Rapport TP1 TLNL : markov

Leader : Emma Mendizabal , Follower : Joey Skaf

5 Novembre 2023

1 Introduction

Ce rapport présente les deux pistes explorées dans le cadre du TP1 du cours de TLNL. L'objectif global de ce TP est de programmer un modèle de langage markovien ngram, d'en mesurer les performances et de réaliser plusieurs tâches telles que la génération d'un nouveau texte. Pour mesurer les performances de ce modèle nous utiliserons principalement l'évaluation extrinsèque par le calcul de perplexité. Pour rappel, le modèle de langage markovien ngram calcule la probabilité d'apparition d'une séquence de n mots au sein d'un corpus donné. L'hypothèse de Markov est la suivante : l'apparition du nème mot dépend seulement des n-1 mots précédents. Ce TP nous permet d'étudier les avantages et inconvénients du modèle.

Nous avons l'occasion d'explorer deux pistes d'approfondissement du sujet : nous choisissons la "parentalité" et la "courbe d'apprentissage". Le rapport nous permet de détailler d'avantage ces pistes : leurs objectifs, notre mise en oeuvre et les résultats obtenus. Ces deux sujets nous ont paru particulièrement intéressants :

- La "parentalité" permet de deviner l'auteur d'un texte, grâce à la comparaison des performances sur de modèles chacun basés sur le corpus d'un auteur. A l'ère de l'IA générative, reconnaître les sources et inspirations d'un écrit s'avère particulièrement utile. Ce sujet nous permet également de réfléchir aux limites de cette détection.
- La "courbe d'apprentissage" permet d'étudier l'évolution des performances d'un modèle en fonction de la taille du corpus sur lequel il a été créé. Faut-il augmenter la taille du corpus pour obtenir de meilleurs résultats ? Le choix de méthode utilisée pour évaluer les performances change-t '-il les résultats ? L'influence du choix des textes est-elle importante ?

2 Modèle Initial

2.1 Piste à creuser 1 : parentalité

2.1.1 Description

La "parentalité" consiste à trouver l'auteur d'un texte en appliquant un modèle de langage sur ce dernier, le modèle ngram ici. Pour cela nous disposons de

plusieurs auteurs, chacun liés à un corpus composé de plusieurs de leurs textes. Nous détaillons ici leur contenu et leur utilité :

Trois auteurs du XIXe siècle :
- *Alexandre Dumas*, avec un corpus composé de 4 textes : Le comte de Monte Cristo, La Reine Margot, Le Vicomte de Bragelonne, Vingt ans après et Les Trois Mousquetaires . Il s'agit du corpus de référence livré avec le sujet.
- *Jules Verne*, avec un corpus composé de 8 textes :Le tour du monde en 80 jours, Autour de la lune, Aventures du capitaine Hatteras, Cinq semaines en ballon, Face au drapeau, L'archipel en feu, Le docteur Ox, Le pilote du danube. Comme Dumas, Verne aime l'action et les dialogues vivants.
- *Honor de Balzac*, avec un corpus composé de 3 textes : La fille aux yeux d'or, L'elixir de longue vie, et Eugénie Grandet. Contrairement à Dumas et Verne, de Balzac préfère la narration, la description des détails sociaux.

Des écrits du XXIe siècle :
- Trois articles issus de *Wikipedia* : Ia wkipedia, Science wikipedia, et Informatique wikipedia. Ces trois textes ont des auteurs différents, mais un même principe explicatif. Leur écriture n'est pas romancée, ni axée sur des sentiments, et ne contient pas de dialogue ni effets de style.
- *Marie Lebert*, avec un texte : le Projet Gutemberg. Comme pour les articles de Wikipédia, le style n'est pas romancé. Le texte aborde l'histoire de ce projet d'accès gratuit à des oeuvres littéraires sur le net.
- Un fichier contenant les paroles d'un titre de *Jul* : En bande organisée. Le style, le vocabulaire utilisé sont très différents des textes présentés précédemment.

Pour créer notre programme de paternalité, nous émettons les hypothèses suivantes :

1. **Première hypothèse** : deux textes d'un même auteur sont davantage similaires que deux textes d'auteurs différents. Pour tester cette hypothèse, nous créons 2 modèles ngram M1 et M2, respectivement à partir d'un texte T1 d'auteur A1 et texte T2 d'auteur A2. Puis nous prenons deux autres textes T1' et T2', toujours des auteurs A1 et A2 respectivement. Pour valider l'hypothèse, la perplexité de T1' doit être meilleure sur M1 que sur M2 et celle de T2', meilleure sur M2.
2. **Deuxième hypothèse** : deux auteurs avec styles similaires auront des scores de perplexité plus proches. Pour tester cette hypothèse, nous comparons les perplexités sur plusieurs textes d'auteurs différents sur un même corpus. Certains auteurs ont des styles similaires (même époque, thèmes abordés, rythme et temps des phrases). Si les perplexités sur ces auteurs sont proches, l'hypothèse est validée.
3. **Troisième hypothèse** : de la même manière que nous devinons l'auteur à partir d'un texte, nous pouvons deviner le titre d'un texte à partir d'une phrase. Pour vérifier cette hypothèse, nous prenons deux textes T1 et T2 et en extrayons une phrase P1 et P2 de tailles similaires. Puis nous

créons un modèle M1 sur le texte restant de T1, idem pour M2. Enfin nous calculons la perplexité de P1 et P2 sur les 2 modèles : la perplexité de P1 doit être plus basse sur M1, idem pour P2 sur M2.

4. **Quatrième hypothèse** : la présence de noms propres dans les textes influe sur le résultat, si l'on calcule la perplexité d'un texte sur un modèle contenant les mêmes noms de personnages, la perplexité sera plus basse. C'est un point important, car nous souhaitons trouver un auteur par rapport à son style et non par rapport aux personnages utilisés. Pour tester cette hypothèse, nous allons comparer les perplexités obtenues avant et après masquage des principaux noms propres sur les textes utilisés pour créer un modèle.
5. **Cinquième hypothèse** : le grammage du modèle a une influence sur la recherche des auteurs. Plus globalement, nous pouvons imaginer qu'un modèle unigram permet plutôt de repérer des champs lexicaux similaires, et que le modèle bigram ou trigram met davantage en valeur des styles d'écriture similaire. Pour tester cette hypothèse, nous calculons les perplexités de deux textes T1 et T2 sur les modèles uni, bi et trigram des corpus d'auteurs A1 et A2. L'hypothèse est validée si nous observons les bons résultats ou si, effectivement, les mauvais résultats sont liés à un mauvais choix de grammage (tenter d'observer un style similaire à l'aide d'un unigram par exemple).

2.1.2 Mise en œuvre

Pour la mise en œuvre de cet algorithme, nous avons effectué plusieurs étapes :

- **Alimenter les corpus de textes**. Pour cela, nous avons majoritairement utilisé le site *www.gutemberg.org*, qui propose des oeuvres entières à lire ou télécharger gratuitement. Nous avons ensuite nettoyé tous ces textes : suppression des caractères indésirables, des mentions de chapitres, etc. Après cette longue étape, nous avons pu tokeniser les textes grâce au script *tokenizer.py* fourni avec le sujet.
- **Développer le code de parentalité**. Pour cela, nous avons fait appel aux scripts déjà réalisés pour ce TP à savoir *ngram.py* et *perplexite.py*. Nous avons adapté notre code pour que les hyperparamètres de création du modèle soient des textes seuls, des corpus, ou des listes de corpus.
- **Créer les tests adaptés à nos hypothèses**. Pour cela, nous avons recréé le script *mask.py* fourni avec le sujet pour l'adapter à nos besoins. Nous avons également extrait des phrases de textes pour tester la troisième hypothèse.

Pendant cette mise en œuvre, nous avons été confrontés à divers challenges :

- **La perplexité**. Joey et moi avons effectué le code chacun de notre côté pour comparer nos résultats. J'obtiens les perplexités exactes sur unigram, et des perplexités proches de celles attendues pour bigram et trigram. Cependant lorsque je teste mon programme sur d'autres textes, je n'obtiens pas un résultat cohérent. Par exemple, la perplexité de T1 sur un modèle lui aussi créé avec T1 donne une perplexité plus grande que sur T2. De son côté, Joey obtient des perplexités proches de celles

attendues sur unigram, bigram et trigram. Cependant, les résultats restent cohérents. **Solution apportée** : Nous choisissons donc l'algorithme de Joey, que nous avons essayé d'optimiser jusqu'à date de rendu pour obtenir l'algorithme le plus juste possible. Pour cela nous avons notamment effectué des tests avec calculs à la main, et vérifié la logique de l'algorithme.

- **La compréhension du script** *masque.py* qui ne servait pas à masquer les noms propres, mais à générer les textes à trous. **Solution apportée** : Nous avons écrit l'algorithme de *masque_noms_propres.py* afin de l'utiliser pour nos nouveaux textes et tester l'hypothèse 4.
- **Le choix des textes**. Difficile de savoir si ces choix sont pertinents sans une étude plus théorique, voire statistique. **Solution apportée** : Nous avons essayé de choisir des textes variés, en grand nombre, et de les nettoyer un maximum (même typologie pour les dialogues, suppression des annexes et notes de vocabulaire, etc ...) pour nous concentrer sur le contenu et l'enchaînement des mots. Nous avons enfin tenté d'équilibrer la taille des corpus des différents auteurs pour une meilleure comparaison des perplexités.

2.1.3 Résultats

Voici les résultats obtenus à l'issue du lancement de notre script "paternalite.py", sur des modèles trigram (sauf pour la vérification de l'hypothèse 3, qui requiert la création de modèles uni, bi et trigrams) et avec une perplexité calculée par estimateurs de Laplace.

Résultats hypothèse 1 :

	Modèle créé sur :		
Perplexité	La Reine Margot (Dumas)	Autour de la lune (Verne)	Auteur trouvé :
Monte Cristo (Dumas)	2140.87	2942.00	Dumas
80 jours (Verne)	2383.45	1700.34	Verne

TABLE 1 – 1er test, Dumas et Verne

	Modèle créé sur :		
Perplexité	La Reine Margot (Dumas)	l'Elixir de longue vie (Balzac)	Auteur trouvé :
Monte Cristo (Dumas)	2140.87	1077.87	Balzac
Yeux d'or (Balzac)	3482.11	1076.11	Balzac

TABLE 2 – 2ème test, Dumas et Balzac

Ces résultats nous permettent de contredire l'hypothèse 1. En effet, si dans la **Table 1** les meilleures perplexités sont observées pour des modèles créés sur un texte du même auteur, ce n'est pas le cas dans la **Table 2**. En regardant

les textes de plus près, nous nous apercevons *La Reine Margot* de Dumas présente beaucoup d'intrigue de cour et de légereté contrairement à l'oeuvre *Monte Cristo* et à l'*Elixir de longue vie*, plus sombres, axés sur la vengence et le pouvoir. Grâce à cette méthode, nous pouvons donc supposer que les bons auteurs sont détectés sous certaines conditions seulement. Vérifions par exemple si l'hypothèse 2 est bonne, soit si deux auteurs aux styles similaires ont des perplexités proches. Nous utiliserons cette fois-ci un corpus pour mieux cerner le style global de l'auteur.

Résultats hypothèse 2 :

Perplexite	Corpus Verne
Monte Cristo (Dumas)	4635.52
Yeux d'or (Balzac)	4381.43
Autour de la lune (Verne)	2947.84
Science wikipedia (Wikipédia)	5183.51
Le projet Gutemberg (Lebert)	4867.57
En bande organisée (Jul)	12855.07

TABLE 3 – Proximité entre les perplexites, sur corpus Jules Verne

Ces résultats nous permettent de valider l'hypothèse 2 : la perplexité de Balzac est plus petite que celle de Dumas, car le style se rapproche davantage de celui de Verne, mais les valeurs restent assez proches car il s'agit de deux romans du même siècle. L'article de wikipédia a un style et un vocabulaire assez différent d'où sa haute perplexité. Enfin le fameux titre de Jul est très loin de celui de Jules, ce qui lui vaut la plus haute perplexité.

Résultats hypothèse 3 :

	Modèle créé sur :		
Perplexité	Le Vicomte de B	La Reine Margot	Oeuvre trouvée :
Phrase Vicomte de B	1135.00	1878.99	Vicomte de B
Phrase Reine Margot	3493.14	1538.47	Reine Margot

TABLE 4 – Parentalité des phrases sur deux textes de Dumas

Ce tableau permet de vérifier l'hypothèse 3. Nous pouvons donc, de manière similaire, déterminer le texte d'origine d'une phrase. Cette tâche paraît par ailleurs plus facile que la détection d'un auteur : on peut imaginer que le corpus formé par le texte contient beaucoup de phrases et permet sûrement de mieux cerner le style de l'ouvrage. Cette question sera abordée dans le deuxième approfondissement.

Résultats hypothèse 4 :

	Modèle créé sur :		
Perplexité	Corpus Dumas	Corpus Balzac	Auteur trouvé :
Le Comte de MC	2878.44	3225.62	Dumas
MC avec corpus masqués	2875.59	3219.97	Dumas
La fille aux yeux d'or	4656.11	2976.00	Balzac
La fille corpus masqués	4699.95	2970.65	Balzac

TABLE 5 – Effet du masque sur les corpus Dumas et Balzac

Ces résultats nous permettent de valider l'hypothèse 4 : masquer les noms propres permet de modifier la perplexité, même faiblement. Nous pouvons également observer que la nouvelle perplexité est plus haute si les noms propres sont également présents dans le texte étudié, plus basse sinon.

Résultats hypothèse 5 :

	Modèle créé sur :		
Perplexité	Corpus Dumas	Corpus Balzac	Auteur trouvé :
Monte Cristo n=1	521.95	725.97	Dumas
Monte Cristo n=2	631.57	1389.89	Dumas
Monte Cristo n=3	2878.44	3225.62	Dumas
Yeux d'or n=1	647.56	526.26	Balzac
Yeux d'or n=2	1021.69	1150.38	Dumas
Yeux d'or n=3	4656.11	2976.00	Balzac

TABLE 6 – Proximité des scores, Dumas et Balzac

Ces résultats nous permettent de valider partiellement l'hypothèse 5 : le grammage a effectivement une influence sur les résultats, comme le montre l'erreur sur le texte « La fille aux yeux d'or » de Balzac pour un modèle bigram. Cependant nos suppositions sur les grammages adaptés à certaines situations ne fonctionnent pas ici. Par exemple, les modèles bigram et trigram donnent des résultats opposés, et mettent donc en valeur des éléments différents. De même les perplexités unigram entre les corpus varient peu alors que les champs lexicaux diffèrent (aventure et amitié contre psychologie et décadence). Le modèle unigram semble donc capter le style également, grâce au repérage des temps de verbes, des mots caractéristiques de l'époque, etc. Seul l'agencement dans les phrases ne peut être capturé avec l'unigram.

2.2 Piste à creuser 2 : Courbe d'apprentissage

2.2.1 Description

L'approfondissement "courbe d'apprentissage" consiste à étudier l'évolution des performances d'un modèle en fonction de la taille du corpus sur lequel il a été créé. Pour cela nous disposons des différents textes présentés précédemment. Nous utilisons les corpus détaillés ci-dessous :

- $Corpus\_Verne$ composé des 8 textes dans cet ordre : Autour de la lune, Aventures du capitaine Hatteras, Cinq semaines en ballon, Face au drapeau, L'archipel en feu, Le docteur Ox, Le tour du monde en 80 jours, et Le pilote du Danube.
- $Corpus\_Verne\_reversed$ composé des mêmes 5 textes, dans l'ordre inverse (de Le pilote du Danube jusqu'à Autour de la lune)
- $Corpus\_Full$ composé des textes de tous les corpus en notre possession : les corpus de Dumas, Balzac, Verne et Wikipedia pour un total de 18 textes.

Pour créer notre programme, nous émettons les hypothèses suivantes :

1. **Première hypothèse** : dans le corpus d'un auteur donné, plus le nombre de textes d'un même style est élevé, plus le modèle permet d'obtenir une faible perplexité. La condition du même auteur nous permet de ne pas regarder plusieurs paramètres en même temps, mais de nous concentrer sur la taille du corpus seulement. Pour tester cette hypothèse, nous utilisons N textes du même auteur et créons N modèles ngram : le 1er basé sur le 1er texte seulement, le 2ème basé sur les 1er et 2ème textes, etc ... Nous calculons la perplexité d'un autre texte du même auteur sur ces corpus et traçons la courbe d'apprentissage.
2. **Deuxième hypothèse** : si l'on modifie l'ordre dans lequel on ajoute les textes au corpus, on observe une évolution des résultats différente. En effet, certains textes du corpus peuvent avoir une proximité plus grande avec le texte de test : proximité de thème, importance du dialogue ou texte très narratif ... Pour cela, nous modifions l'ordre d'ajout des textes dans le corpus et comparons les résultats. Nous ajoutons également des masques pour réduire les correspondances entre noms propres.
3. **Troisième hypothèse** : si l'on augmente la taille du corpus en mélangeant les styles, on obtient une performance différente sur la perplexité et sur le texte à trou. Par exemple, nous pouvons supposer que la variété des phrases rencontrées bénéficie au remplissage du texte à trou, mais que la variance des possibilités est trop élevée pour obtenir une bonne perplexité. Pour vérifier cette hypothèse, nous créons un corpus avec des textes de plusieurs auteurs différents. Puis sur le texte d'un autre auteur, nous calculons la perplexité et les performances du texte à trou selon la taille du corpus et comparons l'évolution.
4. **Quatrième hypothèse** : si l'on essaie de générer un texte, on obtient de meilleurs résultats à mesure que l'on remplit le corpus. Pour tester cette hypothèse, nous lançons le programme genere.py sur des corpus de taille croissante.

2.2.2 Mise en œuvre

Pour la mise en œuvre de cet algorithme, nous avons effectué plusieurs étapes :

- **Alimenter les corpus de textes**. Pour cela, nous avons à nouveau utilisé les textes nettoyés pour l'approfondissement "parentalité".
- **Développer le code**. Pour cela, nous avons fait appel aux scripts déjà réalisés pour ce TP à savoir *ngram.py*, *perplexite.py*, *devine.py* et *genere.py*. Nous avons adapté notre code pour que les hyperparamètres de création du modèle soient des listes de corpus.
- **Créer les tests adaptés à nos hypothèses**. Pour cela, nous avons nommé nos différents corpus et procédé de manière méthodique pour afficher nos résultats.

Pendant cette mise en œuvre nous avons été confrontés à un challenge en particulier :

- **Le choix de la composition des corpus**. Difficile de savoir si ces choix (quels textes, quel nombre de textes) sont pertinents sans une étude plus théorique, voire statistique. **Solution apportée** : Nous avons essayé d'utiliser des textes variés, pour réaliser différents types de tests. Nous aurons ainsi des résultats à discuter. Nous avons également retiré des corpus les textes dont on calcule la perplexité, pour éviter de biaiser les résultats.

2.2.3 Résultats

Résultats hypothèse 1 :

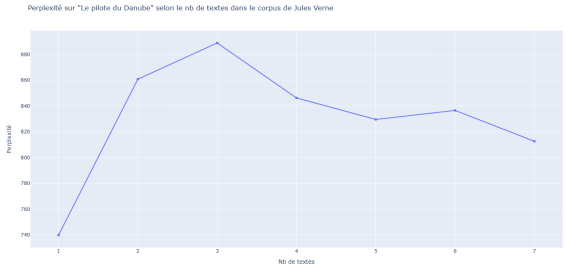

FIGURE 1 – Perplexité sur Le Pilote du Danube / nb de textes corpus Verne

Ces résultats nous permettent de réfuter l'hypothèse 1. Ici sur un corpus de Jules Verne, la perplexité sur le texte *Le pilote du Danube* augmente, puis baisse. L'études des prochaines hypothèses nous permettra d'y voir plus clair sur les causes de cette évolution.

Résultats hypothèse 2 :

FIGURE 2 – Corpus Verne Reversed

FIGURE 3 – Corpus Verne Shuffled

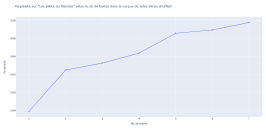

FIGURE 4 – Corpus Verne Shuffled trigram

Ces résultats nous permettent de valider l'hypothèse 2 : changer l'ordre des textes dans le corpus nous permet d'observer des évolutions de perplexité différentes. Pour le corpus reversed, la tendance est à la hausse. Pour le corpus shuffled, la descente se fait par pics. Nous avons également l'influence du grammage, déjà mise en avant dans la partie d'approfondissement « parentalité ». Cependant en regardant les **Figures 1 à 4**, on observe que la perplexité a tendance à se stabiliser. Un autre essai sur un plus grand corpus nous permettrait de valider cette nouvelle hypothèse :

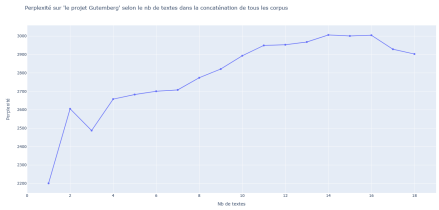

FIGURE 5 – Perplexité sur Le Projet Gutemberg / nb de textes corpus full

Ces résultats nous permettent de valider la nouvelle hypothèse : plus le nombre de textes dans le corpus augmente, plus la perplexité se stabilise. Ainsi, un bon moyen de comparer des modèles est de prendre des corpus de grande taille à chaque fois. Cela n'assure pas la meilleure perplexité, mais assure une certaine stabilité.

Résultats hypothèse 3 :

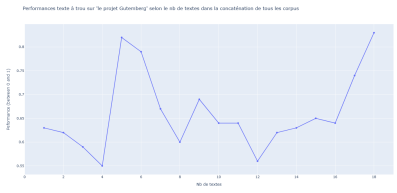

FIGURE 6 – Performances texte à trou Projet Gutemberg

FIGURE 7 – Performances texte à trou 2 Projet Gutemberg

Ces résultats nous permettent de valider partiellement l'hypothèse 3 : l'évolution des performances observées avec le texte à trou et la perplexité ne sont pas les mêmes, mais l'exemple proposé dans l'hypothèse est mauvais. Sur la

Figure 6, qui utilise exactement les mêmes corpus que la **Figure 5** on observe que l'évolution est par pics et que 18 textes ne suffisent pas à la stabiliser. Le résultat dépend donc vraiment des textes ajoutés, et pas forcément de leur nombre. Par ailleurs si l'on change les indices des trous à remplir, l'évolution change totalement (**Figure 7**).

Résultats hypothèse 4 :

Génération sur modèle Trigram avec corpus full
1 texte : '<s> <s> — oui , dit - il . </s> le de la mole en s' inclinant devant elle ce' 2 textes : '<s> <s> — je ne sais pas . </s> le il dit , et que vous avez raison ; mais' 3 à 10 textes : '<s> <s> — je ne sais pas ce que vous avez raison , dit - il . </s> le de' 11 à 18 textes : '<s> <s> — je ne sais pas ce que vous avez raison , dit - il . </s> le et'

Ces résultats nous permettent de valider l'hypothèse 4 : à mesure que l'on ajoute des textes, la génération de phrases est plus pertinente. Nous observons également qu'ici, la génération se stabilise rapidement. Cependant, nous sommes loin du niveau de performance proposé par ChatGPT : même avec un grand nombre de texte, les limites du modèle ngram empêchent la génération de textes cohérents. A travers quelques essais, nous avons par exemple été confrontés aux problèmes de polysémie : le "don" Juan est différent du "don" de la divination.

3 Conclusions et perspectives

En conclusion, ces approfondissements nous ont permis de réfléchir à la pertinence d'un modèle ngram selon la taille et le contenu du corpus, pour diverses tâches.

Notre travail sur la "parentalité" nous a fait questionner les capacités d'un modèle ngram à détecter un auteur à partir de calculs de perplexités. Nous avons découvert que la perplexité seule sur un petit modèle ne nous permettait pas de conclure sur le nom d'un auteur (hypothèse 1), mais que les valeurs pouvaient permettre de détecter proximités de styles et sujets d'écritures (hypothèse 2). De la même manière, l'appariemment d'une phrase à son texte est possible voire plus facile (hypothèse 3). Pour améliorer notre méthode, il est pertinent de supprimer les noms propres de nos modèles (hypothèse 4). Enfin, le grammage a une forte influence sur nos résultats. Difficile d'imaginer quel grammage serait le plus pertinent pour un cas donné, donc tester plusieurs grammages semble être une bonne idée (hypothèse 5). Attention cependant aux problèmes de rareté pour des grammages trop élevés (loi de Zipf).

Quand à la "courbe d'apprentissage", elle nous permet d'observer que la qualité intrinsèque et extrinsèque d'un modèle est bien entendu liée au nombre de textes utilisés pour le créer, mais aussi à leur qualité, leur ordre dans le corpus, et au grammage (hypothèse 1 et 2). Cette courbe d'apprentissage n'est pas monotone, mais semble se stabiliser à mesure que l'on augmente le nombre de textes. Le type d'évaluation a aussi son importance : pour un texte à trou, les résultats varient beaucoup et un corpus plus grand serait nécessaire pour conclure réellement (hypothèse 3). Nous pouvons seulement supposer que pour le texte à trou, seule la nature des textes change le résultat, non leur nombre. Enfin, la génération de texte semble avoir de meilleurs résultats lorsque l'on augmente la taille du corpus, mais d'autres modèles sont à privilégier pour obtenir de vraies phrases cohérentes (hypothèse 5).

Ainsi, le modèle ngram avec hypothèse de Markov semble présenter des limites sur le sujet de parentalité. La détection de plagiat d'auteur par exemple peut s'inspirer de la méthode présentée, mais ne doit pas s'en contenter. Détecter les proximités de styles entre auteurs semble ici plus accessible. Quand à la taille du corpus,. si la nécessité d'augmenter le corpus dépend de la tâche visée, elle est importante pour l'évaluation par perplexité notamment.

Pour aller plus loin dans ce travail, nous pourrions méler ce modèle ngram à des modèles plus récents, tels que les LLMs, pour croiser les résultats et apporter des conclusions encore plus précises. Peut-être le modèle ngram nous permet-il de nous concentrer sur des éléments plus simples, et donc plus proches de la compréhension du langage par le cerveau humain (par example, détecter les similarités entre textes nous est plus facile que de détecter l'auteur lui-même). Nous pourrions aussi, pourquoi pas, tenter une génération en chaîne : à partir du corpus d'un auteur (le plus grand possible), générer un nouveau roman (avec un générateur optimisé basé sur plusieurs modèles). Puis ajouter ce dernier au corpus, et recommencer. Il serait intéressant de voir si le style des nouveaux romans finit par se distinguer des autres.