Traitement automatique des langues
TD (NER)

Youna Froger, Alice Gyde et Mathis Quais

15 décembre 2024

1 GitHub

Voir le lien GitHub : https://github.com/younafroger/TAL$_H NM2$.

2 Explications du sujet

Pour ce TP, nous avons travaillé sur une base de code où la bibliothèque spacy été utilisée. On vient récupérer un modèle qui va faire la tokenisation et l'attribution de la nature grammaticale de chaque mot pour ensuite le replacer dans le contexte de la phrase. Pour charger le modèle, on utilise :

```
nlp = spacy.load("fr_core_news_md")
```

On applique ensuite ce nlp sur les parties de texte a analyser. Avec le modèle de base, on pourra reconnaître de personnes et des endroits (PER et LOC) avec décomposition en cas d'entité en plusieurs mots pour une seule entité.

L'affichage peut également se faire par le biais de boite colorée avec

```
from spacy import displacy
displacy.render(doc, style="ent", jupyter=True)
```

Un traitement nommé Process vient par la suite nettoyer les informations et les afficher sous un certain format.

Une dernière fonction va jouer le rôle d'évaluateur en utilisant les données et en calculant l'efficace des modèles. Le programme calcule les ratios de bonnes/mauvaises prédictions, la précision, le recall et le F1 score.

3 Développement du code

Notre partie consistait à récupérer des données depuis trois fichiers de données, importés sur le notebook, puis a entraîné le modèle pour qu'il puisse reconnaître de nouvelles entités. En se basant sur le code déjà fait, nous avons

décidé d'entraîner le modèle avec le corpus train jusqu'a ce que le f1 score dépasse 0,558, signe d'atteinte d'un palier de réussite convenable. Pour cela, le programme calcule les métriques sur le corpus dev a chaque epoch. Il y avait aussi possibilité de regarder au global a quel moment le modèle devenait le plus efficace en comparant avec les quelques résultats précédents, cependant, nous n'avons pas su comment implémenter cette solution.

Enfin, le programme affiche les rendus sous forme de boite avec le corpus test.

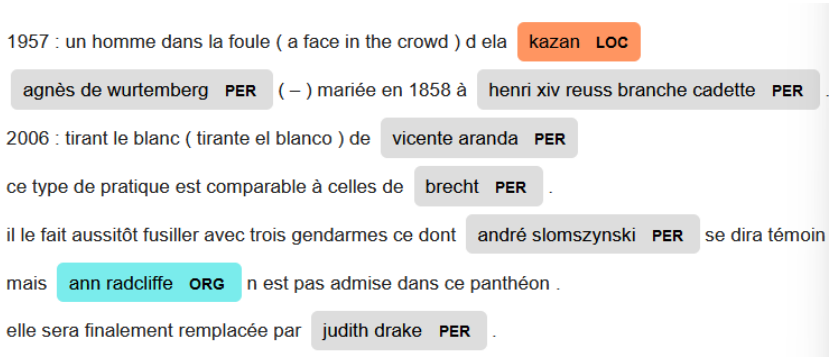
4 Résultats

Comme indiqué précédemment, nous cherchions a obtenir un F1 score assez élevé avant d'arrêter l'entraînement. Cependant, la boucle s'arrête de suite, avec un résultat dépassant notre limite de loin. Et même en forçant sur plusieurs epoch, les résultats restent très haute :

```
Epoch 0, Losses: {'ner': 112.9955089539634}
0.8492021073528289
Epoch 1, Losses: {'ner': 99.66983925506156}
0.8643834569222536
Epoch 2, Losses: {'ner': 104.16887066288095}
0.8736141906873615
Epoch 3, Losses: {'ner': 70.91427545888814}
0.8787461773700306
Epoch 4, Losses: {'ner': 72.7045590693105}
0.8825328846742123
Epoch 5, Losses: {'ner': 90.53508423295852}
0.8861863240018357
Epoch 6, Losses: {'ner': 68.04154189345763}
0.8822944550669216
Epoch 7, Losses: {'ner': 87.90817796268533}
0.8837831638504473
Epoch 8, Losses: {'ner': 64.51329687996568}
0.8847506882838788
Epoch 9, Losses: {'ner': 74.79095042571453}
0.888565965583174
Epoch 10, Losses: {'ner': 74.65948401668132}
0.8833652007648184
Epoch 11, Losses: {'ner': 78.20861535459196}
0.8863984088127295
Epoch 12, Losses: {'ner': 49.31279130246583}
0.8864070986001683
```

Les résultats sur la matrice de confusion sont tout aussi étrange. Après révision du code, nous ne voyons pas comme améliorer le traitement.

Lors de l'affichage des résultats avec displacy, des entités ont été rajoutées : LOC et ORG et l'entité PER est toujours présente.



5 Conclusion

Lors de ce TP, nous avons manipulé les entités nommées par le biais de modèles existants ainsi que des données récupérées sur Internet. L'application des méthodes d'apprentissage sur ces données reste floue, au regard de nos résultats. Cependant, ce TP nous a permis de mettre en pratique des calculs vus lors de précédentes séances et de nous familiariser avec de nouvelles notions.