Segmentation des fichiers logs

Hassan Saneifar, Stéphane Bonniol, Pascal Poncelet, Mathieu Roche

Segmentation des fichiers logs

Hassan Saneifar*,**, Stéphane Bonniol **, Pascal Poncelet*, Mathieu Roche*

*LIRMM, CNRS, Université Montpellier 2; **Satin Technologies

Résumé. Avec la méthode de segmentation appelée passages de discours, la reconnaissance des divisions logiques de documents est essentielle. Cela s'avère plus difficile dans les documents ayant des unités logiques différentes de celles trouvées dans les textes classiques comme les paragraphes ou les sections. Ainsi, nous proposons une méthode automatique pour caractériser les unités logiques complexes propres à ce type de document en fonction de certaines caractéristiques. Ensuite, un processus d'apprentissage supervisé est mis en place afin de pouvoir reconnaître les unités logiques. Les résultats obtenus en utilisant des données issues du monde industriel sont encourageants.

1 Introduction

Des études antérieures montrent que l'extraction d'information dans les fragments de textes donne, en général, de meilleurs résultats que la Recherche d'Information dans l'ensemble du document (Llopis et al., 2002),(Kaszkiel et Zobel, 2001). Ainsi la Recherche de Passages (RP) est considérée comme un moyen pertinent de localiser les informations dans les documents. Cependant, il n'existe pas un accord général sur la manière dont ces passages doivent être définis afin d'obtenir une performance optimale. Les principales différences entre les systèmes de RP résident dans le choix des passages en traitant trois types de questions : Comment définir un passage ? Comment identifier ses limites ? Comment évaluer sa pertinence ?

Dans les méthodes de RP, nous distinguons deux phases principales : (1) la segmentation, (2) le classement (scoring). La segmentation consiste à déterminer les segments de texte qui sont considérés comme des passages candidats. Ici, nous devons nous interroger sur la définition d'un passage et la manière de le reconnaître. Le classement consiste, quant à lui, à évaluer la pertinence des passages candidats en fonction d'une requête.

Dans nos travaux, nous traitons des données textuelles complexes, en particulier des fichiers logs générés par des programmes de EDA [1]. EDA se réfère à des applications de conception assistées par ordinateur pour l'électronique. Elles sont utilisées dans le processus de conception, de test virtuel et de documentation. Les fichiers logs représentent une source principale d'information sur la conception, sur les produits ou même sur les causes de problèmes rencontrés. Les informations contenues dans ces fichiers logs sont utilisées dans le but de vérifier la qualité.

Nous avons précédemment proposé dans (Saneifar et al., 2010) une approche pour évaluer la pertinence des passages candidats (segments) dans les fichier logs EDA. Dans cet article,

1. Electronic Design Automation

nous nous intéressons à la problématique de la segmentation de ces données. La segmentation des fichiers logs s'avère difficile dû à la présence de structures complexes et hétérogènes tels que des tableaux, des listes et des blocs de données. De plus, les aspects multi-sources et multi-structures des fichiers logs doivent être pris en compte dans la segmentation. Ici, nous présentons notre approche qui consiste à caractériser et reconnaître les unités logiques des fichiers logs comme la base de segmentation.

Nous présentons la problématique de la segmentation des fichiers logs ainsi qu'une étude sur les travaux existants dans la section 2. Dans la section 3, nous détaillons les principales étapes de notre approche de segmentation. Nous présentons les résultats des expérimentations dans la section 4.

2 Problématique

La segmentation de texte, utilisée dans des domaines comme les systèmes de Question-Réponse, consiste à diviser un document textuel en plusieurs segments cohérents (Carroll, 2010),(Choi, 2000).

Bien que les fichiers logs générés par des outils différents contiennent *les mêmes informations*, leurs structures et leur vocabulaire peuvent significativement différer en fonction de l'*outil* de conception utilisé. Plus précisément, pour *la même information*, *chaque outil* de conception possède souvent *son propre vocabulaire* et son propre modèle pour structurer les fichiers logs. De plus, les fichiers logs étant différents des données textuelles classiques, les méthodes fondées sur les structures traditionnelles de documents s'avèrent non-pertinentes. Nous présentons par la suite, les travaux existant dans le domaine de la segmentation de texte et étudions la pertinence de chaque méthode dans le contexte des fichiers logs.

Selon (Kaszkiel et Zobel, 2001), Il existe trois catégories principales de méthodes de segmentation : sémantique, fenêtre, discours. La segmentation sémantique consiste à identifier différents sujets véhiculés par le texte, pour le segmenter en unités homogènes formant des blocs thématiques. Ces méthodes utilisent principalement la fréquence des mots ou les relations lexico-sémantiques pour identifier les changements de sujet. Dans notre contexte, ces méthodes ne sont pas nécessairement adaptées. Par exemple, dans les fichiers log, le changement de co-occurrences ne traduit pas nécessairement un changement de sujet. Ce problème est souligné lorsque nous traitons des tableaux ou des données numériques dans des fichiers logs. Il est également possible d'appliquer des méthodes de segmentation fondées sur des fenêtres de taille fixe ou variable. Les fenêtres sont en général constituées des lignes ou des phrases (Kaszkiel et Zobel, 2001). L'un des inconvénients de ces méthodes est le découpage potentiel des passages pertinents (Callan, 1994). Cela peut tout simplement se produire dans le cas des tableaux ou des blocs de données dans les fichiers logs.

Avec l'approche appelée "passage de discours", la segmentation s'effectue selon les unités de discours dans les documents. En effet, les documents ont souvent des unités de discours comme les phrases ou les paragraphes. Ces unités de discours, appelées aussi les unités logiques, peuvent être considérées comme des passages (segments) (Kaszkiel et Zobel, 2001). Afin de les identifier, les éléments marquant le début des unités logiques, comme les lignes blanches ou les alinéas sont souvent exploités. Ces éléments sont appelés les divisions logiques. Comme indiqué dans (Llopis et al., 2002), ces méthodes de segmentation sont plus efficaces car elles utilisent la structure du document. Cependant, les divisions logiques sont

souvent ambiguës avec d'autres types de séparateurs (Tiedemann et Mur, 2008). Par exemple, les contenus des tableaux peuvent être séparés de la même manière que les unités logiques tels que les paragraphes (par ex. en utilisant une ligne blanche).

Dans les fichiers logs, nous considérons des structures complexes (les tableaux, les bloc de données, les listes) comme des unités logiques car elles sont utilisées pour recueillir des idées et des informations. L'identification de ces unités nécessite la reconnaissance exacte des divisions logiques correspondantes. Jörg Tiedemann note dans (Tiedemann et Mur, 2008) que les approches classiques de passages de discours sont souvent inefficaces quand les structures textuelles complexes tels que les tableaux sont mélangées avec d'autres unités classiques comme les paragraphes. Nous montrons par la suite, que nous surmontons ce problème et celui de l'ambiguïté des divisions logiques grâce à notre approche de segmentation qui s'appuie sur une nouvelle méthode de représentation de ce type de données textuelles.

3 Représentation des unités logiques des fichiers logs

Dans cette section, nous présentons notre approche de reconnaissance des divisions logiques fondée sur un système d'apprentissage supervisé. Dans ce cadre, nous développons notre méthode d'acquisition automatique des descripteurs caractérisant les divisions logiques.

Afin d'identifier le début des unités logiques, nous cherchons à identifier des patrons syntaxiques qui différencient une ligne marquant une division logique des autres lignes. Les lignes 86 et 92 de la figure 1 représentent, par exemple, le début de deux unités. De manière simple, nous pouvons caractériser la première division logique par un patron tel que "<- - -><string><fin :>" qui signifie *une ligne commençant par une série de "-", suivi par une chaîne de caractères finissant par un " :"*. Nous considérons ce patron comme un *descripteur*. Pour caractériser une division logique, nous avons besoin d'un ensemble de descripteurs. Nous

84	Total IO Pad Cell Area	: 1076208.64
85		
86	- - - - - - Design Statistics :	
87		
88	Number of Instances	: 13628
89	Number of Nets	: 14293
91		
92	****** Signal Coverages ******	
93	Number of Primary I/O Ports	: 388

FIG. 1 – *Deux unités logiques dans un fichier logs.*

cherchons donc à concevoir une méthode *automatique* d'acquisition de ces descripteurs.

3.1 Acquisition automatique des descripteurs en utilisant les vs-grammes généralisés

Pour créer l'ensemble des descripteurs, nous avons d'abord décidé d'utiliser les n-grammes pour caractériser les divisions logiques. Un n-grammes correspond à une série d'items dans un texte où les items peuvent être des lettres ou des mots. Les n-grammes sont souvent utilisés

comme des descripteurs dans des tâches de classification de documents textuels (Tan et al., 2002). Les n-grammes permettent de modéliser le contenu ainsi que l'enchainement des mots dans un document. Or, dans notre contexte, nous ne cherchons pas à identifier les unités logiques selon leur contenu (les mots ou les lettres) mais selon leurs structures textuelles (les mises en page, les ponctuations, les symboles, etc.). Cette nécessité nous a conduit à définir et proposer un *nouveau type original* de grammes que nous appelons *vs-grammes généralisés*. Les vs-grammes généralisés permettent de modéliser la structure textuelle (la composition des caractères et la mise en page) d'un document tout en étant insensibles au contenu de ce dernier.

Les vs-grammes généralisés

Nous choisissons le terme vs-gramme comme abréviation de "Variable Size Grams". Afin de mieux comprendre la notion de vs-grammes, nous expliquons d'abord les besoins qui nous ont conduit à définir les vs-grammes à partir d'un exemple.

Dans l'exemple précédant (figure 1), la deuxième division logique (ligne 92) est caractérisée par une chaîne de caractères précédée et suivie par une série du symboles "*". En extrayant les n-grammes de taille fixe (par exemple n=3), nous obtenons les tri-grammes suivants sur le premier segment : "`***`","` si`","`gna`","`l c`","`ove`","`rag`","`es `".

Les tri-grammes extraits modélisent bien "*l'enchainement des lettres*" dans les deux lignes représentant le début des unités logiques. Or, la *composition particulière des caractères alphanumériques et non-alphanumériques* qui caractérise ici cette unité logique est difficilement mise en relief par les tri-grammes. Ainsi, les n-grammes de taille fixe ne sont pas pertinents lorsque l'on s'intéresse à la composition des lettres, des ponctuations et des symboles. Afin de prendre en compte cette situation, nous définissons les vs-grammes où les grammes peuvent être de taille variable. Ainsi, un vs-grammes est une série de caractères alphanumériques et non-alphanumériques défini ainsi :

- si le gramme contient une série de caractères alphanumériques, il finit par un caractère non-alphanumérique [2]. Le gramme suivant commence par le caractère non-alphanumérique.
- si le gramme commence par une série de caractères non-alphanumérique, il finit par un caractère numérique. Le gramme suivant commence par le caractère alphanumérique.

En prenant l'exemple de la figure 1, nous obtenons les vs-grammes suivants pour le début du deuxième segment : "`****** S`", "`Signal Coverage *`", "`******`". Contrairement aux tri-grammes précédemment extraits, les vs-grammes modélisent bien la composition des caractères dans cette ligne. Cela signifie que ces vs-grammes expriment une composition de caractères comme une chaîne de lettres entourée par des symboles "*".

Les vs-grammes sont toujours sensibles au contenu des textes. Par exemple, ici, le deuxième vs-gramme extrait, présente une série de lettres *composée* des mots "`signal`" et "`coverage`". Or, la connaissance essentielle à prendre en compte est la présence d'une chaîne de caractères. C'est la raison pour laquelle nous généralisons les vs-grammes en remplaçant les suites de lettres et de symboles par certains symboles représentant leur type et un compteur (par exemple, "+") [3]. Nous remplaçons, par exemple, une chaîne de caractères alphanumériques par le symbole "\w+". Ainsi, sur cet exemple, nous obtenons les vs-grammes généralisés suivants : "*+ S", "\s+\w+ *+", "*+".

2. Les espaces s'ajoutent systématiquement aux grammes.
3. "+" signifie une seule répétition ou plus.

Les vs-grammes permettent de généraliser le contenu des lignes. Ceci constitue l'information essentielle contenue dans les patrons caractérisant les divisions logiques.

Nous constituons l'ensemble des descripteurs en extrayant des vs-grammes généralisés dans une fenêtre de lignes autour des divisions logiques. La taille de la fenêtre constitue un paramètre de notre approche qui est fixée expérimentalement. Chaque descripteur est associé au numéro de la ligne (dans la fenêtre) dans laquelle il est extrait. Le zéro correspond à la ligne même du début de segment (ligne de division logique). Ainsi, en prenant le premier segment de la figure 1, nous obtenons les descripteurs suivants : D_1(\-+ \w+, 0), D_2(\w+ **:**, 0), D_3(\w+ \s+ **:**, -2) et D_4(**:** \w+, -2). Nous procédons de la même manière pour les autres lignes marquant les divisions logiques afin de créer l'ensemble des descripteurs.

3.2 Modèle d'apprentissage des divisions logiques

Une fois l'ensemble des descripteurs déterminé, nous considérons chaque ligne du corpus expertisé comme une instance positive (*division logique*) ou négative (*non division logique*). Chaque ligne est représentée sous forme d'un vecteur booléen dont chaque élément identifie la présence ou l'absence d'un des descripteurs autour de la ligne. Ainsi, nous obtenons un jeu de données d'apprentissage en fonction des descripteurs. Ensuite, un processus d'apprentissage automatique supervisé fondé sur une méthode de classification peut être appliqué. Cela permet de créer un modèle de classification des lignes de corpus en deux classes (début de segment / non début de segment). Ce modèle de classification sera ultérieurement utilisé afin d'associer les lignes d'un nouveau corpus non expertisé à une des classes positive ou négative.

4 Expérimentations

Nous avons évalué la performance de notre approche en terme de précision et de rappel du modèle de classification. Notre corpus d'apprentissage, de taille 1.1 Mo, est constitué de 19 fichiers logs différents issus du monde industriel. Nous présentons ici les résultats obtenus en utilisant les algorithmes de classification qui ont donné les meilleurs résultats : l'arbre de décision (C4.5) et les K plus proches voisins (KPPV). Afin d'évaluer la performance de classification, nous utilisons la méthode de validation croisée (10 niveaux). Le tableau 1 présente la performance des classifieurs. Nous constatons que notre méthode de caractérisation des di-

Classe	Précision	Rappel	F-Score
Pos	0.92	0.74	**0.82**
Neg	0.96	0.98	**0.97**

Classe	Précision	Rappel	F-Score
Pos	0.94	0.75	**0.84**
Neg	0.97	0.98	**0.97**

TAB. 1 – *Performance de classification en fonction de chaque classe - C4.5 (Gauche) et KPPV (Droite)*

visions logiques qui s'appuie sur la notion originale de *vs-grammes généralisés* donne une performance satisfaisante. Avec les KPPV, nous obtenons une précision égale à 0.94 dans la classe positive et égale à 0.97 dans la classe négative. Selon les résultats obtenus, nous confirmons que les vs-grammes généralisés représentent bien les caractéristiques des unités logiques d'un document.

5 Conclusions

Cet article présente une méthode de segmentation des fichiers logs (les textes ayant des unités logiques complexes). Dans notre approche, nous avons constitué un ensemble de descripteurs où chacun présente une des caractéristiques des unités logiques. Nous avons proposé une méthode automatique qui s'appuie sur l'extraction d'un nouveau type de grammes c.-à-d. les vs-grammes généralisés. Les résultats montrent que *les vs-grammes généralisés* peuvent être utilisés pour caractériser les divisons logiques complexes des documents dans le cadre d'une méthode d'apprentissage supervisé.

Références

Callan, J. P. (1994). Passage-level evidence in document retrieval. In *Proceedings of the 17th annual International ACM SIGIR Conference on Research and development in information retrieval*, SIGIR'94, New York, NY, USA, pp. 302–310. Springer-Verlag New York, Inc.

Carroll, L. (2010). Evaluating hierarchical discourse segmentation. In *Human Language Technologies*, HLT'10, Stroudsburg, PA, USA, pp. 993–1001.

Choi, F. Y. Y. (2000). Advances in domain independent linear text segmentation. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics Conference*, San Francisco, CA, USA, pp. 26–33. Morgan Kaufmann Publishers.

Kaszkiel, M. et J. Zobel (2001). Effective ranking with arbitrary passages. *Journal of the American Society for Information Science and Technology 52*, 344–364.

Llopis, F., A. Ferrndez, et J. Vicedo (2002). Text segmentation for efficient information retrieval. In A. Gelbukh (Ed.), *Computational Linguistics and Intelligent Text Processing*, Volume 2276 of *Lecture Notes in Computer Science*, pp. 13–29. Springer.

Saneifar, H., S. Bonniol, A. Laurent, P. Poncelet, et M. Roche (2010). Passage retrieval in log files : an approach based on query enrichment. In *Proceedings of Advances in Natural Language Processing, 7th International Conference on NLP*, IceTAL'10, pp. 357–368. Springer-Verlag.

Tan, C.-M., Y.-F. Wang, et C.-D. Lee (2002). The use of bigrams to enhance text categorization. *Information Processing and Management 38*, 529–546.

Tiedemann, J. et J. Mur (2008). Simple is best : experiments with different document segmentation strategies for passage retrieval. In *Coling 2008*, IRQA'08, USA, pp. 17–25.

Summary

Several application areas require methods for segmenting documents. Depending on the characteristics of our domain, we choiced the segmentation method called "discourse passages" which is based on the identification of "logical units of documents". Thus, we propose here a method to characterize complex logical units found in this type of documents according to their characteristics. Then, a supervised learning process is used to recognize these logical units. Experimental results on the recognition of complex logical units in the log files from the industrial world are encouraging.