Un protocole d'expérimentation sur les propriétés graphémiques avec l'algorithme SOM

Otman Manad*, Nourredine Aliane*, Gilles Bernard*

*Laboratoire d'Informatique Avancée de Saint-Denis (LIASD),
Université PARIS 8, 2 Rue de la Libertée, Saint-Denis, France
{manad, nourredine}@ai.univ-paris8.fr, gilles.bernard@iedparis8.net

Résumé. [1]Nous présentons une recherche sur la distribution et la classification non-supervisée des graphèmes. Nous visons à réduire l'écart entre les résultats de recherches récentes qui montrent la capacité des algorithmes d'apprentissage et de classification non-supervisée pour détecter les propriétés de phonèmes, et les possibilités actuelles de la représentation textuelle d'Unicode. Nos procédures doivent assurer la reproductibilité des expériences et garantir que l'information recherchée n'est pas implicitement présente dans le pré-traitement des données. Notre approche est capable de catégoriser correctement de potentiels graphèmes, ce qui montre que les propriétés phonologiques sont présentes dans les données textuelles, et peuvent être automatiquement extraites à partir des données textuelles brutes en Unicode, sans avoir besoin de les traduire en représentations phonologiques.

1 Introduction

Cet article présente un protocole d'expérimentation pour découvrir les propriétés orales et scripturales des graphèmes sans aucune connaissance préalable, à partir d'un corpus et à l'aide d'un réseau neuromimétique, les cartes auto-organisatrices (SOM - Self Organizing Map) (Kohonen, 1995). Un graphème est peut être définit comme la plus petite unité qui compose un document textuel, et qui correspond à un seul son en oral (phonème). Un graphème peut être composé de ponctuations ou d'un ou plusieurs caractères, par exemple : « a, c, ch, qu, eaux ».

Nos résultats montrent que les expérimentations sur les propriétés structurelles et distributionnelles des caractères peuvent donner des moyens d'accéder aux propriétés des graphèmes et aux phonèmes sous-jacents ; nous contribuons ainsi à réduire cet écart. Nous avons choisi SOM parmi les algorithmes non-supervisés, pour sa capacité à cartographier et visualiser les résultats en deux dimensions, avec une détection facile des phénomènes analysés.

2 Problématique

Parmi les digrammes, 'ca' est plus fréquent que 'tc' en raison des propriétés des consonnes et des voyelles ; mais le 'ch' est plus fréquent que le 'ct' pour une raison complètement dif-

1. Cet article présente une version remaniée et abrégée de l'article paru dans (Bernard et al., 2015)

férente : le 'ch' est une consonne écrite avec deux caractères. Les propriétés du 'ch' doivent être comparées à celles de 'c' et pas à celles du 'ca'. D'autres propriétés sont purement scripturales : majuscule, minuscule, ponctuation. Les mesures d'entropie habituelles ou les chaînes de Markov ne capturent pas ces distinctions. Cela ne les empêche pas d'être efficaces pour la prononciation des chaînes de caractères, la reconnaissance vocale, la correction d'erreurs, l'identification des langues ou la reconnaissance optique. Mais cela constitue un obstacle pour la recherche de phénomènes linguistiques plus fins, comme nécessité par le déchiffrement d'écritures inconnues. Dans de tels cas, nous avons besoin à des analyses plus raffinées. Les graphèmes semblent être traités comme unités par le scripteur aussi bien que par le lecteur, comme montré par des études psychologiques tels que (Weingarten et al., 2004). Le but de notre approche est de capturer ces propriétés sans faire aucune supposition fondée sur nos connaissances, et voir combien d'informations peuvent être collectées.

3 État de l'art

Les propriétés distributionnelles de données écrites sont étudiées depuis longtemps. Pendant les années soixante, les linguistes et les mathématiciens ont travaillé sur des questions connexes : distinction automatique voyelle/consonne (Sukhotin, 1962), détection des frontières de morphème (Harris, 1968). La majorité des recherches concernant les travaux empiriques à partir de données brutes a davantage porté sur les propriétés des mots que sur les aspects grammaticaux ou les graphèmes. Les données grammaticales étaient conçues de manière très abstraite et rendues obscures par l'usage des soi-disant structures profondes pendant des années (voir (Bernard, 1997) qui utilise les statistiques des données grammaticales pour capturer les propriétés des mots) ; les graphèmes semblaient triviaux. Les travaux fondamentaux comme l'algorithme de Harris pour détecter les unités significatives basées sur leur distribution ont attendu des années avant d'être redécouverts (Bernard et Mariage, 2005), (Pham et Lee, 2015).

Les recherches récentes ont amené de nouvelles approches. (Goldsmith et Xanthos, 2009) et (Goldsmith et Riggle, 2012) ont réussi à attribuer, à partir d'un corpus en représentation phonologique, une catégorie à chaque phonème, comme voyelle/consonne ou voyelle basse/haute, en utilisant des profils distributionnels, les modèles cachés de Markov ou le «spectral clustering». (Mayer et Rohrdantz, 2013) ont montré également les possibilités d'un autre algorithme de clustering (Ward,1967) pour découvrir des propriétés phonétiques, mais pour tous ces travaux les mots doivent être donnés en transcription phonologique (comme l'API - alphabet phonétique international). Cette contrainte sur la représentation phonologique bloque la possibilité d'importer de tels algorithmes pour traiter des données brutes, car les données textuelles numériques n'utilisent pas l'API mais au mieux Unicode.

4 Description du système

4.1 Aperçu sur Graphemix

Notre système vise à encapsuler chaque étape de production de l'expérimentation, depuis la constitution de corpus jusqu'à la visualisation des résultats. En fait, travailler sur un corpus en Unicode est notre objectif de départ, "Graphemix 0.1" ne supporte que les écritures du plan

de base d'Unicode (Unicode Basic Multilingual Plane - BMP), que ce soit en UTF-8, UTF-16 ou CS-4. L'utilisateur peut sélectionner d'autres jeux de caractères pour les écritures qui ne sont pas acceptées par le Consortium Unicode. Notre système est disponible comme un projet gitlab (https ://gitlab.com/harris/charprops). Nous décrivons ci-dessous avec plus de détails les principaux composants de notre système.

4.2 Recherche des candidats

Le texte est divisé en candidats, c'est à dire, en n-grammes de caractères Unicode, dont la taille est comprise entre 1 et N, N est fixée par défaut à 5. La valeur par défaut de 5 semble être une bonne limite supérieure générale. Nous déterminons tous les graphèmes possibles avec leur contexte droit. En fait, nous posons d'abord une fenêtre de dix caractères (le double de la taille limite supérieure), chaque candidat à gauche est combiné avec ses voisins possibles à droite. Le candidat de taille un est combiné avec les candidats de toutes les tailles possibles à droite. Ainsi on répète la procédure avec les candidats de taille deux, et ainsi de suite jusqu'à la taille 5. Avec notre corpus de plus de 5 millions de caractères, il y a 140 milliards de contextes possibles. Cependant, beaucoup de contextes incluent les mêmes candidats, dans la position du candidat ou dans la position du voisin, aussi il faut tenir compte du temps que prend l'identification du candidat et du voisin dans le comptage des co-occurrences.

Afin de réduire le temps de calcul et l'utilisation de la mémoire, nous avons utilisé un arbre préfixe (Trie) de caractères qui contient les n-grammes et leurs contextes, avec un profondeur maximum de 10, et une largeur maximum égale le nombre de caractères possibles d'Unicode dans le BMP (théoriquement 65 536).

4.3 Construction de la matrice

La matrice de contexte est une matrice creuse, donc les candidats pourraient théoriquement se compter en millions. Cependant, les n-grammes qui se produisent peu de fois dans un grand corpus de caractères ne sont pas de probables candidats - graphèmes, donc ils peuvent (et doivent) être éliminés. Heureusement, tous les n-grammes possibles n'apparaissent pas dans le corpus, et la majorité des n-grammes n'ont que quelques occurrences. Mais nous avons fait le choix de ne pas avoir de seuil a priori, de sorte que même les n-grammes en hapax (une seule occurrence) sont intégrés dans la matrice.

4.4 L'algorithme Self Organizing Maps

Dans les modèles neuronaux non supervisés, SOM étant probablement le plus populaire de nos jours. Voir (Kohonen, 1995) pour une description complète de l'algorithme.

La réduction de la dimensionnalité dans la carte permet de découvrir et analyser des relations entre les classes de données, qui seraient impossibles à détecter dans leur espace original. La cartographie bi-dimensionnelle fournit de plus un interface commode de visualisation.

X étant le vecteur d'entrée, j un index sur les neurones, W le vecteur de mémoire des neurones, le gagnant, J* est déterminé par l'équation suivante, où d() est la distance ou une mesure de similitude :

$$d(X, W_{j^*}) = \min d(X, W_j) \tag{1}$$

Un protocole d'expérimentation sur les propriétés graphémiques avec l'algorithme SOM.

for each j in $\{n\ ;\ p\}$

Dans la phase d'apprentissage, le gagnant et chaque neurone de son voisinage, sont modifiés selon l'équation 2, où t est le temps (de l'itération), j un indice sur les neurones, x et w sont les coordonnées de X et W :

$$W_j^{(t+1)} = W_j^{(t)} + \alpha^{(t)} V_r^{(t)}(j, j^*)(X_i^{(t)} - W_j^{(t)}) \tag{2}$$

Le taux d'apprentissage α suit l'équation 3, où α_0 est le taux d'apprentissage initial.

$$\alpha^{(t)} = \alpha_0(1 - \frac{t}{t_{max}}) \tag{3}$$

L'apprentissage du voisinage V du gagnant suit ici la variante gaussienne (équation 4), avec de meilleurs résultats que le chapeau mexicain ou d'autres variantes.

$$V_r^{(t)}(j - j^*) = e^{-\frac{dm(j,j^*)}{2\sigma^{2(t)}}} \tag{4}$$

dm : distance Manhattan

avec σ dans l'équation 5, où *i* = valeur initiale et *f* = valeur finale.

$$\sigma^{(t)} = \sigma_i(\frac{\sigma_i}{\sigma_f})^{\frac{t}{t_{max}}} \tag{5}$$

Pour évaluer la qualité de la classification, nous calculons l'inertie intra-classe qui mesure l'homogénéité à l'intérieur des groupes, selon HI (equation 6). S est la mesure de la similitude ou la distance choisie, m le nombre de groupes et n_j le cardinal de chaque groupe.

$$HI = \frac{1}{m}\sum_{j=1}^{m}(\frac{1}{n_j}\sum_{i=1}^{n_j} S(X_i, W_j)) \tag{6}$$

Avec les mesures de distance, il faut que HI soit plus proche de 0 et plus proche de 1 avec les mesures de similitude de cosinus.

5 Experimentations et résultats

Notre corpus complet, de taille 5.2 MB composé de neuf romans français extraits à partir du projet Gutenberg, a rapporté 374108 n-grammes avec 123 caractères ; 125384 n-grammes n'ont qu'une occurrence.

La génération de l'arbre préfixe contenant tous les voisinages rencontrés ne prend pas beaucoup de temps (entre cinq et dix minutes sur un ordinateur de milieu de gamme). La charge porte sur l'identification de quel candidat est voisin de quels autres, et dans les transactions (même par batch) avec la base de données : cette partie du processus prend quelques heures.

Nous avons résolu le problème de la grande dimensionnalité des vecteurs en adaptant l'algorithme SOM pour gérer les vecteurs creux. Par exemple notre matrice de données contient 374108 lignes et 374108 colonnes, et presque 1 élément différent de 0 sur 6622 éléments. Pour cela, nous avons adapté le calcul des distances (euclidienne, manhattan, cosinus) pour la détermination du neuronne gagnant et la mise à jour de son vecteur, ainsi que ses voisins.

Dans notre corpus, un seuil de 25000 a donné 88 candidats. Pour SOM, les meilleurs résultats sont avec une topologie hexagonale (6 voisins), un voisinage commençant par 25% de la carte et finissant avec 0 (le gagnant lui-même), avec la distance euclidienne, et 30000 itérations. Dans presque tous les résultats que trois classes des données émergent : (a) voyelles (y compris les graphèmes vocaliques) ; (b) consonnes (y compris 'ch') ; (c) les ponctuations.

Sur la figure 1, les consonnes sont situées, en bas, dans la même région sur la carte (encerclées en rouge), ainsi les ponctuations sont encerclées en vert (à droite). Les voyelles, y compris les graphèmes vocaliques, sont situées en haut de la carte (encerclées en bleu)

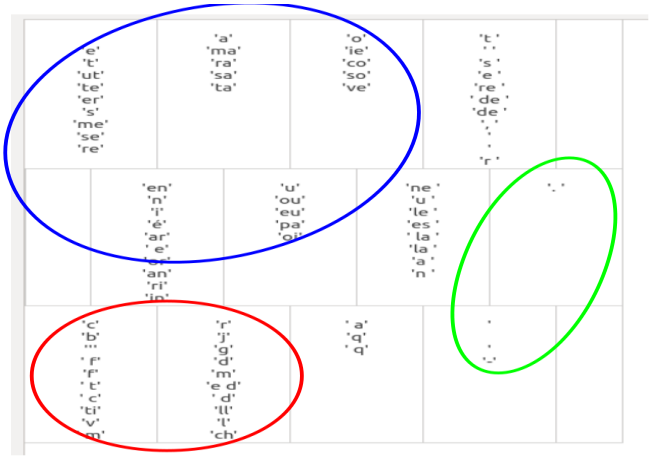

FIG. 1 – *SOM 5x2, 30 000 itérations, distance euclidienne.*

La similitude cosinus fonctionne presque aussi bien que la distance euclidienne. La distance de Manhattan ne donne pas de bons résultats (groupe plus de données dans moins de neurones et confond les catégories).

6 Conclusion

La faisabilité de l'approche basée sur un corpus de texte brut en Unicode pour détecter au moins les propriétés phonémiques des graphèmes est illustrée par les résultats obtenus. Il reste évidement beaucoup à faire, comme la généralisation de l'approche dans d'autres langues. Nous nous sommes concentrés ici sur le protocole d'expérimentation, en nous assurant de ne pas violer notre première contrainte : ne pas mettre d'information dans les données dont on veut tirer de l'information. Nous avons commencé des essais sur les fichiers utf-8 en langue française du projet Gutenberg (près de 330 Mo), et concevons une stratégie pour pouvoir le traiter dans un temps raisonnable. Les résultats de notre corpus complet étant déjà beaucoup plus au point, nous espérons que nous obtiendrons un meilleur taux de classification (en supprimant les données parasites et l'influence du style de l'auteur). Nous avons en perspective immédiate une extension en dehors du Plan Multilingue à Base de l'Unicode (UBMP). Mais cela va complexifier le code, car les surrogates haut et bas ont chacun la taille d'un caractère

Unicode et l'arbre préfixe verra sa largeur (mais non sa profondeur) modifiée pour supporter cela. Une extension comprenant la classification spectrale et des modèles de Markov cachés est prévue dans un futur proche. Notre but est de pouvoir reproduire les expériences de (Goldsmith et Xanthos, 2009) et permettre de croiser le clustering avec les réseaux neuromimétiques.

Références

Bernard, G. (1997). Experiments on distributional categorization of lexical items with self organizing maps. In *WSOM'97 International Workshop on Self Organizing Maps. Espoo, Finland*.

Bernard, G., N. Aliane, et O. Manad (2015). An experimentation line for underlying graphemic properties acquiring knowledge from text data with self organizing maps. In *International Workshop on Artificial Neural Networks and Intelligent Information Processing*.

Bernard, G. et J.-J. Mariage (2005). Post-processing of grammatical patterns produced by self organizing maps. In *8th Conference on Pattern Recognition and Information Processing PRIP'05, Minsk, Belarus*.

Goldsmith, J. et J. Riggle (2012). Information theoretic approaches to phonology: the case of finnish vowel harmony. *Natural Language & Linguistic Theory 30*(3).

Goldsmith, J. et A. Xanthos (2009). Learning phonological categories. *Language 85(1)*.

Harris, Z. (1968). *Mathematical Structures of Language* (1st ed.). Interscience Publishers New York.

Kohonen, T. (1995). *Self-Organizing Maps* (1st ed.). Springer, Berlin.

Mayer, T. et C. Rohrdantz (2013). Phonmatrix : Visualizing co-occurrence constraints of sounds.

Pham, M. et J.-L. Lee (2015). Combining successor and predecessor frequencies to model truncation in brazilian portuguese.

Sukhotin, B. (1962). Experimental'noe vydelenie klassov bukv s pomosçju evm. *Problemy strukturnoj lingvistiki 236*.

Weingarten, R., G. Nottbusch, et U. Will (2004). Morphemes, syllables and graphemes in written word production. *Multidisciplinary Approaches to Language Production*.

Summary

We present an experimentation line that encompasses various stages for research on graphemes distribution and unsupervised classification. We aim to help close the gap between recent research results showing the abilities of unsupervised learning and clustering algorithms to detect underlying properties of phonemes and the present possibilities of Unicode textual representation. Our procedures need to ensure repeatability and guarantee that no information is implicitely present in the preprocessing of data. Our approach is able to categorize potential graphemes correctly, thus showing that not only phonemic properties are indeed present in textual data, but that they can be automatically retrieved from raw-unicode text data and translated into phonemic representations.