Geindre Colin
Bouquerel Louis
Hayoun Alexandre
Jaouen Denis

DATA challenge 2026

Equipe : Les vieux briscards

Projet : CHU

Achats courants : processing et embedding

Notre problème initial est le suivant : faire correspondre des libellés de deux datasets, alors que les nomenclatures sont fondamentalement différentes. Au CHU, un logiciel de gestion trop permissif cause des libellés sans nomenclature spécifique, les données n'ont pas été saisies avec une nomenclature spécifique en tête. Un coup d'œil nous permet de dire qu'elles sont de qualité médiocre avec un orthographe et une syntaxe à qualité variable. On voit également beaucoup de données manquantes ou dupliquées.

Pre-processing

Tout d'abord, nous avons concaténé les colonnes pertinentes, mis les textes à la même échelle et supprimer les mots de liaison qui menaient régulièrement à des hallucinations.

Ensuite, nous avons effectué un processing par LLM en local. Pour éviter les hallucinations, nous avons mis en place des garde-fou, comme supprimer automatiquement les labels créés si ceux-ci sont plus grands que les originaux, ou s'ils comportent des mots comme « est… ».

Nous avons comparé plusieurs modèle et nous sommes arrêtés sur Mistral-7B-v0.3.

Embedding

Nous avons générer des données vectorielles (embedding) pour chacun des labels du dataset source et cible. De nombreux modèle ont été testés : des modèles spécifiquement entrainés sur le domaine médical, d'autres plus généralistes. Ceux qui se sont imposés sont miniLM et mlnet de Huggingface. Etonnemment, Qwen et Mistral ont mal performé.

Matching

Pour faire correspondre les vecteurs ainsi générés, nous avons adopté une approche hybride : un score de proximité composé de la similarité cosinus (via FAISS, qui permet un indexage pour plus de vitesse), et un matcheur lexical (blm20).

Plusieurs formules de score ont été testés, et nous nous somme arrêtés sur un poids de 80% pour le sémantique, et 20% pour matcheur lexical.

Avec plus de temps, nous aurions aimé ré entrainer notre modèle sur les sorties les plus fiables, ou garder les 10 meilleurs matchs et faire une validation croisée en suivant.

Résultat

Au final, après toutes ces itérations, entre 30 et 50% du dataset est identifié avec une fiabilité supérieure a 70%. C'ests beaucoup plus qu'espéré !

La mobilité du personnel : Une carte interactive des communes avec les modes de transports possibles associés

Fonctionnement de l'outil :

Les temps et distances sont calculés depuis les « centroïdes » des communes concernées, c'est-à-dire depuis les centres géographiques.

GeoApify donne les routes exactes et les temps de trajet moyens associées pour ce qui est des trajets en **voiture**.

Les trajets à **vélo** ont été calculé selon la distance à vol d'oiseau et une vitesse moyenne de 15 km/h (vitesse moyenne à vélo).

Les temps de trajet en **bus** sont calculés ainsi : temps de trajet passé à marcher d'un centroïde à l'abribus le plus proche avec une vitesse moyenne de 5 km/h (avec les données GTFS du réseau STAR). Auquel on ajoute le même calcul qu'en vélo avec également une moyenne de 15 km/h (la vitesse moyenne d'un bus en milieu urbain)

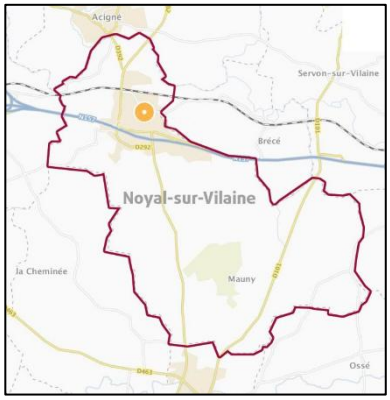
Idéalement, nous utiliserions l'API SNCF pour calculer les temps de trajets en **train**.

Bilan :

Bien qu'améliorable, nous constatons qu'en comparant avec google maps, notre outil donne des estimations de temps de trajet proches de la réalité pour la majorité des communes de la métropole de Rennes.

Améliorations visibles :

Nous avons utilisé les centres géographiques des communes plutôt que les centres de population. Pour les communes les plus proches de Rennes, cela peut mésestimer les temps de trajets de vélo ou de bus lorsque ce centre est particulièrement différent du centre de population. Ci-joint, l'exemple de Noyal sur Vilaine :



En général, pouvoir utiliser les distances de trajet exactes permettrait une meilleure estimation des temps à vélo et en bus. Additionnellement, avoir accès aux vitesses moyennes des différentes lignes de bus, prendre en compte les changements de ligne et les fréquences à chaque station affinerait notre outil. Tout ceci est possible avec l'API google maps (malheureusement payante).

Pour aller plus loin : Nous avons commencé a développer une application qui optimise des routes de covoiturage, sur un modèle temps de trajet versus émission de CO2.

Flotte de Véhicules : IA SCRAPEUSE de fiche technique

Disclaimer : Notre développeur a succombé à la maladie hier et n'a pu développer cette partie.

Notre stratégie d'attaque est de développer une IA qui serait entrainé pour trouver une fiche technique à partir d'un libellé de véhicule (comme ceux disponibles sur les données des véhicules) sur un ensemble de sites web comme lacentrale.fr, autotitre.com ou caradisiac.com. Cette partie est la plus difficile mais ne nous est pas hors de portée actuellement.

Ensuite, à l'aide de scraping, cette IA doit extraire de cette fiche technique le poids du véhicule et le type du véhicule.

Enfin, elle doit remplacer les données manquantes par une consommation carbone prévue et le type du véhicule déduit.

Dispositifs médicaux – Embedding et NLP

Notre stratégie consiste à utiliser une méthode de traitement du langage naturel (NLP) pour diviser chaque dispositif en différents composants. Ensuite, nous voudrions utiliser les mêmes méthodes d'embedding que pour la méthode des achats afin de déterminer les matériaux et ainsi leur poids environnemental.

Conclusion

Ce fut une expérience enrichissante à laquelle nous sommes heureux d'avoir participé. Nous avons pu voir toute l'étendue de nos capacités et ce qu'il nous reste à apprendre. Bien que nous ayons abattu beaucoup de travail, beaucoup reste à accomplir.

Nous aimerions continuer.