Université de Montréal

IFT6285 – Traitement automatique des langues naturelles

Devoir 3

---

par:

Eugénie Yockell
(20071932)
Bassirou Ndao
(0803389)

Date de remise: 2 octobre 2020

Nous avons implémenté un programme Python `corrige` qui permet de corriger une liste de mots mal orthographiés. Pour s'y faire, notre programme prend un texte contenant différents mots et leurs fréquences en guise de corpus d'entraînement. Cet ensemble permet de créer un dictionnaire représentant l'ensemble de mots existants ainsi que leurs fréquences. On peut ainsi associer une probabilité à chaque mot définit par $\frac{fréquence\ mot}{total\ mot}$. Afin de trouver la correction d'un nouveau mot, on mesure la distance du nouveau mot avec chaque mot du dictionnaire. On sélectionne ensuite les mots avec la plus petite distance et la plus grand probabilité. Les distances sont définies à l'aide de la librairie `jellyfish`. De plus, nous avons retiré tous les mots qui contenaient des nombres (85 mots) et retiré toutes les ponctuations des mots sauf - et '.

Nous avons mesuré l'effet de l'utilisation de différentes distances sur un ensemble test de 3935 mots. Pour mesurer la précision, nous étudions les cinq meilleurs candidats de correction. Si le mot corrigé attendu est le meilleur candidat, on lui accorde une valeurs de 1, mais si le mot attendu se trouve dans les quatre meilleurs candidats suivant, on lui accorde une valeur de 0.5. Nous avons un rapport de mot correctement corrigé sur le nombre de mot total. Cette mesure de précision semble plus logique qu'une précision où on observerait seulement le meilleur candidat. Regarder seulement le premier candidat est trop sévère, particulièrement dans un cas où on utilise un algorithme plutôt simple. Il n'est pas nécessaire d'être strict dans la définition de réussite. Étudier les cinq premiers candidats au lieu d'étudier le premier candidats permet d'augmenter la précision d'environs 12% pour chaque distance.

Les résultats de précision de correction pour différentes distances sont présentés à la table 1. On conclue que, dans notre cas, la distance de Damerau-Levenshtein est la plus efficace pour mesurer les distances. La distance de Levenshtein représente le nombre d'opérations d'insertion, délétions et substitutions nécessaire pour passer d'un mot à un autre. Tandis que la distance de Damerau-Levenshtein compte aussi l'opération de transpositions. Il est logique que cette distance soit plus efficace, parce qu'on a un plus grand ensemble de mots voisins.

On note aussi que les temps sont plutôt semblables pour chaque type de distance. Ce résultat concorde avec le fait que les fonctions de distance proviennent tous de la même librairie. De plus, la distance de Damerau-Levenshtein est tout de même un peu plus longue que les autres, car elle calcule aussi les opérations de transpositions.

Distance	**Précision** [%]	**Temps** [s]
Levenshtein	72.6 %	346.3
Damerau-Levenshtein	75.7 %	432.5
Jaro	66.4 %	391.2
Jaro-Winkler	65.4 %	385.2
Hamming	39.2 %	280.2

Table 1: Pourcentage de mots correctement corrigés sur un ensemble test de 1000 mots ainsi que le temps nécessaire pour faire les calculs. À noter que les méthodes de Jaro et Jaro-Winkler sont des mesures de similarités.

Comme mentionné plus haut, lorsqu'on reçoit un mot incorrect, on mesure sa distance avec chaque mot du lexique. Nos candidats de correction n'ont alors aucun restriction. Nous avons tenté de restreinte les candidats pour la correction en créant un ensemble de mots phonétiquement semblables et mesurer la distance avec ces mots à l'aide de la librairie `Soundex`. On réduit très fortement le temps de calcul de l'ordre de la minute pour 1000 mots. Par contre, cette méthode diminuait fortement la précision. En effet, sur un ensemble test de 1000 mots, avec la distance de Damerau-Levenshtein, nous avons 76.5% sans `Soundex`, et 61.0% avec `Soundex`. `Soundex` contraint trop fortement les candidats et souvent la correction du mot n'est pas dans les candidats.

Une fois qu'on définit une bonne mesure de distance, notre hypothèse était qu'une des meilleurs façons d'augmenter la précision de l'algorithme est d'augmenter le lexique. Nous nous sommes intéressé à la relation entre la grandeur du lexique et la précision. On remarque sur la figure 2 en Annexe que le précision augmente à mesure qu'on augmente le corpus. À noter que nous avons randomiser le lexique pour sélectionner les tranches. Nous avons alors tenté d'augmenter le lexique avec différent ensemble de données. Ces ajouts diminuaient dramatiquement la précision. Nous avons aussi tenté d'utiliser le `1Bwc` mais en prenant les mots ayant une fréquence minimale de 10. Encore une fois la précision diminuait. Nous avons alors coupé le lexique selon différentes fréquences minimales. Comme on voit à la figure 1, la fréquence minimale optimale est à 200. On ne s'attendait pas à ce résultat. Ça peut être expliqué par le fait que le corpus `1Bwc` peut contenir des mots incorrects plusieurs fois. Par exemple, le mot "olmertÃªÂs" est présent 40 fois. Ces résultats signifient aussi que la probabilité est une facteur très important. Il se pourrait aussi qu'un plus petit lexique fonctionne mieux sur l'ensemble test simplement parce que l'ensemble test est semblable au lexique. Ainsi plus ce dernier est petit avec les mots les plus probables, meilleur est le modèle.

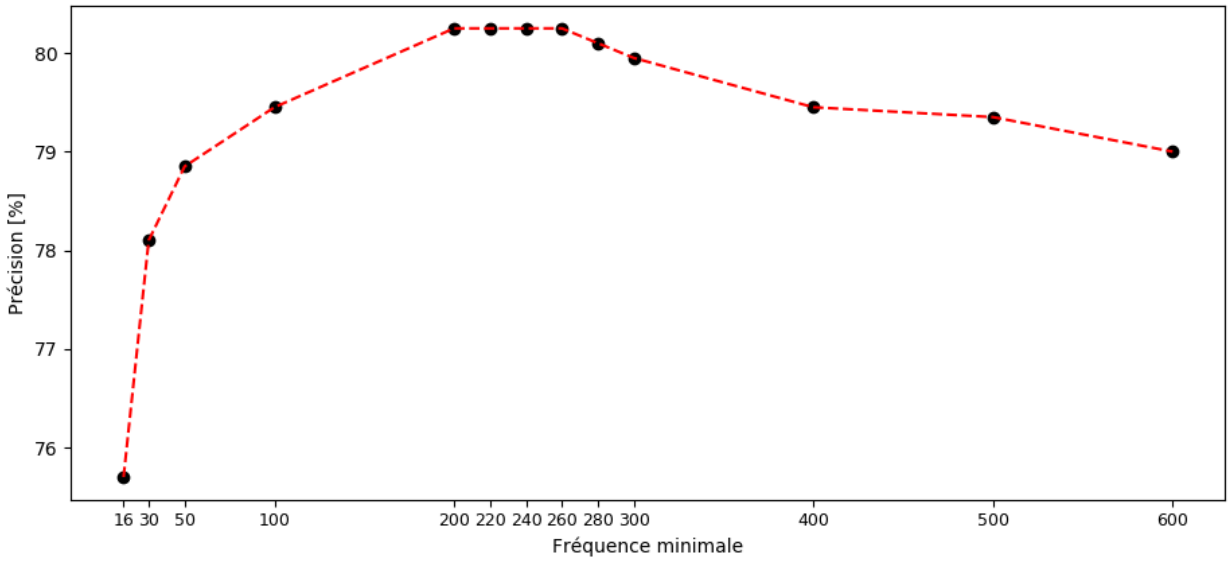

Figure 1: Évolution de la précision en fonction de la fréquence minimale prise pour le lexique. La précision est mesurer sur 1000 mots.

Notre meilleur modèle utilise la distance de Damerau-Levenshtein et utilise les mots du `1Bwc` ayant une fréquence minimale de 200. Sur l'ensemble test complet, nous obtenons une précision 80.2%.

En conclusion, il aurait été intéressant d'explorer d'autres algorithmes phonétiques. On pourrait aussi utiliser des *BK-trees* pour la recherche efficace de candidats. Une autre méthode de sélection de candidats aurait été de considérer tout les mots se trouvant à une distance d'un rayon limite choisie par nos soins. Ces différentes méthodes aurait permis de faire une présélection pertinente de candidats à la correction. D'autre part, une importante amélioration qui pourrait être appliqué serait de prendre en compte le contexte à l'aide de modèles de langue vu en cours comme les modèles n-grammes ou des modèles neuronaux. En effet, la correction d'un mot dépend très souvent de son contexte.

Annexe

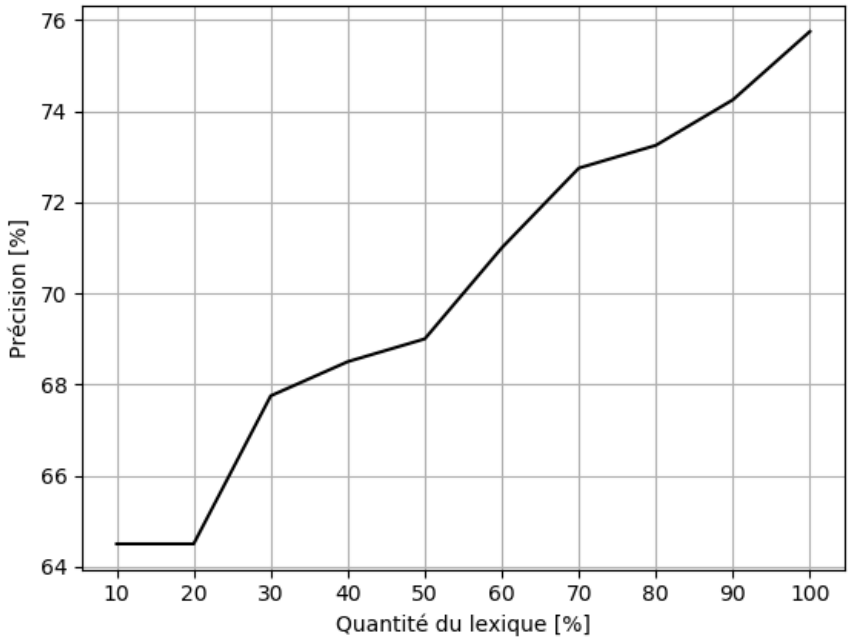

Figure 2: Évolution de la précision en fonction de la portion en pourcentage du lexique. La précision est mesuré sur 200 mots.