

### RÉUNION 03/07/2023

Valentin BONSI

### PASSER D'UN JSON (GPT) AU 'FORMAT EMOJIS'

- > On prend le texte brut et le JSON des entités renvoyé par GPT3.5
- On replace les entités dans le texte, mais pas en utilisant des REGEX : parfois GPT corrige légèrement le texte lors de l'extraction
- > On utilise donc la distance de Levenshtein, avec les propriétés suivantes :
  - Une distance de 0 (aucun écart) dans le cas d'un mot de 3 caractères maximum
  - Une distance max de len(mot) // 3 sinon
- Les entités étant traitées de manière séquentielle par rapport au texte, le résultat devrait être acceptable.
- => Problème : beaucoup de faux positifs trouvés en utilisant cette méthode.

### ASTUCE 1 : ON DÉCOUPE LE TEXTE EN PARAGRAPHES

Paragraphe 1 : Date et heure du mariage

Paragraphe 2 : Mari et ses parents

Paragraphe 3 : Mariée et ses parents

Paragraphe 4 : Nom et prénom des époux

Paragraphe 5 : Témoins et adjoint au maire

=> On évite des faux positifs si par exemple le père du mari et le 2nd témoin ont la même profession.

# ASTUCE 2 : ON UTILISE LE PRÉNOM D'UNE PERSONNE COMME DÉLIMITEUR

- Les entités sont traitées l'une après l'autre (Mari, Pere-mari, Mère-mari, Mariée, Père-mariée, Mere-mariee, Temoin-1, Temoin-2, Adjoint)
- Leur prénom est toujours en premier
- Donc leurs autres caractéristiques sont après leur prénom
- Pour chaque possible occurrence trouvée, on prend en priorité la première après le prénom. Sinon, la dernière avant le prénom (Parfois possible avec les pluriels)

=> On évite les faux-positifs dans un même paragraphe (Mariée et sa mère ont la meme profession)

#### ASTUCE 3: ON FORCE A MATCHER DES MOTS COMPLETS

- > Pour chaque possible occurrence trouvée, on force une extension jusqu'au premier caractère spécial trouvé (ponctuation ou emoji)
- > On refait ensuite un test de Levenshtein pour voir si le match est toujours compétitif face aux autres de la liste, et on le supprime le cas échéant.

=> On évite ainsi à la fois de couper des mots, mais aussi de matcher Jean dans **Jean**ne.

## ASTUCE 4 : LES MATCHS DES PRÉNOMS DOIVENT ÊTRE VIERGES.

- Les prénoms servant de clé pour les autres caractéristiques d'une personne, ils ne peuvent eux-mêmes se référer à rien.
- On met en place un système de stockage de tous les tags précédemment relevés :
  - On stocke la position de départ (avec les emojis)
  - La longueur (avec les emojis)
- > Si un potentiel match pour un prénom est trouvé, on teste s'il se trouve 'entre les bornes' d'une autre caractéristique. Si oui, on l'ignore.
- => On ne matche plus Jean dans l'Avenue **Jean** Jaures.

### ASTUCE 5 : ON TRAITE LES PLURIELS APRÈS COUP

- > On ne cherche pas à détecter directement des cas de pluriels.
- > On ajoute les occurrences une à une, puis on utilise des REGEX pour remplacer les suites d'emojis correspondantes dans un second temps.

### **RESULTAT**

wingt-huit wingt-huit in initial initi
publiquement en la maison Commune:
© ☐ Gilbert Alexandre Jean
Et (P) ( Marcelle Fernande ( M) ( M
Aucune opposition n'existant, les futurs époux les père et mère de la future épouse déclarent qu'il n'a pas été fait de contrat de mariage.;  Gilbert Alexandre Jean  Alexandr
En présence de:     Jeanne   LOUSSOUARN   Jeanne   Louis   Jeanne   Louis   Levallois Perret   Levallois   Levallois Perret   Levallois   Levallois Perret   Levallois   Le

#### GPT3.5 TURBO 16K

- > Fenêtre de contexte multipliée par 4 :
- > Plus de place pour cerner les différentes formulations
- ➤On en profite pour inclure 5 nouveaux exemples (9 au total), en se focalisant sur les erreurs rencontrées jusqu'à maintenant :
- Les pluriels (notamment 'avec ses père et mère' étaient mal reconnus.
- On avait des errements sur les villes/département/pays
- Certains découpages en paragraphes étaient imprécis ou complètement ratés

## GPT3.5 TURBO 16K / RÉSULTATS

