Segmentation de texte non-supervisée pour la détection de thématiques à l'aide de plongements lexicaux

Alexandra Benamar

Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 31e 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

Segmentation de texte non-supervisée pour la détection de thématiques à l'aide de plongements lexicaux

Alexandra Benamar[1, 2]
(1) Université Paris-Saclay, CNRS, LIMSI, 91405 Orsay
(2) EDF R&D, 7 Boulevard Gaspard Monge, 91120 Palaiseau
alexandra.benamar@{limsi.fr,edf.fr}

RÉSUMÉ

Cet article présente les principales méthodes de segmentation automatique de documents textuels spécifiques. La tâche de segmentation thématique de texte consiste à analyser un document pour en extraire des sections cohérentes. Les méthodes de segmentation non supervisées cherchent à optimiser une fonction de probabilité de segmentation ou une fonction de similarité qui peut être calculée entre les blocs ou au sein des blocs. Elles sont réparties en trois catégories : les méthodes statistiques, les méthodes à base de graphes et les approches neuronales. Parmi les approches neuronales utilisées, nous nous intéressons tout particulièrement à celles qui utilisent des plongements lexicaux pour représenter des phrases et définir des segments thématiques. Tout d'abord, nous montrons que les plongements lexicaux permettent une amélioration nette des performances par rapport à des méthodes statistiques. Ensuite, nous évaluons l'impact du choix de la représentation vectorielle des phrases pour cette tâche de segmentation non supervisée.

ABSTRACT

Unsupervised text segmentation for topic detection using embeddings

This paper presents the state of the art of automatic segmentation of domain-specific documents. The task of topic segmentation consists in analyzing a document in order to extract coherent sections. The aim of unsupervised segmentation methods is either to maximize a probabilistic function or to use a similarity function that can be maximize between or within segmented blocks. They are divided into three categories : statistical methods, graph-based methods and neural approaches. Among the last ones, we are particularly interested in those which use latent representations of words to define segments. First, we show that embeddings allow a significant improvement in performance compared to statistical methods. Next, we assess the impact of sentence representation on unsupervised text segmentation methods.

MOTS-CLÉS : segmentation de texte ; méthodes non-supervisées ; plongements lexicaux.

KEYWORDS: text segmentation ; unsupervised methods ; embeddings.

1 Introduction

La segmentation de texte en thématiques cohérentes est une tâche fondamentale en traitement automatique des langues (TAL), à différentes granularités. Un ensemble d'approches appelées segmentation thématique (*topic segmentation* en anglais) s'emploie à diviser un document en segments thématiquement cohérents. Ces approches sont souvent considérées comme un pré-requis pour d'autres

Volume 3 : Rencontre des Étudiants Chercheurs en Informatique pour le TAL, pages 1–14.
HAL : hal-02786182.

tâches telles que l'analyse du discours (Polanyi *et al.*, 2004) et l'analyse de sentiments (Mu *et al.*, 2012). La segmentation en thèmes permet de répondre à des problématiques variées comme le résumé automatique (Chuang & Yang, 2000; Angheluta *et al.*, 2002), la recherche d'information (Huang *et al.*, 2003; Prince & Labadié, 2007) et l'analyse de dialogues (Song *et al.*, 2016). À un niveau plus fin, la segmentation de texte consiste à segmenter des phrases en unités élémentaires du discours (EDU pour *Elementary Discourse Units*) (Marcu, 2000) définies comme des unités similaires à des clauses servant de blocs élémentaires pour la segmentation du discours.

Afin de répondre à ces problématiques de segmentation de texte, des méthodes supervisées et non supervisées de fouille de texte sont généralement mises en œuvre. Les méthodes supervisées nécessitent un grand volume de données d'apprentissage spécifiques à un domaine d'application très structurées (documents juridiques, rapports médicaux, ...) et sont sensibles au bruit. Dans cet article, nous nous intéressons à la segmentation de texte non supervisée, où la spécificité de la méthode réside dans le choix de la fonction objectif et dans la représentation des phrases du corpus (dense ou latente). Depuis de nombreuses années, différentes approches ont été explorées pour segmenter des textes comme la détection de cooccurences de mots entre des segments, la détection de thématiques, l'utilisation de représentations de phrases ou de mots sous forme de graphes et l'utilisation d'architectures neuronales. Ici, nous nous intéressons à l'utilisation de plongements lexicaux pour la segmentation automatique de texte en sections thématiques pertinentes dans des corpus de données spécifiques. Les plongements lexicaux ont connu un grand succès ces dernières années sur plusieurs tâches d'apprentissage non supervisé, en permettant une représentation syntaxique et sémantique des mots dans un corpus. Les évaluations seront réalisées sur le corpus construit lors de la compétition Défi fouilles de texte (DEFT) de l'édition 2006 (Heitz *et al.*, 2006) composé de discours politiques et de textes juridiques. Les méthodes testées dans cet article ont été retenues pour cette tâche spécifique à partir d'un état de l'art assez large des méthodes neuronales de segmentation de texte et comparées à des méthodes statistiques fréquemment utilisées. Après avoir présenté un état de l'art des méthodes de segmentation thématique de documents et de représentations du langage (section 2), nous décrivons le corpus DEFT utilisé (section 3), puis nous présentons les expériences réalisées pour cette tâche (section 4) avant de conclure et de définir nos perspectives de recherche (section 5).

2 État de l'art

Dans cette section, nous présentons tout d'abord les grandes familles de méthodes non supervisées de segmentation de texte. Ensuite, nous nous intéressons aux évolutions récentes qui permettent d'améliorer les représentations de texte : les plongements lexicaux. Puis, nous détaillons deux méthodes proposées ces dernières années pour utiliser des plongements de mots pour la segmentation automatique de texte. Enfin, nous nous intéressons aux méthodes courantes d'évaluation des modèles de segmentation.

2.1 Méthodes de segmentation non supervisée

Parmi les approches répandues de segmentation linéaire de texte, les méthodes non supervisées constituent un champ de recherche majoritaire. Ces méthodes englobent la cohésion lexicale, la modélisation statistique, des méthodes par propagation d'affinité et des approches de *topic modeling*.

2.1.1 Objectif des méthodes

On distingue deux stratégies d'optimisation de la segmentation de texte : des méthodes probabilistes et des approches basées sur la similarité de contenu. La première approche définit un modèle de langue à travers l'étude de la distribution de probabilités des mots d'un corpus en vue d'affecter chaque mot à une thématique. En 2001, une approche probabiliste (Utiyama & Isahara, 2001) consiste à maximiser la probabilité de segmentation d'un texte. Formellement, si on considère une séquence de mots $W = w_1 w_2 \dots w_n$ et une segmentation $S = s_1 s_2 \dots s_m$ de W, l'objectif est de maximiser :

$$P(S|W) = \frac{P(S)P(W|S)}{P(W)} \tag{1}$$

Cela revient à trouver la séquence de segments $S = argmax_s P(W|S)P(S)$, $P(W)$ étant un terme constant. Afin d'évaluer cet objectif, les auteurs émettent deux hypothèses :

1. Les segments sont statistiquement indépendants les uns des autres.
2. Étant donné un segment, les mots au sein d'un segment sont statistiquement indépendants les uns des autres.

Cette méthode est la première a avoir été proposée pour traiter la segmentation comme un problème d'optimisation probabiliste. Elle a permis d'obtenir des résultats à l'état de l'art dans des applications de résumés automatiques. Cependant, cette approche ne semble pas adaptée à une application de recherche d'information. En effet, l'algorithme découpe les gros documents en peu de segments, ce qui est intéressant pour le résumé automatique (où on veut détecter le moins de thématiques pertinentes possibles), mais ce qui n'est pas pertinent pour l'extraction de petits segments de texte.

La deuxième approche consiste à étudier la similarité lexicale entre différentes unités de texte. Les unités de texte peuvent être des mots, des phrases, des séquences de mots ou des paragraphes. Le plus souvent, la similarité est mesurée entre des blocs de textes, sous forme de similarité cosinus $S(s_i, s_j)$ entre deux segments $s_i = w_1 w_2 \dots w_n$:

$$S(s_i, s_j) = \frac{s_i \cdot s_j}{\|s_i\| \|s_j\|} \tag{2}$$

Dans l'équation, $s_i \cdot s_j$ est le produit vectoriel des deux vecteurs et $\|s_i\|$ est la norme L2 du vecteur s_i. La plupart des algorithmes de segmentation supposent que des fragments de texte ayant des distributions de mots similaires appartiendront à la même thématique. Dans les approches de similarité, on distingue des blocs similaires et homogènes en utilisant la similarité des éléments au sein d'un même bloc. Deux approches sont étudiées pour évaluer l'homogénéité des blocs : la maximisation de la similarité au sein des blocs (Reynar, 1998; Choi, 2000) et la maximisation de la dissimilarité entre des blocs (Reynar, 1998). Idéalement, les deux stratégies seraient combinées pour trouver des frontières de segmentations optimales.

2.1.2 Familles de méthodes

Les grandes familles de méthodes de segmentation de texte non supervisées sont :

1. Les méthodes statistiques : elles formulent l'hypothèse que des segments différents contiennent des mots différents. Elles caractérisent les mots par des méthodes statistiques (*ie.* TF, IDF, ...).

2. Les modèles à base de graphes : les nœuds représentent les phrases, et les arêtes peuvent représenter le nombre de mots qu'elles partagent (Pourvali & Abadeh, 2012) ou encore des liens de similarité vectorielle entre leurs mots communs (Glavaš *et al.*, 2016).
3. Les méthodes basées sur des réseaux de neurones.

Dans la suite de cet article, nous détaillerons les méthodes statistiques, servant de base à notre étude, et les méthodes neuronales au cœur de celle-ci.

Modèles statistiques Un document est souvent constitué de sections cohérentes, contenant elles-mêmes plusieurs unités textuelles (paragraphes, phrases, segments, etc.) (Salton *et al.*, 1996). Une hypothèse sous-jacente à la plupart des algorithmes de segmentation de documents consiste à dire que les répétitions lexicales signifient la continuité d'une thématique, alors que le changement de distribution lexicale indique une transition vers une autre thématique. Ce principe a été formalisé par Halliday & Hasan (1976) dans la théorie de la cohésion. Généralement, si deux thématiques sont suffisamment différentes, le vocabulaire significatif associé à chacun d'entre-eux sera également différent. De plus, si des mots apparaissent dans des contextes similaires, ils seront probablement liés sémantiquement, ce qui permet l'utilisation de modèles basés sur la cooccurrences de mots. Reynar (1998) observe que les anaphores pronominales ont tendance à survenir plus fréquemment au sein de segments qu'entre différents segments et montre ainsi que la segmentation en thématiques pourrait être une étape cruciale pour la résolution d'anaphores pronominales. Les premières méthodes de segmentation de textes consistaient à représenter la cohésion lexicale dans un espace vectoriel, en explorant des modèles basés sur la cooccurences de mots. Parmi elles, TextTiling (Hearst, 1997) est une technique de segmentation de texte basée sur la similitude entre des blocs de mots adjacents, en se basant uniquement sur la cooccurence des mots et leur distribution. Cette approche sous-entend qu'une baisse de la similarité entre blocs adjacents correspond à un changement de thématique. L'inconvénient majeur de cette méthode est qu'elle considère la similarité entre des blocs adjacents, sans considérer des dépendances à long terme. Choi (2000) améliore TextTiling avec l'algorithme C99 et calcule un score de cohérence entre toutes les paires d'unités (phrases ou segments) d'un texte au lieu de seulement étudier la cohérence entre les unités voisines. L'algorithme recherche les frontières qui optimisent une fonction objectif basée sur le score de cohérence, en effectuant des coupures successives du texte, similaire à du *clustering* hiérarchique descendant. Plus récemment, des méthodes de segmentation basées sur des modèles de *topic modeling* ont vu le jour. Parmi elles, la méthode PLDA (Purver *et al.*, 2006) calcule des frontières de segmentation et un modèle d'allocation de Dirichlet latente.

Réseaux de neurones Les réseaux de neurones profonds (*Deep Neural Networks*) ont connu un grand succès en TAL, notamment avec l'utilisation de méthodes de *transfer learning* et de modèles de langues. Ces architectures ont permis des gains de performances impressionnants, notamment en classification supervisée, en reconnaissance d'entités nommées ou encore en *Question Answering*. Depuis, les réseaux de neurones ont été introduits pour des tâches de segmentation de données textuelles. Par exemple, les réseaux de neurones récurrents LSTM ont été utilisés pour la segmentation de textes (Koshorek *et al.*, 2018) en considérant ce problème comme une tâche d'apprentissage supervisé, où chaque phrase de fin de segment est étiquetée par un marqueur spécifique. Les auteurs ont utilisé un premier LSTM bidirectionnel pour construire des représentations de phrases à partir de mots et un deuxième LSTM pour attribuer une probabilité d'appartenance de chaque phrase aux labels. Au vu du récent succès des plongements lexicaux, un champ de recherche s'intéresse à leur incorporation à la tâche de segmentation automatique de textes.

2.2 Représentation du langage

Les représentations vectorielles de mots, qui ont permis des gains de performance importants sur plusieurs tâches comme la classification, consistent à associer un mot à un vecteur de taille fixe. Parmi les modèles les plus connus figurent Word2Vec (Mikolov *et al.*, 2013), fondé sur un modèle de prédiction d'un mot à partir du contexte de mots environnants, et GloVe (Pennington *et al.*, 2014) basé sur la prédiction des cooccurences de mots dans un document. Ces approches permettent d'appréhender des représentations figées des mots à partir de leur contexte environnant. Toutefois, les résultats obtenus ne rendent pas compte de la richesse syntaxique et sémantique de la langue. Les mots polysémiques seront représentés par un seul vecteur, indépendamment de leur sens réel dans la phrase. De plus, les mots n'appartenant pas au vocabulaire de l'ensemble d'apprentissage (*Out Of Vocabulary* en anglais ou OOV) ne seront pas reconnus par le modèle ce qui oblige l'utilisateur à construire de nouveaux modèles pour le traitement de domaines spécifiques.

Plusieurs méthodes tirent parti des réseaux de neurones récurrents pour construire des représentations plus riches basées sur des modèles de langues statistiques. Ceux-ci consistent à calculer à partir d'un enchaînement de mots la probabilité d'apparition du mot suivant. ELMo (*Embeddings from Language Models*) (Peters *et al.*, 2018) construit des plongements de mots contextuels qui utilisent la richesse syntaxique et sémantique des mots. Ce modèle combine des réseaux de neurones récurrents pour l'analyse des mots et des réseaux de neurones convolutifs pour l'analyse des chaînes de caractères. Chaque caractère est ainsi représenté par un vecteur de petite taille et chaque mot est représenté par la concaténation des vecteurs des caractères le composant. Cela permet de représenter tous les mots qui n'ont jamais été vus par le système (OOV). De plus, l'utilisation de réseaux récurrents bidirectionnels permet l'ajout d'informations syntaxiques et sémantiques à travers l'enrichissement du contexte pour chaque mot. En utilisant des méthodes de combinaison linéaire et de concaténation vectorielle des couches cachées pour obtenir de nouvelles caractéristiques, ELMo permet une amélioration significative des résultats sur des tâches de *Question Answering* (QA), de détection d'informations sémantiques « qui a fait quoi à qui ? », de résolution de coréférence, etc.

Méthode sémantique Alemi & Ginsparg (2015) démontrent l'utilité des plongements lexicaux sémantiques pour la segmentation de texte, à la fois dans les algorithmes existants et dans de nouveaux algorithmes de segmentation. Les auteurs ont incorporé à l'algorithme C99 (présenté en section 2.1) des plongements de mots GloVe. Cela a permis d'améliorer de quelques pourcents les résultats obtenus avec l'algorithme natif. Le corpus est tout d'abord nettoyé en supprimant la ponctuation et les mots vides. Puis, chaque mot est représenté par son vecteur GloVe associé et pondéré par l'Inverse Document Frequency (IDF) du mot dans l'ensemble du corpus. Enfin, chaque phrase est représentée par le vecteur moyen de l'ensemble des mots la composant. La segmentation est une forme de *clustering* donc un choix naturel pour la fonction de score est la somme des écarts carrés par rapport à la moyenne du segment, comme utilisée dans l'algorithme k-means. En général, la similitude cosinus est utilisée pour le vecteur de mots, mais les auteurs ont choisi de normaliser les vecteurs afin d'utiliser la métrique euclidienne, ce qui se rapproche plus de l'algorithme k-means par défaut. Enfin, les auteurs présentent différentes stratégies d'optimisation de la fonction objectif : une méthode d'optimisation dite « gourmande »(*greedy* en anglais) et une méthode d'optimisation dynamique. La méthode gourmande consiste à diviser le texte itérativement à l'endroit où le gain est le plus grand, jusqu'à ce que ce gain soit inférieur à un seuil de pénalité donné. Le gain est défini comme la somme des normes des segments gauche et droit moins la norme du segment à diviser. La méthode gourmande consiste à construire de manière itérative une structure de données qui stocke les

résultat d'un fractionnement optimal. Il en résulte une matrice stockant un score pour un segment de la position i à j, étant donné une segmentation optimale jusqu'à i. La méthode dynamique a permis une amélioration des résultats de l'algorithme C99 couplé à des plongements lexicaux.

Plus récemment, les architectures neuronales *transformer* (Vaswani *et al.*, 2017) se sont imposées comme une alternative performante aux réseaux de neurones récurrents (Cho *et al.*, 2014), grâce à l'efficacité de leur apprentissage et à leurs performances supérieures en termes de capture des dépendances longue distance. Les modèles *Universal Sentence Encoder* (USE) sont des modèles pré-entrainés d'*embeddings* à différents niveaux : mots, phrases et paragraphes et ont été testés sur des tâches supervisées telles que la classification de sentiments ainsi que sur des tâches de similarité entre les documents (Cer *et al.*, 2018). Parmi les nombreux modèles proposés par USE, le plus performant est le modèle transformer qui utilise les mécanismes d'attention pour calculer des représentations vectorielles sensibles au contexte des mots. En 2018, Devlin *et al.* (2018) proposent BERT, un modèle transformer bi-directionnel à l'état de l'art sur 11 tâches de TAL, dont le QA où il dépasse même l'annotation humaine. BERT contient deux nouvelles tâches de pré-entraînement : l'une à l'échelle des mots et l'autre à l'échelle des phrases.

2.3 Plongements lexicaux pour la segmentation de texte

Méthode séquentielle D'autres travaux (Karus, 2019) présentent un modèle de segmentation séquentiel qui utilise une fenêtre glissante sur les phrases d'un corpus pour détecter les frontières de segmentation (Figure 1). Tout d'abord, l'auteur représente chaque phrase du corpus par un plongement de phrase, calculé avec Word2Vec, en calculant la moyenne des vecteurs de mots la composant. Ensuite, l'algorithme observe les plongements des n premières phrases (n correspond à la taille de la fenêtre) et calcule la plus grande distance cosinus entre les segments de phrases puis divise le texte au niveau de cet indice. Puis, il traite ensuite les n phrases après la scission décidée et répète le processus jusqu'à ce qu'il atteigne la fin du texte. L'auteur a également essayé de réduire la dimension des algorithmes avec une analyse en composantes principales, ce qui lui a permis de gagner en performance.

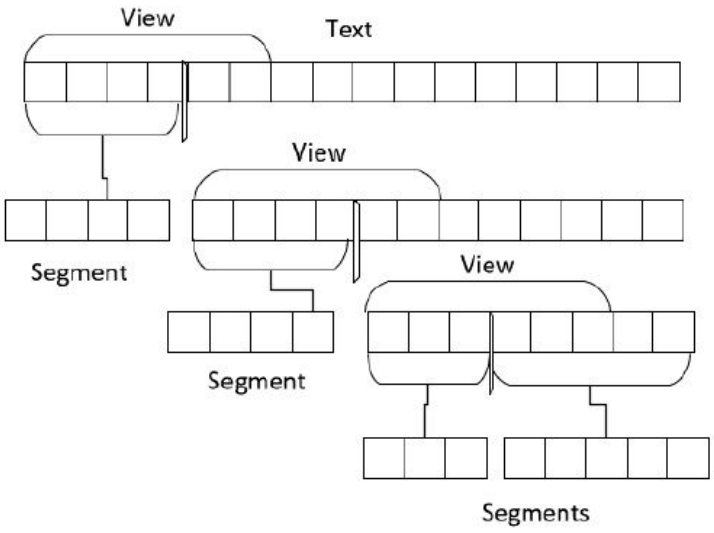


FIGURE 1 – Segmentation séquentielle de textes

2.4 Méthodes d'évaluation

L'évaluation en segmentation de texte est une tâche difficile qui peut prendre en compte différents paramètres. Parmi eux, certains permettent de détecter des erreurs de labels (précision, rappel et f-mesure), d'autres calculent des erreurs de labels et de frontières (P_k, WindowDiff, ...) et les dernières calculent des erreurs de frontières et de types (SER). En segmentation automatique, des étiquettes 0-1 sont associées aux phrases du corpus. L'étiquette 1 signifie que la phrase est la première d'un segment et l'étiquette 0 signifie que la phrase se trouve au sein d'un segment.

Habituellement utilisées en recherche d'information (Van Rijsbergen, 1979), les méthodes de précision et de rappel ont été appliquées à l'évaluation d'algorithmes de segmentation de textes. Le rappel est défini comme le pourcentage de frontières correctement détectées par l'algorithme et la précision est le pourcentage de frontières identifiées par l'algorithme qui correspondent à des coupures réelles du corpus. Formellement :

$$\text{Précision} = \frac{\textit{Nombre de segments correctement étiquetés}}{\textit{Nombre de segments fournis}} \tag{3}$$

$$\text{Rappel} = \frac{\textit{Nombre de segments correctement étiquetés}}{\textit{Nombre de segments étiquetés dans la référence}} \tag{4}$$

Néanmoins, la relation entre la précision et le rappel tendent à rendre cette tâche difficile. En effet, l'ajout de frontières tendra à améliorer le rappel et à diminuer la précision. En plus du rappel et de la précision, la f-mesure a également été utilisée par Baeza-Yates *et al.* (1999) afin de combiner la précision et le rappel mais l'inconvénient de cette approche est que les résultats sont difficiles à interpréter (Baeza-Yates *et al.*, 1999). De plus, le rappel et la précision ne tiennent pas compte des pénalités liées à la distance à la segmentation réelle obtenue par les algorithmes. Par exemple, soit une segmentation réelle $S = 10101010$ et deux algorithmes de segmentation $Seg_1 = 11000110$ et $Seg_2 = 11110000$, on devrait considérer que l'algorithme Seg_1 est meilleur que Seg_2 parce qu'il construit des séquences plus similaires aux segments réels que Seg_2, ce que ces méthodes ne permettent pas de déterminer. De plus, les segments produits par Seg_1 sont plus similaires à S en termes de taille que ceux produits par Seg_2.

En 1997, une nouvelle méthode d'évaluation P_k est proposée par Beeferman *et al.* (1997) afin de pallier ces problèmes et inclure un système de pénalités plus juste. Elle est calculée en fixant k comme étant la moitié de la moyenne de la taille des segments réels, et en déplaçant une fenêtre de taille k sur le jeu de données. A chaque endroit, l'algorithme détermine si les segments en début et fin de fenêtre appartiennent au même segment ou non, et incrémente un compteur si la réponse est fausse. Le nombre résultant est mis à l'échelle entre 0 et 1 en divisant par le nombre de mesures prises. Un algorithme qui attribue correctement toutes les frontières reçoit un score de 0. Pour justifier cette méthode, Beeferman *et al.* (1999) insistent sur leur volonté d'empêcher les algorithmes de s'adapter aux méthodes d'évaluation. Pour cela, les algorithmes dégénérés qui placent des frontières à chaque position ou ne placent aucune frontière obtiennent approximativement le même score. De plus, les auteurs définissent un faux négatif (également appelé un « raté ») comme un cas où une frontière est présente dans la segmentation de référence mais manquante dans la segmentation hypothétique de l'algorithme, et une assignation faussement positive d'une frontière qui n'existe pas dans la segmentation de référence. L'interprétation de P_k est la suivante : plus il est petit, plus le résultat obtenu se rapproche du résultat de référence.

Plus récemment, Pevzner & Hearst (2002) ont mis en avant des problèmes liés à la méthode P_k : les faux-négatifs sont plus pénalisés que les faux-positifs, le nombre de frontières n'est pas considéré, la méthode est difficilement interprétable lorsqu'elle est supérieure à 0, etc. Ils proposent alors *WindowDiff*, inspiré de P_k et modifient la mesure d'erreur de l'algorithme tout en conservant la caractéristique souhaitable de pénaliser les presque-accidents moins que les faux positifs-purs et les faux-négatifs purs. Le correctif fonctionne comme suit : pour chaque position de la fenêtre, il suffit de comparer le nombre de frontières de segmentation de référence tombées dans cet intervalle (r_i) par rapport au nombre de frontières attribuées par l'algorithme (a_i). L'algorithme est pénalisé si $r_i = a_i$ (qui est calculé comme $|r_i - a_i| > 0$). Formellement :

$$\textit{WindowDiff}(ref, hyp) = \frac{1}{N-k} \sum_{i=1}^{N-k} (|b(ref_i, ref_{i+k}) - b(hyp_i, hyp_{i+k})| > 0) \tag{5}$$

où $b(i, j)$ représente le nombre de frontières entre les positions i et j dans le texte et N représente les fausses pénalités négatives observées dans la méthode P_k. Il capture également les faux positifs et les faux négatifs dans des segments de longueur inférieure à k. Comme pour P_k, WindowDiff est une erreur, donc plus sa valeur est petite, meilleur est le résultat.

Enfin, le *Slot Error Rate* (SER) (Makhoul *et al.*, 1999), fréquemment utilisé en détection d'entités nommées, combine et pondère différents types d'erreurs : insertions (I), délétions (D), erreurs de frontière (F), erreurs de type (T) et erreurs de type et de frontière (TF) et est calculé comme ceci :

$$\textit{SER} = \frac{I + D + TF + 0,5 * (T + F)}{\textit{Nombre de segments étiquetés de la référence}} \tag{6}$$

Cette méthode permet une évaluation rigoureuse de blocs de segments. Cependant, la particularité de la segmentation est qu'il ne s'agit pas de faire de distinction de labels entre deux segments A et B, toutes les phrases d'un segment sont représentées par des 0 et les frontières sont représentées par des 1. En conclusion, cette méthode, bien que complète et interprétable, ne semble pas pertinente pour une tâche de segmentation de texte.

3 Corpus

Le corpus de données utilisé dans cet article a été construit dans le cadre de l'édition DEFT 2006 (Heitz *et al.*, 2006) pour effectuer de la reconnaissance automatique de segments thématiques dans des textes rédigés en français. Nous avons choisi d'utiliser des corpus de domaines spécifiques pour effectuer de la segmentation thématique ce qui nous a conduit à sélectionner les genres suivants : discours politiques et textes juridiques.

Le premier corpus est composé de transcriptions manuelles de discours politiques prononcés par différents Présidents de la République française (Valéry Giscard d'Estaing, François Mitterrand et Jacques Chirac). Les en-têtes des discours comprenant le titre, la date et l'orateur ont été supprimés. Les auteurs ont capitalisé sur les discours de Valéry Giscard d'Estaing, quasiment tous présents dans le corpus, contrairement aux autres Présidents. Cet ensemble de données contient également des entretiens politiques avec des journalistes ou d'autres hommes politiques, tant que l'un des interlocuteurs est l'un des Présidents listés précédemment. La spécificité de ce jeu de données est

qu'il est entièrement en majuscules. La segmentation de référence consiste dans les paragraphes des textes originaux (volumétrie : environ 70 Mo).

Le deuxième corpus est composé d'articles de lois de l'Union Européenne. Les références aux images, les en-têtes, l'article final de signature et les lois de moins de 10 phrases ont été supprimés par les auteurs. Les références sont écrites sous la forme [REFERENCE], comme par exemple [EMPLACEMENT TABLE]. Les numéros des articles, chapitres, titres et annexes ont été remplacés par la lettre « X », comme par exemple « Article X ». Les segments thématiques de référence sont les lois qui peuvent être composées de plusieurs articles (volumétrie : environ 110 Mo).

La Table 1 présente des statistiques descriptives du corpus effectuées par les auteurs (Hurault-Plantet *et al.*, 2006). Les corpus sont très différents en terme de contenu des segments thématiques à retrouver. Les discours politiques contiennent beaucoup de segments courts alors que les textes juridiques contiennent deux fois moins de segments de plus grande taille (1). Malgré une volumétrie nettement plus grande pour les textes juridiques en termes de nombre total de phrases (1) et de mots (1), on note que le vocabulaire est moins riche que dans les discours politiques (1), ce qui signifie que la richesse lexicale est faible dans les articles de lois.

	#phrases	**#segs**	**#phrases/seg.**	**vocab.**	**#mots**	**#mots/phrase**	**#mots/seg.**
Discours	303 373	18 929	16	62 465	8 186 044	27	432
Lois	433 456	9 934	44	57 763	11 555 852	27	1 163

TABLE 1 – Statistiques descriptives sur les mots du corpus d'apprentissage. seg(s). : segment(s) ; vocab. : vocabulaire.

4 Expériences

Afin d'évaluer l'impact des plongements lexicaux sur une tâche de segmentation de texte, nous implémentons les méthodes sémantiques et séquentielles présentées en section (2.3). Cette approche sera comparée à des méthodes statistiques usuelles avec les algorithmes TextTiling et C99 présentés précédemment. Le deuxième objectif de l'article est de comparer des représentations vectorielles pour la segmentation. Pour cela, nous utiliserons la méthode séquentielle et nous comparerons la représentation non contextuelle Word2Vec (utilisée dans le papier d'origine) avec les représentations contextuelles ELMo et Universal Sentence Encoder.

4.1 Pré-traitements

Pour toutes les méthodes, la ponctuation a été supprimée. Les méthodes statistiques testées dans cet article (TextTiling et C99) sont appliquées dans la littérature après une phase de *stemming* de données. En français, cette méthode n'est pas toujours pertinente. Pour cette raison, nous avons remplacé l'étape de *stemming* par une lemmatisation du corpus avec l'outil TreeTagger [1]. De plus, ces documents sont rédigés en majuscules. Dans cet article, nous avons fait le choix de transformer les textes en minuscules. L'étape de lemmatisation a été réalisée avec l'outil sur les textes après transformation en minuscules de ceux-ci.

1. https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/

4.2 Vectorisation

Les modèles vectoriels utilisés dans cet article sont Word2Vec, ELMo et Universal Sentence Encoder. Le modèle Word2vec choisi a été réalisé par Fauconnier (2015) et a été appris sur une sortie Wikipédia contenant 600 millions de mots. Il s'agit d'un modèle skip-gram contenant des vecteurs de taille 1000 et un vocabulaire de mots apparaissant au moins 200 fois dans le corpus. Le modèle ELMo utilisé (Che *et al.*, 2018) a été pré-entraîné sur 20 millions de mots en français sur un corpus Wikipédia et sur le corpus Common Crawl.

4.3 Paramètres

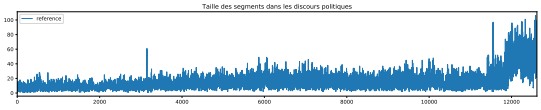
Les méthodes de segmentation ont été lancées sur l'ensemble de données d'apprentissage afin de trouver les paramètres permettant d'optimiser les résultats obtenus. Ces paramètres de calcul ont ensuite été appliqués au jeu de données de test pour l'évaluation des performances des méthodes. Les méthodes TextTiling et C99 ont été utilisées avec des fenêtres de calcul des frontières de taille 10 pour les discours politiques et 35 pour les textes juridiques. La taille optimale des segments utilisée pour la méthode de plongements sémantiques a été fixée à 16 pour les discours politiques et 40 pour les textes juridiques. Enfin, pour la méthode séquentielle, la fenêtre glissante est de taille 35 pour les discours politiques et elle est de taille 20 pour les textes juridiques. La réduction de dimension a été réalisée sur 100 composantes principales pour tous les jeux de données et pour toutes les méthodes de représentation. La variance expliquée par ces axes est comprise entre 69,8% et 89,3%, en fonction des représentations utilisées.

5 Résultats

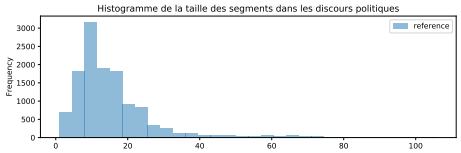
Etape 1 : Comparaison de méthodes neuronales et de méthodes statistiques usuelles. La Table 2 présente les résultats obtenus pour la segmentation de texte sur le corpus (discours politiques et textes juridiques). Les plongements lexicaux permettent une amélioration nette des performances par rapport à des méthodes statistiques usuelles. En comparant la segmentation de référence dans les corpus de discours politiques (cf. Figures 2a et 2b) et de textes juridiques (cf. Figures 3a et 3b), on s'aperçoit que les segmentations sont très différentes. En effet, les coupures permettent d'obtenir des blocs plus homogènes dans les discours politiques que dans les textes juridiques. Etant donné que la méthode séquentielle divise les documents en blocs assez homogènes (cf. Figures 2c, 2d, 3c et 3d), cette méthode est la plus performante pour les discours politiques. La réduction de dimension a permis une légère amélioration des résultats. Ce résultat est intéressant car cela signifie que l'on peut gagner en temps de calcul et en puissance de calcul sur cette méthode en utilisant de plus petits vecteurs, sans dégrader les résultats obtenus. Contrairement à la méthode séquentielle, la méthode sémantique génère des segments moins homogènes, surtout avec l'optimisation gourmande (cf. Figures 2e, 2f, 3e et 3f). Sur les textes de lois, cette méthode permet d'obtenir les meilleures performances de segmentation. Enfin, les indicateurs P_k et WindowDiff (WD) sont concordants, contrairement à la F-Mesure qui, décrite en section (2.4), ne rend pas compte des similarités de segments obtenus mais uniquement des frontières brutes.

	Discours			Lois		
	P_k	**WD**	**F-Mesure**	P_k	**WD**	**F-Mesure**
Text Tiling	0,431	0,431	0,101	0,404	0,404	0,052
C99	0,325	0,614	**0,229**	0,565	0,565	0,083
Sémantique + OG	0,274	0,282	0,187	**0,095**	**0,095**	0,116
Sémantique + OD	0,328	0,282	0,183	0,122	0,097	**0,128**
Séquentiel	0,272	0,277	0,101	0,192	0,194	0,074
Séquentiel + ACP	**0,269**	**0,270**	0,103	0,268	0,188	0,096

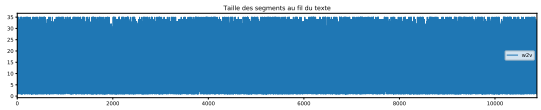
TABLE 2 – Résultats de segmentation obtenus sur les corpus de discours et les textes juridiques. Les méthodes ont été évaluées sur des fenêtres de taille 3-5 pour calculer P_k et WindowDiff et la plus petite erreur a été récupérée. OG : Optimisation Gourmande ; OD : Optimisation Dynamique



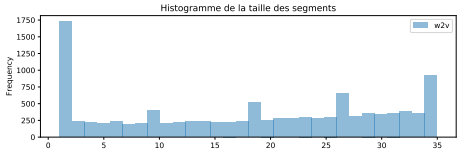
(a) Segments de référence



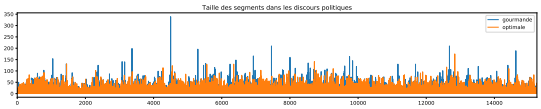
(b) Taille des segments de référence



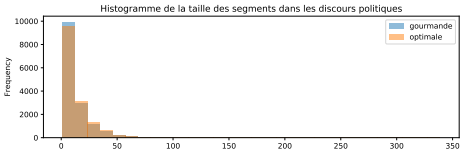
(c) Segments obtenus avec la méthode séquentielle



(d) Taille des segments avec la méthode séquentielle



(e) Segments obtenus avec la méthode sémantique



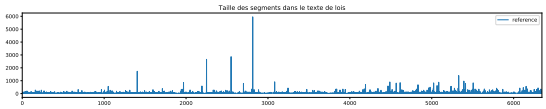
(f) Taille des segments avec la méthode sémantique

FIGURE 2 – Résultats obtenus sur les discours politiques

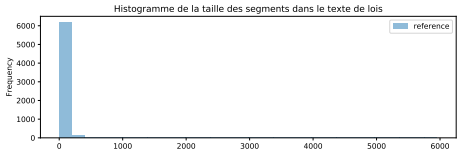
Etape 2 : Représentations contextuelles et non contextuelles. Ici, nous comparons l'utilisation de plongements lexicaux non contextuels (Word2Vec) et des plongements lexicaux contextuels (ELMo et USE) pour la segmentation de textes. L'utilisation de représentations vectorielles complexes n'a pas permis d'améliorer les résultats obtenus avec la méthode sémantique sur les textes juridiques (cf. Table 4). Quelle que soit la représentation utilisée, l'optimisation gourmande est plus performante que l'optimisation dynamique. *A contrario*, l'architecture transformer du modèle USE améliore les résultats obtenus avec la méthode séquentielle sur les discours politiques (cf. Table 3).

6 Conclusion

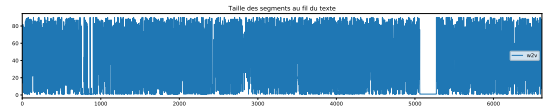
Dans cet article, nous nous sommes intéressés à la segmentation non supervisée de texte pour la détection de thématiques. Nous avons présenté un état de l'art des différentes méthodes utilisées pour



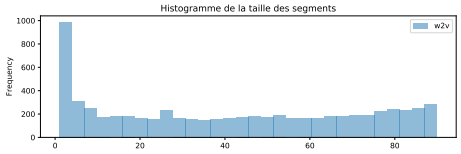
(a) Segments de référence



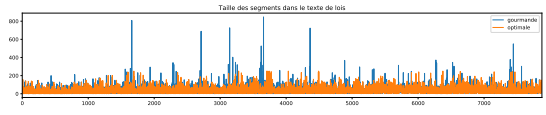
(b) Taille des segments de référence



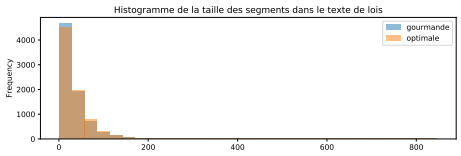
(c) Segments obtenus avec la méthode séquentielle



(d) Taille des segments avec la méthode séquentielle



(e) Segments obtenus avec la méthode sémantique



(f) Taille des segments avec la méthode sémantique

FIGURE 3 – Résultats obtenus sur les textes juridiques

	P_k	**WD**	**F-Mesure**
w2v	0,272	0,277	0,101
w2v + ACP	0,269	0,270	**0,103**
ELMo	0,273	0,276	0,098
ELMo + ACP	0,273	0,271	0,097
USE	**0,239**	**0,239**	0,067
USE + ACP	0,273	0,274	0,093

TABLE 3 – Résultats obtenus avec la méthode séquentielle sur les discours politiques.

	P_k	**WD**	**F-Mesure**
OG + w2v	**0,095**	**0,095**	0,116
OD + w2v	0,122	0,097	**0,128**
OG + ELMo	0,113	0,113	0,076
OD + ELMo	0,148	0,117	0,067
OG + USE	0,101	0,101	0,109
OD + USE	0,135	0,106	0,088

TABLE 4 – Résultats obtenus avec la méthode sémantique sur les textes juridiques.

cette tâche, en particulier pour leur application à des données spécifiques. Plus particulièrement, nous avons évalué l'impact des plongements lexicaux sur la performance des méthodes de segmentation. Nous avons vu que ces méthodes permettent une amélioration nette des résultats en comparaison avec des méthodes statistiques usuelles en segmentation automatique. Ensuite, nous nous sommes intéressés à la performance de deux algorithmes en fonction de la représentation des mots dans le corpus, contextuelle et non contextuelle. La méthode de calcul de plongements lexicaux n'a pas eu d'impact important sur le calcul des frontières de segmentation et semble dépendre de la méthode elle-même. Dans la suite de ce premier travail, nous allons nous intéresser à la mise au point d'une méthode basée sur des plongements de mots en essayant de contourner l'utilisation d'une fenêtre de calcul. Cela est un défaut de la méthode car fixer une fenêtre nous force à calculer un certain nombre de segments figé. Une fois les plongements lexicaux calculés, notre objectif serait de formuler une fonction objectif qui permettrait de maximiser la similarité au sein des phrases d'un segment et entre les segments. Enfin, nous souhaiterions appliquer la méthode de segmentation à des tâches de QA, et d'évaluer l'impact de notre méthode en recherche d'information.

Références

ALEMI A. A. & GINSPARG P. (2015). Text segmentation based on semantic word embeddings. *arXiv preprint arXiv :1503.05543*.

ANGHELUTA R., DE BUSSER R. & MOENS M.-F. (2002). The use of topic segmentation for automatic summarization. In *Proceedings of the ACL-2002 Workshop on Automatic Summarization*, p. 11–12.

BAEZA-YATES R., RIBEIRO-NETO B. *et al.* (1999). *Modern information retrieval*, volume 463. ACM press New York.

BEEFERMAN D., BERGER A. & LAFFERTY J. (1997). Text segmentation using exponential models. *arXiv preprint cmp-lg/9706016*.

BEEFERMAN D., BERGER A. & LAFFERTY J. (1999). Statistical models for text segmentation. *Machine learning*, **34**(1-3), 177–210.

CER D., YANG Y., KONG S.-Y., HUA N., LIMTIACO N., JOHN R. S., CONSTANT N., GUAJARDO-CESPEDES M., YUAN S., TAR C. *et al.* (2018). Universal sentence encoder. *arXiv preprint arXiv :1803.11175*.

CHE W., LIU Y., WANG Y., ZHENG B. & LIU T. (2018). Towards better UD parsing : Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task : Multilingual Parsing from Raw Text to Universal Dependencies*, p. 55–64, Brussels, Belgium : Association for Computational Linguistics.

CHO K., VAN MERRIËNBOER B., GULCEHRE C., BAHDANAU D., BOUGARES F., SCHWENK H. & BENGIO Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv :1406.1078*.

CHOI F. Y. (2000). Advances in domain independent linear text segmentation. *arXiv preprint cs/0003083*.

CHUANG W. T. & YANG J. (2000). Extracting sentence segments for text summarization : a machine learning approach. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, p. 152–159.

DEVLIN J., CHANG M.-W., LEE K. & TOUTANOVA K. (2018). Bert : Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv :1810.04805*.

FAUCONNIER J.-P. (2015). French word embeddings.

GLAVAŠ G., NANNI F. & PONZETTO S. P. (2016). : Association for Computational Linguistics.

HALLIDAY M. & HASAN R. (1976). 1976 : Cohesion in english. london : Longman.

HEARST M. A. (1997). Texttiling : Segmenting text into multi-paragraph subtopic passages. *Computational linguistics*, **23**(1), 33–64.

HEITZ T., AZÉ J., ROCHE M., MELA A., PEINL P. & MEZAOUR A.-D. (2006). Présentation de deft 06 (dÉfi fouille de textes).

HUANG X., PENG F., SCHUURMANS D., CERCONE N. & ROBERTSON S. E. (2003). Applying machine learning to text segmentation for information retrieval. *Information Retrieval*, **6**(3-4), 333–362.

HURAULT-PLANTET M., JARDINO M. & BERTHELIN J.-B. (2006). Ajustement des frontières de segments thématiques détectés automatiquement. *Actes du 2ème dé fouilles de textes (DEFT), Fribourg, Suisse*.

KARUS K. (2019). Using embeddings to improve text segmentation.

KOSHOREK O., COHEN A., MOR N., ROTMAN M. & BERANT J. (2018). Text segmentation as a supervised learning task. *arXiv preprint arXiv :1803.09337*.

MAKHOUL J., KUBALA F., SCHWARTZ R., WEISCHEDEL R. *et al.* (1999). Performance measures for information extraction. In *Proceedings of DARPA broadcast news workshop*, p. 249–252 : Herndon, VA.

MARCU D. (2000). *The theory and practice of discourse parsing and summarization*. MIT press.

MIKOLOV T., CHEN K., CORRADO G. & DEAN J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv :1301.3781*.

MU J., STEGMANN K., MAYFIELD E., ROSÉ C. & FISCHER F. (2012). The acodea framework : Developing segmentation and classification schemes for fully automatic analysis of online discussions. *International journal of computer-supported collaborative learning*, **7**(2), 285–305.

PENNINGTON J., SOCHER R. & MANNING C. (2014). Glove : Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, p. 1532–1543.

PETERS M. E., NEUMANN M., IYYER M., GARDNER M., CLARK C., LEE K. & ZETTLEMOYER L. (2018). Deep contextualized word representations. *arXiv preprint arXiv :1802.05365*.

PEVZNER L. & HEARST M. A. (2002). A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, **28**(1), 19–36.

POLANYI L., CULY C., VAN DEN BERG M., THIONE G. L. & AHN D. (2004). Sentential structure and discourse parsing. In *Proceedings of the Workshop on Discourse Annotation*, p. 80–87.

POURVALI M. & ABADEH P. D. M. S. (2012). A new graph based text segmentation using wikipedia for automatic text summarization. *International Journal of Advanced Computer Science and Applications (IJACSA)*, **3**(1).

PRINCE V. & LABADIÉ A. (2007). Text segmentation based on document understanding for information retrieval. In *International Conference on Application of Natural Language to Information Systems*, p. 295–304 : Springer.

PURVER M., GRIFFITHS T. L., KÖRDING K. P. & TENENBAUM J. B. (2006). Unsupervised topic modelling for multi-party spoken discourse. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, p. 17–24 : Association for Computational Linguistics.

REYNAR J. C. (1998). Topic segmentation : Algorithms and applications.

SALTON G., SINGHAL A., BUCKLEY C. & MITRA M. (1996). Automatic text decomposition using text segments and text themes. In *Proceedings of the the seventh ACM conference on Hypertext*, p. 53–65.

SONG Y., MOU L., YAN R., YI L., ZHU Z., HU X. & ZHANG M. (2016). Dialogue session segmentation by embedding-enhanced texttiling. *arXiv preprint arXiv :1610.03955*.

UTIYAMA M. & ISAHARA H. (2001). A statistical model for domain-independent text segmentation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, p. 499–506.

VAN RIJSBERGEN C. J. (1979). Information retrieval.

VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER Ł. & POLOSUKHIN I. (2017). Attention is all you need. In *Advances in neural information processing systems*, p. 5998–6008.