Apprentissage Automatique

Projet 3 : Lemmatisation

Manon FLEURY-BENOIT

M1 Sciences Cognitives

2023-2024

Projet 3 Lemmatisation

On cherche à créer un modèle permettant de lemmatiser des mots. Pour cela, comme on veut déceler les relations pouvant relier des évènements arbitrairement distants, on utilise un réseau de neurone récurrent (RNN), de type transducteur c'est-à-dire qui lit une séquence (le mot à lemmatiser) et en produit une autre (le lemme prédit).
Le lemme est la forme canonique d'un mot (par exemple le lemme de « mangeront » est « manger »).
Le réseau prend en entrée des séquences de caractères représentés sous la forme de vecteurs one-hot.
On se focalise sur 3 catégories grammaticales : adjectif (A), nom (N) et verbe (V).

On va créer trois types de lemmatiseurs. Premièrement, le lemmatiseur catégoriel ne prendra en compte que des mots d'une même catégorie grammaticale, on va donc en faire trois, un pour chaque catégorie d'intérêt. Deuxièmement, on va créer un lemmatiseur général qui prend en compte la catégorie. Et troisièmement, on fera la version la plus réaliste du lemmatiseur qui ne suppose pas de connaître la catégorie de la forme à lemmatiser.

On fait l'hypothèse qu'on obtiendra la meilleure performance pour le premier modèle (lemmatiseur catégoriel).

I. Lemmatisateur catégoriel

Hypothèse : Meilleure performance pour le modèle de la catégorie des Noms

Pour chaque catégorie (A, N, V), on lit les fichiers et on transforme chaque mot en une liste de vecteurs one-hot pour chaque lettre (dont l'indice du 1 du vecteur correspond à la position de la lettre dans le fichier alphabet.txt). L'ensemble des mots d'une colonne sera alors représenté par un array, donc on obtient deux arrays : un pour la colonne de gauche des mots à lemmatiser et un pour la colonne du milieu pour les lemmes correspondants.

Après construction du modèle GRU, on l'entraîne avec les données d'apprentissage pour une catégorie, puis on l'évalue avec les données test de la catégorie concernée. On clone donc le modèle initial de façon à obtenir 3 modèles distincts car entraînés par une catégorie spécifique.

Performance des modèles

On obtient :

Evaluation du modèle A : Test loss: 0.0236 | Test accuracy: 0.9949
Evaluation du modèle N : Test loss: 0.0124 | Test accuracy: 0.9957
Evaluation du modèle V : Test loss: 0.0469 | Test accuracy: 0.9835

Sachant qu'une perte plus faible et une précision plus élevée indiquent généralement un meilleur modèle, on peut dire que le **modèle de la catégorie Noms est le plus performant**, ce qui est en accord avec notre hypothèse. On remarque qu'en deuxième position se trouve le modèle des adjectifs, et enfin le modèle le moins performant est celui pour les verbes.

Mesure de performance sur le nombre d'erreurs :

A : 254 lemmes mal prédits sur 2414, soit un taux d'erreurs de 0.105 (11%), et donc une exactitude de 0.895 (90%)
N : 530 lemmes mal prédits sur 8408, soit un taux d'erreur de 0.063 (6%), et donc une exactitude de 0.937 (94%)
V : 1081 lemmes mal prédits sur 4185, soit un taux d'erreur de 0.26 (26%), et donc une exactitude de 0.74 (74%).

On peut critiquer ce dernier résultat car on avait une accuracy de 98% (il doit y avoir une erreur dans le code, notamment dans la génération du « true_lemma » qui semble ne pas correspondre toujours au mot dans le fichier, ce qui expliquerait donc un taux d'erreur accru). L'exactitude de 90% pour le modèle A est aussi éloignée de l'accuracy de 99%.

Ces résultats confirment bien que le modèle pour les Noms est plus performant que les autres, avec un taux d'erreurs de prédiction plus petit, soit une exactitude supérieure.
Ils confirment aussi que le deuxième modèle le plus performant est celui des Adjectifs, avec une exactitude de 90%, alors que celle des verbes est bien plus basse (74%) et indique donc que le modèle des verbes n'est pas aussi performant.

Analyse des erreurs commises par ces lemmatiseurs

A : on remarque quelquefois l'erreur de remplacer un 'e' par '#' (par exemple « excellente » -> « excell#nt »), une erreur récurrente.
N : on remarque l'erreur récurrente de remplacer un 's' par '#' (par exemple « synopsis » -> « synop#is »)
V : on remarque des erreurs d'accent. On peut expliquer cela par le fait qu'il peut souvent y avoir une différence d'accent entre le verbe conjugué et le verbe à l'infinitif, en tout cas plus que pour d'autres catégories grammaticales. Par exemple « naissent » et « naître » ou « accru » et « accroître ». Les accents sont plus irréguliers chez les verbes que les autres catégories.

On a donc validé l'hypothèse selon laquelle le lemmatiseur de la catégorie N est le plus performant. Voyons maintenant un lemmatiseur général qui accepte en entrée des formes de toutes les catégories.

II. Lemmatiseur général avec prise en compte de la catégorie

Cette version du lemmatiseur prend en entrée un nom, un verbe ou un adjectif ainsi que l'information de sa catégorie (on prend toujours ces 3 catégories, et non les 14 présentes dans les fichiers lemmatisation_train et lemmatisation_test). Il faut donc trouver un moyen d'indiquer la catégorie dans l'entrée du réseau.
Pour cela, on va encoder la catégorie du mot en vecteur one-hot (vecteur de dimension 3, avec un 1 à l'indice de la catégorie, dont l'ordre a été défini dans le code). Et on va concaténer ce vecteur de dimension 3 avec le vecteur one-hot des lettres, de dimension 66 : on obtient donc un vecteur de dimension 69. On a donc ajouté l'information de la catégorie au niveau de chaque lettre.

Après construction du modèle GRU, on l'entraîne avec les données d'apprentissage générales, puis on l'évalue avec les données de test générales.

Performance du modèle :

On obtient :
Test loss : 0.037
Test accuracy : 0.989

Ce modèle a une précision un peu supérieure à celle du lemmatiseur de la catégorie des verbes (0.984), mais est plus faible que celles des lemmatiseurs adjectif et nom. Le lemmatiseur général prenant en compte la catégorie est donc moins performant que le lemmatiseur de la catégorie nom et que celui de la catégorie adjectif. Voyons maintenant un lemmatiseur qui ne prend pas en compte la catégorie.

III. Lemmatiseur général

Il s'agit de la version la plus réaliste du lemmatiseur car elle ne suppose pas de connaître la catégorie de la forme à lemmatiser. Il prend en entrée des formes de toutes les catégories représentées sous la forme d'une séquence de vecteurs one-hot.

Après construction du modèle GRU, on l'entraîne avec les données d'apprentissage générales et on l'évalue avec les données test générales.

Performance du modèle :

On obtient :
Test loss : 0.0654
Test accuracy : 0.9804

On a une précision inférieure à celle du lemmatiseur général avec prise en compte de la catégorie (0.989), ce qui semble être cohérent : sans indication supplémentaire sur la catégorie, le lemmatiseur a moins d'information pour trouver les bons lemmes, étant donné que la catégorie est une information importante pour déterminer le lemme d'une forme.

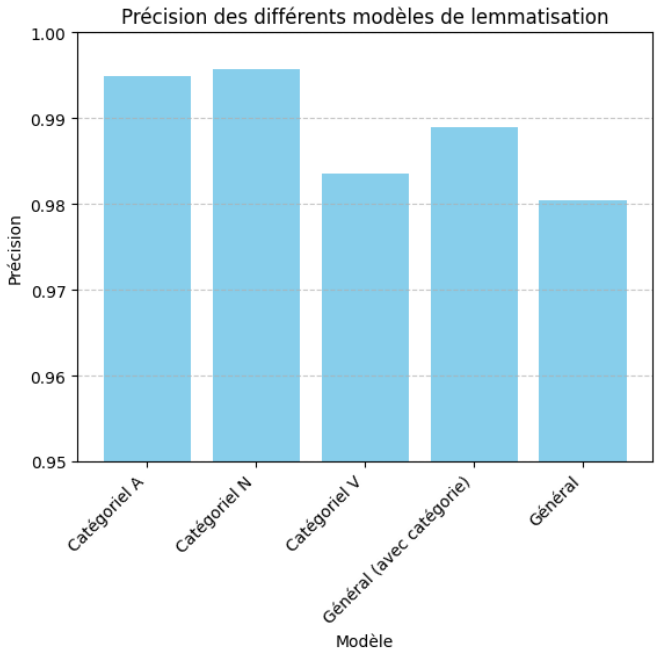

Figure 1 : Histogramme de la précision de tous les modèles de lemmatiseur

Conclusion

Le modèle le plus performant que l'on a créé est celui de la catégorie des Noms avec une précision de 0.9957. Parmi les trois lemmatiseurs catégoriels, le moins performant a été celui des Verbes, avec une précision de 0.9835. On peut expliquer cela par le fait que les verbes ont des formes beaucoup plus irrégulières en français, avec les conjugaisons. Par exemple, c'est plus difficile pour le lemmatiseur d'identifier que le lemme de « suis » est « être », sachant qu'en plus il peut y avoir des problèmes d'ambiguïté où ici « suis » peut également venir du verbe « suivre ». Pour ces raisons, la lemmatisation des verbes est plus difficile que celle des noms et des adjectifs.
De plus, le lemmatiseur général avec prise en compte de la catégorie a une précision supérieure (0.989) à celle du lemmatiseur des verbes. On pourrait expliquer cela par le fait que, dans le lemmatiseur général avec catégorie, la moins bonne performance des verbes est compensée par la meilleure performance pour les noms et les adjectifs, donc en moyennant on a une précision de 0.989 qui reflète la prédiction pour les 3 catégories mélangées.
Enfin, le lemmatiseur général a une précision de 0.9804, inférieure à celle du modèle précédent prenant en compte la catégorie, étant donné que la catégorie aide à déterminer le lemme d'une forme.
On confirme donc notre hypothèse principale selon laquelle le modèle à la meilleure performance est le lemmatiseur catégoriel (des Noms, plus précisément). Ce résultat est logique étant donné que le modèle est entraîné à une catégorie spécifique donc il est performant dans sa catégorie, mais il n'est pas général.

IV. 1. Néologismes

Le lemmatiseur est en théorie capable de traiter des mots qu'il n'a jamais vu et en particulier des mots nouveaux (des néologismes). Imaginons des nouveaux mots des trois catégories étudiées et testons notre lemmatiseur dessus (fichier neologisme.txt).

On obtient :
Loss : 0.109
Accuracy : 0.97

Le lemmatiseur général, testé sur des néologismes, a une précision de 0.97, soit la plus basse qu'on ait pu observer ici. On peut expliquer cela par le fait que les néologismes arbitrairement créés peuvent ne pas être similaires aux mots que le modèle a rencontré lors de son apprentissage. De plus, la taille de l'ensemble de test des néologismes peut influencer la performance du modèle, étant donné qu'on en a créé quelques-uns mais dont la quantité est bien inférieure à celle des données test qu'on avait précédemment. Le modèle a tout de même 0.97 de précision montrant qu'il est capable de lemmatiser des mots inconnus.

2. Généralisation

Le processus de lemmatisation est en bonne partie régulier, est-ce que le lemmatiseur a besoin de beaucoup de données pour apprendre ces régularités ?

Prenons notre lemmatiseur général et sélectionnons seulement une partie des données d'apprentissage (grâce à un pourcentage).

On obtient, pour différents pourcentages :

- 10% (p = 0.1) :
  Loss : 0.222
  Accuracy : 0.958
- 30 % :
  Loss : 0.112
  Accuracy : 0.971
- 50 % (x_train.shape : on a bien 114154/2 = 57077 exemples d'entraînement) :
  Loss : 0.076
  Acuracy : 0.979
- 90 % :
  Loss : 0.05
  Accuracy : 0.985

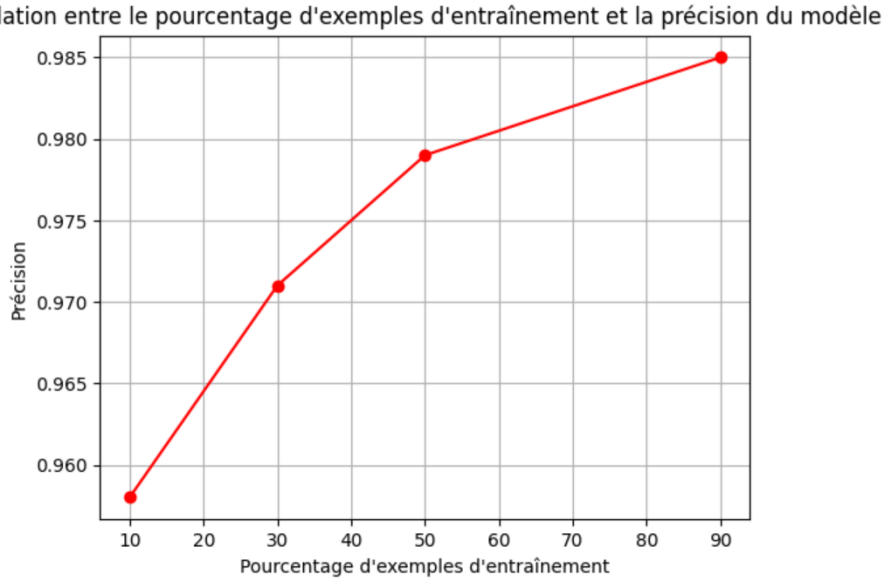

Figure 2 : Précision du lemmatiseur général en fonction du pourcentage d'exemples d'entraînement

On vient bien une corrélation avec une précision qui augmente avec le pourcentage. Quand on ne lit que 10% des lignes des catégories A, V, N, le modèle du lemmatiseur général est précis à 0.958 : cette précision est en effet plus faible car le modèle n'est entraîné que par un faible nombre de données. Et on voit que plus on lui donne des données, plus la précision augmente, plus il arrive à lemmatiser les mots, avec une précision de 0.985 pour 90%. Il semblerait donc que le modèle a besoin de beaucoup de données pour repérer des régularités et les apprendre, et donc pour être très performant.