a modern GPU. It makes those models unpractical in operational tools. Reimers et al. propose to resolve this issue by having the Deep Neural Network model produce the embedding for one sentence and then use metrics (e.g. cosine similarity) to compare embeddings with each other. Unfortunately pooling embeddings from BERT achieve results worse than averaging GloVE embeddings. So they come up with 3 different ways to train the model :

Classification Objective Function. The embeddings produced by the model are concatenated together along with their difference and a prediction is made :

$$o = \text{softmax}(W_t(u, v, |u - v|)) \tag{2.1}$$

They use cross-entropy loss function to optimize.

Regression Objective Function. They compute the cosine similarity of the embeddings and optimize with mean-squared-error.

Triplet Objective Function. Given a triplet of sentence a, p, and n for which a is an anchor, p is a positive sentence, and n a negative one. They optimize the model so that the distance between a and p is lower than the distance between a and n :

$$\max(||s_a - s_p|| - ||s_a - s_n|| + \epsilon, 0) \tag{2.2}$$

s_x is the embedding of the sentence x, $||\cdot||$ a distance metric (euclidean distance in their experiments), and ϵ the minimum distance that we want ($\epsilon = 1$ in their experiments). The objective function used depends on the dataset. The first two functions require a Siamese network (weights are shared), while the third requires a triplet networks. Figure 2.5 shows

the architecture of SBERT for training with the Classification Objective Function, and figure 2.6 shows its architecture for inference.

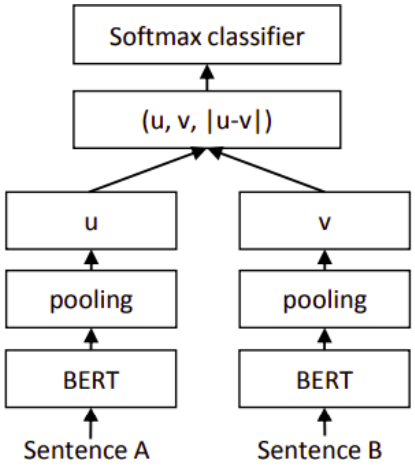


FIGURE 2.5 – SBERT architecture for training with the Classification Objective Function (source [47]).

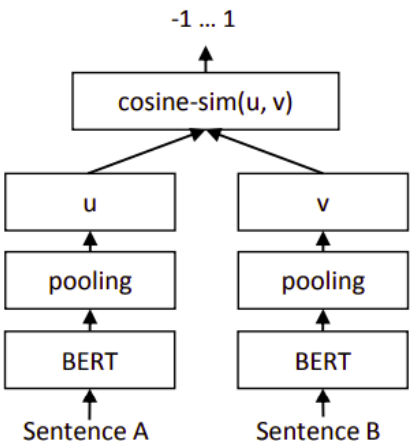


FIGURE 2.6 – SBERT architecture for inference (source [47]).

They also experiment with three different pooling methods to retrieve the sentence embedding from the model :

- CLS pooling : they only consider the output vector for the CLS token.
- Mean pooling : they compute the mean of every output vectors.
- Max pooling : for each element of the vector they keep the maximum from every output vectors.

There experiments demonstrate that mean provide the best results.

2.2 Recommender Systems

Recommender systems are a type of technology that helps users discover new items or services that may be of interest to them. These systems use algorithms to analyze data about the user and their interactions with the system, such as their previous purchases or ratings, in order to generate personalized recommendations. Recommender systems are commonly used in a variety of fields, including online retail, music and video streaming, and social media. They can improve user experience and increase customer satisfaction by providing relevant and tailored recommendations, as well as reduce the overload of information and choices that users face. However, the effectiveness of recommender systems

is dependent on the quality of the data and the algorithms used, as well as the user's perception and acceptance of the recommendations.

Resnick et al. [49] introduced the term "Recommender System" back in 1997, but these type of systems were in use much earlier. For example, Golberg et al. [18] present such a system using a Collaborative Filtering algorithm. This kind of algorithm is one of the most widely-used techniques to make recommendation services [1]. Aggarawal et al. [2] classify Recommender Systems in 4 categories :

- Collaborative Filtering : systems that rely on implicit and explicit feedbacks from users (e.g. clicks or rating) to find relevant items.
- Content-based : systems that analyze items attributes.
- Knowledge-based : systems that use a knowledge base, such as an ontology, for deriving the recommendation.
- Hybrid : any combination of the above categories.

Nowadays most internet platform include some form of Recommender Systems to provide their users with adequate content. Popular examples are Youtube, Amazon, Netflix, LinkedIn, etc.

Collaborative Filtering systems, despite their popularity and great performances, suffer from a major drawback, the cold start problem. This is when the system do not have enough interaction data on its users and products to make recommendations. Knowledge-based systems that rely on ontology have been proposed to tackle this issue [58, 38, 37]. Items and users are mapped to an ontology and the system use the relationships between the concepts to recommend items to users. An ontology, as defined by Nicola Guarino et al. [20], can be (sense 1) "a logical theory which gives an explicit, partial account of a conceptualization", or (sense 2) a synonym of conceptualization. Their definition of a conceptualization is : "an intensional semantic structure which encodes the implicit rules constraining the structure of a piece of reality". In the context of recommender systems, this "piece of reality" refers to the domain or subset of it from which the recommended items originate. For instance, in the case of a system recommending movies, an ontology could encompass movies and actors, along with the relationships between them, such as "actor X played in movie Y".

Content-based recommender systems rely on the content of the items to recommend them. In the case of a Job Recommender System it can look at the skills presents in resumes and job offers to match them together. This kind of algorithm is well suited for systems

that do not produce enough interactions to rely on Collaborative Filtering techniques. It can happen if the user base is too small, or do not interact with enough items, or if items do not stay available long enough to gather enough interactions.

New users can be a problem for Recommender Systems. Collaborative Filtering techniques fallback to recommending popular items, but content-based and knowledge based methods require to know some stuff about the user. This is why with many Recommender Systems, users are first met with a Preferences Acquisition phase. For example, you can see TikTok preferences acquisition in figure 2.7. Some systems (e.g. TikTok) propose to the user to pick his interests among a list, but Narducci et al. [41] show that letting the user freely chose his preferences can lead to better recommendations.

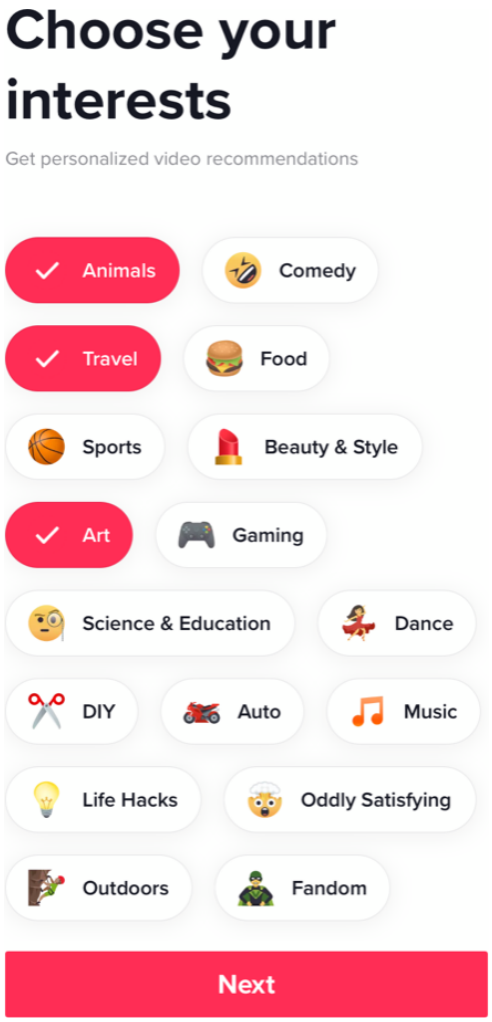


FIGURE 2.7 – The preferences acquisition screen for TikTok (source : uxdesign.cc).

The preference acquisition process can also be automatized. For example, the ontology-based recommender system proposed by Middleton et al. [38] looks at a user's publications to build a list of interests from the ontology. Because a human can understand the topics represented in the ontology, their system come with a the possibility for users to visualize and edit their profile.

2.2.1 Job Recommender Systems

Job recommender systems are a type of technology that helps companies identify potential job candidates who may be a good fit for open positions. These systems use algorithms to analyze data about the job requirements, the candidates' qualifications and experience, and other factors, in order to generate personalized recommendations. Job recommender systems are increasingly being used by companies as a tool for their recruitment process, as they can save time and effort by identifying qualified candidates who may not have been found through traditional methods, such as job postings or networking. Moreover, these systems can help companies achieve diversity and inclusion goals by providing a wider pool of candidates to consider, and can also improve the candidate experience by providing personalized job recommendations that align with their interests and skills. This introduction provides an overview of the role and importance of job recommender systems in the recruitment process, as well as the potential challenges and opportunities associated with their use.

Surveys

Freire et al. [15] surveys job recommendation techniques. They emphasize that these systems often rely on a variety of machine learning techniques and do not fit well into the usual recommendation system categories. They suggest to put many job recommender systems into an "Other Techniques" category. However, De Rujit et al. [50] are critical of this "Other Techniques" category and propose, in their own survey, to regroup this Other Techniques category with the Hybrid category, but divide it into Monolithic and Ensemble categories following the classification of Aggarawal et al. [2]. Monolithic are systems that do not include at least one subsystem which could produce recommendations on its own. Ensemble are systems that include at least one subsystem which could produce recommendations on its own.

Because recruitment is so crucial to company, we can find many industrial publications. For example, Lavi et al. [29] from Randstad (a company specialize in human resources management) propose to use embeddings produced by a Transformer to match candidates to vacancies. Unfortunately, because of the industrial nature of the publication, they do not include how they actually trained the model to produce the embeddings, but it showcase that documents' embeddings produced by transformers are precise enough to base industrial recommender systems on them. They also highlight some problems in the field like risk of discrimination, or the vocabulary gap between Resume and Job Offer.

Vocabulary Gap

To the best of our knowledge, Schmitt et al. [52] are the first to identify the vocabulary gap between job offers and resumes. You can see in figure 2.8 a visualisation of job offers and resumes in a LSA-space (Latent Semantic Analysis). Job offers are represented by circles and Resumes by stars. Separate clusters between circles and stars of the same color denote a vocabulary gap between job offers and resumes. This is the case for graphic designers and waiters.

This gap is mentioned in many publications but is rarely taken into account. Probably because a lot of Job Recommender System publications focus on engineering fields for which skills are clearly named hence there is no gap between offers and resume, e.g. offers will explicitly ask for skills like python which is named the same in resumes. This is the case for us because we only use resumes and small queries from project leaders that use the same vocabulary, so we disregarded this issue in our work. Yet it is an important issue to mention, and it should be taken into account when undertaking the task of building a more generalist job recommender system.

Example of Job Recommender Systems

Wenxing Hong et al. [26] propose a system that classify users and adapt its recommendation strategy based on the user group. Their groups refer to the job seeking strategy employed by users :

— Proactive : they make requests, and engage in various activities referred as active.
— Passive : they mostly look at position that are popular among other users.

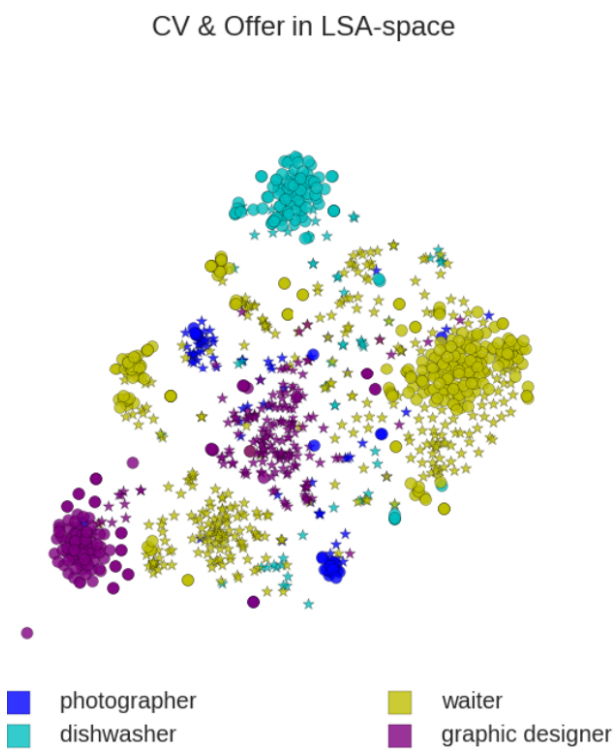


FIGURE 2.8 – Visualisation of vocabulary gap between job offers (circles) and resumes (stars) in a LSA-space (source [16]). Separate clusters between circles and stars of the same color denote a vocabulary gap between job offers and resumes. This is the case for graphic designers and waiters.

— Moderate : this is an in-between of the two others.

They use content-based strategy for proactive users, collaborative-filtering for passive users, and for moderate they use HyR which is a combination of collaborative filtering and knowledge-based. Because of our specific business process, we only want proactive users, so our system mostly rely on content based strategies.

Freire et al. [16] propose a framework for a Job Recommender System that range from preferences acquisition to the collection of feedbacks and their integration in the recommendation process. Their system is designed for both recruiters and candidates at the difference of ours that is focused solely on recruiters. But other than that, the architecture of their framework (Figure 2.9) and ours have a lot in common.

Gugnani et al. [21] propose a skill extractor for unstructured resumes and job descriptions along an interesting approach to data augmentation for the job market. They infer skills that do not appear explicitly in the job description but may be expected in the context

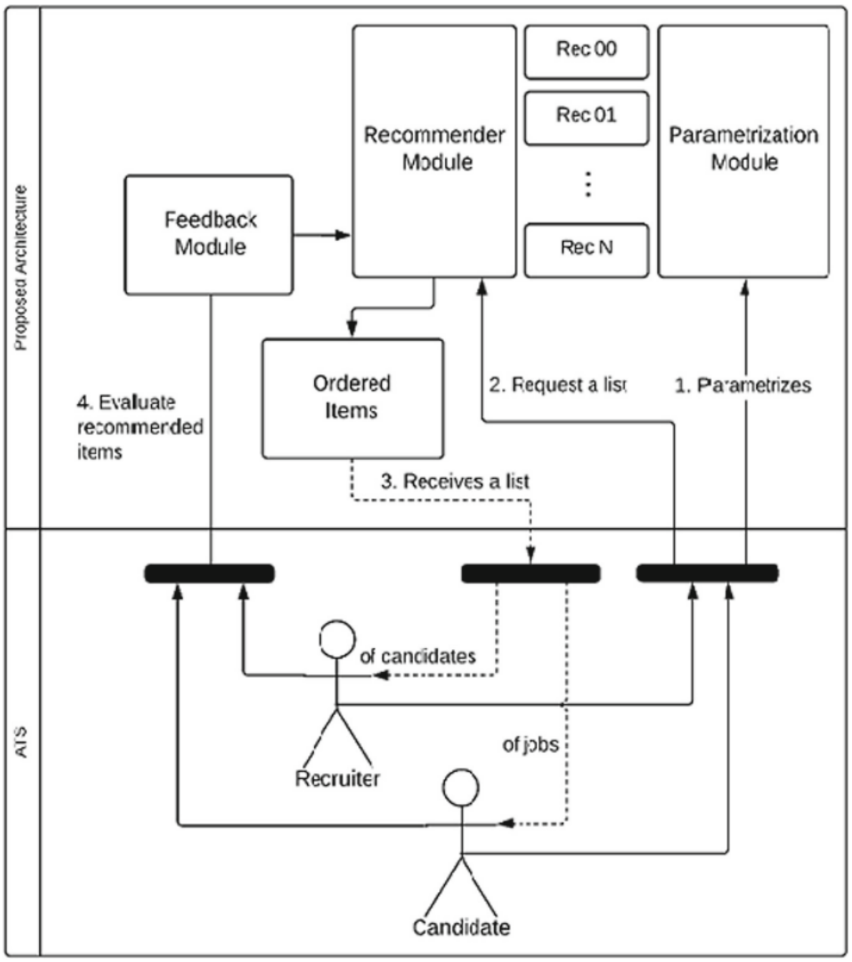


FIGURE 2.9 – The architecture of the Job Recommender System Framework proposed by Freire et al. [16]

of geographic location, industry, or position. They mine those implicit skills by looking at job description with a similar Doc2Vec embedding. The improvement on their job recommender system with implicit skills compared to without is significant : the accuracy goes from 0.68 (without) to 0.88 (with). This approach is particularly interesting to apply to skill-based JRS when working with documents that do not contains many skills.

2.2.2 Recommender Systems evaluation

A recommender system quality can be evaluated according to several criteria. First of all, there is the accuracy of the recommendations, their precision, and their recall. Then

there is the user experience and engagement. The evaluation of recommender systems is a complicated task. In this section, we present what we can find in the scientific literature about this topic.

Ge et al. [17] advocate that accuracy is not enough to evaluate Recommender Systems, and propose to also use coverage and serendipity. They define two types of coverage, prediction coverage and catalog coverage. Prediction coverage (equation 2.3) measures how much of the catalog can be recommended by the system.

$$\text{Prediction coverage} = \frac{|l_p|}{|l|} \tag{2.3}$$

l denotes the set of available items, and l_p the set of items for which the model can make a prediction. It can be used to evaluate how vulnerable a system is to the cold-start problem. Catalog coverage measures how much of the catalog is actually recommended by the system (equation 2.4). Catalog coverage is usually measured on a set of recommendation sessions (e.g. every session for a period of time).

$$\text{Catalog coverage} = \frac{|\bigcup_{j=1}^{N} l_L^j|}{|l|} \tag{2.4}$$

l_L^j denotes the set of items contained in the list L returned by the j^th recommendations, and N is the total number of recommendations. Catalog coverage is especially important when considering the job market. Because in our societies a job is pretty much mandatory to live, a system that would never recommend a person for any job could be considered unethical. Given a sufficient number of position and people a job recommender system should aim to have a catalog coverage equivalent to its prediction coverage. In other words, every person for whom the system can technically produce a recommendation should be recommended for at least one job/position.

Serendipity measures how novel interesting recommendations are. A model should make recommendations that are unexpected by the user yet interesting. Following the approach of [40], Ge et al. measure the unexpectedness of a recommendation as not being recommended by a primitive model that only produce expected recommendation. RS denotes the recommendations of the system we evaluate, and PM denotes the set of recommen-

dations made by a primitive model. The set of unexpected recommendations is noted $UNEXP$ and is calculated as follows :

$$\text{UNEXP} = \text{RS} \setminus \text{PM} \tag{2.5}$$

The usefulness of a recommendation r is noted $u(r)$. $u(r) = 1$ when the recommendation r is useful, and $u(r) = 0$ when it is useless. The serendipity is calculated as follows :

$$Serendipity = \frac{\sum_{r \in \text{UNEXP}} u(r)}{|\text{UNEXP}|} \tag{2.6}$$

About evaluating recommender systems, Pu et al. [43] propose ResQue a user-centric evaluation framework. Their framework consist in 31 questions categorized in 15 constructs. For example : the question "I feel in control of modifying my taste profile" belong to the construct "Control". The questions can be answered using a 5-point Likert scale ranging from "strongly disagree" to "strongly agree". They also suggest the use of a shorter version of their ResQue that ask only on question per construct. This is enough because the questions belonging to a same construct are highly correlated. Also, they discovered in their previous work that the subjective accuracy (i.e. how the the user perceive the model accuracy) is strongly correlated with the objective accuracy and has the advantage of being much easier to obtain [5].

2.2.3 Explainable Recommendations

Producing intuitive explanation of recommendations is a major challenge faced by recommender systems. There are two types of explainable recommendation models [64] :

— Model-intrinsic : consists in interpretable models which feature transparent decisions mechanisms that can be naturally explained.

— Model-agnostic : this approach allows "black box" decision mechanisms, and instead focus on the development of models which can generate an explanation afterwards. This method is also called post hoc.

As highlighted by Zhang et al. though the term "explainable recommendation" is recent (2014), the concept is not. Explanations of the type "we recommend this item because you liked a similar item" date back to at least 1999 [51] and are popular in e-commerce.

Figure 2.10 shows an example of such explanation on the video game marketplace Steam. The system recommend a game because the user recently played to a similar game. We can note that there is no explanation as to why both games are similar.

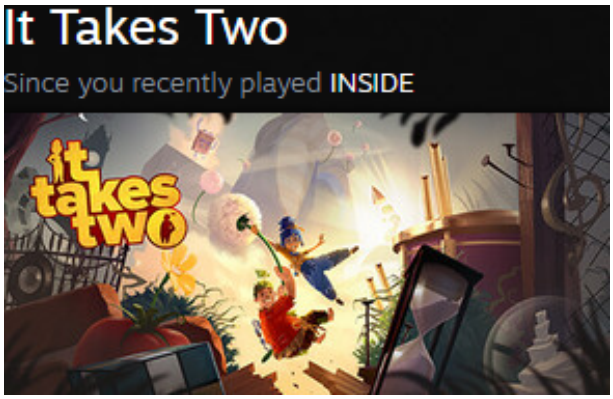


FIGURE 2.10 – An example of an explanation of recommendation on the video game marketplace Steam (source : Steam). The system recommend the video game "It Takes Two" because the user recently played to the video game "INSIDE" which is deemed similar.

The explanation of algorithm can pursue many goals. In the case of Recommender Systems, and a presented by Nava Tintarev et al. [60], the usage of explanation can aim at improving :

- Transparency : explain how the system works.
- Scrutability : allow users to tell the system it is wrong.
- Trustworthiness : increase users' confidence in the system.
- Effectiveness : help users make good decisions.
- Persuasiveness : convince users to try or buy.
- Efficiency : help users make decisions faster.
- Satisfaction : increase the ease of usability of enjoyment.

It is neither possible nor desirable to pursue all of these at once. The authors cite the example of [30] in which users preferred graphic explanations (satisfaction) but actually made better (effectiveness) and quicker (efficiency) choices with textual explanations.

Explanations can sometimes be detrimental. Ferwerda et al. [13] made an experiment on ProgramGenie, a TV program recommender system. The results are that the explanation had a negative on trust, choice satisfaction, and understandability. They also experimented with a content-based algorithm and a collaborative filtering one that is based on demographic. The CB algorithm had a more negative impact than the explanation. They

also noted that both the explanation and the CB algorithm had a positive impact on users with a high activity. Their hypothesis is that with higher activity come more interactions (rating of program) that the system can use to make better recommendations. They also encountered issues during their experiment with participants dropping off and the platform/website being slow and buggy.

Figure 2.11 shows an example on an explanation on ProgramGenie. The explanation taking the form of an histogram is motivated by Herlocker et al. works [24] on explaining Collaborative Filtering recommendations.

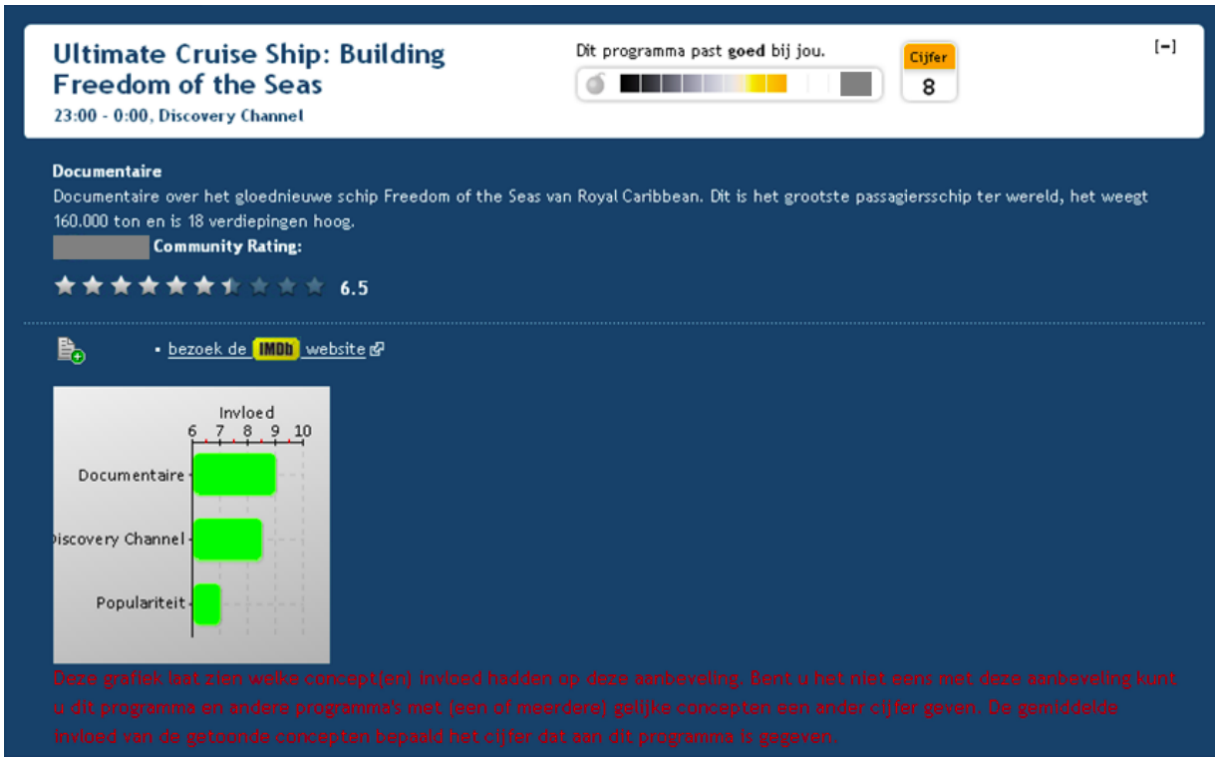


FIGURE 2.11 – An example of an explanation of recommendation on ProgramGenie. It takes the form of an histogram that shows which features contributed to the recommendation and how much. In this example the item "Ultimate Cruise Ship : Building Freedom of the Seas" is recommended mainly because in it a documentary and it airs on "Discovery Channel", but also because it is popular. Source [13].

2.2.4 Ontology-based Job Recommender Systems

By providing a structured representation of knowledge and relationships within a domain, ontologies offer a means to bridge the gap between machine-based recommendations and

human understanding. This section delves into the use of ontologies in the realm of job recommender systems. As seen in Section 2.2 and defined by Nicola Guarino et al. [20], an ontology can be any semantic structure that encodes the underlying rules governing the organization of a piece of reality.

Zheng Siting et al. [57] conducted a comprehensive survey on Job Recommender Systems, where they highlighted an ontology-based system called "Proactive" proposed by Danielle H. Lee et al. [33]. The Proactive system utilized two ontologies, one focusing on job categories and the other on company information such as industry classification and company size. However, the specific ontologies employed in the system remain ambiguous, as the authors only mention that they were based on Yahoo ! HotJobs, which unfortunately ceased operations in 2011. The paper mentions the utilization of ontologies to statically represent data, and it suggests that ontological relationships were instrumental in calculating weight values for each category during the training process. Regrettably, the publication lacks detailed explanations regarding how the ontologies effectively represent data, the specific computation formula for weight values, and the precise role played by relationships in this process.

Saman Shishehchi et al. [55] built a dedicated ontology for their Job Recommender System for Disabled People (JRDP). This system is specifically designed to assist individuals with disabilities in finding job positions suitable to their profiles, taking into account their specific disabilities. The ontology encompasses various attributes such as gender, age, and disability type, among others. The authors provide comprehensive insights into the construction process of their ontology, highlighting the utilized tools and methodologies in detail.

Patsakorn Singto et al. [56] developed an ontology specifically designed for semantic searching of IT careers. Their system demonstrated improved precision and recall compared to a keyword-based system when evaluated on a set of five queries, with precision featuring the highest improvement. The constructed ontology incorporates Computer Fields Knowledge sourced from ACM/IEEE-CS as skills and education information, while leveraging the International Standard Classification of Occupations [2] (ISCO) for categorizing careers. Notably, the ontology incorporates relationships such as "is an operating system" or "is a programming language". To our knowledge, the ontology proposed by the authors has not been publicly released.

2. `https://www.ilo.org/public/english/bureau/stat/isco/`

Mihaela-Irina Enăchescu [11] developed an e-Recruitment ontology for a prototype platform, distinguishing it from other ontologies by focusing on platform users rather than skills or occupations. This ontology encompasses users' educational backgrounds, work experiences, and even practical details such as email addresses and passwords. Notably, unlike traditional ontologies, it does not establish relationships between skills. Instead, skills are treated as properties of work experiences. The author opted not to assign knowledge levels to skills and justified this decision by highlighting the potential for employees to overstate their competencies. Instead, they suggested using the job position as a determinant. For instance, a senior developer would be expected to possess greater expertise in the programming languages used in their work experiences compared to a junior developer.

From these works, we observe that various approaches have been employed in utilizing ontologies in job recommender systems, with no two studies utilizing the same ontology. Several factors could account for this disparity, including the unavailability of certain ontologies, authors choosing not to make their ontologies public, or the inclusion of sensitive user information within the ontologies.

2.3 Research questions refinement

In this section we propose to revisit our research questions to refine them using what our related work taught us.

Q1. How can data-driven techniques, specifically ontologies and text embeddings, be utilized to develop an effective job recommender system that supports recruiters during the screening process ?

In Section 2.2.4, we have explored several works that employed ontologies for job recommender systems, but unfortunately, either these ontologies are no longer available or they have not been made publicly accessible to our knowledge. To construct a knowledge-based job recommender system, it is desirable to utilize an ontology that comprehensively describes the labor market, encompassing a wide range of skills, occupations, and education, along with their relationships. Fortunately, such an ontology exists : ESCO, developed by the European Union, which will be presented in Section 4.1. Our objective is to leverage this ontology to create a job recommender system capable of utilizing ESCO concepts and relationships to generate and explain recommendations, specifically tailored for ALTEN's

non-annotated data (ALTEN skills are not ESCO skills, so a mapping is necessary). Since ALTEN's field of activities encompasses various engineering domains, not limited to IT, and demands high precision in terms of available skills and their relationships, potential challenges may arise. If ESCO proves unsuitable for our context, we will explore the alternative of using text embeddings, which could mitigate the challenges but may reduce the explainability of the recommendation process.

In light of these considerations, our refined research question is as follows : **Q1. How can we effectively utilize the ESCO ontology or text embeddings to develop an Explainable Job Recommender System that is well-suited for a context with a wide array of highly specialized skills ?**

Q2. What is the impact of providing explanations for the recommendations produced by the job recommender system on the efficiency of the screening process ?

To address this question, we require a recommendation platform with real users, where we can measure the time taken by users to make hiring decisions based on the presence of an explanation. To accomplish this, the platform needs to record the duration between the opening of a recommendation and the moment the user reaches a decision. Such an evaluation framework can also be utilized to assess various formats of explanations, allowing us to compare their effectiveness and impact on decision-making.

Q3. To what extent does the availability of explanations enhance recruiters' trust in the job recommender system ? How does the provision of explanations influence recruiters' perception of system accuracy and their confidence in the recommended candidates ?

This question pertains to the users' perception of the recommender system. During the review of related work, we came across ResQue [43], a framework specifically designed to evaluate the overall user experience in a recommender system, encompassing aspects such as trust and confidence. Considering the existence of this framework, we can expand the question to : **Q3. What is the impact of the presence of explanations on the quality of the user experience ?**

Q4. How can the explainable job recommender system be further optimized and customized to accommodate the specific needs and preferences of

recruiter ? How can the system gather user feedback on the quality of the recommendations it produces, and leverage this feedback to improve its performances ?

In our related research, we discovered that explanations offer an opportunity for users to provide feedback on the recommendation process. This feedback can be more precise than simply indicating whether the recommendation is good or not ; it can actually pinpoint which aspect of the system's "reasoning" is correct or incorrect. For instance, the system may incorrectly associate two skills, but still provide a satisfactory recommendation due to the relevance of other skills to the job offer. Our objective is to determine how a job recommender system can effectively utilize explanations to gather more precise user feedback and leverage this feedback to enhance the system. This leads us to the following question : **Q4. How can a job recommender system practically utilize explanations to collect more precise user feedback, and how can this feedback be leveraged to effectively improve the system ? And to what extent did users engage with the feedback system, and how significant was the feedback in improving the system ?**

Our research and the remainder of this document are focused on answering the questions outlined above.

Chapitre 3

The resumes dataset of ALTEN

This chapter introduces the raw resumes dataset that was made available to us by ALTEN and how it was used to build a new training dataset suitable to our needs. This raw resumes dataset is presented in Section 3.1 ; it only contains a structured resume collection and is not directly suitable to our needs. This is why a protocol to build a training dataset from this raw dataset is presented in Section 3.2. This new dataset matches resumes on the one hand, and job offers on the other hand. This newly constructed dataset is later used to train and evaluate Job Recommender Systems. As noticed in the previous chapter, the field of Job Recommender Systems research lacks publicly available datasets, as data used for recruitment is often of personal nature for candidates, and of strategic nature for the recruiting companies. As such this type of dataset is confidential and cannot be disclosed. For this reason, our dataset cannot be made publicly available either.

3.1 ALTEN's structured resume format : Technical Documents

A resume, also named curriculum vitae (CV), is a document written by candidates intended for recruiters in which their educations, skills and accomplishments are listed. The resume is a crucial item of the recruitment process as recruiters usually decide if they want to meet with a candidate on the basis of this document alone. In order to facilitate processing, ALTEN uses a digital format for resumes which they call Technical Document (TD). When a new collaborator joins the company he is asked to fill in his TD using a dedicated software. This is a rich and structured format designed to hold any information that can be of use to illustrate the consultant's professional capacities. It contains information about the candidate education, including degrees, qualifications, and MOOCs (Massive

Open Online Courses). Each entry includes the formation premises (it can be online or a school), the formation name, and the graduation year. It also contains multiple lists of the candidate's skills. For example, there is a list dedicated to languages skills which includes the candidate proficiency level in the language. Finally, DTs contains the full list of the candidate's professional experiences. Each experience has a title, a time span, the context (e.g. the IT department of a sportswear company), the assignment goal (e.g. designing an API for an e-commerce platform), the position held, a list of tasks accomplished during the experience, and a list of skills used.

When exported to PDF, Technical Documents can feature 10 or more pages. They are as such far more detailed than a conventional resume. It is common among companies, especially consulting companies, to use similar elaborated and structured resume formats.

ALTEN stores its TDs in a SQL database, and internal tools can access them through the use of dedicated APIs. Responses to queries are in JSON format. DTs are retrieved using this API to train and evaluate our various models presented in this manuscript, and the Recommender Platform from section 6 also uses this API.

3.2 Building a Job Recommendation Benchmark Dataset for learning and evaluating models

In order to evaluate our Job Recommender Systems and establish a fully supervised Deep Learning model that can serve as a baseline, we need an annotated dataset of resumes that are matched, positively or negatively, to job offers. As previously noticed, such a benchmark dataset is not available in the public domain. However, ALTEN's resumes dataset availability gives us the opportunity to build one. We hence present in the remainder of this section the procedure to build this annotated job recommendation dataset.

The procedure is built upon the intuition that, on the one hand, a resume almost always contains at least two successive experiences. On the second hand, it can be assumed that the last experience was started following a recruitment process for which the experience information available to the recruiter at the time was the candidate's full experience list, but, crucially, without the last job included.

A further assumption can then be made that the last job experience can be viewed as a job offer for which the previous experiences are satisfactory for a recruiter. This last assumption can finally be used to construct our annotated dataset : the last experience represents the "job offer" for which the remainder of the experiences is the "resume". This how positively matched pairs are formed. Figure 3.1 shows this process.

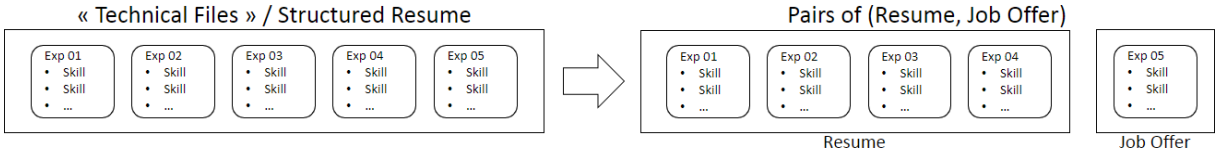


FIGURE 3.1 – We split the list of experiences from a TD to produce a pair of a resume and a job offer.

Similarly, if the job offer and the resume come from different TDs, a negatively matched pair is created, that the system should not recommend. Negative pairs are generated by pairing resumes from one DT with a job offer coming from another randomly selected DT. Figure 3.2 describes this process. We ensure the number of positive and negative pairs is the same by making two pairs per TD.

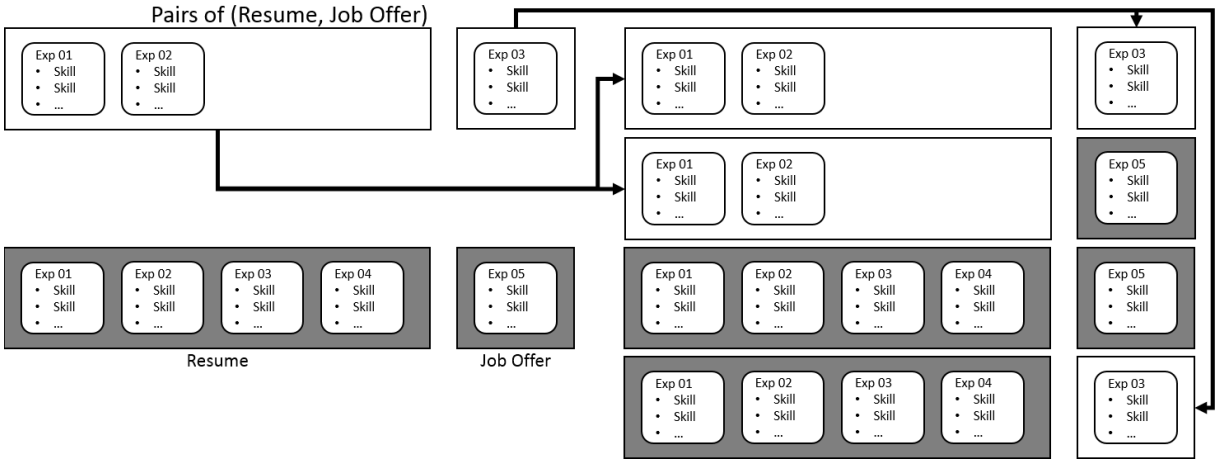


FIGURE 3.2 – This figure explain the process of building a dataset to evaluate Job Recommender System from a list of structured resume (Technical Documents in our case).

This solution can be transposed to any dataset for resumes if there is a way to extract experiences from them. In our case it is trivial because resumes are in JSON format, and experiences are inside an array, but if the resumes were to be in a document format like PDF or ODF, a parsing system would be required.

There are some limitations to this method, first and foremost the fact that negative pairs are matched at random. This means that if resumes are from people working in fields that have nothing in common, it can be trivial for the system to make the difference. Take for example an experience as a cook being paired with the resume from a software engineer. Similarly, if the scope of jobs is too narrow, we can end up with negative pairs in which the resume is actually a good match for the job. In our case, ALTEN employing consultants from most engineering fields, we believe the scope to be appropriate for this methodology.

To build our dataset 30,907 Technical Documents were used from which 61,814 pairs are generated that are divided into 3 subsets :

— a **Train set** composed of 49,452 pairs, used to train models.

— a **Dev set** composed of 6,181 pairs, used to fine tune models parameters.

— a **Test set** composed of 6,181 pairs, used to make a final evaluation of the models.

We use this dataset in section 4.4 to train a Deep Learning based Job Recommender System and compared it with an ontology based one.

Chapitre 4

Conversational Recommender System based on the ESCO ontology

This chapter investigate our first research question, and the potential of the ESCO [1] ontology to make an explainable job recommender system well-suited to ALTEN context. This work was published[36] at the RecSys 2021 conference in the KaRS workshop.

Our initial goal is to propose a Job Recommender System usable on ALTEN's data, and that can produces explainable recommendations. A great way to achieve an explainable Recommender System is to make it Knowledge-based and relying on an ontology composed of concepts and relationships which can be easily presented and understood by the users. ALTEN's resumes do not use such an ontology, so we decided to use ESCO because it is made by the European Union as a standard to describe the entirety of the skills, qualifications, and occupations that can be found in the labor market, and also because most concepts inside the ontology contain multiple labels in 28 languages allowing to train a classifier on it. Such a classifier could be used to project skills found inside ALTEN's resumes into the ontology.

The chapter begins with a presentation of the ESCO ontology in Section 4.1 and its structure in Section 4.2. To use this ontology with ALTEN data, we first need to project the skills and occupations inside it. The classifier we trained to do this is based on the neural network BERT[10], and is presented in Section 4.3. Once ALTEN's resumes and job offers are represented as a set of concepts from ESCO, we can design a Conversational Recommender System that relies on the structure of the ontology to make recommenda-

1. https://esco.ec.europa.eu

tions, explanations, and offer the user the ability to fine-tune those recommendations ; this system is presented in Section 4.4. Finally, in Section 4.5 we will take a look at ESCO Explorer, a tool we developed to visualize and navigate the ESCO ontology that has been particularly helpful to us.

4.1 Presentation of ESCO

ESCO stands for European Skills, Competences, and Occupations. It is a multilingual ontology developed by the European Commission that provides a standardized framework for describing and comparing the skills and occupations of workers in the European Union (EU).

The purpose of ESCO is to facilitate the recognition and transfer of qualifications and skills between countries. It is designed to be used in a variety of contexts, including education, training, and employment, and is intended to help individuals and employers make informed decisions about career development and mobility.

ESCO is organized around a set of core concepts that define the structure of the classification : skills, occupations, and qualifications. Those concepts are linked using relationships such as : "skill X is optional/essential for occupation Y".

ESCO also provide a public API for tools that wish to use the ontology. Currently the ESCO website [2] claims that 113 organisations use the ontology across the world. Its usage is not limited to the European Union though it is here it is the most popular.

4.2 ESCO Structure

In our work, we focus on occupations and skills. We did not take the time to look into qualifications. The data we present here may be outdated because ESCO keeps evolving to better reflect the labor market. For example the ESCO website tell us there are currently 3008 occupations at level 5 and below. At the time we worked on ESCO, we used the version 1.0.3 and we observed 2942 occupations and nothing below level 5.

2. `https://esco.ec.europa.eu/en/about-esco/esco-stakeholders/esco-implementers`

4.2.1 ESCO Occupation

ESCO organizes occupations in a tree composed of 5 levels (not counting the root). The first 4 levels correspond to ISCO (International Standard Classification of Occupations). ESCO Skills are located at level 5. There is not a single branch that skips a level. ISCO nodes are duplicated across levels if required. If we look at figure 4.1, rather than linking the node C directly to H and I if there is no intermediary occupation class between them, ISCO will instead create an occupation E that is a copy of C. For example, the path "Armed forces occupations" [3] → "Non-commissioned armed forces officers" [4] → "Non-commissioned armed forces officers" [5] → "Non-commissioned armed forces officers" [6] → "sergeant" [7] (the footnotes correspond to the URIs to identify the occupations and access them in the ESCO website).

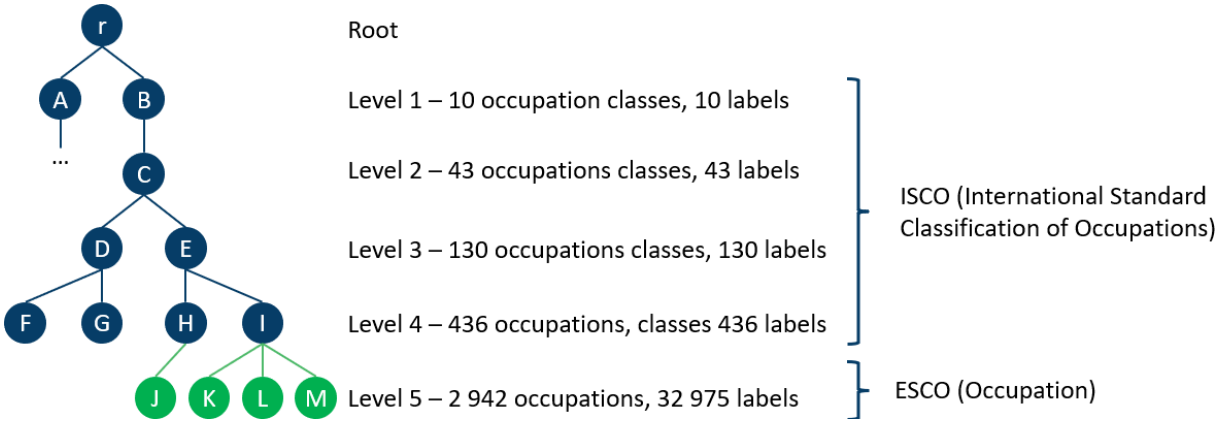


FIGURE 4.1 – The structure of ESCO occupation. The first 4 levels correspond to ISCO and the level 5 are occupations. The number of labels correspond to the number of preferred and alternative English labels at each level.

Every item/concept in ESCO (e.g. an occupation) has a "preferred label", a list of "alternative label", an URI to be identified, and a "description". The labels and description are in 28 languages, all official European Union languages plus Icelandic, Norwegian, Ukrainian, and Arabic [8]. In figure 4.1 the number of labels correspond to the sum of all available

3. http ://data.europa.eu/esco/isco/C0
4. http ://data.europa.eu/esco/isco/C02
5. http ://data.europa.eu/esco/isco/C021
6. http ://data.europa.eu/esco/isco/C0210
7. http ://data.europa.eu/esco/occupation/cb3b2662-774a-4e6f-841e-f120244d7031
8. https ://esco.ec.europa.eu/en/about-esco/what-esco

English labels at each level (preferred and alternative labels). There are no alternative labels for ISCO classes.

4.2.2 ESCO Skills/Competences

ESCO Skills currently is a Directed Acyclic Graph, i.e. a node can have multiple parents. When we worked on it (version 1.0.5), it was a tree and it contained 13 485 skills organized on 5 levels. Figure 4.2 shows a part of ESCO and its current organization (version 1.1.1). ESCO Skills now goes deeper than when we worked on it and skills can have multiple parents.

ESCO skills can be too broad to be used effectively in IT and other highly specialized fields. Some crucial skills are grouped under the same skill or not mentioned at all. For example "Docker" and "Kubernetes" are both included in the description of the skill "manage ICT virtualisation environments", it would be preferable to have them as children. Some other skills are simply missing, for example, there is no mention of "Java Enterprise Edition", it would be adequate to have it as a child of the ESCO skill "Java (computer programming)". ESCO is continuously evolving, but with only 400 skills added since we worked on it more than a year ago, it may not be enough to keep up with the appearance of new skills on the job market.

4.3 ESCO classifiers

In this section, we present the conception of two Deep Learning models to match a text input to an ESCO occupation or an ESCO skill. We published this work in the workshop KaRS at RecSys 2021 [36]. The goal is to use ESCO relationships between occupations and skills to match resumes with job offers. The first step toward this goal is to map non-ESCO occupations/skills from resumes and job offers into ESCO.

In this section, we use the term "node" to denote an occupation or a skill from ESCO.

To achieve our goal, we rely on the numerous labels associated with nodes (every node has one preferred label and can have many alternative labels). Because the structure of ESCO is hierarchic, the classification happens at every level. Each node is a class and every label

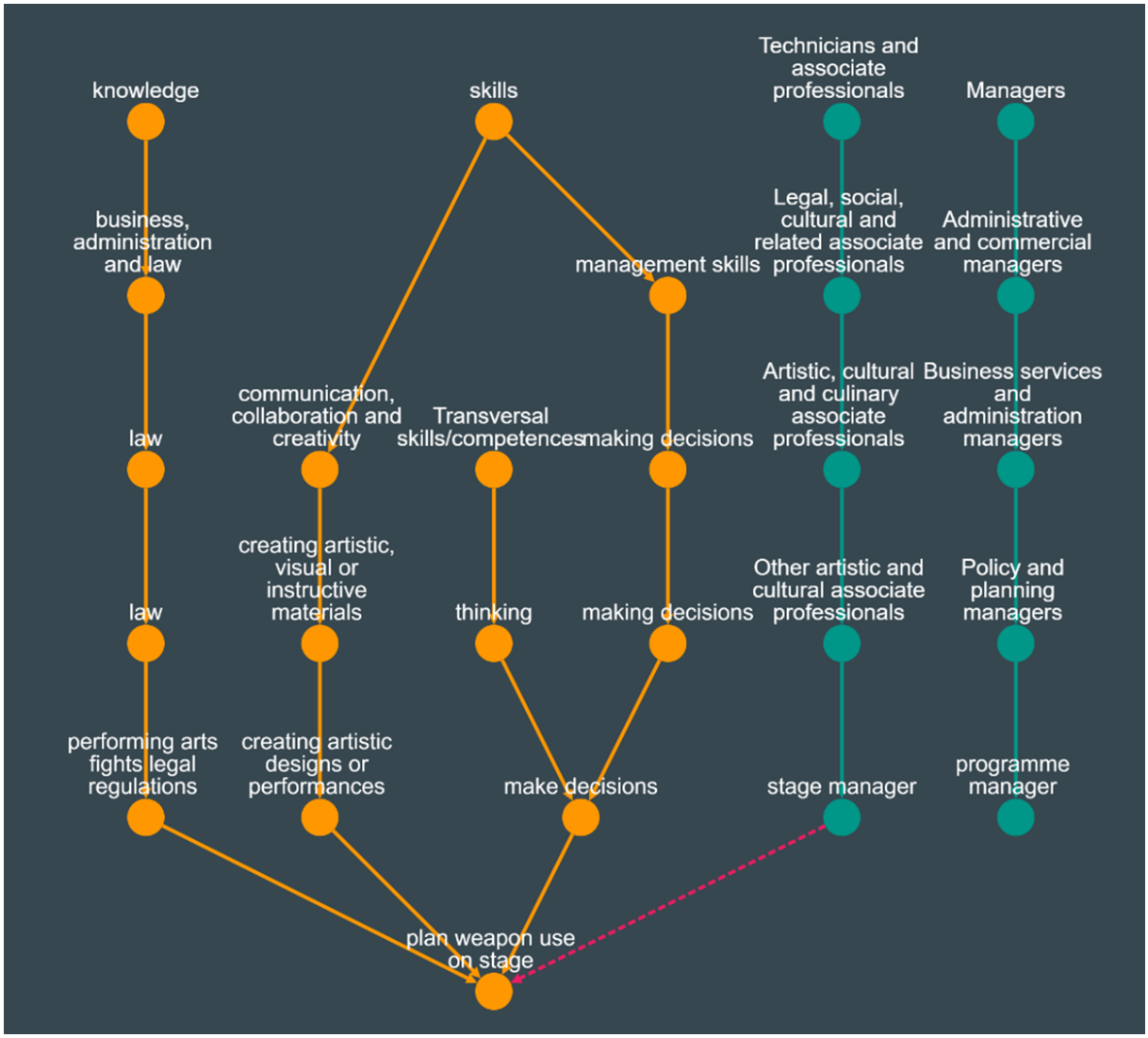


FIGURE 4.2 – An example of the structure of ESCO Skill and Occupation. On the left in orange we have skills, and on the right we have occupations. We notice that the skill "make decisions" had two parents, this does not happen with occupations. The pink dotted link indicate that "plan weapon use on stage" is an optional skill for the occupation "stage manager".

(preferred and alternative) from the node and its descendants are associated with that class. For example, in the case of figure 4.1, every label from nodes E, H, I, J, K, L, M should be associated with node E when classified on level 3. In the case of ESCO skills, a label can be associated with multiple nodes.

4.3.1 ESCO occupation classifier

We use the model BERT[10] as a base for our classifier. We experimented with two pre-trained versions : "bert-base-uncased" which is trained on English data and is case insensitive and "bert-base-multilingual-cased" which is trained on multilingual data (104 languages). The usage of the multilingual model is motivated by the multilingual aspect of ESCO and because ALTEN is a multinational corporation.

Inspired by the work from Liu et al. [34] we experiment with 2 versions of the classifier :

1. **Single-Head Model** : one model classifies on one level, so if there are 5 levels we need 5 models.
2. **Multi-Head Model** : one model classify on every level.

The intuition is that by being trained to do more tasks and "seeing" more data the multi-head version should perform better.

We program the model in the language Python with the libraries Pytorch and Transformers [9]. The model features 180 million parameters, is trained with the optimizer BertAdam and uses a Cross-Entropy loss function. The training continues until the accuracy on the devset hasn't improved over 3 epochs, and keeps the version with the highest accuracy (so 3 epochs before stopping). The training time is between 3 and 4 days for about 20 epochs on a good CPU (we did not had access to a GPU with enough memory to fit the model).

The performances (accuracy) of the different versions of the model are visible in table 4.1. The differences between models are the number of heads, the pre-trained version of BERT, and the languages used to train them. Column 1 shows the accuracy of a single-head model trained in English, whereas column 2 shows the results of the same model but with multiple heads. It can be noticed that the multi-head version performs better than the single-head on every level. The performance gain combined with the simplicity of having one model for the entire hierarchy instead of one per level makes it a better choice. Hence a choice is made that subsequent models all use the multi-head architecture.

Columns 2 and 3 compare bert-base-uncased and bert-base-multilingual-cased trained on English data. The BERT model pre-trained only in English performs better. However, when bert-base-multilingual-cased is being trained on multilingual data, including En-

9. https ://github.com/huggingface/transformers

TABLE 4.1 – Comparison of the accuracy (in percent) of the different versions of our ESCO occupation classifier. The head indicates if this is Single-Head (one model per level) or Multi-Head (one model for every level). The Model indicates the pretrained version of BERT used as a base for the classifier. Train Set indicates on which language the classifier is trained. Test Set indicates the language on which the classifier is evaluated. The last row is simply a numbering of the columns to refer to them more easily in the text.

Head	Single		Multi							
Model	bert-base-uncased			bert-base-multilingual-cased						
Train Set	en			Multi (en, de, it, es, fr)						fr
Test Set	en			Multi	de	it	es	fr		
Level 0	89.2	89.5	89.0	97.1	97.5	98.1	96.8	98.0	97.8	94.4
Level 1	87.5	88.3	86.9	96.6	97.0	97.9	95.9	97.5	97.4	93.1
Level 2	85.8	85.9	85.1	96.4	96.8	97.5	96.0	97.2	97.4	91.5
Level 3	83.5	84.2	82.9	96.1	96.4	97.1	95.3	96.8	97.0	89.7
Level 4	78.0	78.2	75.9	92.3	92.8	92.9	91.8	92.4	95.1	86.5
Average	84.8	85.2	84.0	95.7	96.1	96.7	95.2	96.4	97.0	91.1
Column #	1	2	3	4	5	6	7	8	9	10

glish, it performs much better (column 4). We make a similar observation with the French language (columns 9 and 10).

We train our multilingual model only in English, German, Italian, Spanish, and French, because using the 28 languages in ESCO would have taken too much time and resources.

In conclusion, to get the best performances at classifying occupations into ESCO, we should use the same model for every level and every language.

4.3.2 ESCO skill classifier

The ESCO skill classifier is an application of what we learned in section 4.3.1 to the Skill/Competence part of ESCO. Olivier Dedocoton constructed the dataset and trained the model under our supervision during his internship at ALTEN. It has the same properties as the occupation model, table **??** presents it performances.

TABLE 4.2 – ESCO Skill classifier performances (accuracy in percent).

Level	0	1	2	3	4
Accuracy	99	96	94	93	91

4.4 ESCO Recommender System

In this section, we present how to generate explainable job recommendations based on the skills present in the resume and the job offer using ESCO.

The first step is to extract skills from the documents. In our case, it is trivial because ALTEN uses a structured resume format named "Technical Document" (TD) that contains fields dedicated to skills. We present this format in section 3.1.

The next step is to map skills extracted from resumes and job offers with skills from ESCO using the classifier presented in section 4.3.

Now that resumes and job offers are represented by a set of ESCO skills, we can leverage the structure of ESCO to assess the similarity between the resume and the offer. We do this using the similarity function proposed by Panagiotis et al. [35] :

$$Sim(a,b) = \frac{d_{max} - depth(lca(a,b))}{d_{max}} \tag{4.1}$$

This equation measures the similarity between the skills a and b. d_max denotes the maximal depth of ESCO (4 in our case, it starts at 0) and $lca(a,b)$ the lowest (highest depth) common ancestor of a and b.

To compute the similarity between a job offer o and a resume r we propose the following equation :

$$Score(o,r) = \frac{\sum_{a \in S_o} \max_{b \in S_r} Sim(a,b)}{|S_o|} \tag{4.2}$$

For each skill of the job offer $a \in S_o$ we search for the most closely related skill from the resume $b \in S_r$ according to equation 4.1 and compute the average of those associations. The higher the score the better the resume r match the job offer o.

We can come up with different strategies to make recommendations using the score. For example, we may present to the user the n resumes with the highest score for a job offer. Or we may present every resume with a score superior to a threshold. This threshold can be set manually or learned. We can also combine both strategies : present the n resumes with the highest score that is superior to a threshold.

We can explain recommendations made by this system to the user. Figure 4.3 presents an explanation of such a recommendation. It is a graphical explanation, but we can also pro-

duce a textual explanation. In section 2.2.3 we presented why explaining recommendations is important.

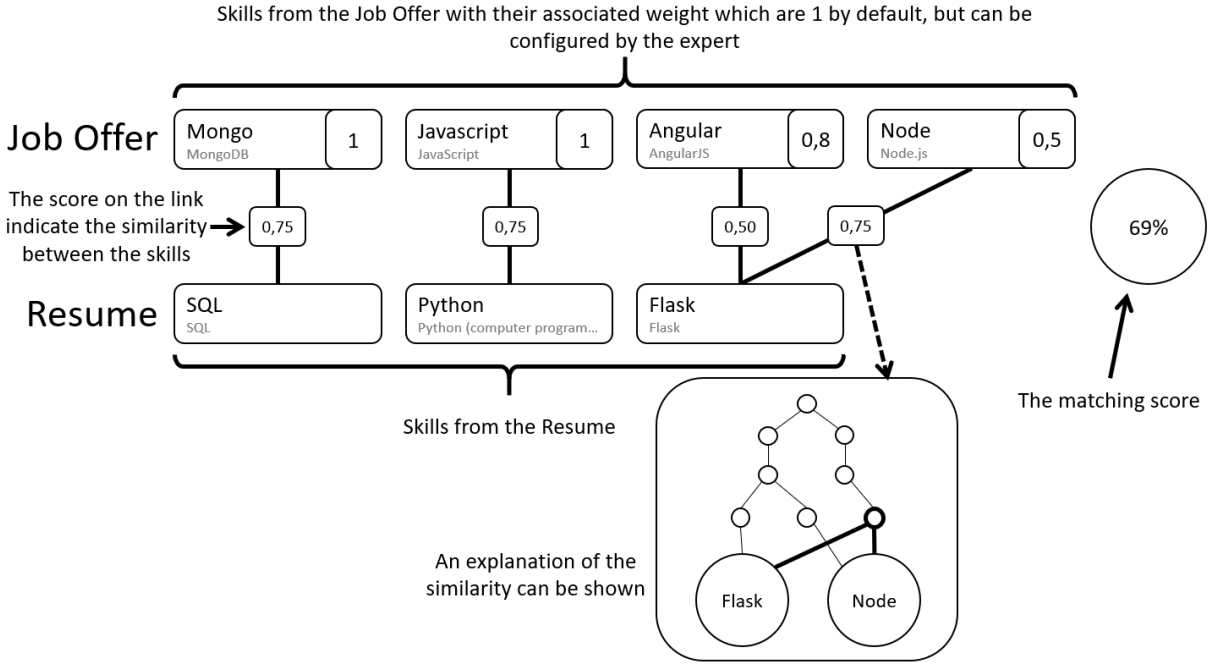


FIGURE 4.3 – An example of a graphical explanation of a recommendation made by the system. We can see how important each skill from the job offer is (this value can be adjusted by the recruiter), we can see with which skill from the Resume it has been matched, how similar both skills are, and how this similarity has been estimated based on the ontology.

Explanation of recommendations is not everything. We also want to evaluate our system on the dataset presented in section 3.2. It is a binary classification task : the resume is either recommended or not recommended to the associated job offer. The strategy we use is to set a threshold : the system recommends the resume if the score is higher than the threshold. We determine the threshold using linear regression on the scores from the train set and evaluate the model on the test set.

Before looking at the performances of our model, we want to present a full deep learning baseline we trained on the same dataset.

4.4.1 Full Deep Learning Baseline

We call the model we present in this section "Full Deep Learning" because it only relies on Deep Learning in opposition to the ontology-based model that uses Deep Learning only to map skills extracted from documents to ESCO skills.

This model is based on BERT and is quite straightforward. We pass a resume and a job offer (both represented by a list of skills) as text to the model. They both end with the special token "<sep>" and the input start with the special token "<cls>", it is at this position that the model will make the binary classification : should we recommend the resume for the given job offer. Figure 4.4 presents this process.

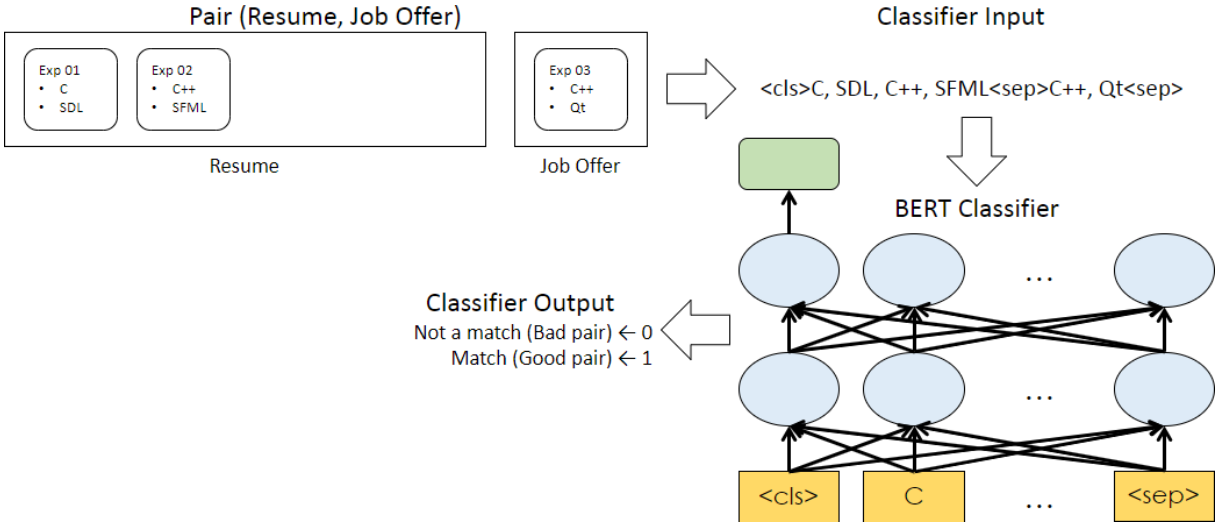


FIGURE 4.4 – The process to predict is a resume should be recommended for a given job offer using only a Deep Learning model based on BERT.

We train the model on the train set from the dataset presented in section 3.2. We stop the training if the performance on the dev set has not improved after 3 epochs. Finally, we evaluate the final performances on the test set.

4.4.2 Recommendation Performances

Table 4.3 presents the performance of the ontology-based Recommender System that use ESCO and the Full Deep Learning system on our Job Recommendation Dataset. Unfortunately our system that use ESCO severely under-performing compared to the

Metric	Full Deep Learning	Ontology-based (ESCO)
Accuracy	83.03%	62.76%
Precision	86.96%	61.29%
Recall	77.92%	70.32%
F1-Score	82.20%	65.50%

TABLE 4.3 – Recommender Systems performances. This table show the accuracy, precision, recall and F1-Score on our binary classification dataset for a full Deep Learning model based on BERT and our ontology-based recommender system base on ESCO.

Full Deep Learning system. However the ontology-based system is explainable and the user can act on it to improve the recommendations. This is what we present in the next section.

4.4.3 Conversational/Interactive Recommendation

Because ALTEN is a consulting company and is continuously recruiting consultants, we consider the scenario in which a recruiter is looking for an employee to pursue a mission. The recruiter has an idea of which kind of profile he needs and can fine-tune the recommendations through actions like adding/removing skills from the offer or giving more importance to certain skills.

To allow the user to give importance to skills we can transform the average from equation 4.2 into a weighted average. It looks like this :

$$Score(o, r) = \frac{\sum_{a \in S_o} (W_a \times \max_{b \in S_r} Sim(a, b))}{\sum a \in S_o W_a} \tag{4.3}$$

W_a is the weight associated to the skill a from the job offer o.

To evaluate the maximal potential of this weighting we design the following test. For each pair, we test a thousand random weight matrices and measure the percentage of those that lead to a good prediction. With this setup, we can produce at least one good weighting for 87% of the pairs. It is a bit higher than the 83% accuracy of the Full Deep Learning system and much higher than the 63% accuracy of the non-weighted version of the ontology-based model. However, even if the recruiter does not choose the weighting randomly, there is no guarantee he can find a good weighting for 87% of the pairs.

To evaluate what accuracy a recruiter could realistically achieve we design the following simulation. The recruiter wishes to find the right consultant for a job out of a hundred. The weighted ontology-based system recommends the five candidates with the highest score. If the recruiter does not find the right consultant in those five recommendations he can take the following actions :

- Add a skill : we consider the recruiter has an idea of the kind of profile he wants, and that profile is the right resume for the offer. So if a skill from the correct resume is infrequent among recommended resumes, it can be added to the offer.
- Increase the weight of a skill : if a skill from the offer is in the correct resume but is infrequent among recommended resumes, its weight can be increased.
- Decrease the weight of a skill : if a skill from the offer is not in the correct resume but is frequent among recommended resumes, its weight can be decreased. If its weight reaches 0, this action is equivalent to removing the skill from the offer.

To create this environment we pick 100 correct pairs from our test set at random, and for each job offer we produce five recommendations, if the resume paired with the offer (the right/correct resume) does not appear among those five, the user can take one of the possible actions and the system will present new recommendations to him. We continue the process until the correct resume is recommended or the user has taken 100 actions. To determine which action the user takes at each step, we use the following heuristic :

- Add skill : for each skill in the correct resume but not in the offer, the probability of adding it to the offer is the inverse of its frequency among recommended resumes (equation 4.4).
- Increase the weight of a skill : for each skill in the correct resume and in the offer, the probability to increase its weight, is also the inverse of its frequency among recommended resumes (equation 4.4).
- Decrease the weight of a skill : for each skill in the offer but not in the correct resume, the probability to decrease its weight is its frequency among recommended resumes (equation 4.5).

$$P_{add/increase}(s) = \frac{|R|}{|r \in R \wedge s \in r| + 1} \tag{4.4}$$

$$P_{decrease}(s) = \frac{|r \in R \wedge s \in r|}{|R|} \tag{4.5}$$

R denotes our catalog (the set of available resumes), r is a resume from R, and s is a skill. At each step, we take the action with the highest probability.

This simulation produces the following results : for 74% of the offers the user finds the correct resume in an average of 8.69 steps. So the interaction permitted by an explainable and configurable recommender system allows for improvement upon the raw performances of the system while providing transparency for the user.

4.5 ESCO Explorer

While working on the ESCO ontology we encountered an issue : the interface of the ESCO website is not adapted to visualize a graph. As you can see in Figure 4.5 when looking a node, the skill computer programming in this example, you cannot see not all its ancestors and descendants. You can only see his direct parents and children, so to visualize the path leading from the root of the classification to this node you have to click the parent recursively until reaching the root. This become increasingly complicated for elements with multiple parents.

To make navigating ESCO more manageable for us, we developed an application we named "ESCO Explorer". We thank Alexis Andreani who developed the first version of this tool under our supervision during his internship at ALTEN. The application allows to search for skills and occupations and visualize them as nodes in a graph. It makes identifying relationships much easier as we can see in figure 4.5. The application is developed in JavaScript with the library electronjs [10].

4.6 Conclusion and limitations

In this chapter, we proposed a configurable and explainable ontology-based recommender system that uses ESCO. However, despite its adaptability, its performances stay low compared to a Full Deep Learning model. We have two possible explanations.

As presented in section 4.2.2 ESCO is not specific enough. It is missing specialized skills like "Docker" or "Java Enterprise Edition" that are important to the job market but are

10. `https://www.electronjs.org/`

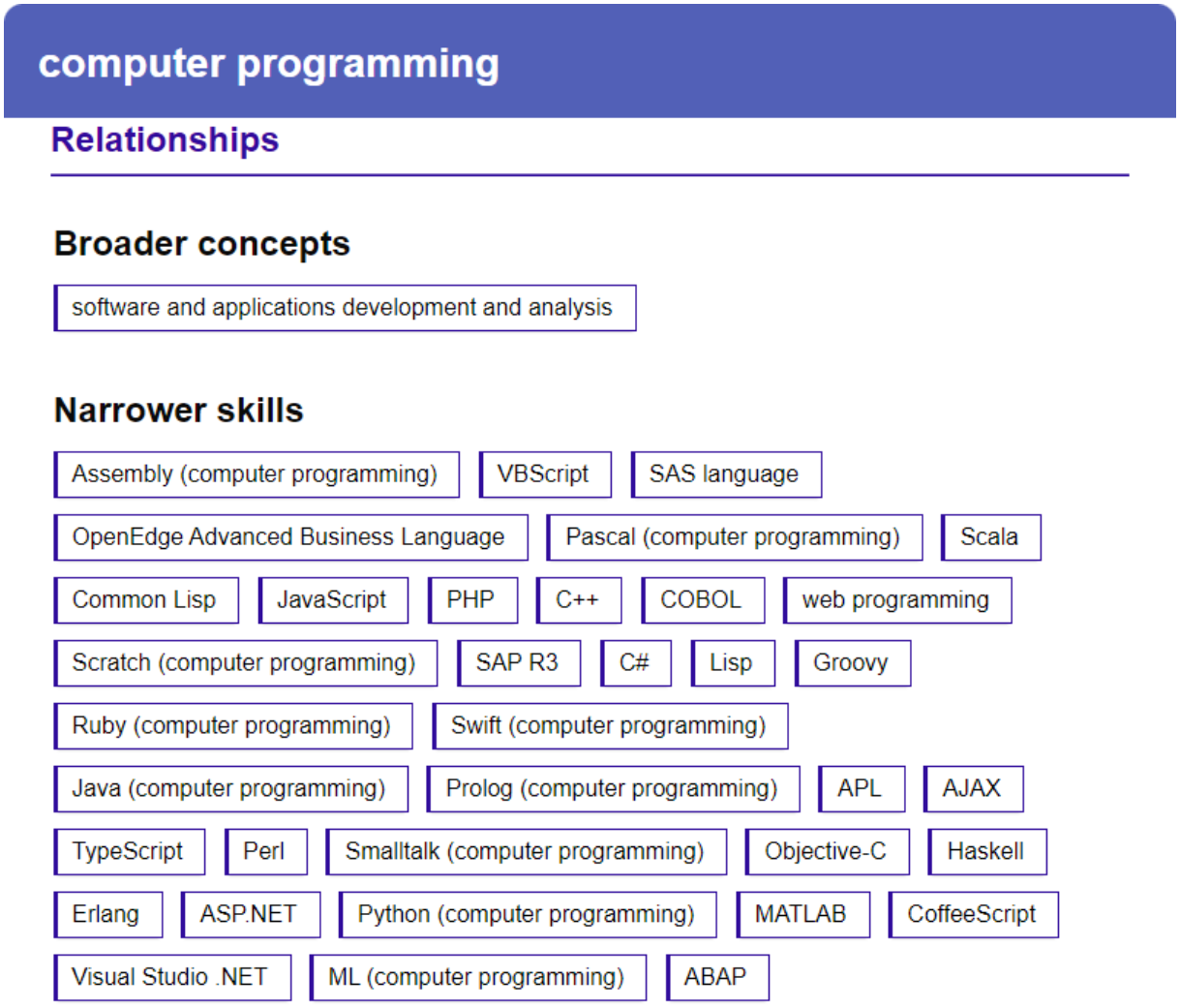


FIGURE 4.5 – Visualization of a skill's relationships on the ESCO website, we can only see its parents and children, but not all its ancestors and descendants.

either grouped with other skills inside meta-skills that are too broad or not mentioned. Extending ESCO could help with this issue, but it is a lengthy process.

Another explanation is the accuracy of the mapping process between skills extracted from documents and skills from ESCO. Our classification model achieves an accuracy of 91% on the deepest level of ESCO skill but is evaluated on a test set composed of labels from ESCO. If we take a closer look at the mapping of a hundred skills from ALTEN TDs, only 69% are somewhat relevant, and a good part of those are imprecise or too broad to be relied upon. We could resolve this issue by integrating ESCO into ALTEN tools :

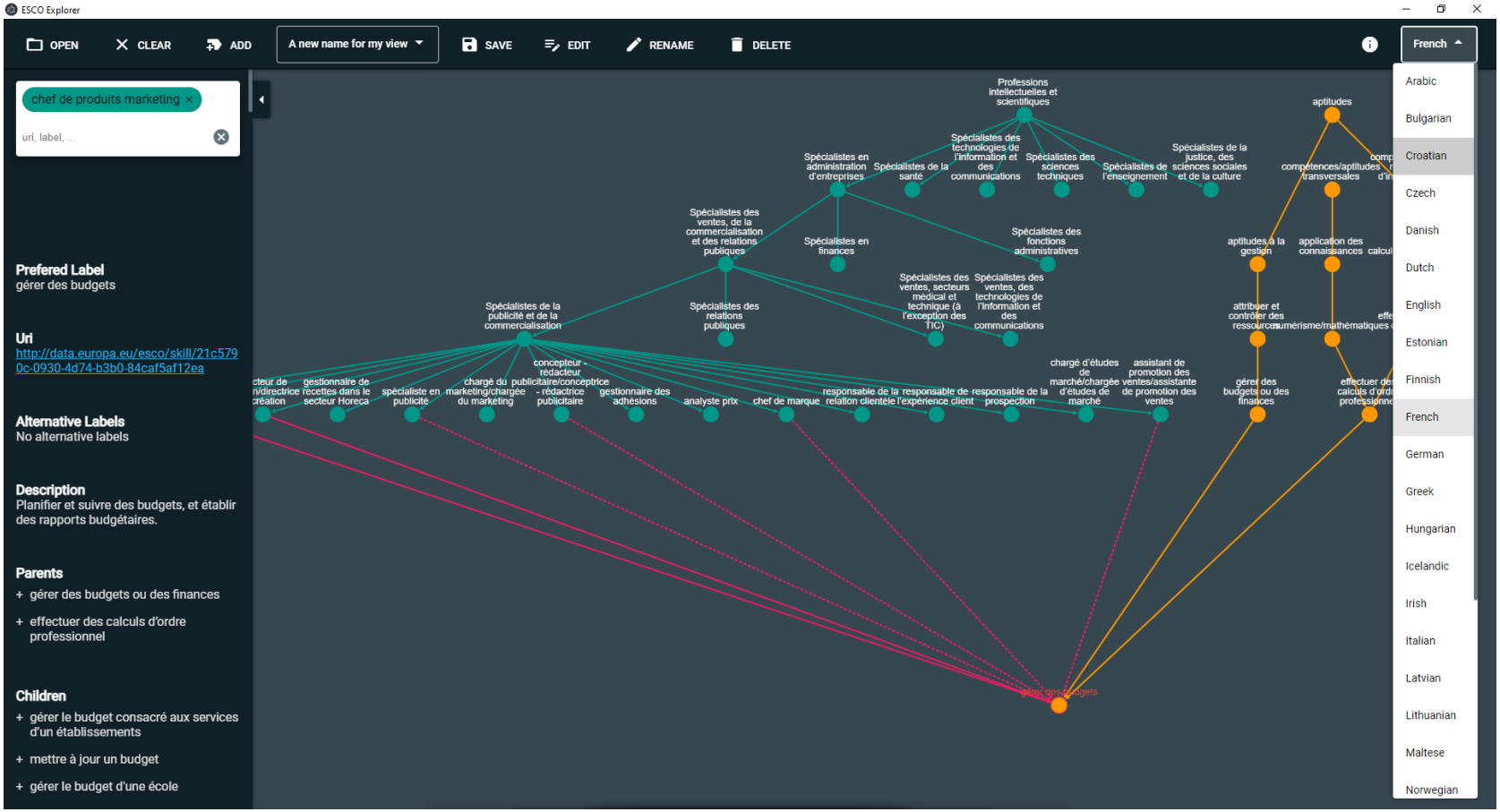


FIGURE 4.6 – A quick look at the interface of ESCO Explorer. We can notice the ease of identifying relationships when using a graph compared to lists like the ESCO website does (figure 4.5).

consultants would pick skills directly from ESCO to fill their TD. However, because ESCO is not specific/complete, ALTEN cannot rely solely on it. It would require developing an extension to ESCO.

Chapitre 5

Classification of ALTEN missions inside an ontology

Being a consulting company with tens of thousands of engineers, it is challenging for ALTEN to know exactly in which fields the company has a presence in. To answer this need of information, ALTEN's analysts developed an internal classification that contains every field the company is an actor of. This classification is then used by analysts to classify the missions conducted by ALTEN. There are tens of thousands of missions to classify every year, performing this task manually is labor intensive. As the classification is an ontology and as the missions are represented by a list of fields redacted in natural language, it was proposed to use the architecture of our ESCO BERT-based ontological classifier (see Section 4.3) to attempt to partially automatize the task. The resulting tool, was a success and is currently in use inside the company.

In this chapter, the structure of the cartography and the missions dataset is presented in Section 5.1. The first version of the classifier is presented in Section 5.2, and finally, Section 5.3 presents how multiple datasets annotated with different versions of the cartography can be used to improve classification accuracy on every versions.

5.1 Data

The cartography developed by ALTEN, characterizes missions using 3 classes :

- **Sector** in which the mission takes place. It contains 11 class labels.
- **Generic Theme** indicates the field of the work done during the mission. It contains 31 class labels.
- **Specific Theme** is a child of the Generic Theme. It adds precision but is missing

for a third of the missions. It contains 92 class labels including the "unspecified" label we added for missions without a Specific Theme.

For example, in a mission to conduct a security audit of the network of a banking company, the sector would be banking, the generic theme would be computer security, and the specific theme would be security audit.

Our initial dataset contains 22 854 missions which can have up to 30 features. 3 of those features are the classes mentioned in the previous section. Only 15 800 missions both have a sector and a generic theme. It is the condition that indicates they are classified and can be used to train and evaluate the model. Our dataset evolves : analysts classify missions and add them over time.

Features are the name of the company the mission comes from, the mission's geographic location (zip code, city name), the mission's description, and some information about the consultant conducting the mission (degree, school, skills, etc.).

Discussions took place with the analysts to determine which features the model can use as inputs, and various combinations are experimented with. The first version of our model only uses the features "Final Customer", "Project Description", and "Missions Technical Description". But in the following version, "Mission Technical Description" is removed because analysts informed us they fill this field at the same time they classify the mission. However, during this study, they decided that the "Mission Technical Description" could be filled in before classification so the model can use it.

The features used as inputs to the model to make predictions impact the portion of the dataset we can use to develop the model because with fewer features there are more missions that either have no usable information (all the features used as input are missing) or are indistinguishable from other missions (i.e. they look like duplicates because the features that differentiate them are not part of the inputs).

5.2 Initial model and results

The model is similar to the one we use for ESCO and that we presented in section 4.3. We use the transformer [1] python library from Hugging Face with pytorch. It is based on the

1. https ://huggingface.co/docs/transformers/index

pre-trained model "bert-base-multilingual-cased" which contains 180 million parameters and is trained in 104 languages. We use a multilingual model because our dataset contains data in English and French. The same model predicts the 3 classes, one head (i.e. output layer) per class, and the head for the specific theme includes a class label "unspecified" for missions that do not have a specific theme. There is one label to predict per class. We use BertAdam as an optimizer, and a cross-entropy loss function. We compute one cross-entropy loss per head and sum the result from each head. The model is trained until the accuracy on the dev set stopped improving for 3 consecutive epochs.

Table **??** presents the different versions of the model. The difference between those versions is the mission's features they take in inputs which in turn impact the number of missions that we can use for the training and the evaluation. The first row of the table correspond to the accuracy we would have if always predicting the majority label. We can notice that despite having the most class labels, the Specific Theme as the most represented label. It corresponds to the label "unspecified", 30.33% of the missions in our dataset do not have a Specific Theme. Another observation we make is that the accuracy on the Specific Theme is systematically better than the one on the Generic Theme. The gap is most notable in the versions 2, 3 and 4 of the model. Once again we attribute this to the "unspecified" label from the Specific Theme which is 2.45 times more represented than the most represented label from the Generic Theme. It allows the model, when in doubt about the Specific Theme, to simply predict the "unspecified" label with a much higher probability to be right than if it does the same with the majority label from the Generic Theme.

The accuracy of our model consistently improve from a version to another, there are two factors that can explain this : more inputs help the model to make the right prediction, and more missions results in a better trained model. The two most notable improvements happen between versions 1 and 2 and between versions 4 and 5. The version 2 gained 2 important inputs (Contractual Description and Consultants Skills), but also lost one of the most important input, Mission Technical Description which is a thorough description of the mission made by an analyst. So the most notable difference between versions 1 and 2 is probably the number of missions used which increased by 66%. For the improvement between versions 4 and 5, we attribute it to the return of the Missions Technical Description as an input. The number of missions also increased by 3% but it does not seems as significant. From those observations we conclude that the first big improvement can

mostly be attributed to more samples, while the second big improvement can be mostly attributed to more inputs. Both of those factors seem to have a significant and similar impact on the performances of the model.

5.3 Multi-year model

2 years after our initial work Niels Quere trained a new version of this classifier during his internship at ALTEN. We thank him for his work. We train this version on data from 2019, 2020, and 2021. However, the cartography changed a lot between 2019 and 2020 and a bit between 2020 and 2021. The classes stay the same, but the classes' labels for Generic Theme and Specific Theme changed. We want the model to make predictions for the latest version of the cartography (so 2021 at the time) and leverage the data from 2020 because as seen in the previous section the number of missions has an impact on the model's performance.

The first solution envisaged is to project labels from 2020 to 2021 and to leave out 2019 because there are too many changes between 2019 and 2021. The issues with this solution are that the data from 2019 are left out, we cannot map labels that are split into multiple labels, and analysts must manually make the projection which takes time and reflection. We train this model with data from 2020 to predict labels from the cartography of 2020, but at the time of inference we project the prediction into the cartography from 2021. So if a label from 2020 is split into 2 labels in 2021, we will present the 2 labels to the analysts and they will decide which is the correct one.

The second solution is to double down on the multi-head aspect of the model. Instead of having one head per class, the model can have one head per class per year. During training, we compute the loss for the heads that correspond to the year of the mission passed as input. The same batch can contain missions from different years. At inference time we look at the prediction from the heads for 2021. Figure 5.1 shows the overall architecture of the model.

Table 5.2 presents the performance of both model. Projection 2020 is only trained and evaluated on data from 2020 because very few data from 2021 were available when we started working on this model, so there is effectively no projection. We would use the projection to classify missions from 2021 with this model. The projection 2020 model

TABLE 5.1 – Performances and attributes of the different versions of the mission classifiers. The model "Majority" consists in predicting the most represented label for each class. The column "missions" indicates the number of missions used to develop (train, evaluate, and test) the model. The column "inputs" tells us which features the model takes as input, a name in bold is a new addition compared to the previous model, and a stroke through the name indicates it has been removed. The columns Sector, G. Th., S. Th., and Avg contain accuracy in percentage. G. Th. and S. Th. stand for Generic Theme and Specific Theme respectively.

Model	Missions	Inputs	Sector	G. Th.	S. Th.	Avg
Majority	-	-	28.27	12.36	30.33	23.65
v1	8 447	Final Customer, Project Description, Mission Technical Description	80.63	65.44	66.33	70.80
v2	13 983	Final Customer, Project Description, ~~Mission Technical Description~~, **Contractual Description**, **Consultants Skills**	96.21	67.12	74.12	79.15
v3	16 472	Final Customer, **City**, **Department Director Division**, Project Description, Contractual Description, Consultants Skills	96.42	68.75	76.88	80.68
v4	16 848	**Project Company Name**, **Department Code**, Final Customer, City, ~~Department Director Division~~, Contractual Description, **Project Denomination**, Project Description, Consultant Skills, **Consultant School**, **Consultant Career Path**, **Graduation Year**, **Degree Level**	**96.50**	70.27	76.68	81.15
v5	17 261	**Mission Technical Description**, Project Company Name, Department Code, Final Customer, City, Contractual Description, Project Denomination, Project Description, Consultant Skills, Consultant School, Consultant Career Path, Graduation Year, Degree Level	95.95	**86.22**	**86.39**	**89.52**

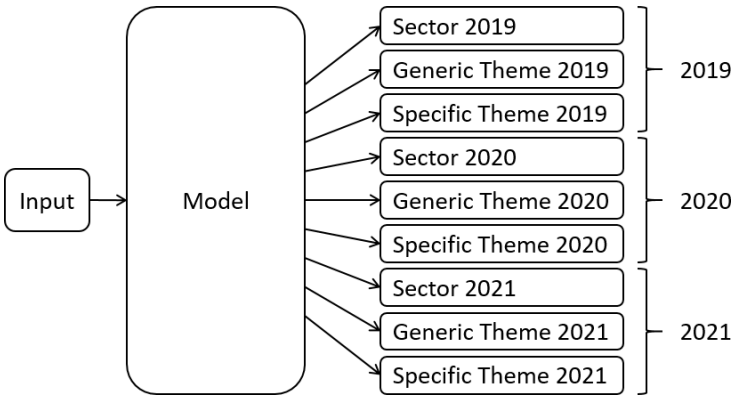


FIGURE 5.1 – The overall architecture of the multi-year missions classifier.

was quickly followed by our first versions of the multi-year model which achieve a net 12.74% accuracy gain on 2020 while being trained with the same number of missions from this year. This gain can be attributed to additional "knowledge" the model gained by also learning to classify 2019 and 2021 ans that is transferable. Because of this gap, we focused our effort on the multi-year model, and the difference between the v3 and the v1 is more data from 2021 (analysts were classifying new missions that we added to the dataset) and fine-tuning on the dev-set of the batch size and the learning rate. Also, the final version of the multi-year model achieve a net 1.41% accuracy gain on the year 2019 compared to our best from the previous section.

TABLE 5.2 – Performances of the model on 2020 and the multi-year model. This is the accuracy in percentage.

Model	2019	2020	2021
Projection 2020	-	70.80	-
Multi-Year v1	83.75	83.54	81.13
Multi-Year v3	90.96	**91.51**	90.56

5.4 Conclusion

Following this experimentation we developed a web service that use this model to make predictions and ALTEN's analysts use it every year to assist them in classifying missions. The model produced does not suffer from the same limitations as our ESCO classifier because it is directly trained on ALTEN data, and the cartography is proprietary and

contains every concept required to characterize ALTEN missions. Furthermore ALTEN's analysts make improvement to the cartography every year so it does not suffer from the incompleteness issues of ESCO. However what we are doing is only possible because the same analysts invested the time to develop the cartography and annotate thousands of missions.

The design of the multi-year model allows to easily leverage data from previous years even if the cartography change from year to year and improve the accuracy of the model.

Chapitre 6

AI4HR Recruiter

In this chapter we answer our first research question by successfully developing an embedding-based job recommender system that is well suited to ALTEN context. This success allow us to build a recruitment platform that we use as an experimental playground to answer the rest of our research questions.

In consulting companies like ALTEN, it is common to assign consultants without a client mission to an internal project. In this chapter, we present AI4HR Recruiter, a tool designed to support its users in a specific internal recruitment process. Our methods exploit the content of the Technical Documents presented in section 3.1 with the help of transformers-based embedding. Our recommendation techniques are content-based and provide the user with an explanation of the recommendation. These explanations highlight the parts of the resume that are relevant to the user. Users can provide explicit feedback to the system through those recommendations. This feedback is effectively used by the recommender system to improve recommendations. The tool also collects implicit feedback (e.g. recording of actions such as opening a resume), so we can analyze usage and further improve the system.

We evaluate our methods and tools in a real context. Our tool is in use in a real (internal) recruitment process at ALTEN. Our experiments show that recommendations explanation translates into a significant gain in efficiency. We also compare our embedding based recommendation techniques to a simple keyword based search. Our experiments show that the users prefer our recommendation techniques. Also, users have a good overall perception of the tool, which they prefer it to the previous system, that was used for the internal recruitment, which did not include recommendations.

The chapter is organized as follows. Section 6.1 presents the context and business process we aim to improve through recommendations. Section 6.2 discusses our recommendation

techniques and our AI4HR Recruiter tool. Section **??** present our experimental evaluation, and Section 6.4 concludes the chapter.

6.1 Business Process Mercato

We present here the context and the business process in which our tool AI4HR Recruiter is used. The details of this process might be specific to the company ALTEN [1], however other companies can have similar business processes.

At the company, consultants work on missions or projects for clients. The latter's goals are to perform tasks for the company and serve as a training opportunity during which the consultants can acquire competencies, strengthen their skills and gain valuable experience. Consultants without a client project are placed in a pool of available consultants. Project leaders hire consultants from this pool regularly. Specifically, at ALTEN, there is a weekly event called "mercato" where project leaders look at the resumes of available consultants.

During this weekly event, project leaders express who they wish to hire. Based on these wishes, the managers decide whom to hire for which project, especially if multiple project leaders want to hire the same person. Our recommendation tool AI4HR Recruiter helps project leaders identify adequate candidates. This internal recruitment process is called at ALTEN the "Mercato".

The company wants a maximum of consultant engaged in a mission. Ideally, none should be left unoccupied. At the end of the wishes phase, managers will ask project leaders who want to voluntarily hire consultants left without any wishes. Most end up with a mission.

Before our tool was available, another one was in use. This predecessor would only display the list of available consultants and allow users to make wishes by putting a certain amount of credits on consultants. They had a limited number of credits to spend each week. The system came with some issues, one of which was the all-in strategy : a user would find a consultant he wants, put all his credits on him, and stop the process there. It is an issue because there are many more consultants than users, and the policy is to hire a maximum of consultants.

In general, there is a conflict between the goal of the process and the interest of project

1. `https://www.alten.fr/`

leaders. The company wants a maximum of consultants occupied, so managers ideally need every consultant to have a wish from a project leader. But project leaders want to spend as little time as possible on the "Mercato" process. An easy way to accomplish this is to look at as few consultants as possible.

What does this conflict means in term of recommendation ? It is in the best interest of the process to maximize the recall of recommendations made to project leaders. But it is in the best interest of project leaders to maximize the precision of recommendations. A system with good recall and good precision would be ideal.

We insist that our job candidate recommender system does not automate the recruitment process. Its goal is to help project leaders to identify the candidates and to improve the above business process.

Two more limitations of the previous tool are the impossibility to view a consultant's resume inside it. You need to click a link to open the resume in another tool. And no research nor recommendation functionalities. AI4HR Recruiter drastically improve the "Mercato" process by providing both of those.

6.2 Recommendation Platform

This section describes our tool AI4HR Recruiter and its recommendation strategies. Here is a non-exhaustive list of its features :

- User's preferences acquisition (Section 6.2.2)
- Exploration of the catalog of consultants
- Consultation of a consultant's profile
- Search the catalog of consultants
- Filter consultants based on a few features
- Receive recommendations
- Bookmark consultants (Section 6.2.5)
- View other users' bookmarks
- Take notes about a consultant

You can have a look at what the tool looks like in Figure 6.1.

Description of Figure 6.1 :

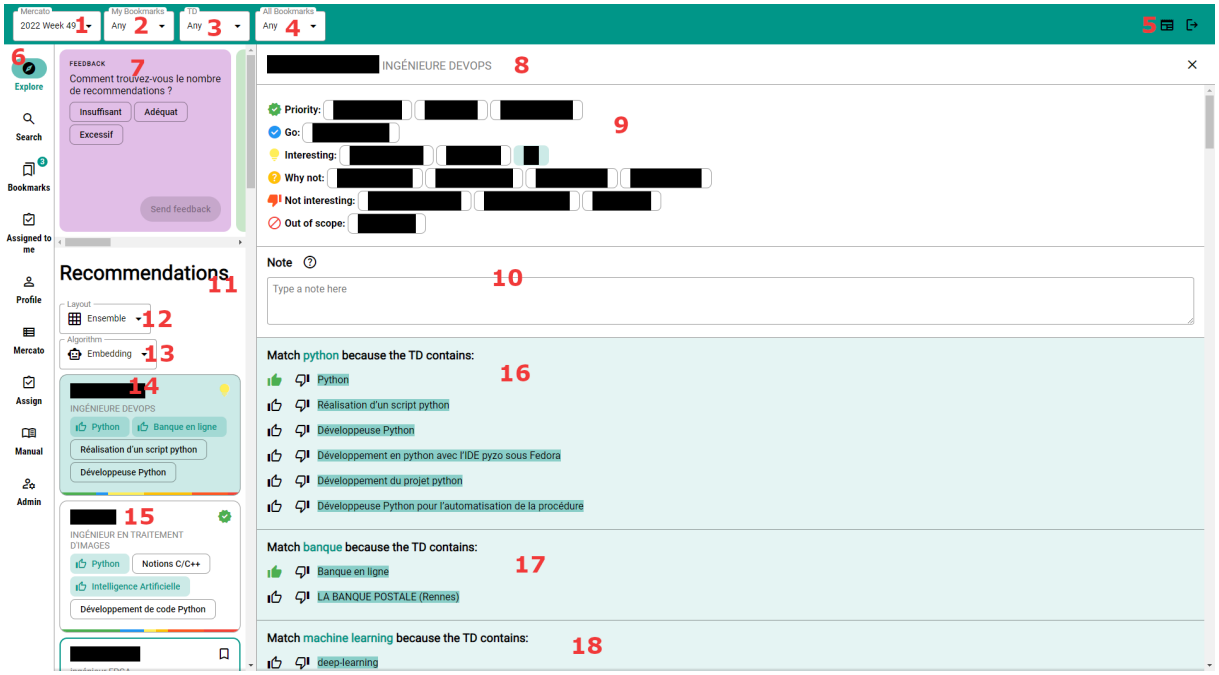


FIGURE 6.1 – AI4HR Recruiter interface

— **1** is a dropdown menu to select the week to display.
— **2**, **3**, and **4** are filters users can use to hide some consultants.
— **5** is next to a button that opens a dialog informing users of the newer functionalities and another button to log out of the tool.
— **6** is a navigation rail to switch between the various panels. Currently, we are in the explore panel that mostly contains recommendations.
— **7** is a card that asks a question to the user. We use those to retrieve explicit feedback for this study. It is significantly more efficient at surveying users than asking them to fill out a form in a separate tool.
— **8** is a side panel showing the selected consultant. The black rectangle hides the consultant's name for confidentiality. The side panel is closed if there isn't any consultant selected. In this case, the main panel fills the entirety of the screen. It contains various information about the consultant, including his resume (you need to scroll down past the bookmarks and recommendations to see it).
— **9** is a section containing all of the consultant's bookmarks. The black boxes hid the name of the users who made them.
— **10** allows the user to add a note to the consultant. This note is visible to the managers and is used to make the final assignment. Typically, a user would use

this note to explain to the managers why they should assign the consultant to him.
— **11** is the list of recommended consultants.
— **12** is a dropdown menu to select the layout used to display recommendations. "Carousel" make a sub-list (carousel) for each user's interests so a consultant can appear in multiple lists. "Ensemble" combine all the interests in the same list, so a consultant only appears once.
— **13** is a dropdown menu to select the recommendation algorithm. Its value can be enforced and hidden from the user for experiments.
— **14** and **15** are consultants cards. They contain the name of the consultant (hidden by a black box), the title of the consultant (i.e. his typical job), the bookmark button, and a list of chips that represent fields that match some of the user's interests.
— **16**, **17**, and **18** are recommendations and explanations for the consultant. In this example, the system recommends the consultant for "python", "banque", and "machine learning".

To obtain recommendations, users first define their interests. We discuss this preference acquisition phase in section 6.2.2 while section 6.2.3 presents our recommendation algorithms and techniques. Section 6.2.4 describes the explanation of the recommendation, how we use this explanation to gather explicit feedback from users, and how we leverage those feedbacks to improve the recommendation. Finally, in section 6.2.5, we describe how users specify which consultant they wish to hire using bookmarks.

6.2.1 Structured Resume

As mentioned in the introduction, ALTEN uses a standardized resume format named Technical Document (TD). It is a rich and structured format designed to hold any information that can be potentially useful to staff a consultant. It contains every formation, from a university degree to a MOOC (Massive Open Online Course), with various information like the school, the university course, or the year of graduation. It contains multiple lists of skills. One is dedicated to known languages and includes the proficiency level in the language. The format also contains a list of professional experiences. Each experience has a title, a period, the position held, the goal (e.g. designing an API for an e-commerce platform), the context (e.g. the IT department of a sportswear company), a list of tasks accomplished during the experience, and a list of skills used.

Those resumes are provided to us through an API in JSON format. They are stored as received in our database to be displayed to users. For the recommendation, we reduce a resume to a list of fields. A field can contain any information from the original resume in a natural language format (e.g. a task from an experience or a known language with its level of proficiency). The structure is lost in the process : we do not retain to which experience a task or a skill belongs. Every field is considered equally during the recommendation process. In other words, to our recommender system, a resume is reduced to a list of strings that we call fields.

The motivation to reduce a resume to a list of strings for the recommender system is that it can be adapted to any format with relative ease. It is important because different companies use different resume formats. The trade-off is that we lose information in the process.

Our solution could be improved by associating a weight to the fields. This weight could be determined by the nature of the field (a position may be more important than a skill), the date of the experience (a recent experience may be more important than an older one), or a combination of both. We do not explore this possibility in this publication.

6.2.2 Preference Acquisition

The first time a user opens AI4HR Recruiter, the tool asks him to fill his profile with his interests so the system can start making recommendations. Interests are texts the user can write as he pleases without any restriction or indication. Leaving the user to freely chose his preferences can lead to better recommendations, Narducci et al. [41]. A user can have as many interests as he wants. We discuss how users effectively use interests in section 6.3.1.

6.2.3 Recommendation/Search Algorithms

Once the tool acquires user preferences, it can make recommendations. AI4HR Recruiter has two content-based recommendation strategies : keyword-based and word embedding-based. These recommendation techniques do not suffer from the cold start problem.

Both recommender systems associate a score with every consultant for every user's inter-

Metric	Keyword	Embedding
# Events	441	2342
Mean catalog size	50.52	51.59
Mean # interests	6.46	6.27
Mean # recommendations	37.78	62.85
Mean # recommendations per interests	5.85	10.03
Mean # unique recommendations	18.36	29.86

TABLE 6.1 – A comparison of the number of recommendations produced by each algorithm. The number of events indicates the number of recommendation processes registered by the tool. A recommendation process is initiated every time a user connects to the tool or changes his list of interests. Every other metric is calculated based on what was available when the recommendation was made.

ests. The tool shows every consultant with a score greater than 0 to the user. There is no limit to the number of recommended consultants other than the catalog size. The tool present recommended consultants in descending order based on their scores.

Table 6.1 presents a variety of metrics about the number of recommendations made. Because the catalog size and the mean number of interests are very similar between both algorithms, results are easily comparable. The keyword algorithm makes fewer recommendations (37.78) than the embedding algorithm (62.85), because it is more restrictive. The number of recommendations made by the embedding algorithm is higher than the catalog size because the systems can recommend the same consultant for multiple interests.

There are two layouts to present recommendations to the user. The first one is the carousel layout in which a carousel of consultants is made for each interest. This layout can show the same consultant multiple times. When a user clicks a consultant from a carousel, only the explanation of the recommendation for the interest associated with the carousel is shown. This cause an issue because users rely a lot on the recommendation explanation to make their decision. So a consultant rejected in a carousel may be missed in a later carousel despite being a good match.

The second layout (figure 6.1) is "ensemble". There is only one list of consultants, and when the user clicks a consultant he is presented with explanations for every interest the consultant was recommended for. This layout reduces the number of recommendations presented (mean number unique recommendations instead of mean number recommenda-

tions in table 6.1) to the user, and when reviewing a consultant he is presented with every explanation relative to the consultant. The recommendations explanations are sorted in descending order based on the score of the corresponding recommendation.

Keyword Algorithm

The keyword algorithm tokenizes the query and counts occurrences of the tokens in the Resume. For a recommendation, the query is an interest of a user. The more occurrences, the higher the Resume is ranked. The tokenization split the query into keywords based on the presence of white space and quotes (words inside quotes are not separated). If a query contains multiple keywords but does not contain any words inside quotes, the system will also search for occurrences of the entire query : for the query *Artificial intelligence* it looks for *artificial*, *intelligence*, and *artificial intelligence*. This algorithm is case insensitive.

This algorithm also uses a filter based on explicit feedback from the user. More about that in section 6.2.4.

Embedding Algorithm

In Natural Language Processing, embeddings are a way to represent text in a vector of numbers with a fixed size. Those embeddings usually carry semantic information and permit the use of mathematical functions. It is common to compare two embeddings semantically using the cosine similarity. The first two embedding techniques to become very popular in the field are Word2Vec [39] and GloVe [42]. Nowadays, embeddings based on neural networks are the most popular. The sentence-transformers framework [2] makes more than a hundred pre-trained embedding models readily available. Those models are all based on the work of Reimers et al.[47, 46] with sBERT and are only models we experimented. You can find an evaluation of 30 of those models on our data in Section 6.2.4.

The embedding algorithm embeds the query and the fields of the resume and looks for fields that have a similar embedding to the one of the query. Two embeddings are similar if their cosine similarity is greater or equal to a threshold. We choose the embedding model and compute the threshold using explicit feedback from users, more about that in section

2. https ://www.sbert.net/

6.2.4. The consultant with the most similar field to the query is ranked the highest. The case sensitivity of this algorithm depends on the embedding model.

Similarly to the keyword algorithm, this algorithm also uses a filter based on explicit feedback from the user. More about that in section 6.2.4.

6.2.4 Explanation of Recommendation

The explanation is a list of fields that lead to the consultant recommendation (Figure 6.1). We propose this explanation of the recommendation with the goal to :

- improve **Efficiency**. By highlighting relevant fields, we hope to reduce the time a recruiter spends reading resumes. We experiment on this in section 6.3.4.
- provide **Scrutability**. By allowing users to like/dislike highlighted fields, we hope to improve recommendations and build a dataset to evaluate models.

An explanation highlights a maximum of 6 fields. Highlighting more could overwhelm the user. For the keyword algorithm, the system selects fields that contain the most occurrences of the query's keywords. For the embedding algorithm, the system selects fields that are the most similar to the query (defined in section 6.2.3).

Feedback Filter

As illustrated in figure 6.1 (16, 17, and 18), users can like/dislike fields in the explanation. When it happens, feedback is registered in the database and provides us over time with a dataset of items composed of :

- The **query** for which the system recommended the consultant.
- The **field** for which the user provided feedback.
- If the field was **liked/disliked** hence is it relevant to the query.
- The **user** who provided the feedback.
- The **system** that made the recommendation (keyword/embedding).

The recommender system uses this feedback to provide an additional filter on top of the keyword and embedding algorithms. If a user disliked a field for a given query, we filter out the field without running it through the keyword/embedding algorithm. On the other hand, liked fields are further highlighted in the explanation by being shown as such. Feedbacks are tied to users because we hypothesized that different users may associate

different meanings to the same query. A user can use the query "bank" to find bankers, while another user may use it to find developers of banking software.

Embedding models evaluation

We use the feedback dataset created by the users actions of liking/disliking fields of the explanation (section 6.2.4) to evaluate 30 embedding models and find their optimal cosine similarity threshold. Those models were chosen for being the most liked sentence transformers models on HuggingFace Model Hub [3].

Each model embeds the query and the field of every feedback then their cosine similarity is computed. Finally, we train a linear regression model to predict if the pair is liked or disliked using only the cosine similarity. For the keyword algorithm, we use the number of occurrences in the field of keywords extracted from the query instead of cosine similarity. It is essentially a binary classification on a single parameter. We can determine the optimal threshold to use with our embedding algorithm (section 6.2.3) by looking at the weight and bias of the linear model. Performances all the evaluated models can be found in table 6.2 along with our keyword algorithm, and our original model with our original threshold. We went from a configuration a 47.9 F1-score which performs worse than the keyword algorithm (68.0), to a configuration with a 76.5 F1-score.

6.2.5 Bookmarks

Users can bookmark consultants to specify who they wish to hire. It is done via a button as seen in figure 6.2. Managers use those bookmarks to assign consultants to project leaders. There are seven types of bookmarks :

- **Priority** : the user wants the consultant absolutely.
- **Go** : the user wants the consultant but it is not critical.
- **Interesting** : the user is interested in the consultant.
- **Why not** : the user is not so interested in the profile, but since the company wants them to hire every available consultant, he could envisage taking him if nobody else has a job for him.

3. Link to the most liked sentence transformers models on Hugging Face : `https://huggingface.co/sentence-transformers?sort_models=likes#models`

TABLE 6.2 – Evaluation of embedding models. Results are sorted from best F1-score to worst. We add the keyword algorithm and the original model with its original threshold both indicated by a * for comparison. This evaluation is detailed in section 6.2.4.

Model	F1-score	Accuracy	Precision	Recall	Threshold
multi-qa-MiniLM-L6-cos-v1	**76.5**	**80.0**	81.3	**72.3**	38.2
all-MiniLM-L12-v2	74.8	78.7	79.8	70.4	41.1
paraphrase-MiniLM-L6-v2	72.5	76.7	77.3	68.2	44.2
distilbert-base-nli-stsb-mean-tokens	72.2	76.7	77.7	67.4	49.5
all-MiniLM-L6-v2	71.5	76.5	78.8	65.5	42.2
LaBSE	71.2	75.5	75.4	67.5	42.4
multi-qa-mpnet-base-dot-v1	71.2	75.9	77.1	66.0	53.7
msmarco-distilbert-dot-v5	70.2	75.2	76.4	64.9	79.5
paraphrase-albert-small-v2	70.2	75.2	76.2	65.1	43.1
paraphrase-mpnet-base-v2	70.2	73.4	70.8	69.6	53.3
stsb-bert-base	69.7	74.8	75.7	64.6	50.1
paraphrase-MiniLM-L3-v2	68.9	74.7	76.8	62.5	39.1
all-mpnet-base-v2	68.9	74.1	75.0	63.7	43.6
paraphrase-multilingual-mpnet-base-v2	68.3	72.1	69.9	66.8	56.8
*keyword	68.0	74.4	77.7	60.4	56.2
all-roberta-large-v1	66.6	72.4	72.9	61.3	42.3
nli-mpnet-base-v2	66.5	72.6	73.7	60.6	53.1
paraphrase-distilroberta-base-v2	66.4	70.4	67.8	65.0	42.7
all-distilroberta-v1	65.4	71.9	73.2	59.1	38.1
roberta-base-nli-stsb-mean-tokens	64.8	71.0	71.3	59.4	47.0
paraphrase-distilroberta-base-v1	64.3	71.2	72.6	57.7	41.2
distiluse-base-multilingual-cased-v1	60.0	65.9	63.4	57.0	44.4
distilbert-multilingual-nli-stsb-quora-ranking	55.1	61.5	57.9	52.5	84.9
xlm-r-bert-base-nli-stsb-mean-tokens	52.4	61.2	58.3	47.6	63.7
bert-base-nli-mean-tokens	52.2	61.5	59.1	46.6	75.7
paraphrase-multilingual-MiniLM-L12-v2	51.8	57.7	53.1	50.5	62.4
allenai-specter	51.0	60.5	57.7	45.7	84.6
*original (paraphrase-MiniLM-L6-v2)	47.9	68.0	**89.2**	32.8	60.0
paraphrase-xlm-r-multilingual-v1	47.1	59.1	56.0	42.1	55.7
distiluse-base-multilingual-cased-v2	43.0	57.0	52.8	40.8	51.8
distiluse-base-multilingual-cased	43.0	57.0	52.8	40.8	51.8
distilbert-base-nli-mean-tokens	42.2	60.3	60.1	35.1	79.7

— **Not interesting** : the user is not interested in the consultant and does not have a job for him.
— **Out of scope** : the profile is unrelated to the user's activities.
— **Missing information** : there is not enough information in the consultant's profile to decide.

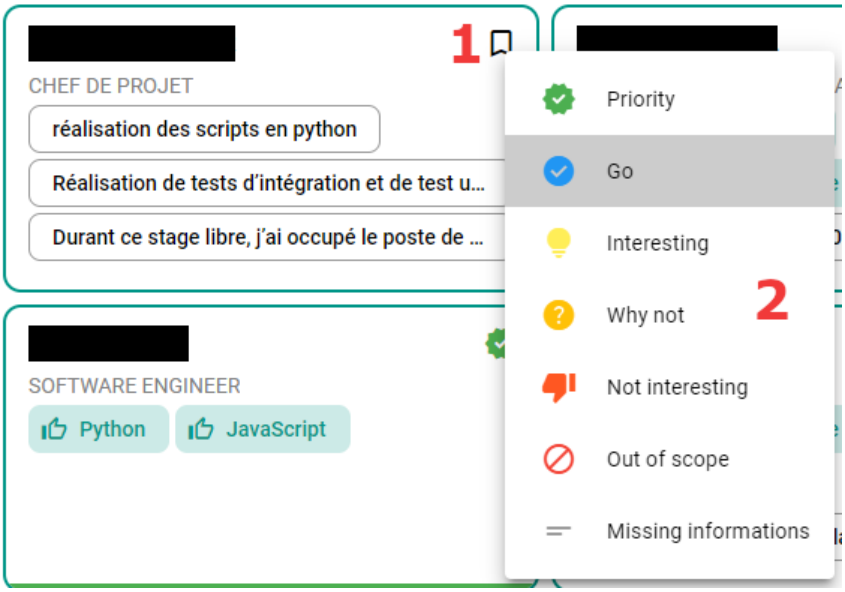


FIGURE 6.2 – Consultants have a bookmark button (1) located at the top right corner of their card. Clicking this button opens a menu (2) that contains the list of possible bookmarks (see section 6.2.5 for more details). Clicking a button on this menu will bookmark the consultant accordingly.

A user can change (e.g. from "Priority" to "Go") or delete a bookmark at any time. For our research, we always register an event when a user creates, changes, or deletes a bookmark. This event stores the user, the consultant, the bookmark type, the moment in time, and the card's location in the tool. If its location is the explore panel, we know it comes from a recommendation, and we register the interest leading to the recommendation along with the explanation. If it comes from the search panel, we register the search query if there is one. Those events are part of implicit feedback produced by users' behaviors.

We qualify "Priority", "Go", "Interesting", and "Why not" bookmarks as "positive bookmarks". They indicate that the user wishes to hire the consultant (the managers make the final decision based on all the bookmarks made during the process). It is similar to buying a product or consuming content (e.g. watching a video or reading an article) in other recommender systems.

We qualify "Not interesting", and "Out of scope" bookmarks as "negative bookmarks". They indicate that the user does not wish to hire the consultant. A "negative bookmark" does not always denote a bad recommendation. The user may reject the consultant because he already has enough consultants to work on his projects. He cannot manage an infinite number of consultants.

"Missing information" is a special case. It often indicates that the consultant's resume is missing or not filled out properly. The tool never recommends those consultants. The "Missing information" bookmark is useful so a user can indicate to others that there is no point in opening the consultant's resume (it is empty). It is a form of collaborative filtering. Also, it indicates to the managers that they need to manually provide the consultant's resume.

6.3 Experiments

In this section, we evaluate how AI4HR Recruiter performs and analyze how users effectively use it. The results presented here are based on 6 months of active usage, during which the tool has seen :

— 61 users (not all active recruiters).
— 1,700 consultants.
— 9,000 bookmarks.
— 3,500 explicit feedbacks.
— 69,000 implicit feedbacks.

The number of users corresponds to the total number of people with an account. Some are not even allowed to access the tool : account creation happens during the first connection, but it cannot access until an administrator approves it. Some users are inactive, some are recruiters, some are managers, and some are people who need access to present the tool to others inside the company. For each experiment, we specify the exact number of active users involved.

We start by looking at users' preferences (interests), how many they have, and how they are formatted (section 6.3.1). Then we look at the time users spend on the tool (section 6.3.2). And finally, we performed two cross-over trials, one on the impact of the explanation (section 6.3.4) and one on the differences between the keyword and the embedding

recommender system in terms of performance (section 6.3.4).

6.3.1 Analysis of users' preferences

For 23 users we have a total of 268 interests. The average number of interest an user have is 11.65, the median is 11, the minimum is 4 (so every user filled their interests), and the maximum is 24. The distribution can be observed on the left plot of Figure 6.3. An interest has an average length of 1.4 words, the median is 1, the maximum is 7. The distribution of the interests length can be observed on the right plot of Figure 6.3.

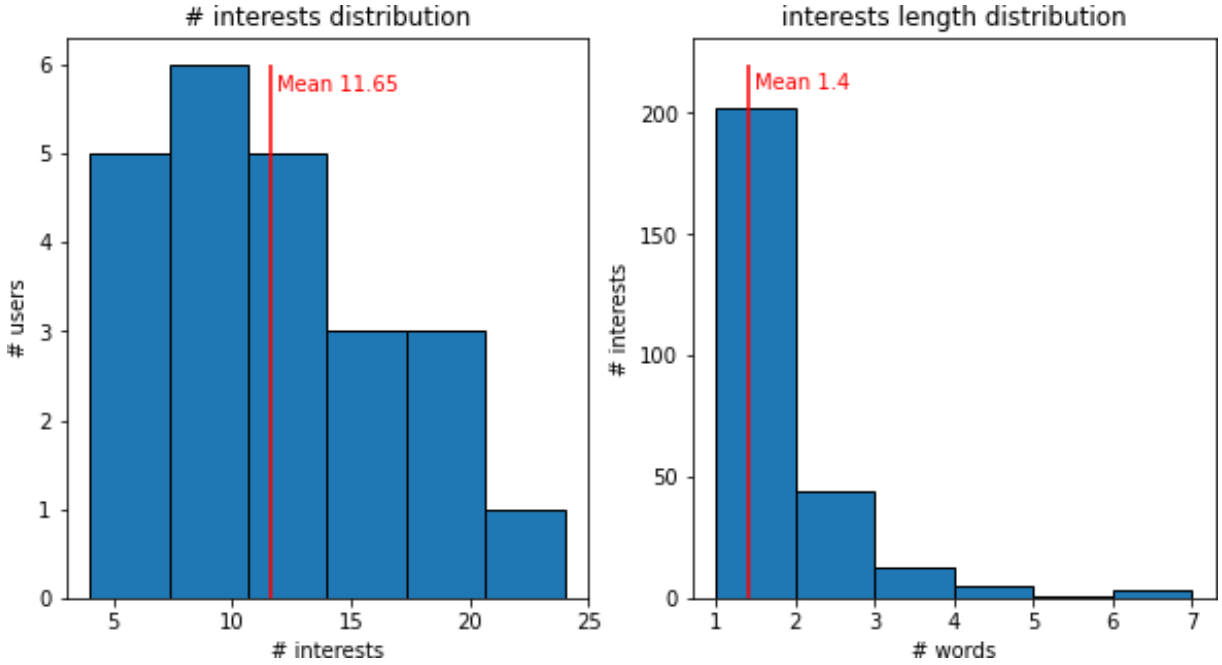


FIGURE 6.3 – Distributions of the number of interest per user and the interests length (in words)

The most common interest are : "Python" with 14 appearances, "Java" with 5 appearances, "PMO" with 4 appearances, and "ROS" with 4 appearances. The computation of the number of appearances is case insensitive : For example "DevOps" is written 3 different ways : "DevOps", "devops", and "Devops" (the official case is "DevOps" [4]). If an interest is included in another interest but is not alone, it doesn't count as an appearance. For example, "java" is included in the interest "java eclipse" but it is not counted as a "java" appearance.

4. https ://en.wikipedia.org/wiki/DevOps

6.3.2 Time spent on AI4HR Recruiter

We first look at user events during a Mercato. The average duration between the first event of a user and its last is 1 hour 2 minutes and 15 seconds, but it is not representative of the time spent in the tool. Some users perform the Mercato in one go without interruption, while others do it in multiple sessions. Those users may or may not close AI4HR Recruiter between sessions. We also notice a few other behaviors that are not exclusive. For example, doing a quick check at the end of a Mercato or being interrupted at the start of a session : the user opens the tool, barely does anything then comes back much later.

To get a better idea of the time spent on the tool, we look at the duration between two consecutive events. We find a mean duration of 1 minute and 19 seconds. The maximum duration between two consecutive events is more than 6 hours. We are confident users do not look at the same resume for that long. So we arbitrarily define a maximal duration between events of 3 minutes. If the duration between an event and the next is longer than 3 minutes, the user took a pause and it is now a different session. Using this method, we find a mean of 2.87 sessions per Mercato per user, with an average session duration of 4 minutes and 51 seconds and an average total duration of 13 minutes and 56 seconds. Those times may seem short, but users are experienced with the standardized resume format used by ALTEN, hence knowing where to look.

We want to estimate the time saved by AI4HR Recruiter. MPL (Mon Petit Lab), the predecessor of AI4HR Recruiter, did not include consultants resumes. The user had to open them in another tool. On average it takes 7.90 seconds between clicking a resume link and being able to read it while it is instantaneous on AI4HR Recruiter. Using AI4HR Recruiter data, we find the average number of consultant per mercato to be 42.27, and the average number of resume opening per consultant to be 0.78 (not all user will open every resume, and the same resume can be opened multiple time). So we can compute an average time saved of 4 minutes and 21 seconds. The computation is visible in Figure 6.4. The time saved represents 31% of what is currently spent on AI4HR Recruiter.

We also asked users : "how do you estimate the time spent doing your mercato on AI4HR Recruiter compared to MPL ?". Their answers are :

- 14 "I spend less time on AI4HR Recruiter than MPL."
- 3 "I spend more time on AI4HR Recruiter than MPL."
- 3 "I spend the same time on both."

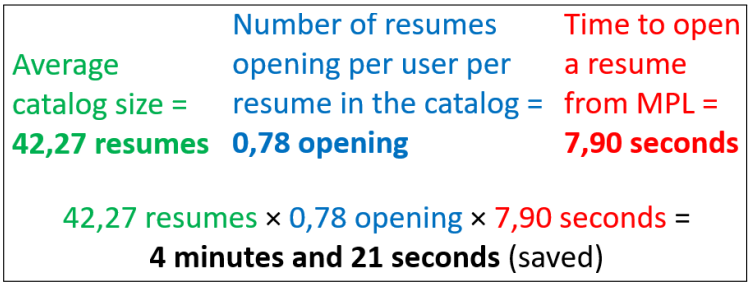


FIGURE 6.4 – Estimation of time saved by being able to instantaneously open resumes in AI4HR Recruiter compared to the previous tool.

TABLE 6.3 – Results from the comparison between MPL (previous tool) and AI4HR Recruiter (our tool)

Tool	Avg # Users	Wishes per user	Chosen consultants per user
AI4HR Recruiter	14.5	**6.28**	**1.76**
MPL	22.5	2.36	1.28

We can conclude without a doubt that, in terms of efficiency, our tool represents a significant improvement to the Mercato business process. We attribute this improvement to regrouping more features inside a faster tool and providing users with recommendations.

6.3.3 Business process improvements

The business process' goal is to engage a maximum of consultants on internal projects. To do so, managers need as many wishes from project leaders as possible. In our tool, a wish is essentially a positive bookmark ("why not" and upward), while in MPL (the previous tool), it was putting credits on the consultants. We analyze the process results on both tools during 4 weeks where they coexisted. The tools share a part of their user base. The results are visible in Table 6.3. We can see that at the time, there were more users on MPL than AI4HR Recruiter, but users from our tool express more wishes and want more consultants. The user base in MPL is higher because, at the time, both tools coexisted and people using AI4HR Recruiter had to report their wishes in MPL. More wishes help managers to make affectation, hence helping the business process. These figures show some quantitative improvements in the business process. Unfortunately, we lack sufficient data, especially related to the tool MPL, to do a more thorough analysis of these improvements.

6.3.4 Cross-over trials

What is a cross-over trial

A cross-over trial is a study in which each subject is his own control [27]. It is most commonly used in medicine because of physical characteristics differences between patients which can impact the effect drugs or other medical care can have. The main advantages of cross-over trials compared to A/B testing of parallel-group trials are that it requires fewer subjects to obtain the same number of observations, and it removes the impact differences between groups can have on the results [54].

It is relevant to use cross-over trials in our situation because the number of active recruiters on AI4HR Recruiter is limited (about 20), and differences in behavior can have a significant impact on metrics such as the number of resumes opened or average time spent on a resume.

There are a few disadvantages associated with cross-over trials [54]. First, there is the problem of dropouts. Because the trial takes at least twice as long as an A/B testing trial, the risk of having subjects leaving the study is higher. However, because the use of AI4HR Recruiter is mandatory for recruiters to perform their job, this risk in our case is greatly reduced. The second disadvantage is that cross-over studies are not suited for conditions that can significantly impact patient health, in our case we can disregard this issue. Third, there is the issue of treatment interaction and persistence in their effect. In our case, this could manifest in the form of users changing their behavior or interests because of the algorithms or parameters we study. Those changes would carry over between stages of the trial. Finally, results can be difficult to analyze, especially with studies that include three or more treatments. In our case, only two parameters/algorithms are compared per study to avoid such difficulties.

Two cross-over trials are performed, one on the impact of the explanation and one on the differences between the keyword and the embedding algorithm in terms of performance. The specifics are discussed in the following section.

Experimental Protocol

To ensure that results are not too heavily affected by the restrained pool of users and ever-changing catalog, a cross-over trial is carried out. Our user population and trial period are both split in two. A group will test one feature during the first part of the trial, while the other group can test a second feature we wish to compare with the first. Then we swap both groups during the second part of the trial. So each group is its own control group, and doing so should ensure results when comparing two features are independent of the groups and catalog compositions.

We also employ the ResQue evaluation framework from Pearl Pu et al. [43] which evaluates the experience a user has with a Recommender System. This framework is composed of 31 questions (e.g. "I became familiar with the recommender system very quickly.") spread through 15 constructs (e.g. "Perceived Ease of Use"). Users answer questions using a 5-point Likert scale ranging from "strongly disagree" (1) to "strongly agree" (5). Following the authors' advice, we ask only one question per construct to make the form faster to fill (questions inside the same construct are strongly correlated).

Explanation

For this experimentation, we are going to look at :

— Efficiency, using the average time a user spends on a resume.
— Trustworthiness, using the ResQue framework [43].
— Satisfaction using both ResQue and asking the question inside of the tool.

We use the keyword algorithm (Section 6.2.3). This algorithm is easy to explain. The explanation shows the user which fields contain occurrences of tokens from the query and highlights them in the document.

Thirteen users participated in both weeks. The overall time spent on resumes is significantly higher during the second week, regardless of the groups. The main factor is probably the second week having more consultants. Because of it, recruiters could tire during the process and take more time. Also, a user notified us the tool got slow (the server lacked memory, we solved the issue by increasing memory from 4GB to 8GB). The increase in time is an issue because the composition of the groups is unbalanced (there are more people in one group than the other). It makes comparing times between weeks compli-

TABLE 6.4 – Results from our experimentation on the impact of an explanation in a Recommender System.

	Explanation	Active Users	Average Time (s)
Week 1	With	4	17.72
	Without	10	29.25
	Any	14	25.96
Week 2	With	13	26.80
	Without	5	45.70
	Any	18	32.05

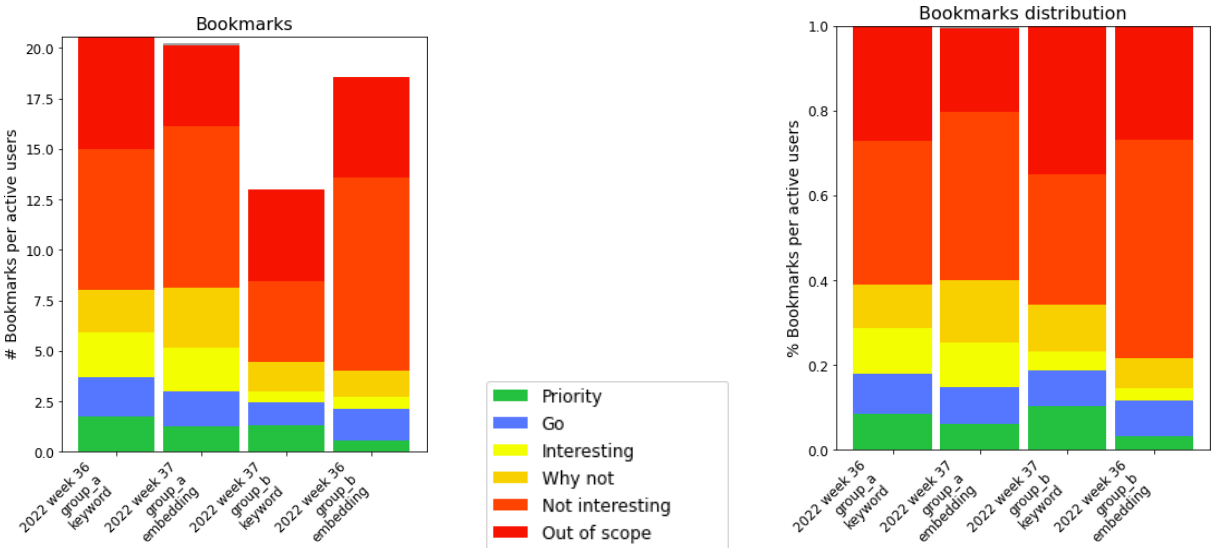
cated. To resolve this, we normalize the times of the second week. We normalize them using the average time per group : each group will contribute to half of the average time spent on a TD regardless of the group size (Equation 6.1). So the second week times are normalized using the equation 6.2.

$$pondered\_average = \frac{\sum times\_with\_explanation}{|times\_with\_explanation| \times 2} + \frac{\sum times\_without\_explanation}{|times\_without\_explanation| \times 2} \tag{6.1}$$

$$normalize(t) = t \times \frac{pondered\_average_{week\_1}}{pondered\_average_{week\_2}} \tag{6.2}$$

We compare the average view time with and without the explanation of each user who participated in both weeks. Whichever of these times belongs to the second week is normalized as explained above. We find that, on average, they spent 20.1% less time viewing resumes with an explanation. It is an improvement in terms of efficiency.

We now look at the results of the ResQue framework. 14 users answered the quiz for the first week, 9 for the second, and 7 of those answered both weeks. Using Student's t-test we determine there is no significant statistical difference for any questions between the groups. The results are similar for a t-test with paired samples using only the 7 users who answered both weeks. So despite a significant time gain, users did not perceive any meaningful difference. However, the number of users who answered the quiz in both weeks may not be enough. It was difficult to have them answer a long quiz outside of the tool, and



(a) Number of bookmarks by mercato and by group

(b) Legend

(c) Percentage of bookmarks by mercato and by group

FIGURE 6.5 – Number/Percentage of bookmarks per type (color, the legend is the same for both graph, but only shown in the first), by mercato (year and week), by group (composition of groups is the same across the experiment) and by algorithm (the algorithm depends of the mercato and the group).

it required asking them over email many times. We recommend adding a single question inside the tool they can quickly answer when they wish to.

Recommender Systems Comparison

During two weeks we split our pool of users in two groups of 13 people, one with the keyword algorithm and another with the embedding one. Users with the keyword algorithm during the first week switched to embedding during the second week and vice versa. The algorithm an user has is hidden to him. Because the tool is important to ALTEN functioning, users have the possibility to opt out of the experiment at any point. None chose to do so.

To evaluate our models we want to compute their precision, recall, accuracy and f1-score. To do so we need true positive, false positive, true negative and false negative. Here is how we proceed :

— **True Positive :** number of consultants recommended and bookmarked positively.

— **False Positive :** number of consultants recommended and bookmarked negatively,

TABLE 6.5 – Comparison of the performances of Recommender Systems. F1 stands for F1-score, Acc. for Accuracy, Pre. for Precision, and Rec. for Recall.

Week	Group	Algorithm	F1	Acc.	Pre.	Rec.
36	A	keyword	50.4	46.8	35.1	89.5
37	A	embedding	46.8	41.9	30.7	98.6
37	B	keyword	40.2	55.8	27.2	76.7
36	B	embedding	34.7	44.8	21.1	97.2

or clicked but not bookmarked.

— **True Negative :** number of consultants not recommended and bookmarked negatively, or clicked but not bookmarked.

— **False Negative :** number of consultants not recommended and bookmarked positively.

Positive bookmarks are : "Priority", "Go", "Interesting" and "Why not". Negative bookmarks are : "Not interesting" and "Out of scope". Bookmarks "Missing information" are considered neutral hence ignored. It is important to consider consultants that have been clicked but not bookmarked, because users do not take the time to negatively bookmarks consultant they are not interested in. For example between 8% and 19% (depending of week and group) of False Positive are consultants that have been recommended and clicked but not bookmarked. The results of this analysis are visible in Table 6.5.

We notice that the keyword Recommender System has a better F1-score, Accuracy and Precision than the embedding one, the latter has a better recall. Recall that is important because a goal of the mercato is to find a mission for a maximum of consultant. The precision may seem very low, but this is because we do not take into account consultants who are not positively bookmarked because the user is not interested in them right now. A user can find a consultant interesting but not bookmark it positively either because he already has someone with a similar profile, or has reached the limit of consultants he can effectively manage at the same time.

In the interest of the mercato and having a maximum of consultants hired, a higher recall is preferred. But in the interest of users and spending less time on the mercato a higher precision is preferred. From this experience alone we cannot declare which system is better, it depends on the perspective.

Now we look at the number and distribution of bookmark types. Those data can be

viewed in Figure 6.5. We use the number of active users in the group during the corresponding week to normalize the data. An active user is a user who bookmarked at least one consultant. For the first week, we have 12 active users for group A (keyword) and 7 for group b (embedding). And for the second week, we have 8 for group A (embedding) and 7 for group B (keyword). The keyword algorithm results in more "priority" bookmarks, but other types seem more dependent on the group composition. Group A uses bookmarks "interesting" and "why not" more than Group B, regardless of the algorithm. This difference between groups highlights the importance of using a cross-over trial. We do not detect any major difference linked to the algorithm apart from a higher number of positive bookmarks for the keyword algorithm, which is correlated to our measure of Precision and has already been highlighted previously.

We also asked the users : "do you like the recommendation by embedding ?". Their answers are :

— 14 "Yes."
— 2 "No."
— 7 "I do not use it."

The question and its answers do not allow the expression of a preference between keyword and embedding, but most recruiters who use the embedding algorithm enjoy it.

6.4 Conclusion

In this chapter we presented a recommendation tool, AI4HR Recruiter that is intended to improve the internal recruitment process of a consulting company. The AI4HR Recruiter relies on content-based recommendation techniques and helps to identify consultants who are relevant to the user's (i.e. recruiter's) needs. The tool is in use at a consulting company and it is used in the context of real internal recruitment. We have evaluated our tool through a number of experiments that indicate improvements in the recruitment process. The recommendations are complemented with explanations that shed light on why a specific consultant has been recommended. The high recall of the embedding-based recommendation helps to achieve the company's goal to provide project for each available consultant. Both the users of the tools and the managers where satisfied with the improvements of the related business processes.

In the future, we want to ask more questions to our users through the tool. It is the most effective way to obtain answers. We want to improve the precision of our recommender systems. This could be achieved by leveraging the ever-growing dataset of explicit feedbacks from the recommendation to train a model instead of relying only on the cosine similarity of embeddings. Also, we want to propose recommendations based on the previous bookmarks of the user.

Chapitre 7

Conclusion and future work

The overall objective of this thesis was to thoroughly investigate the usability of explainable job recommender systems within large consulting companies, with ALTEN serving as a case study. The primary aim was to shed light on the benefits and potential drawbacks of this technology, and to contribute to the advancement of both scientific and industrial research in the field.

Throughout this thesis, we have addressed several key research questions and conducted in-depth investigations into the effectiveness of explainable job recommendations, user perception and engagement, and the utilization of user feedback to improve the system. In this concluding chapter, we aim to provide a comprehensive summary of our research and its implications in the field of job recommender systems. First, we recapitulate our research questions, their answers, and our contributions. Then we reflect on our methodology, highlighting its upsides and potential limitations. Additionally, we outline avenues for our future research. Finally, we conclude with final reflections on the significance and impact of our work in advancing the understanding and application of job recommender systems.

7.1 Research Questions Answers and Contributions

This section recapitulate our research questions, their answers, and the contributions made to answer them.

Q1. How can we effectively utilize the ESCO ontology or text embeddings to develop an Explainable Job Recommender System that is well-suited for a context with a wide array of highly specialized skills ? Throughout the document, we explored different approaches to answer this question. In Chapter 4, we utilized the

ESCO ontology by training a BERT-based skill classifier and building a recommender system based on ontology relationships. While the system demonstrated promising results and showed the advantages of explainability and interactivity, we concluded that ESCO is not precise enough to meet the needs of a company which requires a wide array of highly specialized skills such as ALTEN. However, the approach using embeddings in Chapter 6 proved to be a more suitable option. Leveraging publicly available models and user feedback, the embedding-based technique successfully captured semantic relationships and became a fundamental component of an industrial recruitment platform used by ALTEN. The contributions made to answer this questions are as follow :

Firstly, in Chapter 3, we introduced a novel methodology for constructing a Job Matching Dataset solely using resumes. The motivation was to address the lack of publicly available datasets in the field of Job Recommender System, and the labor-intensive nature of manual annotation. This methodology, outlined in our publication at KaRS [36], offers an approach that requires only a set of resumes and a means of identifying professional experiences. By making dataset creation more accessible and cost-effective, we hope to facilitate research and evaluation of models within the field of Job Recommender Systems.While our method exhibits advantages, such as ease of replication, it does have limitations,including the potential for false negatives and the absence of semantic structure in job offers.

Moving on to Chapter 4, we presented our work on developing a Conversational Job Recommender System based on the ESCO ontology, as featured in our publication at KaRS [36]. To accomplish this, we trained a BERT-based skill classifier to map skills extracted from ALTEN resumes into ESCO, enabling the use of the ontology relationships between skills for producing explainable recommendations. This contribution is particularly valuable for those interested in constructing a Job Recommender System using an ontology, training an ontological classifier, or working with the ESCO ontology. However, our research led us to the conclusion that ESCO may not be the most suitable fit for ALTEN's specific activities due to its lack of precision with engineering skills. This finding can assist others in assessing whether ESCO aligns with their own contexts and potentially motivate the development of more suitable ontologies.

Then, in Chapter 5, we built a BERT-based Deep Learning model that classifies ALTEN missions into an internal ontology. This is mostly an industrial contribution, the model has been used for 3 years already, and helped classify thousands of missions. To train

the model we came up with a methodology that use data from outdated versions of the ontology, and help the model generalize. This method should be useful to anyone looking to train a Deep Learning classifier for an evolutive referential. This contribution does not directly help to answer the research question, but build upon our work on deep ontological classifiers started with ESCO.

Lastly, in Chapter 6, we developed a job recommendation platform for ALTEN to evaluate an embedding-based Job Recommender System. The success of the system led us to the conclusion than embedding is better suited to our use-case than the ESCO ontology, and we used this platform to answer our other questions.

Q2. What is the impact of providing explanations for the recommendations produced by the job recommender system on the efficiency of the screening process ? Chapter 6 addressed this question through a cross-over trial conducted in the recruitment platform we developed. The presence of explanations significantly reduced the time taken by users to make decisions on consultant suitability for available missions. This finding highlights the positive impact of providing explanations in enhancing the efficiency of the screening process within a recommender system-based recruitment context.

Q3. What is the impact of the presence of explanations on the quality of the user experience ? Also examined in Chapter 6 through a cross-over trial in the same platform, the impact of explanations on the quality of the user experience was assessed using a survey based on ResQue [43]. The results were inconclusive, as various factors, such as the form of the explanation, may have influenced the outcomes. Further improvements or deteriorations in user experience could potentially be observed if the explanation played a more significant role in the user assessment of the recommendation. Based on these results, it is suggested that resources would be better invested towards other areas of the recommendation platform for enhancing user experience.

Q4. How can a job recommender system practically utilize explanations to collect more precise user feedback, and how can this feedback be leveraged to effectively improve the system ? To what extent did users engage with the feedback system, and how significant was the feedback in improving the system ? Chapter 6 addressed this question within the context of the AI4HR Recruiter platform. We implemented a straightforward approach to provide explanations and gather user feedback. Explanations consisted of up to six fields from the resume that closely

matched the query and led to the recommendation. Users could provide feedback by liking or disliking each field independently. The collected feedback has different use, such as highlighting liked fields in the interface, excluding disliked fields from the recommendation process, and utilizing all fields to evaluate embedding models and determine the optimal similarity threshold. Over the course of a year and a half, our users provided over ten thousand feedbacks, showcasing their high level of engagement in the process. Ultimately, the use of these feedbacks resulted in a remarkable 60.7% improvement in the F1-score of the embedding-based recommender system. These findings emphasize the value of user feedback in recommender system improvement and the high level of engagement users exhibited in the feature.

7.2 Reflection on Methodology

In this section we reflect on the methodology employed in our study, discussing its strengths and limitations.

In Chapter 3, we proposed an automated process for constructing a job matching dataset, which offered notable advantages such as eliminating the need for costly manual annotation. However, we also acknowledged potential limitations, such as the possibility of false negatives. To measure the quality of the dataset created, it would have been valuable to involve recruiters in manually annotating a small subset of the generated pairs.

In Chapter 4, we developed a conversational recommender system based on the ESCO ontology. While the system was designed for interaction, we faced limitations in terms of a user base and resource constraints that prevented the creation of a full-fledged tool with a user interface. Consequently, we resorted to simulating human-machine interactions to evaluate the system. However, it is important to note that this methodology cannot fully replicate human behavior, and therefore, the results should be interpreted with caution. Integrating the system into the AI4HR Recruiter platform would have been an interesting avenue to explore, but due to the incompatibility of ESCO with ALTEN's context, the experimentation was halted.

In Chapter 6, we introduced the AI4HR Recruiter platform, which was specifically developed to conduct in-vivo experiments and address ALTEN's internal recruitment needs. This platform provided a valuable opportunity to engage with real users in a realistic

setting. However, it is worth noting that the user base remained relatively limited. To mitigate this challenge, we implemented cross-over trials to address issues arising from group variance or environmental changes. While our first experiment encountered the issue of unbalanced groups, this can be rectified during the results analysis phase. We strongly advocate for the use of cross-over methodology when conducting studies involving human participants, as it is a widely accepted standard in fields like medicine but is still underutilized in computer science. Additionally, we encountered difficulties in obtaining survey responses related to user experience, while we did not face the same issue when directly incorporating questions within the platform.

By critically assessing our methodology and highlighting its strengths and limitations, we aim to provide transparency and contribute to the ongoing conversation on research methodology in the field. These reflections serve as valuable insights for future research endeavors, facilitating the adoption of robust methodologies and improving the reliability of findings.

7.3 Future Research

In this section, we outline potential research avenues that could make significant contributions to the field of job recommender systems.

Firstly, the field is greatly affected by the absence of public datasets that could serve to train and benchmark models. It would benefit from the creation of a dataset that is manually annotated by recruitment professionals and of appropriate size to support the training of Deep Learning Neural Networks. Such a dataset could consist of a collection of resumes paired with job offers, annotated with the relevance of each resume to the corresponding job offer, accompanied by explanations. The two main obstacles to such a dataset are the competitiveness of the recruitment industry and the privacy concerns. To tackle the first issue, the dataset creation should be publicly funded, which is doable considering the various efforts countries and international agencies are doing to ensure high employment rates. For example, ESCO is funded by the European Union, and designed to help people find jobs internationally. The privacy issue however, would require extensive anonymization work. This is a difficult task because resumes contain many key information about people life such as their education, career path, occasionally their hobbies, etc. Noise

could be added, but it would decrease the data quality. An alternative method would be to create synthetic yet realistic data using statistic methods. Also, the dataset could contains demographic data about the candidates, such as gender or ethnicity, greatly facilitating bias studies but aggravating the privacy concerns. This type of dataset would serve as a catalyst for high-quality and more generalizable research in the field of job recommender systems.

The ESCO ontology currently stands as one of the most comprehensive ontologies available for the labor market, encompassing numerous occupations, skills, and qualifications across various domains. However, it lacks the necessary granularity and specificity to be effectively utilized in a Job Recommender System focused on engineering fields. There is a clear need for a more detailed ontology. Moreover, to ensure its long-term usefulness, such an ontology should be designed with evolution in mind, considering the constant emergence of new skills, tools, and occupations. Developing such an ontology and establishing a framework for its maintenance would constitute a significant contribution to multiple fields, including job recommender systems.

The field of explainable recommender systems would greatly benefit from a systematic and extensive study focusing on the format of explanations. This study could involve categorizing explanations based on various features, such as completeness, ad-hoc or post-hoc nature, and textual or visual format. Furthermore, it could explore the compatibility between different types of explanations and recommender systems. For example, systems that fully rely on Deep Learning may only produce post-hoc explanations, or systems based on semantic similarity using embeddings may only produce partial ad-hoc explanations (we can tell that two textual entities are semantically similar based on their embeddings, but we cannot explain why the model came up with those specific embeddings). By measuring the positive and negative impacts associated with each type of explanation, valuable insights could be gained. Additionally, investigating the types of user feedback elicited by different forms of explanation and exploring how this feedback could be leveraged to enhance recommender systems would be an important research endeavor.

Addressing these problems can contribute to advancing the field of job recommender systems, enhancing the quality and reliability of research outcomes, and furthering our understanding of explainable recommender systems. These areas offer rich opportunities for future research and hold significant potential for impactful contributions.

7.4 Conclusion

In summary, this research has investigated the usability and potential implications of explainable job recommender systems, with a focus on large companies using ALTEN as a case study. The significance of this study lies in its contribution to the scientific and industrial research in the field of explainable job recommender systems.

The main contributions and outcomes of this research can be summarized as follows :

1. **Methodology for Building a Job Matching Dataset :** A methodology was proposed in Chapter 3 to automatically construct a job matching dataset using resumes. This approach addressed the lack of publicly available datasets in the field of job recommender systems. While the methodology demonstrated strengths such as cost-effectiveness, it also acknowledged limitations, including the potential creation of false negatives. Future research could involve obtaining manual annotations from recruitment professionals to evaluate the accuracy of the dataset creation process, or even manually annotating an entire dataset that could be made publicly available to facilitate further studies in the field.
2. **Conversational Job Recommender System based on the ESCO Ontology :** In Chapter 4, a conversational job recommender system was developed using the ESCO ontology. Although the system showcased promising results in terms of explainability and interactivity, the system was not evaluated in-vivo, and the ESCO ontology was found to lack the necessary precision for ALTEN's engineering-focused context. This research emphasized the need for an ontology that aligns more closely with the specific requirements of engineering fields, while proposing a job recommender system than can efficiently utilize such ontology.
3. **AI4HR Recruiter Platform** : The development of the AI4HR Recruiter platform in Chapter 6 provided a practical means for conducting in-vivo experiments. Our findings emphasized the importance of explanations in improving process efficiency and gathering user feedbacks which are invaluable for enhancing recommendations accuracy.

This research highlights the need for publicly available datasets in the field of job recommender systems, as well as the importance of ontology design and maintenance to ensure their relevance in specific contexts. The study also emphasizes the positive impact of explanations in improving the efficiency of the screening process and the potential for

user feedback to enhance system accuracy. These findings have practical implications for large companies like ALTEN, offering opportunities to optimize recruitment processes and reduce time-to-hire.

In conclusion, this research has made notable contributions to the field of explainable job recommender systems. It has shed light on the significant efficiency gains that can be achieved through useful recommendation explanations, and the importance of user feedback to improve the system. Moving forward, future research can build upon these findings and explore additional avenues, such as studying explanation formats, further investigating the impact of user feedback, and continuing to advance the field of explainable job recommender systems.

Annexe A

Appendix

A.1 Translation of Leonardo Da Vinci "Resume" letter to Ludovico Sforza Duke of Milan

Translation of Leonardo Da Vinci "resume" letter to Ludovico Sforza Duke of Milan (Source : openculture.com). The letter can be seen in Figure 1.1.

Most Illustrious Lord, Having now sufficiently considered the specimens of all those who proclaim themselves skilled contrivers of instruments of war, and that the invention and operation of the said instruments are nothing different from those in common use : I shall endeavor, without prejudice to any one else, to explain myself to your Excellency, showing your Lordship my secret, and then offering them to your best pleasure and approbation to work with effect at opportune moments on all those things which, in part, shall be briefly noted below.

1. I have a sort of extremely light and strong bridges, adapted to be most easily carried, and with them you may pursue, and at any time flee from the enemy ; and others, secure and indestructible by fire and battle, easy and convenient to lift and place. Also methods of burning and destroying those of the enemy.

2. I know how, when a place is besieged, to take the water out of the trenches, and make endless variety of bridges, and covered ways and ladders, and other machines pertaining to such expeditions.

3. If, by reason of the height of the banks, or the strength of the place and its position, it is impossible, when besieging a place, to avail oneself of the plan of bombardment, I have methods for destroying every rock or other fortress, even if it were founded on a rock, etc.

4. Again, I have kinds of mortars; most convenient and easy to carry; and with these I can fling small stones almost resembling a storm; and with the smoke of these cause great terror to the enemy, to his great detriment and confusion.

5. And if the fight should be at sea I have kinds of many machines most efficient for offense and defense; and vessels which will resist the attack of the largest guns and powder and fumes.

6. I have means by secret and tortuous mines and ways, made without noise, to reach a designated spot, even if it were needed to pass under a trench or a river.

7. I will make covered chariots, safe and unattackable, which, entering among the enemy with their artillery, there is no body of men so great but they would break them. And behind these, infantry could follow quite unhurt and without any hindrance.

8. In case of need I will make big guns, mortars, and light ordnance of fine and useful forms, out of the common type.

9. Where the operation of bombardment might fail, I would contrive catapults, mangonels, trabocchi, and other machines of marvellous efficacy and not in common use. And in short, according to the variety of cases, I can contrive various and endless means of offense and defense.

10. In times of peace I believe I can give perfect satisfaction and to the equal of any other in architecture and the composition of buildings public and private; and in guiding water from one place to another.

11. I can carry out sculpture in marble, bronze, or clay, and also I can do in painting whatever may be done, as well as any other, be he who he may.

Again, the bronze horse may be taken in hand, which is to be to the immortal glory and eternal honor of the prince your father of happy memory, and of the illustrious house of Sforza.

And if any of the above-named things seem to anyone to be impossible or not feasible, I am most ready to make the experiment in your park, or in whatever place may please your Excellency – to whom I comment myself with the utmost humility, etc.

BIBLIOGRAPHIE

[1] Gediminas ADOMAVICIUS et Alexander TUZHILIN, « Toward the next generation of recommender systems : A survey of the state-of-the-art and possible extensions », in : *IEEE transactions on knowledge and data engineering* 17.*6* (2005), p. 734-749.

[2] Charu C AGGARWAL et al., *Recommender systems*, t. 1, Springer, 2016.

[3] Tom BROWN et al., « Language models are few-shot learners », in : *Advances in neural information processing systems* 33 (2020), p. 1877-1901.

[4] Daniel CER et al., « Semeval-2017 task 1 : Semantic textual similarity-multilingual and cross-lingual focused evaluation », in : *arXiv preprint arXiv :1708.00055* (2017).

[5] Li CHEN et Pearl PU, « Interaction design guidelines on critiquing-based recommender systems », in : *User Modeling and User-Adapted Interaction* 19.*3* (2009), p. 167-206.

[6] Kyunghyun CHO et al., « On the properties of neural machine translation : Encoder-decoder approaches », in : *arXiv preprint arXiv :1409.1259* (2014).

[7] Jan CHOROWSKI et al., « End-to-end continuous speech recognition using attention-based recurrent NN : First results », in : *arXiv preprint arXiv :1412.1602* (2014).

[8] KR1442 CHOWDHARY, « Natural language processing », in : *Fundamentals of artificial intelligence* (2020), p. 603-649.

[9] Junyoung CHUNG et al., « Empirical evaluation of gated recurrent neural networks on sequence modeling », in : *arXiv preprint arXiv :1412.3555* (2014).

[10] Jacob DEVLIN et al., « Bert : Pre-training of deep bidirectional transformers for language understanding », in : *arXiv preprint arXiv :1810.04805* (2018).

[11] Mihaela-Irina ENĂCHESCU, « A prototype for an e-recruitment platform using semantic web technologies », in : *Inform Econom.* 20.*4* (2016), p. 62.

[12] Dumitru Erhan et al., « Why does unsupervised pre-training help deep learning ? », in : *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop et Conference Proceedings, 2010, p. 201-208.

[13] Bruce Ferwerda, Kevin Swelsen et Emily Yang, « Explaining content-based recommendations », in : *New York* (2018), p. 1-24.

[14] John R Firth, « A synopsis of linguistic theory, 1930-1955 », in : *Studies in linguistic analysis* (1957).

[15] Mauricio Noris Freire et Leandro Nunes de Castro, « e-Recruitment recommender systems : a systematic review », in : *Knowledge and Information Systems* 63.*1* (2021), p. 1-20.

[16] Mauricio Noris Freire et Leandro Nunes de Castro, « A Framework for e-Recruitment Recommender Systems », in : *International Conference on Artificial Intelligence and Soft Computing*, Springer, 2020, p. 165-175.

[17] Mouzhi Ge, Carla Delgado-Battenfeld et Dietmar Jannach, « Beyond accuracy : evaluating recommender systems by coverage and serendipity », in : *Proceedings of the fourth ACM conference on Recommender systems*, 2010, p. 257-260.

[18] David Goldberg et al., « Using collaborative filtering to weave an information tapestry », in : *Communications of the ACM* 35.*12* (1992), p. 61-70.

[19] Alex Graves et Jürgen Schmidhuber, « Framewise phoneme classification with bidirectional LSTM and other neural network architectures », in : *Neural networks* 18.*5-6* (2005), p. 602-610.

[20] Nicola Guarino et Pierdaniele Giaretta, « Ontologies and knowledge bases », in : *Towards very large knowledge bases* (1995), p. 1-2.

[21] Akshay Gugnani et Hemant Misra, « Implicit skills extraction using document embedding and its use in job recommendation », in : *Proceedings of the AAAI Conference on Artificial Intelligence*, t. 34, 08, 2020, p. 13286-13293.

[22] Kaiming He et al., « Deep residual learning for image recognition », in : *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, p. 770-778.

[23] Melissa Heikkilä, *Dutch childcare benefits scandal*, mars 2022, url : `https://www.politico.eu/article/dutch-scandal-serves-as-a-warning-for-europe-over-risks-of-using-algorithms/`.

[24] Jonathan L Herlocker, Joseph A Konstan et John Riedl, « Explaining collaborative filtering recommendations », in : *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, 2000, p. 241-250.

[25] Sepp Hochreiter et Jürgen Schmidhuber, « Long short-term memory », in : *Neural computation* 9.*8* (1997), p. 1735-1780.

[26] Wenxing Hong et al., « A job recommender system based on user clustering. », in : *J. Comput.* 8.*8* (2013), p. 1960-1967.

[27] A Huitson et al., « A review of cross-over trials », in : *Journal of the Royal Statistical Society : Series D (The Statistician)* 31.*1* (1982), p. 71-80.

[28] Qilu Jiao et Shunyao Zhang, « A brief survey of word embedding and its recent development », in : *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, t. 5, IEEE, 2021, p. 1697-1701.

[29] Dor Lavi, « Learning to Match Job Candidates Using Multilingual Bi-Encoder BERT », in : *Fifteenth ACM Conference on Recommender Systems*, 2021, p. 565-566.

[30] Anna S Law et al., « A comparison of graphical and textual presentations of time series data to support medical decision making in the neonatal intensive care unit », in : *Journal of clinical monitoring and computing* 19.*3* (2005), p. 183-194.

[31] Quoc Le et Tomas Mikolov, « Distributed representations of sentences and documents », in : *International conference on machine learning*, PMLR, 2014, p. 1188-1196.

[32] Yann LeCun, Yoshua Bengio et Geoffrey Hinton, « Deep learning », in : *nature* 521.*7553* (2015), p. 436-444.

[33] Danielle H Lee et Peter Brusilovsky, « Fighting information overflow with personalized comprehensive information access : A proactive job recommender », in : *Third International Conference on Autonomic and Autonomous Systems (ICAS'07)*, IEEE, 2007, p. 21-21.

[34] Xiaodong Liu et al., « Multi-task deep neural networks for natural language understanding », in : *arXiv preprint arXiv :1901.11504* (2019).

[35] Panagiotis Mavridis, David Gross-Amblard et Zoltán Miklós, « Using hierarchical skills for optimized task assignment in knowledge-intensive crowdsourcing », in : *Proceedings of the 25th International Conference on World Wide Web*, 2016, p. 843-853.

[36] François Mentec et al., « Conversational recommendations for job recruiters », in : *Knowledge-aware and Conversational Recommender Systems*, 2021.

[37] Stuart E Middleton, Harith Alani et David C De Roure, « Exploiting synergy between ontologies and recommender systems », in : *arXiv preprint cs/0204012* (2002).

[38] Stuart E Middleton, David De Roure et Nigel R Shadbolt, « Ontology-based recommender systems », in : *Handbook on ontologies*, Springer, 2004, p. 477-498.

[39] Tomas Mikolov et al., « Efficient estimation of word representations in vector space », in : *arXiv preprint arXiv :1301.3781* (2013).

[40] Tomoko Murakami, Koichiro Mori et Ryohei Orihara, « Metrics for evaluating the serendipity of recommendation lists », in : *Annual conference of the Japanese society for artificial intelligence*, Springer, 2007, p. 40-46.

[41] Fedelucio Narducci et al., « Improving the user experience with a conversational recommender system », in : *International Conference of the Italian Association for Artificial Intelligence*, Springer, 2018, p. 528-538.

[42] Jeffrey Pennington, Richard Socher et Christopher D Manning, « Glove : Global vectors for word representation », in : *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, p. 1532-1543.

[43] Pearl Pu, Li Chen et Rong Hu, « A user-centric evaluation framework for recommender systems », in : *Proceedings of the fifth ACM conference on Recommender systems*, 2011, p. 157-164.

[44] Alec Radford et al., « Language models are unsupervised multitask learners », in : *OpenAI blog* 1.*8* (2019), p. 9.

[45] Nils Reimers, Philip Beyer et Iryna Gurevych, « Task-oriented intrinsic evaluation of semantic textual similarity », in : *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics : Technical Papers*, 2016, p. 87-96.

[46] Nils Reimers et Iryna Gurevych, « Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation », in : *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, nov. 2020, URL : https://arxiv.org/abs/2004.09813.

[47] Nils Reimers et Iryna Gurevych, « Sentence-BERT : Sentence Embeddings using Siamese BERT-Networks », in : *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, nov. 2019, URL : https://arxiv.org/abs/1908.10084.

[48] Nils Reimers et Iryna Gurevych, « Sentence-bert : Sentence embeddings using siamese bert-networks », in : *arXiv preprint arXiv :1908.10084* (2019).

[49] Paul Resnick et Hal R Varian, « Recommender systems », in : *Communications of the ACM* 40.*3* (1997), p. 56-58.

[50] Corné de Ruijt et Sandjai Bhulai, « Job recommender systems : A review », in : *arXiv preprint arXiv :2111.13576* (2021).

[51] J Ben Schafer, Joseph Konstan et John Riedl, « Recommender systems in e-commerce », in : *Proceedings of the 1st ACM conference on Electronic commerce*, 1999, p. 158-166.

[52] Thomas Schmitt, Philippe Caillou et Michele Sebag, « Matching jobs and resumes : a deep collaborative filtering task », in : *GCAI 2016-2nd Global Conference on Artificial Intelligence*, t. 41, 2016.

[53] Mike Schuster et Kuldip K Paliwal, « Bidirectional recurrent neural networks », in : *IEEE transactions on Signal Processing* 45.*11* (1997), p. 2673-2681.

[54] Stephen S Senn, *Cross-over trials in clinical research*, t. 5, John Wiley & Sons, 2002.

[55] Saman Shishehchi et Seyed Yashar Banihashem, « Jrdp : a job recommender system based on ontology for disabled people », in : *International Journal of Technology and Human Interaction (IJTHI)* 15.*1* (2019), p. 85-99.

[56] Patsakorn Singto et Anirach Mingkhwan, « Semantic searching it careers concepts based on ontology », in : *Journal of Advanced Management Science* 1.*1* (2013), p. 102-106.

[57] Zheng SITING et al., « Job recommender systems : a survey », in : *2012 7th International Conference on Computer Science & Education (ICCSE)*, IEEE, 2012, p. 920-924.

[58] John K TARUS, Zhendong NIU et Ghulam MUSTAFA, « Knowledge-based recommendation : a review of ontology-based recommender systems for e-learning », in : *Artificial intelligence review* 50.*1* (2018), p. 21-48.

[59] *The Dutch childcare benefit scandal, institutional racism and algorithms*, juin 2022, URL : https://www.europarl.europa.eu/doceo/document/O-9-2022-000028_EN.html.

[60] Nava TINTAREV, « Explanations of Recommendations », in : *Proceedings of the 2007 ACM Conference on Recommender Systems*, RecSys '07, Minneapolis, MN, USA : Association for Computing Machinery, 2007, p. 203-206, ISBN : 9781595937308, DOI : 10.1145/1297231.1297275, URL : https://doi.org/10.1145/1297231.1297275.

[61] Ashish VASWANI et al., « Attention is all you need », in : *Advances in neural information processing systems* 30 (2017).

[62] Alex WANG et al., « GLUE : A multi-task benchmark and analysis platform for natural language understanding », in : *arXiv preprint arXiv :1804.07461* (2018).

[63] WIKIPEDIA, *Dutch childcare benefits scandal*, URL : https://en.wikipedia.org/wiki/Dutch_childcare_benefits_scandal.

[64] Yongfeng ZHANG, Xu CHEN et al., « Explainable recommendation : A survey and new perspectives », in : *Foundations and Trends® in Information Retrieval* 14.*1* (2020), p. 1-101.

COLLEGE DOCTORAL BRETAGNE
MATHS, TELECOMS
INFORMATIQUE, SIGNAL
SYSTEMES, ELECTRONIQUE

Université de Rennes

Titre : Systèmes de Recommendations de Poste Explicable pour les Recruteurs et les Compagnies de Services

Mot clés : Systèmes de Recommendations de Poste Explicable, Apprentissage Profond, Traitement du Langage Naturel, Interaction Homme Machine

Résumé : Le recrutement a toujours été une tâche cruciale pour la réussite des entreprises, notamment pour les entreprises de services pour lesquelles l'embauche est un élément central de leur modèle commercial. La croissance du marché du travail ainsi que l'augmentation du nombre de compétences spécialisées requises par les entreprises ont motivé l'exploration de techniques pour optimiser et même automatiser certaines parties du processus de recrutement.

Les nombreux progrès réalisés dans les domaines de l'intelligence artificielle et du traitement automatique du langage naturel au cours des dernières décennies ont offert la possibilité de traiter efficacement les données utilisées lors du recrutement.

Nous examinons l'utilisation d'un système de recommandation d'emploi dans une entreprise de conseil, en mettant l'accent sur l'explication de la recommandation et sa perception par les utilisateurs. Tout d'abord, nous expérimentons avec des recommandations basées sur la connaissance en utilisant l'ontologie européenne des compétences et des professions ESCO qui présente des résultats prometteurs, mais en raison des limites actuelles, nous utilisons finalement un système de recommandation sémantique qui fait désormais partie des processus de l'entreprise et offre la possibilité d'études qualitatives et quantitatives sur l'impact des recommandations et de leurs explications.

Nous relions la disponibilité des explications à des gains majeurs d'efficacité pour les recruteurs. L'explication offre également un moyen précieux d'affiner les recommandations grâce à des retours utilisateurs contextuels. Un tel retour d'information est non seulement utile pour générer des recommandations en temps réel, mais aussi pour fournir des données précieuses pour évaluer les modèles et améliorer davantage le système. À l'avenir, nous préconisons que la disponibilité des recommandations devienne la norme pour tous les systèmes de recommandation d'emploi.

Title: Explainable Job Recommender Systems for Recruiters and Consulting Companies

Keywords: Explainable Job Recommender Systems, Deep Learning, Natural Language Processing, User Machine Interaction

Abstract: Recruitment has always been a crucial task for the success of companies, and especially consulting companies for which hiring is a centerpiece of their business model. The growth of the labor market along with the increasing number of specialized skills that are required by companies has motivated the exploration of techniques to optimize and even automate parts of the recruitment process.

The numerous progress made in the fields of Artificial Intelligence and Natural Language Processing during the past few decades offered the opportunity to efficiently process the data used during recruitment.

We examine the use of a job recommender system in a consulting company, with a focus on the explanation of the recommendation and its perception by users. First, we experiment with knowledge-based recommendations using the European ontology of skills and occupation ESCO which showcases promising results, but because of current limitations, we finally use a semantic-based recommender system that has since become part of the company processes and offers the opportunity for qualitative and quantitative studies on the impact of the recommendations and its explanations.

We link the explanation availability to major gains in efficiency for recruiters. It also offers them a valuable way to fine-tune recommendations through contextual feedback. Such feedback is not only useful for generating recommendations at run-time but also for providing valuable data to evaluate models and further improve the system. Going forward we advocate that the availability of recommendations should be the standard for every job recommender system.