Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition),
Traitement Automatique des Langues Naturelles (TALN, 27e édition),
Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition)
Nancy, France, 08-19 juin 2020

Calcul de similarité entre phrases : quelles mesures et quels descripteurs ?

Davide Buscaldi[1], Ghazi Felhi[1], Dhaou Ghoul[2],
Joseph Le Roux[1], Gaël Lejeune[2] Xudong Zhang[1]
(1) Sorbonne Paris Nord, LIPN, 99 Avenue Jean Baptiste Clément, 93430 Villetaneuse
(2) Sorbonne Université, 1 rue Victor Cousin, 75005 Paris, France
(1)prenom.nom@lipn.univ-paris13.fr,
(2) prenom.nom@sorbonne-universite.fr

RÉSUMÉ

Cet article présente notre participation à l'édition 2020 du Défi Fouille de Textes DEFT 2020 et plus précisément aux deux tâches ayant trait à la similarité entre phrases. Dans notre travail nous nous sommes intéressé à deux questions : celle du choix de la mesure du similarité d'une part et celle du choix des opérandes sur lesquelles se porte la mesure de similarité. Nous avons notamment étudié la question de savoir s'il fallait utiliser des mots ou des chaînes de caractères (mots ou non-mots). Nous montrons d'une part que la similarité de Bray-Curtis peut être plus efficace et surtout plus stable que la similarité cosinus et d'autre part que le calcul de similarité sur des chaînes de caractères est plus efficace que le même calcul sur des mots.

ABSTRACT

Sentence Similarity : a study on similarity metrics with words and character strings

This article details the participation of the Sorbonne team, composed of researchers from Sorbonne Paris Nord (LIPN lab) and Sorbonne University (STIH lab) to the 2020 Deft challenge. We participated in the two tasks involving similarity measurement. We have been interested in two questions : first of all choosing the appropriate similarity measure and secondly choosing the appropriate features to construct the vectors. We show that (I) the Bray-Curtis similarity can be more efficient and more stable than a classical cosine distance and (II) that character n-grams tend to be more efficient for similarity tasks without needing fine-tuning or data description (lemmatization ...).

MOTS-CLÉS : similarité, n-grammes de caractères, distance euclidienne, distance de Bray-Curtis.

KEYWORDS: similarity, character n-grams, euclidean distance, Bray-Curtis distance.

1 Introduction

Cette édition 2020 du défi Fouille de Textes était principalement consacrée aux données médicales et comprenait trois tâches : (I) identification du degré de similarité entre paires de phrases (parallèles et non parallèles), (II) identification des phrases parallèles possible pour une phrase source dans le domaine médical et (III) extraction d'information sur des cas cliniques dans des textes biomédicaux. Les détails sur le processus de collecte et d'annotation des données dans l'article introducteur du défi (Cardon *et al.*, 2020).

Atelier DÉfi Fouille de Textes, pages 14–25.
HAL : hal-02784738.

Notre travail s'est concentré sur les tâches 1 et 2 que nous avons traité sous l'angle des mesures de similarité. Nous avons conçu une architecture extrêmement simple, que l'on peut sans doute qualifier de *baseline* améliorée, exploitant des mesures de similarité sur des vecteurs. Notre contribution est de réfléchir d'une part aux bonnes manières de construire ces vecteurs, quelles caractéristiques ou dit autrement quelles opérandes, et d'autre part sur les meilleures manières de comparer ces représentations, la recherche en quelque sorte des bons opérateurs.

Dans la Section 2 nous présenterons quelques grandes lignes des approches possibles en calcul de similarité puis dans la Section 3 nous exposerons la méthode que nous avons développé pour ce défi et enfin dans la Section 4 nous présenterons les résultats obtenus et quelques éléments de discussion.

2 Approches en calcul de similarité

Le calcul de similarité est une tâche essentielle du Traitement Automatique de données, textuelles ou non, et a toujours reçu une attention particulière de la communauté scientifique. Si l'on se restreint aux données textuelles, le besoin est avant tout venu de besoin pour la recherche d'information. Il s'agit en effet de pouvoir d'une part de mesurer le degré de proximité entre des documents et d'autre part de pouvoir identifier, et ordonner, la liste des documents les plus pertinents à offrir en réponse à une requête dans un moteur de recherches.

Assez classiquement, le défi posé ici est d'identifier les observables permettant de représenter numériquement des documents et de choisir les mesures appropriées pour en déduire la meilleure mesure de similarité pour une tâche donnée. Les données textuelles sont représentées informatiquement parlant comme des séquences de caractères, sans représentation linguistique autre l'ordre des caractères dans la séquence. Dès lors pouvoir comparer deux chaînes autrement que par une opération de recherche d'identité stricte impose d'identifier des caractéristiques internes de ces chaînes qui vont permettre d'identifier des opérandes sur lesquelles la comparaison pourra porter, ce choix n'étant pas sans impact sur les résultats (Mehdad & Tetreault, 2016). Les opérandes peuvent être de deux grandes catégories : (I) les sous-chaînes de caractères elles mêmes, les formes brutes, dont les mots graphiques sont un sous-ensemble, et (II) les redescriptions, généralement calculées à partir des mots, par exemple la racinisation ou la lemmatisation. Le type de redescription utilisé affectera le qualificatif que l'on va donner à la similarité calculée : si la redescription encode des propriétés syntaxiques on parlera plus facilement de similarité syntaxique, si elle encode des propriétés sémantiques alors on parlera de similarité sémantique… Ensuite, pour pouvoir appliquer des opérateurs de comparaison on aura deux grands types d'approches pour la similarité : d'une part la recherche de similarités séquentielles calculées par comparaison, sous-chaînes communes ou encore distance d'édition, et d'autre part la vectorisation qui permet de se placer dans un cadre méthodologique bien adapté au calcul automatique.

3 Quelle mesures de similarité et quels descripteurs ?

Notre approche était fondée sur une représentation assez simple du problème : que pouvaient donner des mesures de similarité très simples appliquées sur la tâche 2, tâche la plus simple puisqu'elle consistait à extraire la phrase la plus proche parmi trois candidates. L'approche la plus immédiate, en tout cas celle qui nous a paru comme telle, a été de calculer une vectorisation en mots et d'appliquer

une simple mesure de similarité cosinus pour classer les phrases candidates. Les premiers résultats étaient très élevés (au-delà de 93% de bons appariements) ce qui laissait à penser que la tâche était assez aisée. Nous avions envisagé d'exploiter des méthodes sophistiquées à base de plongements de mots, spécialisés ou non sur le domaine médical, mais il nous a semblé qu'il serait intéressant de partir de cette *baseline* plutôt convaincante. Plusieurs auteurs se sont intéressés à cette question de la capacité des *baseline*, pour peu qu'on leur porte suffisamment d'attention, à avoir des résultats équivalents (Moreno & Dias, 2014) voire supérieurs (Rendle *et al.*, 2019) à des approches état de l'art. C'est aussi une question qui s'est posé régulièrement dans la communauté DEFT, par exemple lorsque la *baseline* conçue par les organisateurs du Défi 2019 s'est avérée plus performante que des approches plus complexes implantées par les participants du défi (Grabar *et al.*, 2019). La question scientifique est tout à fait intéressante puisqu'il s'agit aussi de mesurer la valeur ajoutée apportée par des méthodes sophistiquées qui sont souvent plus gourmandes en ressources : taille des jeux de données d'entraînement, disponibilité de données linguistiques (lexiques, plongements de mots ...) pour une langue et/ou un domaine donné ou encore tout simplement coût en temps de calcul (Strubell *et al.*, 2019).

Dès lors, nos investigations se sont portées sur deux objectifs : d'une part chercher dans la méthode elle même quelles pouvaient être les points d'amélioration et d'autre part regarder si cette méthode pouvait être adaptée pour traiter également la tâche 1. Nous avons donc exploré la question des opérations appliquées, les mesures de similarités (Section 3.1) d'une part et les opérandes sur lesquelles les mesures allaient porter, mots ou chaînes de caractères, d'autre part (Section 3.2.

3.1 Choix des mesures de similarité

Différentes mesures de similarité peuvent être utilisées pour rapprocher des documents représentés sous forme vectorielle. Nous avons cherché à comparer différentes mesures de similarité s'appuyant sur des vecteurs. Dans tous ces cas, on va comparer des phrases représentées sous la forme de deux vecteurs V et W, V représentant la phrase à apparier et W représentant chacun des candidats tout à tour. Assez naturellement, on va se tourner vers la distance euclidienne mais il peut être intéressant de se tourner vers des variantes telles que la distance de Manhattan ou la distance de Minkovski. Pour rappel, on peut réécrire la distance de Manhattan pour la faire passer de la forme :

$DistManh = \sum_{i=1}^{n} |x_i - y_i|$

à la forme : $DistManh = \sqrt[1]{\sum_{i=1}^{n} (x_i - y_i)^2}$

De sorte que la filiation avec la distance de Minkovski et la distance euclidienne devienne plus évidente :

$DistEucl = \sqrt[2]{\sum_{i=1}^{n} (x_i - y_i)^2}$ et $DistMink = \sqrt[3+]{\sum_{i=1}^{n} (x_i - y_i)^3+}$

Avec les mesures précitées, les segments les plus proches sont ceux qui minimisent la distance. Nous intégrons ensuite des mesures de similarité plus classiques en TAL dont la similarité cosinus, l'indice de Jaccard et le coefficient de Dice qui sont souvent considérés comme une des mesures de référence quand il est question de similarité textuelle (Huang, 2008). Nous avons également intégré la dissimilarité de Bray-Curtis (Bray & Curtis, 1957).

— $Cosinus = \frac{V.W}{||V||.||W||}$
— $Jaccard = \frac{|set(V) \cap set(W)|}{|set(V) \cup set(W)|}$

— $Dice = \frac{2*|set(V) \cap set(W)|}{|set(V) \cup set(W)|}$

— $Bray - Curtis = \frac{2\sum_{i=1}^{n} min(V[i], W[i])}{\sum_{i=1}^{n} (V[i]+W[i])}$

Nous pouvons donc d'ores et déjà tester une première *baseline* qui va exploiter ces distances et similarités en travaillant simplement sur les effectifs des mots graphiques, sans pré-traitement ni élimination de *stop-words*. La tokenisation est effectuée par un simple découpage sur les espaces. Nous avons appliqué cette *baseline* sur les 572 instances du jeu d'apprentissage de la tâche 2 (Cardon *et al.*, 2020). Parmi les trois candidats proposés on choisit celui qui présente la similarité la plus grande (ou la distance la plus faible pour les trois premières mesures décrites ci-dessus) sans seuil d'aucune sorte [1].

Les résultats sont présentés dans le tableau 1, nous pouvons voir deux choses : d'une part la tâche est assez facile et d'autre le choix de la mesure de similarité peut avoir son importance.

	Bons résultats	MAP
Distance Euclidienne	534/572	0,9336
Distance de Minkowski	534/572	0,9336
Distance de Manhattan	536/572	0,9371
Similarité Cosinus	553/572	0,9668
Coefficient de Dice	553/572	0,9668
Similarité de Bray-Curtis	557/572	0,9738
Indice de Jaccard	559/572	0,9772

TABLE 1 – Résultats de l'application des mesures de similarité, au grain mot sans pré-taritement ni pondération, sur la tâche 2 triés par ordre croissant de MAP

Afin d'enrichir légèrement et à faible coût la représentation, nous montrons dans le tableau 2 les résultats obtenus avec des représentations en n-grammes de mots en prenant différents intervalles de valeur de N de 1 à 4. Les résultats en haut à gauche de chaque sous-tableau ($N_{min} = N_{max} = 1$) correspondent donc aux valeurs de la *baseline* du tableau 1. Nous pouvons voir que tenir compte des bi-grammes de mots permet d'améliorer les résultats, même si pris individuellement les bi-grammes offrent une représentation moins dense et moins efficace que les unigrammes. Par contre dans cette configuration l'impact des tri-grammes n'est pas significativementpositif. Nous proposns ensuite dans le tableau 3 les résultats avec une pondération tf-idf, où l'Idf est calculé en prenant compte de l'ensemble du corpus d'apprentissage. Nous avons laissé les résultats avec l'indice de Jaccard à titre d'illustration puisqu'ils ne semble pas pertinent de combiner tf-idf et indice de Jaccard. Nous pouvons voir que la pondération tf-idf ne permet pas d'améliorer les résultats. Il y a certainement plusieurs raisons derrière cela : le fait que le nombre de doucments dans le corpus est peut être trop petit pour que le Tf-Idf puisse offrir une plus-value, peut être que l'utilisation d'Okapi BM-25 améliorerait les résultats comme cela avait pu être montré par (Claveau, 2012) mais ce n'était pas le cas ici. On peut penser aussi que la tâche étant simple, cette *baseline* atteignait tout simplement un plafond de verre. Nous montrerons dans la section suivante qu'il n'en est rien, en travaillant sur des n-grammes de caractères nous arrivons à améliorer encore un peu les résultats.

1. Pas de seuil de score similarité minimale ou d'écart minimal de similarité entre deux candidats

	max=1	max=2	max=3	max=4
min=1	0.9773	**0.979**	0.9773	0.9738
min=2		0.958	0.9545	0.9563
min=3			0.9126	0.9108
min=4				0.8374

(a) Indice de Jaccard

	max=1	max=2	max=3	max=4
min=1	0.9668	**0.972**	0.965	0.965
min=2		0.9545	0.9545	0.9545
min=3			0.9126	0.9108
min=4				0.8374

(b) Coefficient de Dice

	max=1	max=2	max=3	max=4
min=1	0.9668	**0.972**	0.9668	0.9633
min=2		0.9545	0.9545	0.951
min=3			0.9126	0.9108
min=4				0.8374

(c) Similarité cosinus

	max=1	max=2	max=3	max=4
min=1	0.9738	**0.9773**	0.9755	0.972
min=2		0.958	0.9563	0.9563
min=3			0.9126	0.9108
min=4				0.8374

(d) Similarité de Bray-Curtis

TABLE 2 – Résultats en MAP, sans pondération, sur les données d'apprentissage avec des n-grammes de mots et différents intervalles de longueur de 1 à 4

	max=1	max=2	max=3	max=4
min=1	0.4073	0.4091	0.4021	0.4056
min=2		0.4353	0.4283	0.4161
min=3			**0.4371**	0.4196
min=4				0.4266

(a) Indice de Jaccard

	max=1	max=2	max=3	max=4
min=1	0.9633	**0.9668**	0.9633	0.9615
min=2		0.9563	0.9528	0.9493
min=3			0.9091	0.9091
min=4				0.8374

(b) Coefficient de Dice

	max=1	max=2	max=3	max=4
min=1	0.9685	**0.972**	0.9668	0.9668
min=2		0.9545	0.9563	0.9563
min=3			0.9143	0.9108
min=4				0.8374

(c) Similarité cosinus

	max=1	max=2	max=3	max=4
min=1	0.9738	**0.9773**	0.972	0.972
min=2		0.958	0.9563	0.9563
min=3			0.9143	0.9143
min=4				0.8392

(d) Similarité de Bray-Curtis

TABLE 3 – Résultats en MAP, avec pondération Tf-Idf, sur les données d'apprentissage avec des n-grammes de mots et différents intervalles de longueur de 1 à 4 (NB : résultats avec l'indice de Jaccard, pour information)

3.2 Choix des descripteurs : mots ou chaînes de caractères

La dimension que nous avons souhaité examiner ensuite a donc été le choix des des descripteurs, en d'autres termes le choix des opérandes sur lesquelles allait porter le calcul de similarité. Une piste naturelle, et employée par d'autres participants du défi, était certainement d'avoir recours à des représentation plus riches des mots de manière notamment à mieux encoder la synonymie et plus généralement la proximité sémantique. Ceci pouvait prendre la forme d'une racinisation, d'une lemmatisation ou de l'exploitation de plongement de mots. Ici nous avons choisi une méthode qui se rapproche dans une certaine mesure de la racinisation mais qui permet aussi d'encoder des relations séquentielles entre les mots : l'utilisation de n-grammes de caractères.

L'idée, que l'on peut résumer sous la forme « Tout ce que nous savons faire avec des mots, nous

devrions pouvoir le faire avec des chaînes de caractères »[2] (Umemura & Church, 2009), est double : d'une part rechercher les limites des analyses au grain mot d'un point de vue efficacité et d'autre part d'un point de vue plus épistémologique interroger la pertinence de chercher systématiquement à travailler à un grain d'analyse "interprétable" tel que le mot alors que l'utilisation des chaînes de caractères est plus naturelle pour la machine et sachant que la tokenisation n'est pas une tâche de TAL réglée à l'heure actuelle dans tous les contextes. Ce qui va amener à chercher à standardiser les textes comme on le fait souvent pour les *tweets*(Nebhi *et al.*, 2015), les textes bruités (issus d'océrisation par exemple) ou encore les textes anciens (Gabay *et al.*, 2019).

Pour examiner cela nous avons simplement utilisé les mêmes mesures de similarité mais en les appliquant cette fois sur des vecteurs de N-grammes de caractères (Figure 1). Nous pouvons voir que les résultat sont systématiquement supérieurs à ceux obtenus au grain mot dès lors que la représentation inclut les 2-grammes de caractères et ceci reste vrai quel que soit le calcul de similarité utilisé. On peut observer que les résultats avec l'indice de Jaccard augmentent très peu lorsque l'on augmente la taille des n-grammes de caractères. Cela ne semble pas très étonnant puisque le fait de binariser les valeurs des dimensions (0 ou 1) va lisser très fortement l'effet de redondance amené par les n-grammes de caractères. Par exemple pour la chaîne `tototo`, les 4-grammes `toto` et `otot` vont tout deux être représentés de la même manière que le 5-gramme `totot`. Les résultats obtenus avec le coefficient de Dice sont notablement moins stables mais on observe un pic, supérieur à ce que l'on voit avec l'indice de Jaccard, avec $N_{min} = 4$ et $6 \leq N_{max} \leq 8$. En moyenne les résultats avec la similarité Cosinus sont supérieurs mais le pic est moins élevé. Enfin, les résultats obtenus avec la distance de Bray-Curtis, s'ils ne montraient pas le même pic que le coefficient de Dice, offraient selon nous le meilleur compromis entre stabilité et efficacité puisque l'on pouvait s'abstenir de définir un seuil minimal.

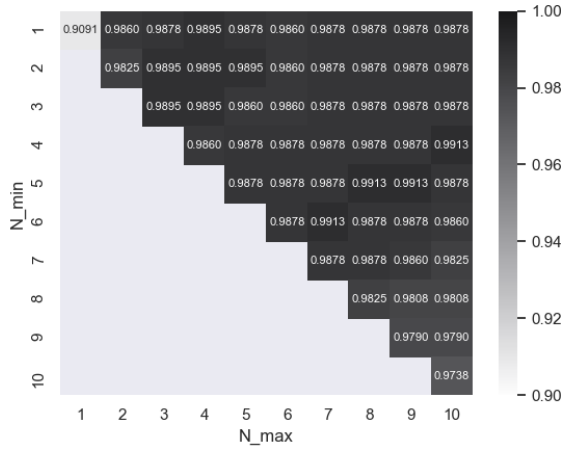
3.3 Passage de la tâche 2 à la tâche 1 et choix des run

Pour passer de la tâche 2 à la tâche 1, nous avons choisi tout simplement de retravailler les scores de similarités nativement normalisés[3] : Jaccard, Dice, Cosinus et Bray-Curtis). Nous transformons le score de similarité en un vote de la façon suivante : $vote = int(Sim * 5)$. Il est évident que des optimisations étaient possibles mais nous avons souhaité conserver la simplicité de l'approche *baseline*.

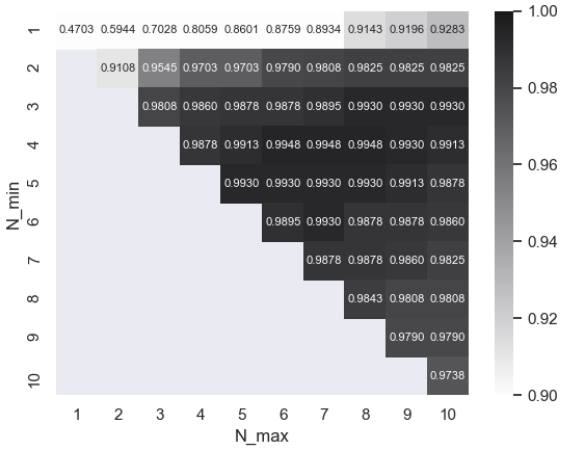
Nous pouvons voir dans la Figure 2 les résultats obtenus avec des n-grammes de caractères sur la tâche 1. La distance de Bray Curtis offre là encore des résultats plus stables même si le meilleur résultat est obtenu avec la distance cosinus. Nous avons tout de même choisi de conserver les configurations optimales que nous avions identifié pour la tâche 2 afin de limiter l'aspect *fine tuning* des valeurs N_{min} et N_{max}. Les configurations choisies sont présentées dans le tableau 4.

Le `run3` est un classifieur SVM à noyau RBF qui utilise comme caractéristiques pour chaque phrase à apparier les résultats de tous les systèmes utilisant les distances Bray Curtis et Cosinus, systèmes qui étaient apparus comme les plus complémentaires. Ce système de vote a obtenu des résultats suivants en validation croisée (10 strates) sur le jeu d'entraînement : 0,99 de MAP en moyenne sur la tâche 2, et 0,73 d'EDRM en moyenne sur la tâche 1.

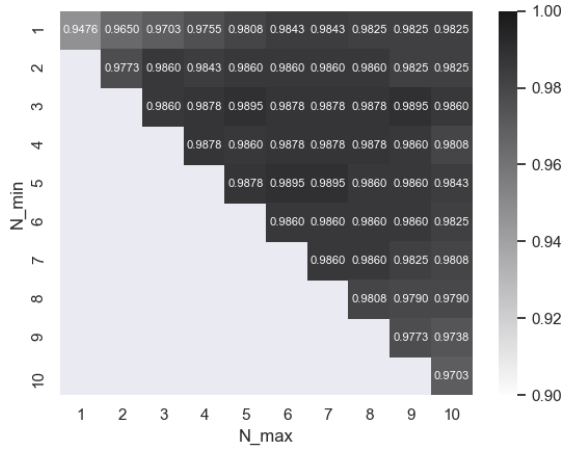
2. *Anything we can do with words we ought to be able to do with substrings"*
3. Nous n'avons a pas exploré la normalisation des autres distances du fait que leurs résultats sur la tâche 2 étaient significativement moins bons



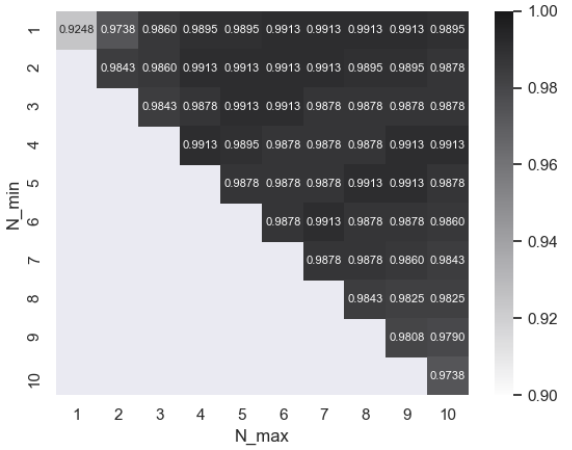
(a) Indice de Jaccard



(b) Coefficient de Dice



(c) Distance cosinus

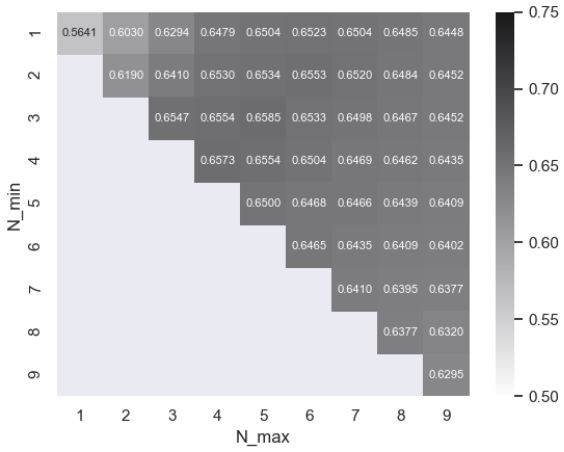


(d) Distance de Bray-Curtis

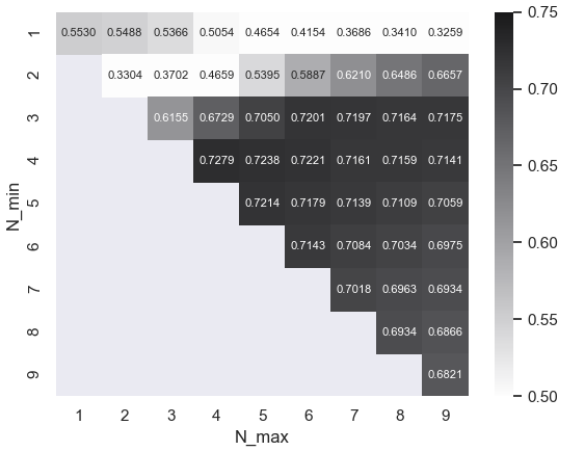
FIGURE 1 – Tâche 2 : résultats (MAP) avec des n-grammes de caractères sur le jeu d'apprentissage

	Distance utilisée	Opérandes utilisées	Tâche1	Tâche 2
run1	Cosinus	n-grams de 3 à 5	min(dist)	int(5*Sim)
run2	Bray-Curtis	n-grams de 1 à 10	min(dist)	int(5*Sim)
run3	Cosinus+Bray-Curtis(*bagging*)	Toutes	SVM radial	SVM radial

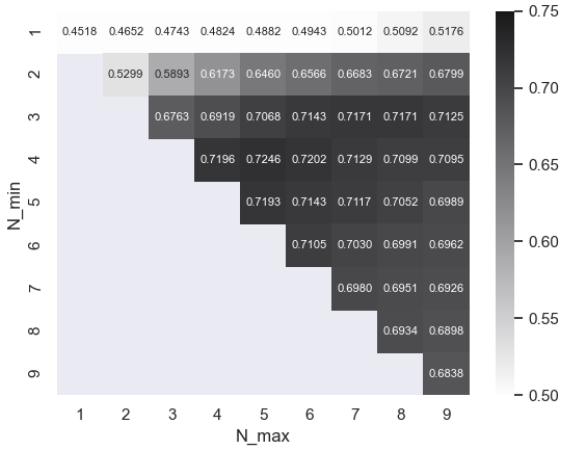
TABLE 4 – Configuration des runs soumis



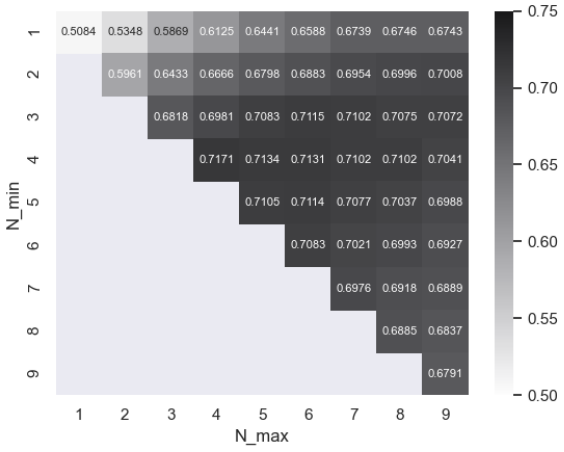
(a) Indice de Jaccard



(b) Coefficient de Dice



(c) Distance cosinus



(d) Distance de Bray-Curtis

FIGURE 2 – Tâche 1 : résultats (EDRM) avec des n-grammes de caractères sur le jeu d'apprentissage

4 Résultats et Discussion

4.1 Résultats officiels

Les tableaux 5 et 6 présentent nos résultats officiels sur les tâches 1 et 2. Les méthodes que nous avons proposé ont été moins performantes sur la tâche 1, avec un `run1` et un `run2` en dessous ou au niveau de la moyenne. Mais, le système de vote (`run3`) a offert une valeur ajoutée très importante à nos résultats sur cette tâche, $+10pp.$ par rapport au `run1`. Ce gain est plus grande que celui que nous avions observé sur le jeu de données d'entraînement. Sur la tâche 2, nos résultats sont globalement meilleurs, entre la médiane et le maximum des résultats soumis. Il est à noter que nos trois `run` ont donné un score strictement égal bien que les fichiers de résultats soient différents.

minimum : 0,653	médiane : 0,795	maximum : 0,822
run1 : 0,709	run2 : 0,673	run3 : 0,815

TABLE 5 – Évaluation officielle de nos trois runs sur la tâche 1 (EDRM) et comparaison avec le minimum, le maximum et la médiane (moyenne des soumissions : 0,762)

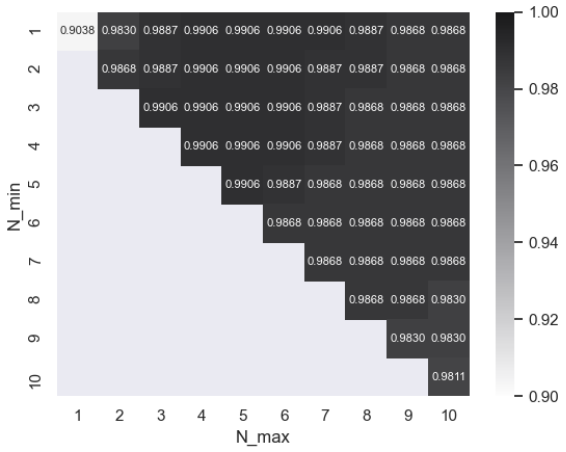
minimum : 0,9396	médiane : 0,9868	maximum : 0,9906
`run1` : 0,9887	`run2` : 0,9887	`run3` : 0,9887

TABLE 6 – Évaluation officielle de nos trois runs sur la tâche 1 (MAP) et comparaison avec le minimum, le maximum et la médiane (moyenne des soumissions : 0,9822)

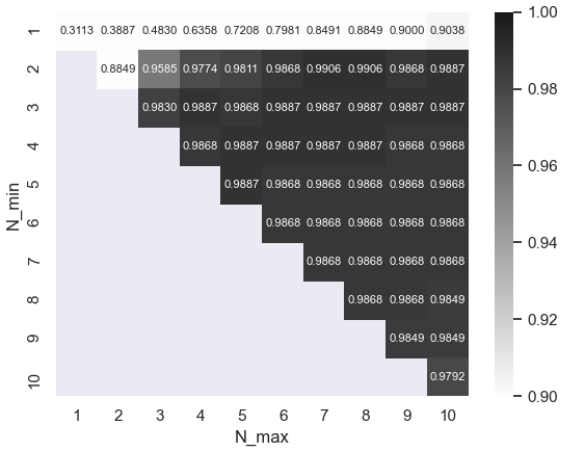
4.2 Variation des résultats sur les données de test

La figure 3 montre les résultats obtenus sur le jeu de test de la tâche 2 en faisant varier les mesures de similarité ainsi que la taille des n-grammes de caractères. Nous pouvons observer que les résultats sont très stables. Le score obtenu par nos 3 `runs` (0,9887) étant trouvé avec de nombreuses configurations ce qui correspond à 524 bons résultats sur 530. Plusieurs configurations amènent un résultat de 0,9906 ce qui correspond au meilleur système répertorié (1 instance bien appariée de plus que ce que nous avons soumis). Enfin, nous avons plusieurs cas avec la distance de Bray Curtis où nous obtenons un score meilleur de 0,9925 ce qui correspond à 526 bons résultats.

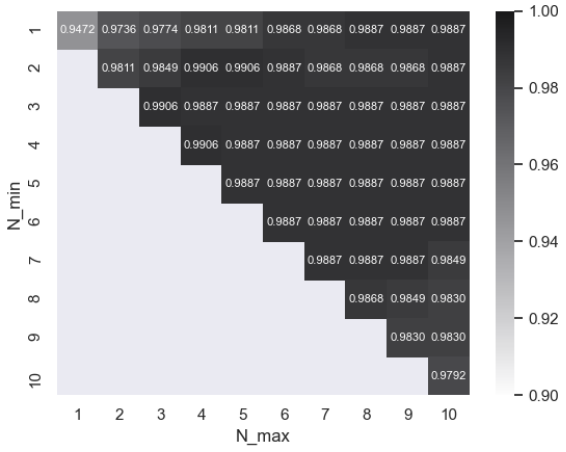
Sur la tâche 1, les méthodes `run1` et `run2` se sont avérées éloignées des meilleurs résultats. De fait, les variations sur les mesures de similarité ou la taille des n-grammes de caractères n'apportaient qu'un bénéfice somme toute relatif comme nous le montrons dans la figure 4. Les résultats plafonnent autour de 0,72 d'EDRM avec une pointe à 0,7294 avec le coefficient de Dice et des N-grammes de taille 4 à 5. Il apparaît que ces différentes variations étaient assez complémentaires ce qui explique pourquoi le système de *bagging* utilisé pour le `run3` a pu apporter 10 points de pourcentage de mieux que le meilleur système soumis (`run1` et 8 points de mieux que le meilleur résultat enregistré (coefficient de Dice et des N-grammes de taille 4 à 5).



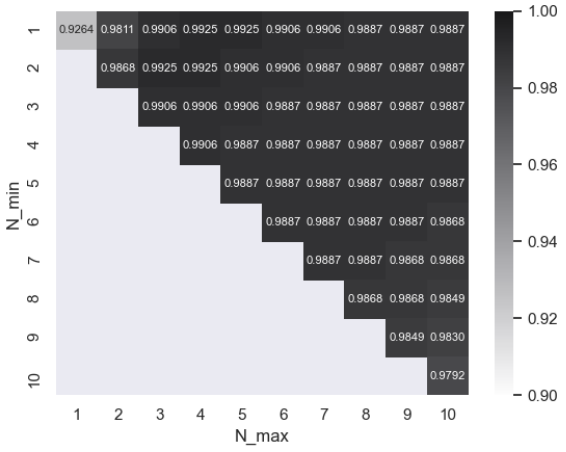
(a) Indice de Jaccard



(b) Coefficient de Dice

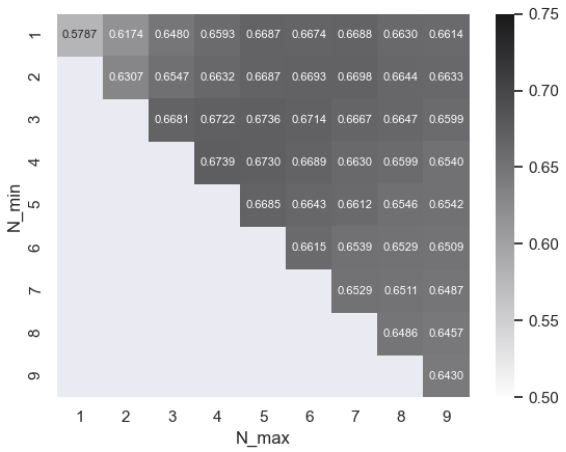


(c) Distance cosinus

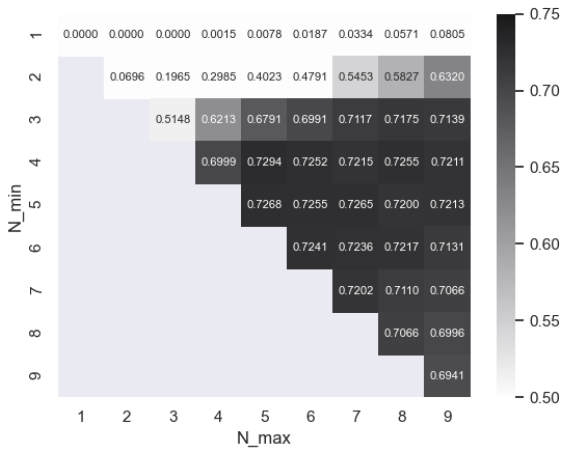


(d) Distance de Bray-Curtis

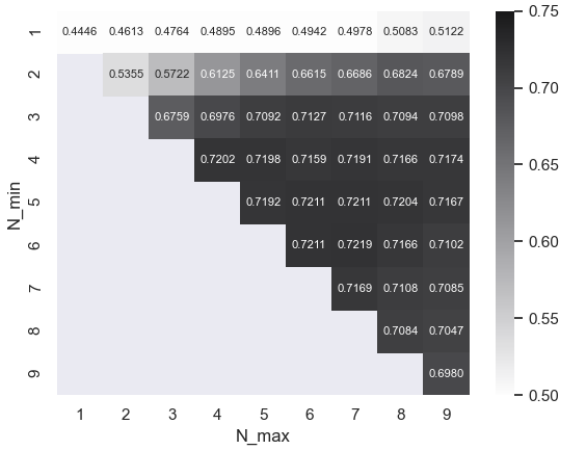
FIGURE 3 – Tâche 2 : résultats (MAP) avec des n-grammes de caractères sur le jeu de test



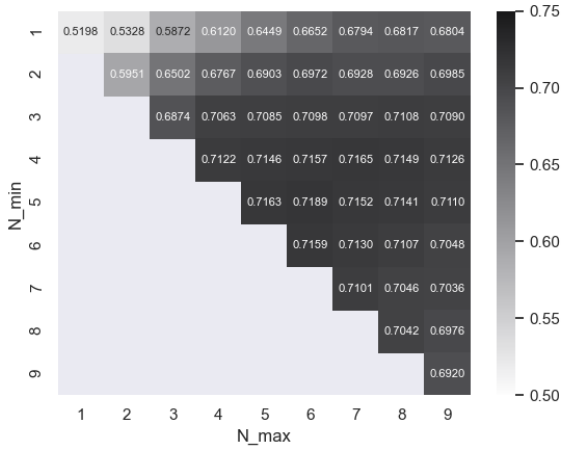
(a) Indice de Jaccard



(b) Coefficient de Dice



(c) Distance cosinus



(d) Distance de Bray-Curtis

FIGURE 4 – Tâche 1 : résultats (EDRM) avec des n-grammes de caractères sur le jeu de test

4.3 Discussion

Les résultats que nous avons présenté montrent l'intérêt de s'intéresser à optimiser des solutions simples de type *baseline*. eN effet, il peut suffire de modifier des paramètres simples pour faire progresser les résultats jusqu'à des niveaux qui apparaissent comme compétitifs vis-à-vis d'approches plus complexes. En particulier, l'utilisation de représentations en n-grammes de caractères plutôt qu'en mots présente l'avantage de diversifier, si l'on ne souhaite pas utiliser le terme « enrichir », à moindre coût la représentation et donc de la rendre plus à même de modéliser finement les relations entre les segments comparés. Ceci est d'autant plus important que les segments sont courts. Il semble évident que dans le cas de ces deux tâches de similarité, la relative stabilité du vocabulaire utilisé était sans doute favorable à une approche en chaînes de caractères. Cette approche permet notamment de capturer des racines et donc de détecter des familles de mots par exemple. Au contraire, quand des éléments censés être proches sémantiquement parlant ne partagent pas une grande proximité formelle, l'approche serait sans doute trop frustre.

Références

BRAY J. R. & CURTIS J. T. (1957). An ordination of the upland forest communities of southern wisconsin. *Ecological Monographs*, **27**(4), 325–349.

CARDON R., GRABAR N., GROUIN C. & HAMON T. (2020). Présentation de la campagne d'évaluation DEFT 2020 : similarité textuelle en domaine ouvert et extraction d'information précise dans des cas cliniques. In *Actes de DEFT 2020 (TALN 2020)*, p. 3–14.

CLAVEAU V. (2012). Vectorisation, Okapi et calcul de similarité pour le TAL : pour oublier enfin le TF-IDF. In *TALN - Traitement Automatique des Langues Naturelles*, p.=-, Grenoble, France.

GABAY S., RIGUET M. & BARRAULT L. (2019). A Workflow For On The Fly Normalisation Of 17th c. French. In *DH2019*, Utrecht, Netherlands : ADHO.

GRABAR N., GROUIN C., HAMON T. & CLAVEAU V. (2019). Information Retrieval and Information Extraction from Clinical Cases. Presentation of the DEFT 2019 Challenge. In *DEFT 2019 - Défi fouille de texte*, p. 1–10, Toulouse, France.

HUANG A. (2008). Similarity measures for text document clustering. In *Proceedings of the Sixth New Zealand Computer Science Research Student Conference (NZCSRSC2008), Christchurch, New Zealand*, p. 49–56.

MEHDAD Y. & TETREAULT J. (2016). Do characters abuse more than words ? In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, p. 299–303, Los Angeles : Association for Computational Linguistics.

MORENO J. G. & DIAS G. (2014). Easy Web Search Results Clustering : When Baselines Can Reach State-of-the-Art Algorithms. In *14th Conference of the European Chapter of the Association for Computational Linguistics*, Gotenburg, Sweden.

NEBHI K., BONTCHEVA K. & GORRELL G. (2015). Restoring capitalization in #tweets. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, p. 1111–1115, New York, NY, USA : Association for Computing Machinery.

RENDLE S., ZHANG L. & KOREN Y. (2019). On the difficulty of evaluating baselines : A study on recommender systems. arXiv preprint : 1905.01395.

STRUBELL E., GANESH A. & MCCALLUM A. (2019). Energy and policy considerations for deep learning in NLP. *CoRR*, **abs/1906.02243**.

UMEMURA K. & CHURCH K. (2009). Substring statistics. In *Proceedings of the 10th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing '09, p. 53–71, Berlin, Heidelberg : Springer-Verlag.