TeamBuilder: D'un moteur de recommandation de CV notés et ordonnés à l'analyse sémantique du patrimoine informationnel d'une société

Démonstration

Patrice Darmon
Umanis
Levallois-Perret, France
pdarmon@umanis.com

Rabah Mazouzi
Umanis
Levallois-Perret, France
rmazouzi@umanis.com

Otman Manad
Umanis
Levallois-Perret, France
omanad@umanis.com

Mehdi Bentounsi
Umanis
Levallois-Perret, France
mebentounsi@umanis.com

ABSTRACT

Teambuilder is a prototype of a recommendation engine based on a Big Human Resources Data platform, allowing to securely retrieve the best candidates (resume) for a specific mission.

KEYWORDS

Big Data, NoSQL, Ressources Humaines, Machine Learning, NLP, Moteur de recommandation, Privacy By Design, RGPD

1 INTRODUCTION

De nombreuses sociétés et en particulier les Entreprises de Services du Numériques (ESN) sont confrontées à l'enjeu de la recherche et de l'appariement automatisé et surtout fiable de Curriculum Vitae (CV), aux multiples formats[1], avec des fiches de mission, elles mêmes pouvant être de formes et de structures très diverses. A ceci doit s'ajouter :

- L'**expérience de l'utilisateur** avec des exigences d'ergonomie et de temps de réponse des interfaces homme-machine par les chargés de recrutement, ingénieurs d'affaires, équipes commerciales, etc.
- La nécessité conjugée à la difficulté de prise en compte de **référentiels** actualisés de compétences métiers et d'entreprises.
- La nécessité de **fusionner des données hétérogènes** issues du Système d'Information de l'entreprise (Données RH, CRM, etc.).
- Des **contraintes de sécurité** de plus en plus poussées en termes de pseudonymisation, chiffrement, gestion des durées de conservation, et des consentements des personnes dans le cadre de la réglementation en vigueur (cf. Règlement Européen sur la Protection des données [14]).

Du point de vue des approches de traitement et en partant du postulat que les CV et fiches de missions prennent souvent la forme de documents textuels et de données non-structurées ou semi-structurées, utiliser des techniques d'intelligence artificielle dans un tel cadre constitue un challenge pour la communauté des chercheurs en fouille de texte [5].

[1] Les CV peuvent être créés à partir d'outils ou de sites Web dédiés (via, e.g., onlinecv.fr).

1.1 Contexte et approches existantes

Les modes de recherche proposés par les CVthèques ou sur les sites de recherche d'emploi sont principalement syntaxiques et à base de mots clés sur le titre du CV et les compétences techniques des profils. Prenons l'exemple de la recherche d'un profil *"ingénieur Big Data dans le secteur de l'automobile"*. Les CVthèques du marché proposent de l'effectuer de deux manières :

(1) Une recherche simple sur les titres des CV. Cela permet de récupérer les profils ayant dans le titre de leurs CV les termes *Big Data* et/ou *automobile*.
(2) Une recherche complexe sur l'ensemble des compétences techniques requises pour un ingénieur Big Data (e.g., Hadoop, Spark, Scala, .etc), en rajoutant à cela d'autres termes, tels que automobile et/ou des noms de constructeurs automobiles. Cette recherche permettra de récupérer les profils disposant des compétences techniques recherchés sur des projets menés dans le secteur recherché.

Ces modes de recherche ne permettent pas d'accéder de manière efficiente et fiable aux meilleurs profils appropriés à une demande. Notamment, dans les cas ou : *(i)* un profil malgré qu'adéquat ne dispose pas du bon titre de CV ; *(ii)* quand la recherche n'est pas faite sur les bonnes compétences techniques ou faite sur des alias, et *(iii)* la liste des constructeurs automobiles renseignée n'est pas complète.

En effet, les outils du marché ne proposent pas de moteur de recherche sémantique des compétences détaillées avec des appariements, scalables et en temps réel, et intégrant les contraintes de sécurité sur les données personnelles de profils de candidats, aux fiches de mission clients [6, 7]. Certains travaux proposent des approches notamment à base d'ontologies [8, 4, 9, 10]. Cependant, ils ne présentent pas des cas d'utilisation permettant un passage à l'échelle et une évolution vers d'autres domaines métiers.

1.2 Motivations et contributions

Suite à ce constat quant aux solutions du marché, le projet "TeamBuilder" a été lancé début 2016 avec dès son origine, des objectifs forts (i) d'évolutivité vers d'autres domaines métiers, (ii) de

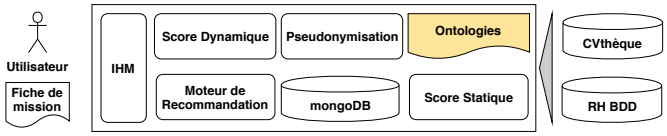

Figure 1: Architecture Fonctionnelle de TeamBuilder.

scalabilité et (iii) de sécurité. TeamBuilder vise à créer une architecture NoSQL de type Data Lake sur MongoDB [3] ; et alimenté par plusieurs sources de données (bases de CV de différentes sources, information de disponibilité des profils, fiches de missions, ...). L'architecture implémentée est à base de micro services et utilise des ontologies combinées à des techniques de fouille de texte, d'apprentissage automatique et de pseudonymisation (Figure 1).

L'un des défis majeurs du projet a été de concevoir et d'intégrer de manière itérative un flux de traitement de données non-structurées à base de microservices évolutifs, et d'automatiser le déploiement avec Docker. TeamBuilder est actuellement en production et propose aux équipes commerciales, aux chargés de recrutements et aux responsables de staffing d'équipes :

- Un moteur de recherche sémantique de CV.
- Un moteur de recommandation de CV à base de scores statiques et dynamiques.
- Un moteur d'appariement temps réel de plusieurs dizaines de milliers de CV aux fiches de missions clients sur la base d'une ontologie principale de 900 compétences.
- Des fonctionnalités de visualisation des données.
- Des fonctionnalités de chiffrement et de pseudonymisation des CV.

2 LA PLATEFORME TEAMBUILDER

Teambuilder propose un moteur de recommandation scalable et sécurisé de CV scorés et ordonnés. Il inclut également des fonctions de désambiguïsation sémantique à base d'ontologies (compétences avec alias, clients sectorisés, ...etc.). Dans cet objectif, plusieurs sources de données hétérogènes sont fusionnées et sauvegardées dans une base de donnée orientée document (i.e., MongoDB). La plateforme développée est constituée de plusieurs processus indépendants et faiblement couplés sous la forme d'images (Figure 1).

Les différents processus constituent un environnement microservices basé sur l'outil Docker. Ces microservices forment une collaboration afin d'assurer le bon fonctionnement de la plateforme. Dans la Figure 2, sont décrites les étapes de :

(1) Préparation et d'initialisation de l'environnement de développement et d'intégration continue. Elle permet le démarrage des services et la mise en place des conteneurs Docker.
(2) Lancement de la plateforme et la communication entres microservices. Il s'agit d'une requête envoyée par le client (i.e., IHM Web) aux microservices constituant la plateforme (e.g., Dynamic Scoring) via le microservice *classic-frontend*. Une authentification est effectuée lors de la première connexion.

2.1 Modélisation

2.1.1 Ontologies. Étant donné qu'un CV est composé d'un ensemble de compétences techniques ou managériales appartenant elles-mêmes à une ou plusieurs catégories de compétences (Table 1) ; et après une analyse de l'état de l'art sur les ontologies de compétences, notamment la norme européenne EN 16234-1:2016 [13] qui décrit 40 compétences pour les professionnels des Technologies de l'Information, nous avons fait le choix de construire, via des interviews d'experts de chaque catégorie de compétences, une ontologie de compétences détaillée représentant plus de **900 compétences**, avec une dizaine d'attributs par compétence. L'ontologie est actualisée tous les 6 mois, avec les alias, les définitions, la catégorisation des compétences sur 4 niveaux maximum, en multi-hiérarchies, etc. Ces travaux ont été complété par des ontologies relatives aux :

Table 1: Exemple de catégories et compétences.

Catégorie	Compétences Techniques / Certifications
Big Data	MongoDB, Hadoop, Cassandra, etc.
BI	Oracle, Talend, Informatica, etc.
CRM	SalesForce, MS Dynamics, etc.

(1) **Noms des entreprises** avec leurs alias (2000 entreprises modélisées) et leur rattachement à un secteur d'activité, e.g., ERDF et son nouveau nom Enedis est une entreprise du secteur Energie.
(2) **Certifications éditeurs** : Prototypage sur les certifications Microsoft avec une ontologie construite automatiquement via un web scraping des pages des éditeurs de solutions, par exemple Microsoft[2], et analyse des compétences associées avec notre ontologie de compétences.
(3) **Diplômes**
(4) **Fonctions** et leurs alias (consultant, ingénieur d'études, ...)

2.1.2 Modèle de données. Dans TeamBuilder, les données sont modélisées dans une base MongoDB sur des documents au format JSON. La collection principale de la base est "person". Elle regroupe l'ensemble des informations, d'une personne physique ou profil, récupérées et calculées par l'ensemble des microservices :

- Informations permettant l'identification : nom, prénom(s), date de naissance, etc.
- Informations de contacts : adresse(s), email(s), numéro(s) de Téléphone, etc.
- Projet(s) et mission(s) passé(s) et courant(s).
- Compétences techniques, certifications et les scores correspondants.
- Pseudonyme calculé à partir des données d'identification.

2.1.3 Requêtes. TeamBuilder dispose d'une IHM permettant d'exécuter différents types de requêtes. En réponse, chaque CV, compétence du CV et catégorie de compétences fait l'objet d'un score dédié (scoring Multi-labels). On considère :

(1) Les requêtes sur les domaines de compétences : A cet effet, l'utilisateur choisit un domaine de compétences dans une liste déroulante afin d'obtenir une liste de **N** CV scorés, ordonnés, et compatibles avec le domaine renseigné.

$$Query_{SkillsDomain} = \bigcup_{0>n>N} (CV^{n}_{(Skills_i;SKScores_i)}; CVScore_n)$$

[2]https://www.microsoft.com/fr-fr/learning/exam-list.aspx

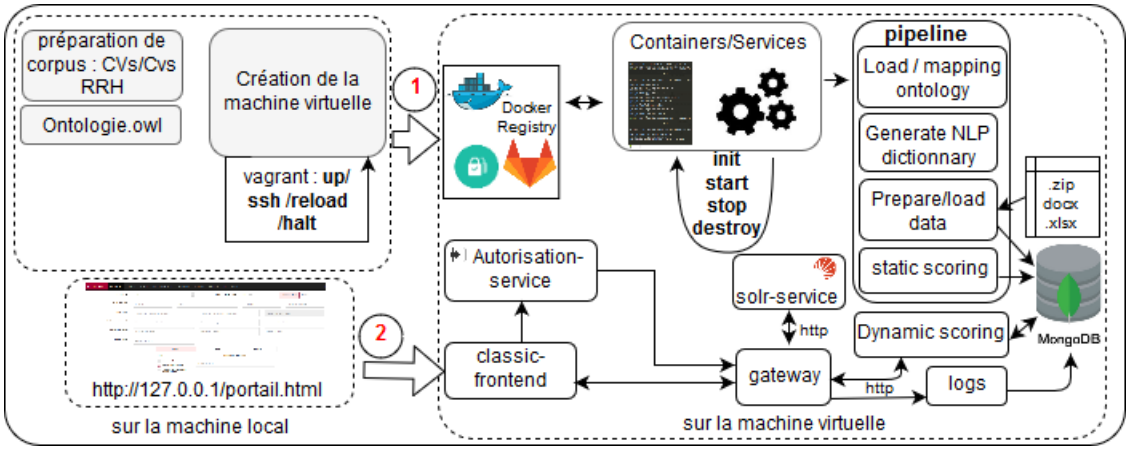

Figure 2: Architecture technique de TeamBuilder.

avec

$$SkillsDomain \in \{BiDData; BI; IoT; :::\};$$
$$Skills_i \in \{MongoDB; Cassandra; Hadoop; :::\};$$
$$Scores_i \in [0; 100]; \; ScoreCV_i \in [0; 100];$$

(2) Les requêtes composées de filtres sur plusieurs paramètres à la fois (domaine de compétence, compétences techniques, secteur d'activité).

$$Query_{SkillsDomain;Skills_i^j;Sector} = \emptyset \quad (CV^n_{(Skills_i;SKScores_i)}; CVScore_n)$$
$$0>n>N$$

avec

$$Sector \in \{Energie; Automobile; Assurance; :::\};$$

2.2 Scoring

Le processus de préparation des données consiste à calculer des scores pour toutes les compétences extraites d'un CV. On distingue deux types de score :

(1) **Scores Statiques** : calculés à partir des données obtenues par l'annotation de chaque CV, notamment la fréquence et la position de la compétence dans le document (e.g., un poids élevé est donné aux compétences mentionnées au début d'un document parce qu'on suppose que celles-ci correspondent au titre du CV). Le calcul prend en compte aussi la note de la compétence, attribuée par les experts, elle représente l'importance de la compétence par rapport au marché du travail. Les notes sont préalablement renseignées dans l'ontologie des compétences.
Scores Radars : Un second score statique particulier est calculé durant l'étape de préparation. Il s'agit d'une agrégation des différents scores statiques des compétences par domaine (e.g., Big Data, Data Science, BI, etc.). Le score radar est utilisé comme marqueur dans une visualisation des données sous la forme d'un radar, où chaque branche représente un domaine de compétences. Ce radar permet de comparer graphiquement, deux ou plusieurs profils (Figure 3).

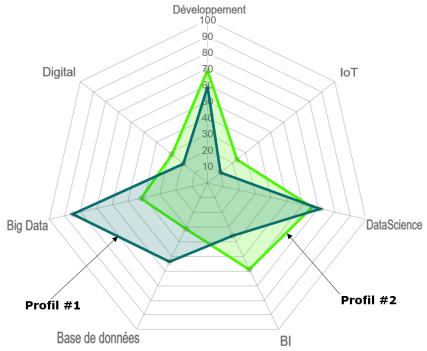

Figure 3: Radars des compétences

(2) **Scores Dynamiques** : il s'agit des scores calculés et utilisés au moment de l'execution d'une requête de recherche. A cet effet, le service de scoring dynamique, et selon certains de ses critères de recherche (compétences requises, compétences optionnelles, secteur, etc.), effectue une pondération des scores statiques afin d'obtenir un nouveau "score dynamique" permettant de trier les résultats et retourner les **N** meilleurs profils.

2.3 Rapprochement CV - Fiche de mission

TeamBuilder permet de faire un rapprochement multi-critères entre une fiche de mission[3] insérée par un utilisateur, avec les CV préalablement stockés et scorés, via le service de scoring statique, dans une *collection mongoDB*. Dans un premier temps, les compétences demandées sont extraites, automatiquement, à partir de la fiche de mission. Ces dernières sont ensuite comparées avec les compétences des CV stockés dans la base en utilisant la formule (1) entre la fiche de mission reçue et les compétences des profils adéquats pour

[3]Une fiche de mission est un document décrivant les besoins fonctionnels et techniques d'un client pour une mission ou un projet.

une mission. À noter que nous avons lancé des expérimentations en utilisant plusieurs mesures de similitude (Jaccard, Levenshtein, Hamming, Jaro, etc...). Ces dernières ne correspondent pas à nos attentes, d'oú vient notre choix de la formule (1).

$$distMatching(P_i, M) = |SkillsPerson_i \cap SkillsMission| + w \quad (1)$$

avec:
$SkillsPerson_i$: représente les compétences d'une personne P_i.
$SkillsMission$: représente les compétences extraites de la fiche de mission d'un client M.
w : une pondération calculée afin de favoriser la personne ayant le plus de compétences dans le cas ou deux ou plusieurs personnes possèdant les compétences demandées.

$$w = \frac{|SkillsPerson_i| * |(SkillsPerson_i \cap SkillsMission)|}{NombreTotalSkills} \quad (2)$$

$NombreTotalSkills$: Le nombre de compétences dans l'ontologie.
Enfin, les CV retournés sont ordonnés par ordre de correspondance, du plus au moins pertinent.

2.4 Préservation de la vie privée

La valeur des informations contenues dans les CVthèques oblige les ESN à la prise en charge de la sécurité des données à caractère personnel, et ce afin de faire face aux conséquences des incidents et violations sur la disponibilité, confidentialité et intégrité des données. En plus des bonnes pratiques de sécurité informatique (i.e., réseau, système d'exploitation, base de données et développement logiciel), la *"Privacy by Design"* rajoute une nouvelle couche de sécurité sur les données elles mêmes, permettant ainsi leur gestion et traitement de manière anonyme [14].

Afin d'implémenter la privacy by design dans le contexte du projet et ainsi répondre plus efficacement aux besoins de sécurité, Teambuilder intègre :

(1) Un service de pseudonymisation à base de chiffrement réversible dans l'objectif de préserver la confidentialité des *documents* dans la base mongoDB. Pour cela, nous utilisons le chiffrement asymétrique (i.e., RSA) afin de chiffrer les données permettant une identification des personnes physiques (nom, prénom, émail, numéro de téléphone) lors de l'étape de préparation des données. Ainsi, seuls les utilisateurs habilités (i.e., disposant de la clef de déchiffrement) peuvent accéder à ces informations.
(2) Un Service de génération de pseudonymes aléatoires permettant d'identifier les profils (via ce pseudonyme) lors des échanges entre utilisateurs non habilités (i.e., externes à l'entreprise) et ceux habilités sans obligation de révéler les identités des personnes.
(3) Un service de contrôle d'intégrité des *documents* mongoDB, à base de chiffrement irréversible afin de calculer des signatures pour les *documents* dans la base mongoDB.
(4) Un flux complet de traitement des données permettant la mise en place d'une architecture Privacy by Design et intégrant les services précédemment cités [1, 2].

3 DÉMONSTRATION

TeamBuilder a été conçu et réalisé pour être déployé en utilisant l'outil Docker [12]. En plus d'automatiser les déploiements, Docker permet d'empaqueter une application et ses dépendances dans un conteneur isolé pouvant être exécuté sur n'importe quel serveur.

Nous allons montrer dans cette section le prototype TeamBuilder dans un ensemble de scénarios effectués par l'utilisateur à partir de l'IHM et illustrés par des copies d'écrans (Figure 4).

Recherche par domaine de compétences et/ou par mots clés. Dans ce premier cas d'utilisation, illustré par la Figure 4a, un utilisateur sélectionne l'ensemble des critères pour sa recherche de profils et indique le nombre de CV qu'il veut voir afficher en résultats (15 par défaut). S'il choisit juste un domaine (e.g., Big data) sans aucune autre précision, TeamBuilder va sélectionner et afficher tous les CV ordonnés et notés qui ont des compétences dans le domaine choisi (ici Big Data) dans leur CV (Figure 4c). Cela permet à l'utilisateur de sélectionner juste un domaine technique sans avoir à connaitre le détail des compétences du domaine. Mais il peut également affiner sa recherche en sélectionnant une liste de compétences, de secteurs d'activités, de sociétés (clients), ...

Recherche par fiche de mission. Dans ce deuxième cas d'utilisation, pour une expérience utilisateur d'appariement sémantique facilitée, le document fiche de mission constitue la requête. En effet, l'IHM de recherche offre également à l'utilisateur la possibilité de déposer, en mode *drag and drop*, une fiche de mission (Figure 4b), (les compétences y seront reconnues en temps réel). L'utilisateur lance ensuite la recherche de profils. Les résultats seront là aussi affichés dans l'onglet "résultat" (Figure 4c) avec les profils possédant les meilleurs scores pour les compétences (ou leurs alias) dans leurs CV. L'utilisateur peut en complément visualiser ou télécharger le CV de chaque profil affiché de manière pseudonymisé ou non selon les habilitations issues du mode d'authentification.

Annotation de documents. Le troisième cas d'utilisation, illustré via la Figure 4d, permet à l'utilisateur de déposer, toujours en mode drag and drop, n'importe quel(s) document(s) pour annotations en temps réel en regard des ontologies de TeamBuilder. Dans le cas du dépôt d'un CV, cette IHM affiche à droite une classification automatique multi-labels sous forme de graphique des profils par domaine de compétences (Big Data, Data Science, BI, ...) entrainée par réseau de neurones. D'autre part, dans la partie gauche, chaque item d'ontologie retrouvé par TeamBuilder dans le(s) document(s) génère l'affichage d'un score.

Cette IHM permet aussi d'analyser (unitairement ou en masse), selon les mêmes modalités, n'importe quel document en lien avec les ontologies créées pour TeamBuilder comme le contenu technique :

(1) de publications scientifiques informatiques afin de donner à l'utilisateur en quelques secondes des scores lui permettant de l'aider à savoir s'il est pertinent pour lui de lire ou non ce(s) document(s) par rapport à son besoin
(2) d'un appel d'offre d'une société, pour faciliter la recherche d'une équipe de réponse adaptée et/ou d'une réponse à Appel d'Offres similaire dans le corpus disponible.

4 CONCLUSION ET PERSPECTIVES

Au delà de son domaine d'application initial présenté dans ce document, les atouts de notre plate-forme sont : *(i)* son architecture innovante, *(ii)* sa scalabilité, *(iii)* sa capacité à analyser sémantiquement différents domaines métiers via des ontologies évolutives, et