Université de Montréal

Système symbolique de création de résumés de mise à jour

par
Pierre-Etienne Genest

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures
en vue de l'obtention du grade de Maître ès sciences (M.Sc.)
en informatique

Avril, 2009

Université de Montréal
Faculté des études supérieures

Ce mémoire intitulé:

Système symbolique de création de résumés de mise à jour

présenté par:

Pierre-Etienne Genest

a été évalué par un jury composé des personnes suivantes:

Philippe Langlais
président-rapporteur

Guy Lapalme
directeur de recherche

Michel Boyer
membre du jury

Mémoire accepté le 17 avril 2009

RÉSUMÉ

Nous présentons dans ce mémoire notre travail dans le domaine de la création automatique de résumés. Nous nous sommes intéressés au problème des résumés de mise à jour d'articles de journaux. Étant donnés un groupe d'articles de journaux et une requête de l'utilisateur en langue naturelle, il faut créer automatiquement un résumé de 100 mots qui répond au besoin d'information énoncé dans la requête. Pour un deuxième groupe d'articles de journaux plus récents et à partir de la même requête, il faut créer un résumé de mise à jour de 100 mots. Un résumé de mise à jour omet toute répétition de l'information contenue dans le plus vieux des deux groupes d'articles.

Nous avons développé le *News Symbolic Summarizer* (NESS), un programme qui crée des résumés de mise à jour. Notre approche est symbolique, dans le sens où nous utilisons un parseur syntaxique pour recuillir des connaissances linguistiques riches. À partir de celles-ci, nous calculons des scores de pertinence pour chaque phrase de chacun des articles et les phrases les plus pertinentes sont extraites pour former le résumé. L'architecture du programme est décrite en détail de même que la méthodologie pour le calibrer.

Nous avons soumis NESS à la compétition de résumés de mise à jour de la *Text Analysis Conference* (TAC) 2008, ce qui permet d'évaluer concrètement la qualité de notre programme et de le comparer avec d'autres systèmes qui tentent de répondre au même problème.

Mots clés : informatique, traitement des langues naturelles, résumés automatiques, résumés de mise à jour.

ABSTRACT

We describe in this thesis our work in the domain of automatic summarization. We were interested in the problem of creating update summaries of newswire articles. Given a cluster of articles and a natural language topic (or query) from the user, we must automatically create a 100-word summary that answers the information need stated in the topic. For a second cluster of more recent articles, the program must create a 100-word update summary. This update summary must omit any repetition of information already included in the older cluster of articles.

We have developed the News Symbolic Summarizer (NESS) which creates such update summaries automatically. Our approach is symbolic, meaning that it relies on a syntactic parser to extract richer linguistic knowledge from the source documents. From this knowledge, the program computes salience scores for each sentence of each article in the cluster, and the most salient sentences are extracted to create the summary. The architecture of the program is described in detail, along with the methodology that we used to calibrate its parameters.

We have participated to the Text Analysis Conference 2008 competition for update summarization with NESS. This has made it possible to compare our program to others which try to answer the same problem. NESS performed well and the evaluation pinpointed to some aspects that we should improve in future versions of it.

Keywords: Computer science, natural language processing, automatic summarization, update summaries

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ANNEXES

REMERCIEMENTS

Merci au professeur Guy Lapalme pour sa direction attentionnée et chaleureuse, sa générosité et sa grande patience. À Fabrizio Gotti pour son aide technique et sa bonne humeur dans le laboratoire. À Myriam et mes parents Bernard-André et Suzanne pour leur appui tout au long de mes études et durant le projet et la rédaction du mémoire.

CHAPITRE 1

INTRODUCTION

L'explosion du *World Wide Web* dans les dernières années a rendu une quantité colossale d'informations disponible à un large public. Les sites d'actualité nous bombardent de nouvelles alors que les blogues se créent et se remplissent à une vitesse fulgurante et que nos boîtes de courriels débordent. Les lecteurs font ainsi face au problème de trier l'information provenant de sources trop nombreuses et trop volumineuses. Dans ce contexte, le besoin d'un accès rapide et simple aux informations pertinentes est grandissant. Les résumés sont une solution appropriée à ce problème.

De plus, les textes arrivent souvent en ligne rapidement et le besoin de les résumer immédiatement existe bel et bien, notamment pour des articles journalistiques. Le site de nouvelles réputé CNN.com [Net09], par exemple, offre déjà au haut de chaque page de nouvelles une section intitulée "*Story Highlights*" qui présente les éléments d'information essentiels de l'article. Ces résumés au style direct et télégraphique (mais nettement plus complets qu'un simple titre) sont créés manuellement, mais un système automatique permettrait à un utilisateur d'obtenir de tels résumés sur n'importe quel site à propos de n'importe quelle nouvelle ; ou mieux encore, d'obtenir un résumé de plusieurs articles provenant de sources différentes mais traitant du même sujet.

Dans le cas des résumés de nouvelles d'actualité, le problème de la redondance d'informations dans les articles subséquents peut survenir. Par exemple, quelques jours après un crash d'avion, le nombre de morts annoncé a possiblement changé, quoique la plupart des informations essentielles de la nouvelle, elles, n'ont pas changé et sont répétées. En supposant que le lecteur visé soit déjà informé sur le sujet, il est désirable d'éviter la répétition des informations connues dans les résumés

suivants. Seuls les éléments d'information qui diffèrent ou s'ajoutent devraient être mentionnés de façon à créer non pas un résumé de l'article lui-même, mais plutôt une mise à jour du sujet, contenant les données nouvelles apportées par les articles les plus récents.

Ce mémoire s'intéresse en particulier au problème posé par la création de résumés automatiques de mise à jour à partir d'articles de journaux. Il s'agit d'un problème bien concret qui possède des applications immédiates. Le système que nous proposons pour y répondre est le *NEws Symbolic Summarizer* (NESS). Notre projet s'inscrit dans un effort conjoint de plusieurs équipes universitaires et industrielles à travers le monde, qui cherchent à solutionner ce même problème précis et avec lesquelles nous avons pu comparer nos résultats.

1.1 Les résumés automatiques

Inderjeet Mani définit de manière très générale le but d'un résumeur dans [Man01] comme étant "de produire une représentation condensée du contenu d'une source pour la consommation humaine". Ici, le résumeur, en tant qu'agent créateur de résumés, peut faire référence à un être humain autant qu'à un programme informatique. Les être humains créent, dans l'état actuel des choses, de bien meilleurs résumés, qui couvrent le contenu de la source de manière plus complète et concise et qui sont plus lisibles. Cependant, les résumés automatiques ont l'avantage d'être généralement accessibles plus rapidement et à un moindre coût. Ainsi, le résumé automatique est une alternative au résumé manuel (humain), non un remplaçant.

Karen Jones définit plus spécifiquement le résumé de texte dans [Jon99] comme étant "une transformation réductive d'un texte source vers un texte résumé à travers une réduction du contenu basée sur une sélection ou une généralisation de ce qui est important dans la source". En d'autres mots, pour résumer, il faut savoir saisir l'essentiel d'un document, séparer l'important du moins important et le rap-

porter dans un texte résumé. Voilà d'ailleurs ce qui fait de la création de résumés une tâche difficile à accomplir automatiquement.

1.1.1 Caractéristiques d'un résumé

Les résumés de texte, automatiques ou non, peuvent s'appliquer à plusieurs problèmes différents. Chaque problème possède des caractéristiques plus ou moins bien définies, que nous présentons dans cette section. Ce qui suit est inspiré du livre *Automatic Summarization* [Man01].

Taux de compression

Le taux de compression, ou taux de réduction, exprime la quantité de texte qui a été compressée par le processus de résumé, en tant que taux par rapport à la taille du document résumé. Notons que certains auteurs préfèrent définir cette même expression comme le complément de notre définition, soit le rapport entre la longueur du résumé et la longueur du texte source. Dans plusieurs applications, c'est plutôt le nombre de mots ou de caractères des résumés qui sera fixé que le taux de compression, qui variera selon la taille des documents sources. Les résumés ont plus souvent un taux de compression aux alentours de 97%.

Auditoire

L'auditoire visé d'un résumé peut être générique ou spécifique à l'utilisateur. Un auditoire générique consiste en un groupe, homogène ou non, connu d'avance et qui peut aller des informaticiens d'un laboratoire de recherche en particulier à la population de toute une université, au grand public en général. Dans le second cas, on tentera d'apprendre des caractéristiques de l'utilisateur dans le but de personnaliser le résumé à ses besoins particuliers.

Relation entre résumé et source

Il existe deux grandes classes de résumés au niveau de la relation entre le résumé et la source : les résumés par extraction et les résumés par abstraction. L'extraction consiste à former un résumé en retirant et réorganisant des groupes lexicaux (mots, propositions, phrases, paragraphes) de la source. À l'opposé, l'abstraction est une reformulation de la source qui peut inclure des entités textuelles n'apparaissant pas dans la source. L'extraction est nettement plus facile à accomplir automatiquement puisqu'aucune génération de texte nouveau n'est requise.

Cohésion et cohérence

La cohérence d'un texte est reliée à sa lisibilité et il s'agit d'un aspect important de son niveau linguistique. Un texte incohérent contient des phrases grammaticalement fautives ou emploie des termes inexistants. On considère que tout résumé créé automatiquement doit être cohérent pour être considéré comme acceptable pour un être humain. Les résumés de texte par extraction sont généralement cohérents lorsqu'ils extraient des phrases entières du document source.

Le degré de cohésion, cependant, peut varier selon le type de résumé que l'on cherche à créer. La cohésion d'un texte s'évalue au niveau de la continuité entre les phrases et de l'organisation des idées contenues. Un résumé doit être fluide, les phrases qu'il contient doivent être reliées entre elles et aucune redondance ne doit être observée. Les résumés de style télégraphique ou même ceux où on ne fait qu'énumérer des mots-clés associés au document source ont une très faible cohésion, quoique cela soit tout à fait acceptable dans certains contextes. Les résumés formés d'un texte continu, en revanche, requièrent généralement une bonne cohésion pour être agréables à lire et du niveau d'un résumé humain. L'ambiguïté référencielle, qui survient par exemple lorsqu'un pronom personnel renvoie à une entité inconnue dans la première phrase d'un résumé, est un problème fréquent, associé à un manque

de cohésion (et même de cohérence) dans les résumés par extraction.

Couverture

Un résumé peut couvrir un seul texte ou plusieurs textes, c'est-à-dire qu'il rapporte les informations essentielles d'un ou plusieurs textes ; dans le second cas, on parle alors d'un résumé multi-documents. Les résumés multi-documents peuvent chercher à faire ressortir les informations communes ou plutôt tenter de répertorier le plus d'éléments différents, tout en évitant les redondances. Les résumés de mise à jour sont une forme particulière de résumés multi-documents pour lesquels certains documents sources contiennent de l'information à éviter d'inclure dans le résumé.

Type de source

Il est possible d'orienter la tâche de création de résumés en fonction d'une connaissance préalable du type des documents sources (articles scientifiques ou de journaux, romans, poèmes, discours politiques, etc.). Le type de source peut permettre de profiter de certaines structures communes pour identifier des stratégies de résumé adaptées. La recherche sur les résumés de documents de type donné obtient d'assez bons résultats, pouvant être commercialisés. C'est le cas notamment dans le domaine des résumés de documents médicaux [AKS05] et des résumés de documents juridiques [Far05].

1.2 Survol du mémoire

Ce mémoire présente le système Ness que nous avons développé dans le cadre d'une participation à la compétition de la conférence TAC 2008. Il s'agit d'un résumeur automatique multi-documents d'articles de journaux capable de créer des résumés de mise à jour sur un sujet donné.

L'organisation du mémoire est la suivante. Le problème que nous avons cherché à

résoudre et le contexte particulier de la conférence TAC sont présentés au chapitre 2. Des travaux connexes ayant inspiré notre recherche ainsi que des approches qui ont été proposées par d'autres équipes sont décrites au chapitre 3. Le chapitre 4 décrit NESS en détail, en incluant les différentes versions qui ont été soumises à la compétition. Les résultats obtenus et leur évaluation sont présentés au chapitre 5, avec également une discussion de ceux-ci. Enfin, au chapitre 6, nous proposerons des travaux futurs suite à notre travail et nous conclurons.

CHAPITRE 2

DESCRIPTION DU PROBLÈME

2.1 À propos de la participation à TAC 2008

Notre recherche a été menée dans le cadre d'une participation à la compétition de la *Text Analysis Conference* (TAC) 2008 [ITL08]. Cette conférence est organisée par le *National Institute of Standards and Technology* (NIST), organisme gouvernemental américain situé près de Gaithersburg, dans l'état du Maryland. TAC supporte la recherche dans le domaine du traitement automatique de la langue naturelle (TALN) en organisant des évaluations à grande échelle. Les buts de TAC incluent la promotion de la recherche en TALN basée sur de grandes collections communes pour effectuer des tests, l'amélioration des mesures et méthodologies d'évaluation en TALN, ainsi que l'accélération du passage de technologies des laboratoires de recherche vers les produits commerciaux en démontrant les améliorations des méthodologies en TALN sur des problèmes concrets.

TAC 2008 est la première édition de cette conférence annuelle. Notons que TAC est née entre autres de l'union de deux prédecesseurs : la branche question-réponse de la *Text REtrieval Conference* (TREC) et la *Document Understanding Conference* (DUC), qui portait sur les résumés automatiques. TAC séparait la compétition 2008 en trois pistes (*tracks*), soient les questions-réponses, les résumés et le *recognizing textual entailment*. La piste sur les résumés comportait une tâche de mise à jour (*update*) et une tâche de résumés de textes d'opinions. Nous avons participé à la première tâche.

2.2 Description de la tâche à compléter

Les consignes de la tâche principale de la branche de résumés de la conférence TAC 2008 ont défini notre problème. Il s'agit de caractéristiques spécifiques proches d'un besoin bien concret, celui de recevoir un résumé automatique de nouvelles de 100 mots (4 à 6 phrases environ) à propos d'un sujet d'actualité précis.

2.2.1 Documents d'entrée

L'entrée du système est constituée d'un groupe de 10 articles reliés de près ou de loin à un sujet d'actualité restreint. Nous utiliserons le mot *cluster* en référence à cette définition précise, parce que c'est le terme employé lors de la compétition et qu'il est plus précis que *groupe* dans ce contexte. Ces articles proviennent de la collection AQUAINT-2, qui inclut des articles de six agences de presse, rédigés en langue anglaise entre le 1er octobre 2004 et le 31 mars 2006. Chaque article compte en moyenne 497 mots, ce qui donne un taux de compression moyen de 98% pour les résumés (100 mots sur 4973 en moyenne par cluster).

Les résumés doivent également répondre à une requête spécifique, en lien avec les informations contenues dans les 10 articles. Les requêtes contiennent un titre et une ou deux phrase(s) complète(s) en forme impérative et définissent un besoin d'information du lecteur. Les requêtes comptent en moyenne un total de 20 mots. On retrouve un exemple de requête à la figure 2.1. Les deux clusters correspondant au même sujet que cette requête contenaient chacun dix articles, pour un total de 4558 et 6227 mots respectivement.

Washington Governor Race. Follow developments concerning the 2004 election for Governor of Washington.

FIG. 2.1 – Requête du sujet D0813B de la collection de TAC 2008.

À chaque cluster est associée une telle requête pour orienter le résumé sur un

besoin d'information spécifique. Un deuxième cluster de 10 articles est fourni pour le même sujet et il fait l'objet d'un second résumé. Les articles du deuxième cluster ont tous parus plus récemment que ceux du premier, ce qui simule un utilisateur cherchant à rester bien informé sur un sujet donné. Le résumé du deuxième cluster doit répondre à la même requête et est un résumé de mise à jour, c'est-à-dire un résumé des informations nouvelles n'apparaissant que dans le deuxième cluster. On doit supposer que le lecteur a déjà pris connaissance de tout le contenu des articles du premier cluster (non seulement du résumé) et, pour répondre à son besoin d'information, éviter la répétition d'informations déjà rencontrées. Le deuxième résumé compte donc en fait 20 articles d'entrée, 10 contenant de l'information nouvelle à résumer, et 10 contenant de l'information à omettre.

La tâche complète requiert donc deux résumés par sujet, chacun sur un cluster de 10 articles d'agence de presse et répondant à la même requête. Le premier est un *résumé standard*, portant sur le *cluster A* et le deuxième est un *résumé de mise à jour*, portant sur le *cluster B* tout en ne répétant rien du cluster A. Nous continuerons d'utiliser cette terminologie dans le reste de ce document.

Lors de la compétition à TAC 2008, 48 requêtes et 960 articles devaient être traités pour rédiger 48 résumés standards et 48 résumés de mise à jour.

2.2.2 Caractéristiques du système

Ayant décrit les entrées du système de résumé, nous savons qu'il doit être multi-documents, orienté sur une requête et capable de compléter un résumé standard et un résumé de mise à jour.

Les résumés doivent de plus être formés de phrases complètes en anglais correct et dans un style d'écriture fluide. On recherche ici à offrir un résumé de la même qualité linguistique qu'un résumeur humain et, comme nous le verrons dans le chapitre 5, les résumés sont en effet comparés à des résumés écrits par des humains lors de l'évaluation.

Notre système emploie une technique d'extraction de phrases décrite en détail au chapitre 4.

2.3 Contexte de la compétition

La compétition s'est déroulée sur seulement 11 jours, du 1er juillet (réception des données) au 11 juillet 2008 (soumissions finales). Trois séries de résultats (*runs*) pouvaient être soumises, pour nous donner la chance de faire évaluer différentes configurations du système. 31 équipes ont participé en 2008 à cette branche de la compétition et 71 séries de résultats ont été reçues. Nos résultats sont présentés au chapitre 5.

CHAPITRE 3

TRAVAUX CONNEXES ET APPROCHES RÉCENTES

3.1 Travaux connexes

Les articles que nous présentons ici ont beaucoup influencé la recherche dans le domaine de la création automatique de résumés. En particulier, les trois premiers sont souvent cités pour leur rôle historique dans le développement du domaine.

3.1.1 Luhn, 1958

Les travaux de Luhn [Luh58] sont généralement cités comme étant les premiers effectués en création automatique statistique de résumés par extraction. L'idée de calculer la fréquence des termes présents dans le texte et dans chaque phrase est à l'origine de toutes les techniques les plus performantes d'aujourd'hui, alors que Luhn fut le premier à décrire une méthode statistique simple pour le faire. Il fait aussi usage de *stemming*, technique couramment utilisée pour regrouper les termes de même famille lexicale.

Son approche utilise la fréquence de termes dans les phrases pour l'extraction, afin de créer des résumés d'articles scientifiques. Son objectif était de déterminer quels mots sont véritablement déterminants dans le document à résumer, soient ceux qui apparaissent fréquemment, sans être simplement des mots outils non significatifs. L'algorithme de Luhn filtre les termes considérés comme communs, parmi ceux présents dans le document, à l'aide d'un anti-dictionnaire (*stoplist*) créé manuellement et contenant les pronoms, articles et prépositions de langue anglaise. Les autres termes sont regroupés par formes similaires : les termes possédant un préfixe commun et pas plus de 6 caractères différents sont considérés comme représentant une même notion. Il compte le nombre d'occurrences de chaque groupe lexical.

Les termes peu fréquents sont éliminés pour ne conserver qu'une liste restreinte de termes fréquents dans le document, mais qui ne sont pas des mots outils. Ces mots sont considérés comme significatifs pour le document, selon son vocabulaire. Les phrases reçoivent un score selon le nombre et la proximité des mots significatifs qu'elles contiennent. Les meilleures phrases sont extraites pour former le résumé.

3.1.2 Edmundson, 1969

L'utilisation de la structure du document en création automatique de résumé mène généralement à une amélioration du système et Edmundson fut le premier à la proposer. Dans ses travaux [Edm69], il teste 3 nouveaux critères d'évaluation des phrases en vue de l'extraction pour la création automatique de résumés, dont deux utilisent la structure du texte. Les meilleurs résultats sont obtenus en combinant plusieurs critères, une innovation à l'époque.

En plus de la méthode par fréquences de mots saillants de Luhn, les 3 critères suivants ont été proposés : la présence de mots clés positifs ou négatifs (la présence d'un mot clé positif dans une phrase, comme "Cet article a pour but" ou "En conclusion", la rend plus susceptible d'être extraite pour le résumé, alors qu'un mot clé négatif, tel que "impossible", accomplit l'inverse) ; la présence dans la phrase de mots appartenant à un titre, sous-titre ou entête du document ; la position de la phrase dans le texte (en supposant que les phrases pertinentes à un résumé se retrouvent au début et à la fin des sections et paragraphes du document). Une méthodologie détaillée a été développée par Edmundson pour comparer les différentes approches. Un corpus est séparé en un ensemble d'articles d'entraînement et un ensemble de test. Dans la phase d'entraînement, des modifications manuelles sont apportées aux paramètres du programme, en plus d'utiliser l'ensemble d'entraînement pour déduire les mots clés. Par comparaison, les trois nouveaux critères ont produit de meilleurs résultats que la fréquence de termes de Luhn. L'approche produisant les meilleurs résumés, selon l'étude d'Edmundson,

est une approche hybride consistant à combiner les trois nouveaux critères en une somme non-pondérée : la fréquence de mots clés, la fréquence de mots de titre et la position des phrases.

3.1.3 Pollock et Zamora, 1975

Les travaux de Pollock et Zamora [PZ75] se distinguent par la spécificité de la tâche qu'ils cherchent à accomplir : dans leur cas, la création de résumés d'articles scientifiques de chimie. Ils montrent les avantages spécialiser un système de résumés à un type prédéterminé de documents et expliquent comment exploiter ces connaissances additionnelles. Ils sont aussi parmi les premiers à faire de la compression de phrase, suite à l'extraction, pour raccourcir le résumé davantage. Il est intéressant de noter que leur programme est devenu le premier système de création automatique de résumés à être commercialisé.

Leur technique est basée sur la présence de mots clés (très similaire au critère du même nom d'Edmundson), dont la liste est cependant spécifique à la chimie. En particulier, ils s'intéressent à des sous-domaines de la chimie, chacun appelant une liste spécifique à ce champ pour la création de résumés.

Un critère de fréquence de termes comme celui de Luhn est également utilisé, mais plutôt que de le combiner à un autre critère pour former un score comme chez Edmundson, ils l'utilisent pour moduler les résultats qui proviennent de l'approche par mots clés. Un mot clé rencontré fréquemment dans le corpus est considéré comme moins indicatif d'une phrase propice à un résumé qu'un mot clé rencontré peu fréquemment.

Pollock et Zamora innovent aussi par l'utilisation d'une méthode pour faire de la réduction de phrase par élimination de propositions. Ils ont compilé une liste paires mot-type (où le type est verbe, nom, adjectif, etc.). Ils se servent ensuite d'une grammaire pour classifier les virgules de chacune des phrases, en supposant que les virgules peuvent indiquer un changement de proposition. Les propositions

sont classifiées selon le type des mots autour des virgules. Les propositions introductives, celles qui se terminent par “that” ou qui débutent par “in”, de même que les appositions, sont éliminées des phrases extraites pour le résumé. Aussi, l'orthographe des mots, les abréviations et les composés chimiques sont standardisés, grâce à une banque de règles de transformation.

3.1.4 Marcu, 2000

Les travaux de Marcu [Mar00] s'intéressent à l'utilisation de l'analyse de la structure du discours (“discourse parsing” ou “rhetorical parsing”) pour la création de résumés automatiques de texte. La théorie du discours attribue aux textes possédant une bonne cohésion une structure interne caractérisable par des relations rhétoriques. Marcu propose d'utiliser un analyseur rhétorique, qui détermine automatiquement des relations de ce type, comme outil principal pour la création automatique de résumés. Il prétend ainsi que ses résumés contiennent les phrases les plus essentielles à la structure du discours du texte résumé. Cet outil peut aussi servir d'auxiliaire à une méthode statistique de création de résumés.

Marcu utilise la terminologie et les relations de la *Rhetorical Text Structure* (RTS) proposée par Mann et Thomson [MT88] et formalisée par Hovy dans [Hov88]. Les relations rhétoriques, telles que “anti-thèse”, “circonstances” et “justification”, unissent deux ou plussieurs parties de texte non-interposées, où l'une des parties joue le rôle de noyau et l'autre ou les autres celui de satellite. À partir de ces relations, un arbre est formé lors d'une analyse complète, dans lequel les noeuds peuvent être abstraits ou explicites. L'analyse peut être effectuée automatiquement en utilisant plusieurs méthodes comme l'identification de propositions et de mots clés ou à partir d'un système de règles dérivées automatiquement par apprentissage.

Selon Marcu, les phrases les plus pertinentes pour un résumé sont celles qui appraissent au haut de l'arbre d'analyse obtenu par une analyse rhétorique du document. Ce sont celles qui possèdent le plus grand nombre de satellites (ou

enfants) servant à mieux les définir. Ceci fait donc un lien avec le concept de base de ce qu'est un résumé : chercher à représenter la même information mais de façon plus concise (donc avec moins d'explications, de contextualisation, etc.). En plus de ce procédé dépendant uniquement de l'analyse rhétorique, il propose également des méthodes où cette analyse sert d'outil pour améliorer la qualité des résumés produits par d'autres systèmes de résumés. Marcu suggère notamment d'utiliser les méthodes statistiques traditionnelles déjà discutées, mais d'imposer que les phrases que l'on peut identifier comme auxiliaires, à cause de leur profondeur dans l'arbre ou du type de relation dont elles sont le satellite, ne soient pas extraites. Cette approche hybride semble produire de meilleurs résumés.

3.1.5 Goldstein et al., 2000

Goldstein et al. [GMCK00] énumèrent les difficultés spécifiques aux résumés multi-documents (un résumé pour une collection de documents possédant le même thème) et proposent des alternatives pour y répondre. Ils s'intéressent à la problématique de la création de résumés automatiques d'articles de nouvelles, sous la contrainte d'une description du sujet d'intérêt pour l'utilisateur (une requête qui décrit le besoin d'information), soit sensiblement le même problème que celui que nous étudions dans le cadre de ce mémoire.

Les difficultés spécifiques aux résumés multi-documents incluent selon eux : une plus grande redondance dans la source, puisque les documents peuvent répéter les mêmes informations alors que le résumé doit éviter cette répétition ; un changement d'information dans les articles plus récents, puisque les articles n'ont pas tous la même date ; un taux de compression plus grand, puisque le résumé doit rester petit mais que le groupe de documents d'entrée peut contenir des dizaines de documents ; et un problème de co-références (le problème où une phrase extraite inclut un pronom dont le référé n'est pas présent dans celle-ci) qui est encore plus important lorsque les documents n'identifient pas toujours les mêmes entités de la même façon.

Leurs travaux proposent une solution possible à ces problèmes en intégrant à une méthode statistique déjà utilisée pour des résumés d'un seul document des critères qui ciblent spécifiquement les difficultés mentionnées. Le critère de couverture donne un score plus élevé à une phrase dont les mots apparaissent dans la plupart des documents de la collection (fréquence dans les documents). Le critère de séquence temporelle favorise les phrases contenues dans les documents plus récents. Un critère de non-redondance utilise la similarité de termes entre les phrases déjà sélectionnées pour former le résumé et les candidates suivantes. Enfin, un critère pénalise la sélection de phrases qui sont situées dans le même document et celles qui sont rapprochées dans le même document.

3.1.6 Radev et al., 2004

Les travaux de Radev et al. [RJST04] utilisent le concept des centroïdes pour créer des résumés multi-documents par extraction. Le but de leur technique est de maximiser la pertinence d'une phrase extraite par rapport au sujet du cluster, tout en minimisant la redondance entre les phrases retenues pour former le résumé. Notre système partage plusieurs éléments avec les travaux de Radev et al. : utilisation des valeurs $tf \cdot idf$, utilisation du rang d'une phrase dans le document, pondération de plusieurs critères pour former un score, mesure de similarité entre deux phrases.

Le cluster D de documents à résumer est représenté par un vecteur nommé centroïde dont chaque composante est un terme qui apparaît dans le cluster. La valeur associée au mot w_i est $v_{w_i} = \text{TF}(w_i) * \text{IDF}(w_i)/|D|$, où $\text{TF}(w_i)$ est la fréquence normalisée du mot w, $\text{IDF}(w)$ est le log de l'inverse du nombre de documents de tout le corpus dans lesquels apparaît w et $|D|$ est le nombre de mots dans le cluster de documents. Le centroïde du document est représenté par $\text{CENTROIDE}(D) = (v_{w_0}, v_{w_1}..., v_{w_{|D|}})$.

Trois critères sont utilisés pour donner un score aux phrases s_j du cluster : la valeur centroïdale, la valeur positionelle et la redondance avec la première phrase.

La valeur centroïdale d'une phrase $C(s_j)$ est calculée en faisant une somme des valeurs centroïdales v_{w_i} de chaque mot de la phrase. La valeur positionnelle est donnée par $(|D| - j + 1)/|D|$ où j est le rang de la phrase dans son document (et non dans tout le cluster). La redondance avec la première phrase se calcule par le produit scalaire entre le vecteur des TF(w_i) de la phrase à considérer et la première phrase du document dans lequelle elle se situe. Les trois critères sont normalisés de sorte qu'ils produisent des valeurs entre 0 et 1 uniquement.

Le score final pour chaque phrase est donné par une somme pondérée des trois critères (le troisième prend une pondération négative). Pour minimiser la redondance entre deux phrases de résumé, une mesure de similarité entre les phrases est utilisée. Cette mesure correspond au nombre de mots communs entre deux phrases, divisé par le nombre total de mots dans celles-ci. Le score final déjà mentionné est donc modifié négativement par la mesure de similarité avec une phrase possédant initialement un meilleur score. Un algorithme itératif permet d'obtenir une liste ordonnée par le score ainsi modifié. Les meilleures phrases sont extraites et forment le résumé, dans lequel elles apparaissent en ordre de date des documents qui les contiennent.

3.1.7 Mihalcea, 2004

Les travaux de Mihalcea [Mih04] utilisent un algorithme à base de graphes pour créer automatiquement des résumés. Les approches par graphes, dont ces travaux servent de base, ont gagné en popularité dernièrement. Par exemple, l'approche de Chen et al., présentée plus loin dans ce chapitre, utilise une les graphes de façon plus complexe mais similaire.

Les sommets du graphe d'un document à résumer sont ses phrases. Ces sommets sont rejoints par des arêtes qui portent une pondération représentant la similarité entre les phrases. Cette similarité est évaluée par la somme des mots communs entre les deux phrases, divisée par la somme des logs de la longueur de chaque phrase.

Grâce à ce graphe, on peut donner un score final aux phrases selon leur "degré d'influence" sur le reste du document. Un processus itératif basé sur le PageRank utilisé dans Google [BP98] permet de calculer le score final. Le score est la somme pondérée (par la similarité entre les phrases) des scores de toutes les phrases qui se rattachent à la phrase d'intérêt. Cet algorithme est garanti de converger en un temps polynomial. Les phrases possédant le meilleur score sont enfin extraites pour former le résumé.

Mihalcea a testé cette approche en la comparant à 5 autres approches récentes à ce moment et obtient les deuxièmes meilleurs résultats. Certaines approches par graphes se retrouvent encore parmi les plus performantes aujourd'hui.

3.1.8 Gagnon et Da Sylva, 2006

Les travaux de Gagnon et Da Sylva [GS06] portent spécifiquement sur une méthode symbolique pour la compression de texte, une sous-tâche de la création automatique de résumés. Il y a un grand intérêt à effectuer de la compression de texte sur les phrases d'un résumé pour améliorer le taux de compression du résumé, ou pour augmenter le nombre de phrases (donc le nombre d'idées distinctes, s'il n'y a pas de redondance) dans un résumé de taille fixe. Leur méthode consiste à réduire la taille des phrases en coupant certaines parties non-essentielles, en se basant uniquement sur une analyse syntaxique de la phrase. Puisque le but de cette compression est de maintenir la grammaticalité et la sémantique de la phrase, il s'agit en fait d'une méthode de résumé, quoique les systèmes qui nous intéressent possèdent des taux de compression nettement plus bas que ce que l'unique compression des phrases peut offrir (de l'ordre de 70% de compression, alors qu'on désire généralement un taux de compression supérieur à 95%). Cette technique peut néanmoins être utilisée de paire avec un système de résumé automatique par extraction, soit en effectuant d'abord la sélection des phrases pertinentes puis ensuite la compression, ou en compressant d'abord toutes les phrases avant d'en faire

une sélection pour l'extraction.

Chaque phrase du texte à compresser est analysée à l'aide d'un parseur syntaxique qui identifie et classifie les dépendances entre les mots de chaque phrase. La sortie du parseur représente chaque phrase sous la forme d'un arbre d'analyse, qui inclut des relations grammaticales entre les mots qu'elle contient. Les phrases peuvent ensuite être compressées en coupant des sous-arbres identifiés comme n'étant pas essentiels à la structure de la phrase. En particulier, la proposition principale de la phrase est toujours conservée, ce qui rend la réduction de phrases très courtes (6 mots et moins) rarement possible. Un anti-filtre prévient par la suite le retrait de certains sous-arbres qui représenteraient une trop importante coupure. De 80 à 90% des phrases peuvent être réduites par leur technique qui s'applique en particulier à la langue française. Environ 25% des réductions étaient jugées erronées par un évaluateur humain, ce qui peut être trop élevé pour plusieurs applications, dont le résumé. Ce taux est très dépendant de la qualité initiale de l'analyse syntaxique automatique de la phrase et serait donc amélioré si les analyses utilisées étaient de meilleure qualité.

3.2 Approches récentes de TAC 2008

Cette section présente un aperçu de certaines approches qui ont reçu de bons résultats à la compétition de la Text Analysis Conference (TAC) 2008. Elles serviront de points de comparaison avec notre système au chapitres 5.

3.2.1 Chen et al. : distance d'information

Deux programmes donnant de bons résultats aux évaluations ont été développés par Chen et al. [CYL+09] de l'Université Tsinghua. Le premier utilise le concept de distance d'information pour créer des résumés. Le groupe propose que la théorie de l'information connue sous le nom de complexité de Kolmogorov permet de mo-

tiver leur approche, quoique celle-ci soit plutôt une nouvelle version des approches statistiques habituelles, la plupart déjà décrites dans ce chapitre.

Selon, la théorie de Kolmogorov, la complexité d'une chaîne de charactères x est proportionnelle à la longueur du plus court programme pouvant produire cette chaîne, $K(x)$. On peut également s'intéresser à la complexité $K(x|y)$ de créer une chaîne x sous la condition d'une autre chaîne y (prenant la seconde chaîne en entrée). La distance maximale entre deux chaînes se définit ainsi par $D_{max}(x, y) = max\{K(x|y), K(y|x)\}$.

Pour appliquer cette théorie au choix d'un résumé S, étant donnés un cluster d'articles A, on peut reformuler le problème comme un problème de minimisation de la valeur de $D_{max}(S, A)$. En pratique, la valeur de cette expression doit être grandement approximée ; ils proposent l'approximation $D_{max}(S_i, A) \simeq |S_i|$, où $|S_i|$ est le nombre de "mots importants" dans la phrase S_i du résumé S.

Les mots importants sont définis comme des mots n'étant pas des stopwords, et dont la fréquence-document (le nombre de documents contenant le mot parmi les dix documents d'entrée) est supérieure à un certain seuil. Une heuristique est utilisée pour éviter de sélectionner des phrases trop similaires et un score additionnel est attribué aux phrases contenant les entités nommées présentes dans la requête. Les 15 meilleures phrases selon cette évaluation sont ensuite combinées dans toutes les permutations (de 100 mots ou moins) possibles pour former un grand nombre de résumés candidats. Le même procédé (compter les mots importants) est ensuite utilisé pour déterminer quel résumé est le meilleur parmi les résumés candidats, ce qui est donné dans leur formalisme par $D_{max}(S, A) \simeq |S|$. Aucune description de la démarche suivie pour éviter la redondance avec le premier cluster n'est donnée.

La théorie de Kolmogorov semble jouer un rôle plutôt limité pour justifier le choix d'heuristique de leur approche, tant les simplifications qui doivent être faites sont grandes.

3.2.2 Chen et al. : centralité des phrases

L'université Tsinghua a proposé une seconde approche, très différente, mais aussi performante que la précédente. Elle se base cette fois sur le concept de centralité des phrases pour créer un résumé, semblable au modèle à base de graphe de Mihalcea. L'approche se distingue par sa manière innovatrice de faire de la mise à jour et par la qualité de la vérification de la redondance. Comme nous le verrons au chapitre 5, cette approche a obtenu globalement les meilleurs résultats lors de la compétition.

Dans cette approche, à chaque mot est attribué un score, calculé par le produit de la fréquence du terme dans tout le cluster et du nombre de documents d'entrée dans lequel le mot apparaît. Uniquement les 300 mots ayant le meilleur score sont conservés pour le reste de l'analyse. Une matrice termes-phrases est construite avec ces mots et un algorithme de *Latent Semantic Analysis* (LSA) est utilisé pour la décomposer en unités sémantiques. Il en résulte une matrice de similarité entre les phrases qui permet de donner un score de similarité à chaque paire de phrases. Ces scores sont utilisés comme poids des arêtes dans un graphe où les sommets sont les phrases elles-mêmes. C'est avec ce type de graphe que le concept de centralité sera utilisé pour déterminer les phrases candidates.

On peut considérer une phrase comme pertinente si elle est similaire aux autres phrases des documents à résumer. En d'autres termes, si une phrase dans le graphe possède de forts poids de similarité avec d'autres phrases importantes dans le graphe, alors cette phrase devrait être elle-même importante. Cette méthodologie permet de faire de la mise à jour. Il faut pour cela relier les phrases en A entre elles et de les faire lier avec des poids négatifs avec les phrases en B.

Les phrases candidates sont sélectionnées si elles possèdent une haute centralité et si elles se situent près du début d'un document (une mesure bien connue dans les résumés d'articles de journaux). Encore une fois, cette équipe choisit de garder

un ensemble de 15 phrases candidates pour former tous les résumés possibles provenant de combinaisons de ces phrases pour faire 100 mots au total. Le score d'un résumé est basé sur la non-redondance des phrases entre elles et la couverture de l'information qu'elles contiennent. Un résumé est ignoré si deux de ses phrases sont très semblables selon la matrice de similarité (selon un seuil de 50% de la similarité maximale). Une formule ad-hoc est utilisée pour compléter l'évaluation des phrases ; celle-ci prend en compte le nombre d'occurrences de chaque mot dans le résumé, ainsi que dans les phrases qui le contiennent et la centralité de celles-ci. Le résultat final est un résumé dont les phrases sont peu similaires, dont les mots sont variés et où les mots fréquents dans le résumé font partie de phrases à centralités élevées. Notons qu'aucune compression des phrases n'est effectuée par cette équipe, selon leur article.

3.2.3 Gillick et al.

Gillick et al., de l'International Computer Science Institute (Berkeley), ont proposé une approche innovatrice basée sur la maximisation du nombre de *concepts* présents dans le résumé [GFT09]. Les concepts sont définis comme des n-grammes de radicaux (mots tronqués de leur terminaison). Cette méthode est inspirée de la métrique d'évaluation automatique ROUGE, qui est basée sur la comparaison des ngrammes (spécifiquement des bigrammes dans ROUGE-2) entre des résumés humains et un résumé automatique sur la même tâche. ROUGE est décrit en plus de détails à la section 4.8. Comme nous le verrons au chapitre 5, ce système est le plus performant dans les évaluations de ROUGE et en particulier de ROUGE-2.

Pour évaluer la pertinence d'un concept, seule la fréquence d'apparition des mots qui le constituent parmi les documents du cluster à résumer est utilisée. Spécifiquement, un concept pertinent est défini comme un bigramme qui apparaît dans plus de 3 documents sur 10. Les mots faisant partie d'un anti-dictionnaire (stop-list) sont ignorés, mais pas les bigrammes contenant au moins un mot perti-

nent. Les phrases qui n'ont aucun mot en commun avec la requête sont ignorées.

Un problème d'optimisation est défini et résolu sous la forme d'un *Integer Linear Program* (ILP). La fonction à maximiser est le nombre de concepts pertinents différents qui apparaissent dans le résumé. Formellement, les contraintes sont le nombre de mots d'un résumé (100), le fait qu'une phrase est sélectionnée si et seulement si tous ses concepts le sont aussi et le fait que les concepts et phrases aient des valeurs de 0 ou 1, pour indiquer la présence ou l'absence du concept et de la phrase dans le résumé. Le nombre d'occurrences d'un concept est donc sans importance tant qu'il apparaît au moins une fois dans le résumé ; cette propriété devrait logiquement tendre à faire éviter les redondances. Le résumé qui maximise la fonction à optimiser tout en respectant les contraintes est sélectionné. Les phrases sont ordonnancées chronologiquement et en ordre d'occurrence dans un article donné, comme dans la plupart des systèmes participant à la compétition.

CHAPITRE 4

LE SYSTÈME NESS

Pour répondre au problème décrit au chapitre 2, nous avons créé le *NEws Symbolic Summarizer* (NESS). Ce chapitre a pour but de décrire son architecture, la méthodologie employée pour calibrer ce système et les différentes versions du système qui ont été soumises à la conférence. Un article le décrivant [GLNW08] a été soumis à TAC 2008 et il est reproduit en annexe.

Le principe directeur de NESS est l'extraction de phrases du cluster d'articles pour créer un résumé avec celles qui sont jugées les plus pertinentes pour combler le besoin d'information exprimé par la requête. Les phrases des textes d'entrée sont donc l'unité lexicale la plus importante de notre application.

Dans ce chapitre, nous décrivons d'abord chaque module du système. La figure 4.1 illustre les étapes du traitement automatique complet permettant de passer de deux clusters de 10 articles de journaux et d'une requête à deux résumés de 100 mots ou moins, de façon à répondre au problème décrit au chapitre 2. Les fléches du diagramme indiquent le flux d'information entre les modules.

L'implémentation informatique de NESS est en majeure partie du *Extensible Stylesheet Language Transformations* (XSLT [Rec99]). Plusieurs outils ont aussi été développés en Java et Python à l'occasion. Le BASH-script a été utilisé abondamment pour traiter efficacement le grand nombre de fichiers de la compétition, pour gérer les appels successifs à différents programmes, ainsi que pour automatiser les tests. Les programmes FIPS et Wordnet ont aussi été employés.

4.1 Pré-traitement

Le rôle du pré-traitement est de nettoyer les données d'entrée (articles et requêtes) et de les transformer dans le format voulu. Des corrections sont également

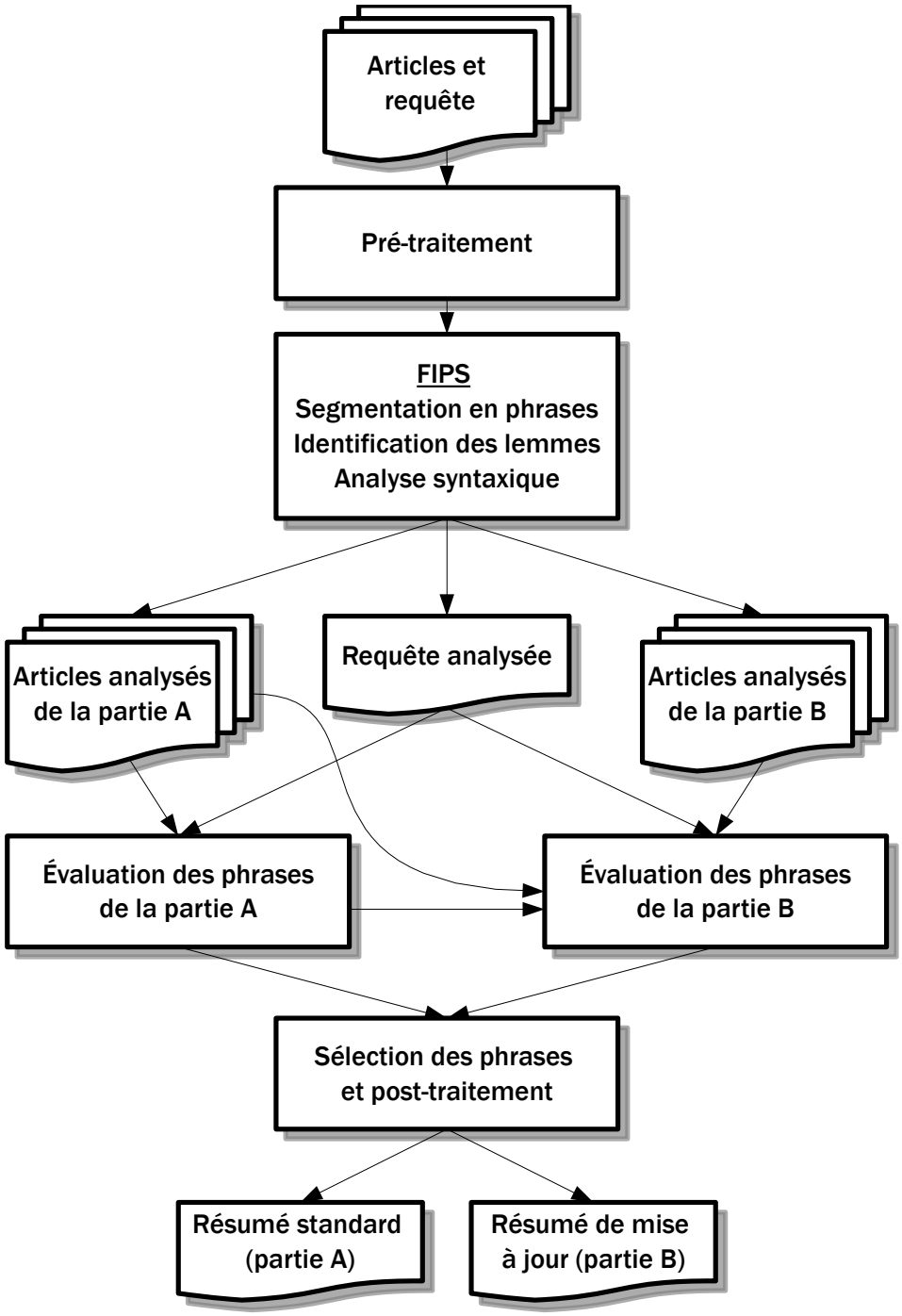

FIG. 4.1 – étapes d'exécution de NESS.

apportées pour éviter certains problèmes qui pourraient survenir dans les phases suivantes.

Divers caractères qui peuvent poser problème lors de l'exécution de FIPS (voir la section 4.2) sont supprimés. Un filtre qui retire l'initiale du milieu de noms propres en anglais (le W. dans George W. Bush par exemple) est entre autres appliqué pour éviter les erreurs de segmentation des phrases que cela générait dans FIPS. Ce nettoyage, qui n'enlève que rarement un élément réellement essentiel d'une phrase, améliore significativement les résumés produits par notre système, en évitant qu'ils contiennent des phrases tronquées et incohérentes.

Des informations contenues dans les fichiers d'entrée mais non pertinentes à notre traitement sont également retirées à ce moment-ci. Notamment, le titre, l'entête, la date et le code d'identification des articles sont retirés pour ne garder que le texte des articles.

Les requêtes sont modifiées de sorte que le titre de la requête soit répété deux fois, suivi d'une unique transcription des énoncés. Cette modification, un choix heuristique, sert à augmenter le poids proportionnel du titre en relation avec le texte de chaque requête.

Suite à ces étapes, les données sont dans le format voulu pour effectuer l'analyse syntaxique.

4.2 Analyse syntaxique et Fips

L'étape que nous appelons analyse syntaxique inclut trois traitements essentiels de notre méthodologie : la segmentation en phrases, l'identification des lemmes et la création d'un arbre syntaxique. Ces trois tâches sont exécutées par l'outil très puissant qu'est FIPS. Il s'agit d'une étape, fondamentale à notre approche symbolique, qui n'est généralement pas effectuée par d'autres systèmes de création de résumés.

4.2.1 Fips, un parseur syntaxique

Fips [Weh07] est un parseur syntaxique développé par Luka Nerima et Eric Wehrli du *Language Technology Laboratory*(LATL) de l'Université de Genève. Il nous a été gracieusement rendu disponible pour compléter ce projet. Nous avons également pu collaborer avec Luka et Eric pour régler quelques problèmes et bugs du programme, spécifiques et non-spécifiques aux textes de TAC 2008.

Fips est un programme fonctionnant uniquement sous le système Windows qui prend en entrée du texte non-formaté à être analysé et produit en sortie un arbre syntaxique de chaque phrase rencontrée. Les lemmes sont identifiés pour chaque mot et fournis à la sortie. Dans la toute dernière version, le genre, le nombre et la fonction grammaticale des mots, là où applicable, étaient également inclus. Ces informations grammaticales n'ont cependant pas été utilisées par Ness.

Le texte reçu en entrée par Fips peut provenir du clavier ou de fichiers. Dans notre cas, ce sont tous les articles et les requêtes qui ont été soumis à Fips pour l'analyse syntaxique. La sortie de Fips est un arbre contenant les informations mentionnées. Nous discuterons davantage des formats à la section 4.3.

4.2.2 Segmentation en phrases

Comme nous l'avons mentionné, les phrases sont l'unité fondamentale de notre système de création de résumés. La segmentation du texte en phrases est donc une étape initiale et essentielle de Ness. Le problème de la segmentation en phrases n'est pas trivial, mais il ne représente pas non plus une tâche insurmontable. Fips effectue lui-même une telle segmentation qui apparaît explicitement à la sortie : chaque phrase est représentée par un arbre syntaxique différent. Notons que les phrases sont à leur tour segmentées en mots distincts, ce qui ne s'avère pas être une tâche difficile en anglais - ni en français d'ailleurs. Il suffit dans la plupart des cas de se fier à la présence d'un espace ou de ponctuation.

> If Rossi continues trending upward, he could pull off a huge political upset and become Washington's first Republican governor in 20 years.

FIG. 4.2 – Phrase extraite de l'article APW_ENG_20041117.0090, du sujet D0813B-A.

4.2.3 Identification des lemmes

La sortie de FIPS inclut les lemmes de chaque mot rencontré, c'est-à-dire dans leur forme non-conjuguée, tels qu'on les retrouve dans le dictionnaire. Ces lemmes apparaissent conceptuellement comme une feuille de chaque mot qui est un élément de l'arbre syntaxique de sortie.

4.2.4 Arbre syntaxique

Pour chaque phrase à analyser, FIPS effectue une analyse syntaxique dont le résultat est produit sous la forme d'un arbre. La structure de cet arbre exprime la structure de la phrase et l'arbre inclut la fonction syntaxique de chaque mot et des syntagmes (groupes de mots). Par exemple, la phrase de la figure 4.2 produit l'arbre syntaxique illustré à la figure 4.3.

FIPS n'arrive pas à analyser toutes les phrases. Dans le cadre de notre projet, environ 72% des phrases ont été analysées entièrement par FIPS. Les phrases qui ne sont pas analysées entièrement seront éventuellement abandonnées et jugées automatiquement comme non pertinentes lors de l'évaluation des phrases (voir la section 4.5).

Notons que même si le programme indique avoir complété l'analyse d'une phrase donnée, celle-ci n'est pas toujours exacte. De nombreuses erreurs d'analyse légères surviennent et parfois même de graves erreurs. Prenons par exemple la phrase de la figures 4.2 et sa sortie par FIPS en 4.3. Notons que les deux VP *pull off* et *become* ont le même sujet *he* en commun et la même dépendance au TP *could*. La structure

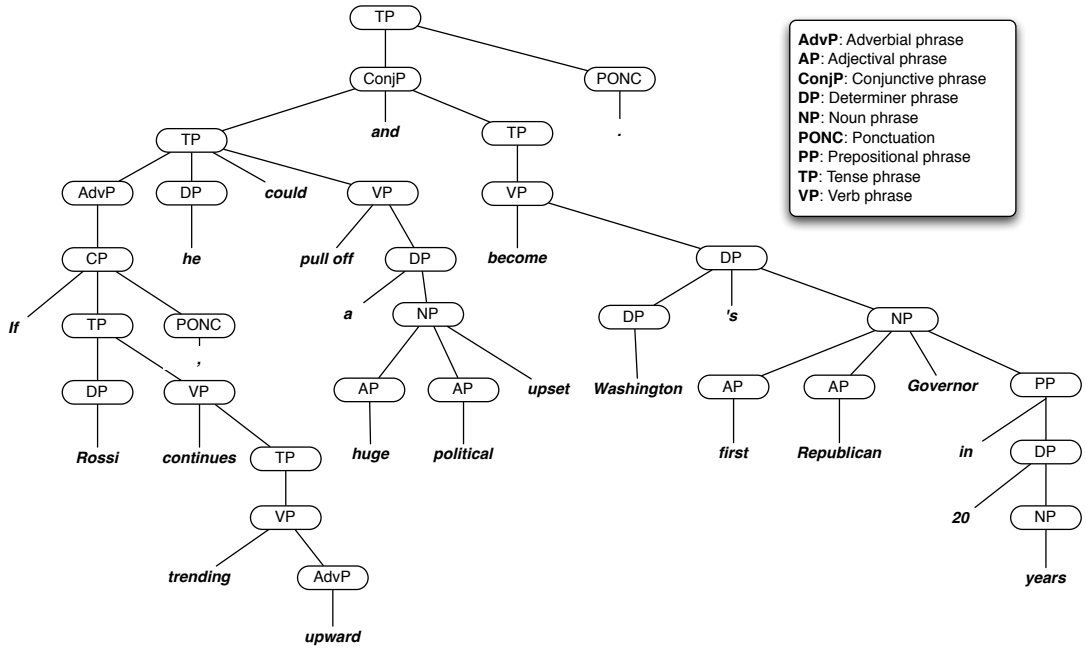

FIG. 4.3 – Arbre syntaxique produit par FIPS sur la phrase de la figure 4.2.

correcte de la phrase placerait donc la conjonction *and* comme enfant de *could* et les deux VP comme enfants de *and*. De telles erreurs sont relativement fréquentes et elles surviennent davantage dans les phrases de taille moyenne ou longue ; celles qui sont justement de bonnes candidates pour les résumés. Ces erreurs compliquent une utilisation intensive des informations syntaxiques par un système automatique. Comme nous le verrons dans la section 4.5, notre utilisation de ces informations est demeurée restreinte, principalement parce qu'il est impossible de se fier à tous les détails de chaque analyse.

4.3 Transformation en format XML

Le format utilisé par NESS pour tout le traitement des textes et pour la formation d'un résumé est le *Extensible Markup Language*(XML). Ce format permet de sauvegarder aisément de l'information sous la forme d'une structure de données

<TP>
<ConjP>
<TP>
<AdvP>
<CP>If<LEX>if</LEX>
<TP>
<DP>
<GF>SUBJ</GF>
<GENDER>mascfemneut</GENDER>
<NUMBER>sing</NUMBER>Rossi<LEX>Rossi</LEX>
</DP>
...

FIG. 4.4 – Extrait de l'analyse présentée à la figure 4.3, en format XML.

extensibles, c'est-à-dire définies par le programmeur. Le langage XSLT permet d'exécuter efficacement des transformations sur des fichiers de format XML pour créer d'autres fichiers XML ou des fichiers plus faciles à lire comme du texte ou du HTML.

Ce choix d'architecture est approprié à notre approche symbolique au résumé automatique, d'autant plus que les arbres syntaxiques produits par FIPS s'expriment facilement sous la forme d'un objet XML et que les manipulations de ces arbres s'effectuent très élégamment dans des feuilles XSLT. Tout le processus d'évaluation des phrases et de création de résumés de la section 4.5 est d'ailleurs programmé dans des feuilles XSLT ; et toutes les informations auxiliaires de la section 4.4, de même que celles provenant de FIPS, sont sauvegardées en format XML. Un exemple du format XML est donné à la figure 4.4.

4.4 Extractions d'informations auxiliaires

Certaines informations servant à l'évaluation des phrases peuvent être obtenues à l'avance pour simplifier les calculs, en particulier les mots d'expansion obtenus avec Wordnet et les valeurs idf des mots dans le corpus d'articles et de requêtes.

4.4.1 Synonymes avec Wordnet

Wordnet est un dictionnaire développé au laboratoire de sciences cognitives de l'université Princeton [MBF+90]. Il regroupe les mots partageant un concept dans des *synsets*, ou ensembles de synonymes. Chaque synset a une définition et des liens sémantiques sont tracés entre les différents synsets. Notre utilisation de Wordnet se limite cependant à la recherche de synonymes dans le but de faire de l'expansion de requête. Nous ne faisons qu'extraire des mots faisant partie du ou des synsets auxquels appartient un terme de la requête.

Les mots des requêtes sont extraits des documents XML de sortie de Fips et sont soumis à Wordnet pour obtenir des synonymes. On peut donner à Wordnet le type (nom, adjectif, verbe ou adverbe) du mot dont on cherche des synonymes en paramètre. Il s'avère que cela est justement une information que l'on peut extraire de l'arbre syntaxique produit par Fips. Cette information additionnelle améliore la qualité des mots d'expansion obtenus.

4.4.2 Poids *idf*

La fréquence de document inversée (*idf*) de chaque mot du corpus des articles et des requêtes sont nécessaires pour établir les poids $tf \cdot idf$ que nous utilisons pour effectuer les calculs de similarité, tel qu'expliqué à la section 4.5.1. La valeur d'*idf* représente la rareté d'un mot dans un corpus, soit l'inverse de sa fréquence.

La formule exacte utilisée est donnée à l'équation 4.1, où $|\mathcal{D}|$ est le nombre de documents dans le corpus et $|\{d_i : t \in d_i\}|$ est le nombre de documents possédant le terme t. Dans notre contexte, la phrase étant l'unité auprès de laquelle la recherche est effectuée, les d_i dans l'équation sont en fait les phrases des articles et requêtes.

$$\mathrm{idf}(t) = \log \frac{|\mathcal{D}|}{|\{d_i : t \in d_i\}|} \tag{4.1}$$

Notons que le poids *idf* doit préférablement être calculé avant d'effectuer les

calculs de similarité, puisque les phrases de tout le corpus sont considérées pour calculer ce poids et qu'il serait inutile de refaire ce calcul pour chaque résumé. Ainsi, effectuer un résumé ne requiert que les informations auxiliaires déjà extraites, les articles à résumer et la requête à laquelle répondre.

4.5 Évaluation de la pertinence des phrases

Puisque NESS compose des résumés par extraction de phrases, le choix des phrases est donc l'aspect fondamental de toute l'application. Dans notre approche, nous évaluons la pertinence des phrases indépendamment, en leur attribuant un score de pertinence par rapport à la requête. Toutes les étapes précédentes du système ont eu pour but d'évaluer la pertinence de chaque phrase pour un résumé, par rapport à la requête.

L'attribution d'un score global combine des scores de plusieurs critères de pertinence de phrase. Le score de chaque critère est pondéré puis additionné. Dans cette section, nous décrivons ces critères de sélection et leur normalisation.

Les critères d'évaluation de pertinence des phrases sont les suivants.

`Word_Sim` Similarité entre les mots de la phrase et les mots de la requête

`Word_Depth_Sim` Similarité entre les mots de la phrase et les mots de la requête, pondérée par la profondeur du mot dans l'arbre

`Lemma_Sim` Similarité entre les lemmes de la phrase et les lemmes de la requête

`Sent_Position` Position de la phrase dans l'article d'origine

`Sent_Weight` Poids informationel des mots de la phrase dans le corpus

`Expansion_Sim` Similarité entre les mots de la phrase et les mots de l'expansion de la requête, incluant ceux de la requête elle-même

Comme nous l'avons expliqué à la section 2.2, les résumés de la partie B de chaque sujet sont des résumés de mise à jour. Nous avons eu recours à deux critères

spécifiques à la tâche de mise à jour. Les scores de ces critères sont soustraits plutôt qu'additionnés pour obtenir le score global.

`Cluster_Sim` Similarité entre les mots de la phrase et l'ensemble des mots des articles de la partie A

`Top_Sents_Sim` Maximum de la similarité entre les mots de la phrase et les mots des phrases les plus pertinentes de la partie A selon NESS

4.5.1 Calculs de similarités

Plusieurs critères de pertinence des phrases reposent sur un calcul de similarité entre les mots ou les lemmes de deux groupes de termes, généralement des phrases. Ce calcul de similarité s'effectue toujours de la même façon par NESS.

Nous calculons la similarité entre deux groupes de termes à l'aide de la méthode du cosinus dans le modéle vectoriel. La méthode d'attribution de poids est par *tf·idf*.

La fréquence de terme (tf) est le nombre d'occurrences du terme dans le groupe de termes qui le contient, normalisé par le nombre total de mots dans ce document. En multipliant la valeur tf par la valeur idf défini en 4.4.2, on obtient un poids $tf \cdot idf$ calculable pour chaque mot d'un document.

On considère ensuite un espace vectoriel de dimension égale au nombre de termes dans le corpus. Chaque groupe de termes peut être représenté par un vecteur dont la valeur est représenté par les poids *tf·idf* des termes qu'il contient. Deux vecteurs dans cet espace peuvent être comparés pour en déterminer la similarité à l'aide d'un calcul cartésien du cosinus de l'angle qui les sépare. La similarité entre deux vecteurs v_1 et v_2 est donc donnée par l'équation 4.2. Cette définition implique que la similarité est de 0 lorsqu'aucun terme n'est commun aux deux groupes et qu'elle est de 1 lorsque tous les termes apparaissent le même nombre de fois dans chaque groupe.

$$\mathrm{SIM}(v_1, v_2) = \cos\theta_{v_1 v_2} = \frac{\vec{v_1} \cdot \vec{v_2}}{\|\vec{v_1}\|\|\vec{v_2}\|} \tag{4.2}$$

Notons que tous les termes ne sont pas considérés dans ce calcul. En particulier, les mots les plus fréquents de la langue anglaise, que l'on appelle stopwords dans le domaine, sont automatiquement ignorés car il est connu qu'ils ne fournissent que peu ou pas de pouvoir discriminant entre différents textes.

Nous avons constaté que les verbes apparaissant dans les requêtes sont presque tous sous forme impérative, et, en conséquence, qu'ils n'offrent en général pas d'intérêt pour la sélection de phrases. Les verbes à l'impératif ne portent pas un sens relié au besoin d'information décrit dans la requête et ne doivent donc pas être utilisés pour représenter le contenu de la requête. Favoriser une phrase contenant le mot *follow*, tiré de l'exemple de la figure 2.1, ne serait pas judicieux puisque la phrase pourrait tout autant être pertinente ou non pertinente à la requête, qu'elle contienne ce mot ou non. Grâce à FIPS, nous pouvons aisément déterminer si un mot a la fonction de verbe ou non, et ignorer tous les verbes lors de la construction des vecteurs servant aux calculs de similarité.

Les critères `Word_Sim` , `Lemma_Sim` , `Word_Depth_Sim` et `Expansion_Sim` sont déterminés grâce au calcul de similarité défini plus haut, de même que les critères de mise à jour `Cluster_Sim` et `Top_Sents_Sim` . Les trois premiers critères sont évalués en utilisant v_1 le vecteur de tous les mots ou lemmes de la requête et v_2 le vecteur des mots ou lemmes de la phrase à évaluer. L'application d'un facteur additionnel pour `Word_Depth_Sim` est expliqué dans la prochaine section. Le critère `Expansion_Sim` utilise à son tour un vecteur de tous les mots de la phrase évaluée comme v_2, mais son vecteur v_1 inclut non seulement les mots de la requête, mais aussi tous les mots obtenus par l'expansion de requête, c'est-à-dire jusqu'à 10 mots supplémentaires par nom apparaissant dans la requête. En effet, dans la version finale de NESS, il a été choisi d'ignorer l'expansion de requête sur les mots de type

autre que les noms. Les résultats de notre méthode utilisant Wordnet sur les verbes, les adjectifs et les adverbes ne semblaient que rarement être appropriés.

Le critère `Cluster_Sim` utilise le vecteur composé de tous les mots contenus dans tous les articles de la partie A comme v_1 et le même v_2 que précédemment.

Enfin, le critère `Top_Sents_Sim` , lui, est calculé en prenant la valeur maximale de plusieurs calculs de similarité effectués entre les mots de la phrases à évaluer et les mots des phrases les mieux évalués par notre système dans la partie A. Ce critère désavantage donc grandement les phrases considérées très similaire à l'une des phrases jugées comme pertinentes pour décrire la partie A, car on suppose qu'il y a des répétitions.

4.5.2 Profondeur d'arbre et expansion de requête

La profondeur d'arbre mesure l'importance d'un mot dans une phrase. Ce qu'on entend par profondeur d'arbre est le niveau d'arborescence occupé par le nœud dans l'arbre syntaxique produit par FIPS. Ainsi, dans l'exemple de la figure 4.3, *and* a une profondeur de 2, alors que *upward* a une profondeur de 10.

Un mot servant à qualifier ou donner du contexte à un autre mot ou à un groupe de mots peut être défini comme étant subordonné à ceux-ci. Plus un mot est subordonné à d'autres mots dans une phrase et plus sa profondeur d'arbre sera élevée. On suppose qu'un mot très subordonné (c'est-à-dire qu'une chaîne de relation relativement longue le relie à la tête de la phrase, soit le verbe de la proposition principale) joue un rôle moins important dans la phrase et qu'il devrait donc être considéré de façon moins importante dans le calcul de similarité. Le critère `Word_Depth_Sim` prend en compte cette heuristique en divisant simplement le poids de chaque mot par sa prodondeur d'arbre avant d'effectuer le calcul de similarité.

4.5.3 Poids informationnel de la phrase

Comme nous l'avons mentionné dans la section 4.4.2, le poids *idf* d'un mot correspond à sa rareté dans le corpus. On peut supposer intuitivement que les mots rares apportent des informations nouvelles, alors que les mots fréquents répètent de l'information jusqu'à un certain point. La quantité de nouveauté contenu dans une phrase peut donc être approximée par les poids *idf* des mots qui la composent.

Nous estimons le poids informationnel d'une phrase par la somme de tous les *idf* des mots qu'elle contient. Le critère `Sent_Weight` , que nous avons justement défini comme représentant le poids informationnel d'une phrase, est calculé de cette façon.

4.5.4 Position de la phrase dans l'article

Il est admis que la position d'une phrase dans un texte permet d'estimer l'importance de son contenu. Les phrases de l'introduction d'un texte peuvent par exemple contenir des informations de synthèse quant aux sujets qui seront discutés. On peut faire cette supposition pour des articles de journaux, qui possèdent souvent une structure où les éléments les plus importants et les phrases contenant le plus d'information sont situées au tout début de l'article. Plus on avance dans la lecture de l'article et plus on rentre dans les détails de la nouvelle racontée, alors que les premières phrases ont tendance à être plus synthétiques et informatives. On observe en particulier cette hiérarchie des phrases dans les textes du corpus fourni, possiblement parce qu'ils proviennent d'agences de presse.

Le critère `Sent_Position` cherche à tirer profit de cette information. Il attribue aux phrases une valeur égale à l'inverse du rang de la phrase dans l'article d'où il provient. La première phrase prend une valeur de 1 alors que la 4^{e} a une valeur de 0.25 pour ce critère.

4.5.5 Normalisation des scores des critères de pertinence

Pour avoir une meilleure compréhension intuitive de la pondération des différents critères mentionnés, il est avantageux de normaliser chacun des critères d'évaluation. Les valeurs du critère `Sent_Weight` peuvent aisément atteindre 100 et plus alors que celles du critère `Word_Depth_Sim` ne dépassent pas 0,05. Si l'on désirait que ces deux critères interviennent à part égale dans le calcul du score global d'une phrase, il faudrait que le coefficient de pondération du premier soit environ 2000 fois plus élevé que celui du deuxième, plutôt que de partager le même coefficient. Ceci est évidemment contre-intuitif et nuisible à l'analyse des résultats finaux.

La normalisation des scores des critères de pertinence s'avère donc être un outil utile à la compréhension humaine du fonctionnement du système quoiqu'elle ne change en rien le traitement et les calculs. La normalisation divise simplement le score de toutes les phrases, pour chaque critère donné, par la valeur du score maximal de ce critère parmi toutes les phrases. Ainsi, tous les critères ont des scores dans le même intervalle de 0 à 1, avec toujours au moins une occurrence de la valeur 1. Ceci facilite le calibrage des critères que nous discuterons à la section 4.8.

4.6 Édition et rejet de phrases

Nous apportons quelques modifications assez simples aux phrases dans le but de sauver quelques mots. Si notre résumé contient des phrases plus courtes, alors il arrive qu'à l'occasion nous puissions inclure davantage de phrases avant d'atteindre la limite de 100 mots que nos résumés doivent contenir. Un résumé qui contient une phrase supplémentaire permet de couvrir davantage d'idées distinctes, en assumant qu'il n'y ait pas de redondance importante.

Les expressions parenthésées, n'apportant généralement pas d'informations nouvelles, sont retirées entièrement des phrases pour l'évaluation et lors de l'inclusion dans le résumé. Aussi, les expressions vagues en référence à une citation sont en-

levées ; ce sont des expressions telles que “, he said”. Enfin, nous ajustons les ponctuations dans le cas où cela causerait un problème de lisibilité. Entre autres, un point terminal est rajouté aux phrases qui n'en ont pas et les guillemets orphelins sont également retirés. Au total, une vingtaine de rêgles d'édition de ce type sont utilisées.

Il nous a semblé que l'évaluation des phrases courtes pouvait être fautive et que les phrases trop courtes pouraient être incluses injustement à cause de la méthode de sélection des phrases de la section suivante. De plus, sans retirer les phrases trop courtes, nous avons observé l'apparition presque systématiques de phrases de seulement deux à quatre mots qui n'avaient pas leur place dans un résumé. Ainsi, les phrases possédant moins de 5 mots sont simplement ignorées lors de la sélection.

4.7 Sélection des phrases et création du résumé

L'évaluation des phrase se calcule par une somme pondérée des scores de chaque critère défini plus haut. La sélection des phrases s'effectue en choisissant les phrases obtenant les scores les plus élevés, tout en respectant la limite imposée de 100 mots par résumé.

L'algorithme que nous utilisons sélectionne les phrases avec les meilleurs scores jusqu'à ce qu'il ne soit plus possible d'en ajouter sans dépasser la limite de mots. Il choisit ensuite la phrase possédant le meilleur score parmi celles qui contiennent moins de mots que l'espace restant sous la limite permise de 100 mots.

Lors de la sélection de phrases pour le premier résumé, nous sauvegardons les phrases qui formeraient le meilleur résumé de 400 mots selon notre système. Ces phrases sont ensuite utilisées lors de l'évaluation de la partie B pour établir le critère `Top_Sents_Sim` , comme nous l'avons déjà mentionné.

4.8 Calibration des paramètres

NESS emploie 8 critères pour l'évaluation des phrases (voir la section 4.5) et chaque critère produit un score qui est ensuite pondéré et addionné aux autres pour donner un score final à chaque phrase des documents d'entrée (voir la section 4.7). Les paramètres les plus importants dans NESS sont ces pondérations associées à chaque critère. Pour des raisons pratiques, nous avons limité les valeurs que peuvent prendre les paramètres à l'intervalle $[0, 1]$.

Nous avons effectué une calibration de ces paramètres pour optimiser la qualité des résumés produits et nous décrivons ici cette démarche. L'évaluateur automatique ROUGE ainsi que des observations manuelles ont guidé la calibration.

4.8.1 ROUGE

Recall-Oriented Understudy for Gisting Evaluation (ROUGE) est une application développée par Chin-Yew Lin [Lin04] qui sert à évaluer des résumés automatiques dans le but d'aider la recherche dans le domaine des résumés automatiques.

Pour évaluer un résumé, ROUGE compare celui-ci à des exemples de résumés écrits par des êtres humains à partir des mêmes données et servant de référence. Plus le résumé automatique est semblable aux résumés humains et plus le score obtenu sera élevé. La proximité lexicale est le critère déterminant pour l'évaluation de la proximité du contenu de cet évaluateur. Dans ROUGE, la proximité lexicale peut être évaluée de différentes façons, qui produisent chacune un score distinct. Elles incluent de compter : le nombre de termes communs entre le résumé à évaluer et les résumés exemples, le nombre de n-grammes (groupes de n termes) communs et le nombre de n-grammes espacés (groupe de termes non-adjacents mais contenus dans une fenêtre d'un nombre donné de termes du texte) communs. Chaque type de proximité lexicale correspond à un score de ROUGE différent. Nous avons utilisé principalement les brigrammes (ROUGE-2) et les bigrammes espacés par de 0 à 4 termes entre eux (ROUGE-SU4) pour calibrer le système Ness.

Des exemples de résumés sont nécessaires en tant que référence pour l'évaluation automatique de résumés par ROUGE. Pour les résumés de mise à jour, le seul ensemble de test comportant des exemples de résumés écrits par des êtres humains disponible était la collection de clusters et de requêtes de la conférence DUC 2007 [ITL07]. Une compétition de résumés automatiques avec mise à jour très similaire à celle de TAC 2008 avait alors eu lieu en tant que tâche pilote de cette conférence. Le nombre de sujets différents (chacun représenté par une requête et accompagné

de 20 documents) n'était cependant que de 10. Les données et résultats (contenant les résumés humains) de la tâche de mise à jour de DUC 2007 nous ont été fournis et nous avons pu les utiliser afin de calibrer notre système avec ROUGE.

4.8.2 Exploration de l'espace des paramètres et ses contraintes

Les 8 paramètres principaux de NESS (la pondération de chaque critère de sélection des phrases) forment un espace à 8 dimensions. Chaque point dans l'espace correspond à une configuration du système, soit un ensemble de valeurs pour les pondérations de chaque critère. Nous pouvons exécuter le programme et utiliser ROUGE pour faire une évaluation des résumés créés sur l'ensemble de test décrit dans la section précédente. Une exploration exhaustive de l'espace des paramètres permettrait d'obtenir un point optimal où les scores donnés par ROUGE seraient près d'un maximum global.

Chaque test requiert cependant l'exécution de NESS sur tous les sujets, ce qui impose une contrainte de temps sur notre exploration. Nous avons développé une façon de conserver les résultats partiels de l'exécution de notre système pour accélérer les exécutions subséquentes sur les mêmes documents d'entrée. Les scores de chaque critère sont conservés et seulement les étapes de calcul du score global de chaque phrase ainsi que la sélection des phrases sont effectuées à nouveau.

La contrainte de temps demeure tout de même importante lorsque l'on considère la taille de l'espace à explorer. Le nombre d'exécutions étant proportionnel au nombre de valeurs différentes à échantillonner sur l'intervalle $[0, 1]$ à la puissance 8, il est inconcevable de bien couvrir l'espace à explorer. De plus, les données d'entraînement (l'ensemble de test présenté plus haut) sont peu nombreuses et le risque de sur-entraînement est très présent. Enfin, des observations préliminaires ont démontré que les valeurs des scores ROUGE étaient très non-linéaires dans l'espace des paramètres lors de l'exécution de notre système, ce qui éliminait la possibilité d'utiliser efficacement les gradients pour trouver des optimums et qui

augmentait les risques du sur-entraînement.

4.8.3 Heuristique

Pour répondre aux problèmes posés par la contrainte de temps (et donc l'incapicité de bien couvrir l'espace des paramètres), le risque de sur-entraînement et la non-linéarité de l'espace, nous avons employé une heuristique.

Nous avons choisi d'imposer que les valeurs relatives des pondérations aux critères communs aux résumés standard et de mise à jour soient identiques (ceci permet donc de calibrer 8 paramètres, tel qu'indiqué, et non 16). Ceci a aussi pour effet de diminuer le sur-entraînement et de compenser pour la faible taille de l'ensemble de test. Il s'agit également d'un choix consistant au niveau du traitement du langage - en excluant les particularités pour la tâche de mise à jour, les phrases sont évaluées de la même façon selon qu'elles se retrouvent dans la partie A ou B. Nous pouvons observer que ce choix a été respecté dans les valeurs retenues présentées dans le tableau 4.1.

Notre recherche d'un optimum intéressant des valeurs relatives des pondérations a suivi une méthodologie pouvant se résumer dans l'algorithme itératif suivant, où n est le nombre de paramètres testés à une étape donnée. Il s'agit d'un algorithme de recherche avare (*greedy search*).

1. Initialisation : On observe le score obtenu en n'utilisant que deux paramètres ($n = 2$) pour l'évaluation. Toutes les paires de paramètres sont testées, dans un certain nombre de proportions entre les deux pondérations de chaque paire. Les quelques meilleures paires et la proportion obtimale de cette paire sont conservées pour l'étape suivante.

2. Boucle : à partir des quelques combinaisons de $n - 1$ paramètres qui sont les plus performantes à l'étape précédente, nous ajoutons, pour différentes valeurs, chacun des paramètres restants à inclure pour une combinaison donnée.

De tous ces tests, nous gardons les quelques meilleures combinaisons de n paramètres qui procurent les meilleures évaluations.

3. Terminaison : Lorsque les meilleures combinaisons finales de 8 paramètres sont obtenues, nous effectuons de petites variations des valeurs de chaque paramètre pour explorer l'espace autour de ces solutions finales. Ceci nous a mené à plusieurs candidats pour la calibration.

L'algorithme présenté permet de limiter les effets de non-linéarité tout en restreignant grandement la taille du nombre de tests qui doivent être effectués pour couvrir l'espace.

Les scores ROUGE obtenus sur les différentes combinaisons de valeurs des paramètres étaient souvent rapprochés malgré des distances parfois importantes dans l'espace des paramètres. Ceci indique que des corrélations entre les paramètres pouvaient exister et également que plusieurs maximums locaux existaient probablement étant donné l'ensemble de test. Une observation manuelle des différences dans les résumés produits a permis de faire un choix final. Nous avons également choisi de garder les proportions entre les paramètres à des entiers faibles pour éviter le sur-entraînement et pour que le résultat soit plus élégant. Notons que les valeurs des paramètres peuvent être comparées entre elles de façon relativement intuitives grâce à l'étape de normalisation effectuée, comme nous en avons discuté à la section 4.5.5. Enfin, la somme des valeurs finales présentées est ramenée à 1.0 pour fin de lisibilité. Les valeurs qui ont été retenues sont présentées dans le tableau 4.1.

4.9 Soumissions à TAC

Comme nous l'avons mentionné au chapitre 2, trois soumissions pouvaient être envoyées à la compétition de TAC 2008. Seules les deux premières (Run 1 et Run 2) ont été évaluées manuellement, ceci menant à des résultats très distincts, comme nous en discuterons au chapitre 5. Nous décrivons ici les caractéristiques des ver-

critère	**Run 1**		**Run 2**		**Run 3**	
	A	B	A	B	A	B
`Word_Sim`	.00	.00	.10	.09	.00	.00
`Word_Depth_Sim`	.10	.09	.00	.00	.10	.10
`Lemma_Sim`	.30	.26	.45	.39	.30	.30
`Sent_Position`	.25	.22	.25	.22	.25	.25
`Sent_Weight`	.20	.18	.20	.18	.20	.20
`Expansion_Sim`	.15	.13	.00	.00	.15	.15
`Cluster_Sim`	.00	.04	.00	.04	.00	.00
`Top_Sents_Sim`	.00	.08	.00	.08	.00	.00

TAB. 4.1 – Pondération de chaque critère d'évaluation des phrases pour les résumés standards (A) et de mise à jour (B), pour les trois soumissions à TAC.

sions de NESS utilisées pour la compétition. La différence principale entre les versions se situe au niveau des valeurs de pondérations pour chaque critère. Les valeurs retenues pour chaque soumission sont présentées dans le tableau 4.1.

4.9.1 Run 1

La première soumission, ou Run 1, représente la version que nous croyions être la meilleure, qui est décrite dans ce chapitre.

4.9.2 Run 2

La version de NESS pour la deuxième soumission à TAC est une version "faible en ressources". Nous avons cherché principalement à vérifier la performance de notre système lorsque celui-ci fait abstraction des analyses syntaxiques de FIPS, mais la ressource WordNet est également ignorée. Nous avons tout de même utilisé FIPS dans cette version, mais uniquement pour la segmentation des phrases et la lemmatisation ; toute autre information a été ignorée. Notons que ceci empêche le retrait des verbes décrit à la section 4.5.1. Cependant, toutes les phrases qui ne sont pas analysables entièrement par FIPS (et qui sont ignorées dans les autres versions

de Ness) sont à nouveau incluses comme candidates potentielles pour les résumés à créer, dans cette version-ci.

4.9.3 Run 3

La troisième soumission est simplement un test pour voir si notre traitement de la mise à jour joue un rôle important sur les résultats obtenus. Les paramètres sont les mêmes pour la partie A et B et les résumés sont donc créés de la même façon dans les deux cas, sans que la création des résumés en B n'utilise les documents de la partie A. Les résumés de la partie A pour les soumissions 1 et 3 sont donc identiques.

CHAPITRE 5

RÉSULTATS ET ÉVALUATION

5.1 Exemples de résumés créés

Un exemple de requête et des deux résumés produits par la Run 1 de NESS, soumise à TAC 2008, sont donnés à la figure 5.1.

5.2 Méthodologie d'évaluation

La méthodologie d'évaluation que nous utilisons est celle utilisée dans le cadre de la *Text Analysis Conference* (TAC) 2008, décrite au chapitre 2. Des experts évaluateurs du *National Institute of Standards and Technology* (NIST) ont manuellement évalué nos deux premières soumissions à la compétition. Les mesures d'évaluation que nous décrivons ici permettent de comparer entre elles les méthodes utilisés pour créer automatiquement des résumés.

Pour certaines mesures d'évaluation, des résumés de référence, écrits par des êtres humains, sont nécessaires. Huit employés de NIST se sont réparti la rédaction de résumés standards et de mise à jour pour les 48 sujets. Pour chacune des 96 tâches, quatre résumés de référence ont été écrits par quatre personnes différentes. Les évaluations manuelles ont été conduites sur 57 soumissions automatiques (incluant nos soumissions 1 et 2) et les évaluations automatiques ont été réalisées sur toutes les 71 soumissions (incluant notre troisième soumission). Les résumés écrits par des humains ont également été évalués de la même façon pour fins de comparaison.

Requête	Washington Governor Race. Follow developments concerning the 2004 election for Governor of Washington.
Résumé A	Two weeks after the election, Washington state still doesn't have a new governor. The closest governor's race in Washington history was forced into a recount as counties finished tallying the ballots and found only a few votes separating the candidates out of 2.8 million cast. Fifty-eight days after the election, Christine Gregoire was declared the governor-elect of Washington on Thursday, the official winner by an infinitesimal margin after two recounts in a historically close race. As political junkies across the country have recovered from their presidential election withdrawal, they have turned to the Washington governor's race for entertainment.
Résumé B	Rejecting claims that fraud and illegal votes wrongly put a Democrat in the governor's office, a judge in this conservative area of eastern Washington dismissed a Republican lawsuit on Monday that had sought to overturn the election of Gov. Christine Gregoire. The battle over Washington's contested governor's election touches on many of the questions that divide this country between rural and urban, Republican and Democrat, red and blue – and echoes frustrations of the past two presidential elections. She's now the governor of Washington state and we're ready to work with her. Vance, for example, calls Gregoire "our fake governor".

FIG. 5.1 – Résumé standard (A) et de mise à jour (B) soumis en réponse à la requête D0813B de la collection de TAC 2008.

5.2.1 Qualité globale

La qualité globale (*overall responsiveness*) d'un résumé est une mesure de la quantité d'information contenue dans le résumé qui permet de répondre à la requête, tout en tenant compte de la qualité de la langue utilisée pour transmettre cette information et de la mesure dans laquelle on a répondu à la requête. Ceci représente le degré de satisfaction global d'un utilisateur. Cette mesure est évaluée manuellement par des évaluateurs sur une échelle de 1 (très mauvais) à 5 (très bon). Cette méthode d'évaluation représente également le niveau d'appréciation des utilisateurs pour les résumés qui leur sont offerts.

5.2.2 Niveau linguistique

Le niveau linguistique du résumé est une mesure combinant cinq évaluations linguistiques : la grammaticalité, la non-redondance, la clarté référencielle, le focus et la cohésion du texte. Sur la base de ces critères, des évaluateurs ont attribué à chaque résumé une note allant de 1 (très mauvais) à 5 (très bon).

5.2.3 Score pyramidal

La méthode pyramidale d'évaluation des résumés [Pas06] est une comparaison du contenu du résumé automatique par rapport au contenu des résumés humains. Les évaluateurs annotent, dans les quatre résumés de référence ainsi que dans tous les autres résumés, les unités d'information saillante (*Summary Content Units*). La méthode pyramidale évalue le degré de superposition de ces unités sémantiques entre le résumé à évaluer et les résumés de référence. Ceci équivaut au niveau de proximité du contenu sémantique des résumés automatiques avec les résumés humains. Le score correspond à la proportion moyenne d'unités d'informations contenues dans les résumés par rapport à ceux des résumés de référence.

	Run 1	**Run 2**	**Run 3**	**Réf.**	**Max.**
A - Qualité globale	2.542	2.792	-	4.620	2.792
A - Niveau linguistique	2.938	3.000	-	4.786	3.000
A - Score pyramidal	0.277	0.308	-	0.664	0.362
A - ROUGE-2	0.082	0.091	0.082	0.118	0.111
B - Qualité globale	2.500	2.458	-	4.625	2.604
B - Niveau linguistique	2.833	2.792	-	4.797	3.208
B - Score pyramidal	0.284	0.276	-	0.630	0.344
B - ROUGE-2	0.082	0.078	0.080	0.117	0.101

TAB. 5.1 – Scores obtenus par les trois versions de NESS pour les quatre méthodes d'évaluation, sur les résumés standard (A) et de mise à jour (B).

5.2.4 ROUGE

Le programme d'évaluation de résumés automatiques ROUGE a été décrit à la section 4.8. La valeur la plus utilisée est celle de la fréquence de bigrammes identiques à ceux des résumés humains, nommée ROUGE-2. Le score correspond à la proportion moyenne de bigrammes des résumés automatiques qui sont communs aux bigrammes contenus dans les résumés de référence.

5.3 Résultats de l'évaluation et discussion

5.3.1 Résultats bruts

Les scores obtenus par les trois versions de NESS qui ont été soumises sont présentés au tableau 5.1. La quatrième colonne à partir de la gauche indique la moyenne optenue dans chaque catégorie pour les résumés de référence écrits par des humains. La colonne de droite contient la valeur maximale atteinte pour cette catégorie parmi toutes les soumissions de résultats de systèmes automatiques lors de la compétition.

On observe globalement que nos scores sont près de ceux des meilleurs systèmes automatiques dans chaque catégorie, mais que nous nous situons très loin du ni-

veau atteint par les résumés composés par des êtres humains. Nos scores de qualité globale et de niveau linguistique se situent aux alentours ou en dessous de 3, ce qui est grandement inférieur aux résultats des résumés humains, qui obtiennent systématiquement des scores moyens au dessus de 4.5. Ces importantes différences sont une démonstration, parmi de nombreuses autres, de l'ampleur du travail qui reste à accomplir avant d'atteindre une lisibilité et un niveau d'information qui se rapprochent de ceux d'un résumé écrit manuellement. Au niveau du score pyramidal, on observe encore une fois un écart très significatif : nos résumés contiennent moins du tiers des unités de contenu, alors que les résumés humains en contiennent environ les deux tiers. Les scores automatiques de ROUGE-2 ne possèdent pas un écart aussi significatif, mais les trois scores manuels sont nettement plus exacts et appropriés pour donner une bonne évaluation de la qualité des résumés produits.

Il existe également un écart entre nos différentes soumissions, correspondant aux versions de Ness décrites à la section 4.9. Cet écart est plus petit, mais puisqu'il provient d'une moyenne sur 48 évaluations par partie, il n'est pas à négliger. Nous observons que la Run 2 obtient des scores nettement supérieurs à ceux de la Run 1 dans la partie A, alors que l'inverse est vrai dans la partie B, et ce pour toutes les méthodes. Les deux principales différences entre les deux versions du système peuvent expliquer cette différence. La Run 2 ne profitait pas de l'utilisation de ressources linguistiques additionnelles, en particulier les analyses syntaxiques de Fips, ce qui limitait le traitement de l'information et donc la détection de pertinence des phrases. En revanche, nous avons retranché les phrases ne pouvant être analysées par Fips des phrases à considérer pour les résumés dans la Run 1, celles-ci représentant 28% des phrases présentes en moyenne. Retrancher autant de phrases nuit à la qualité des résumés créés puisque celles-ci peuvent fort bien inclure une phrase pertinente à inclure dans un résumé, qu'il soit de mise à jour ou non. Pour la tâche fort complexe qu'est la création de résumés de mise à jour, il est probable que l'utilisation de données additionnelles aide à mieux cerner le problème,

	Run 1	**Run 2**	**Run 3**	#
A - Qualité globale	15	1	-	57
A - Niveau linguistique	2	1	-	57
A - Score pyramidal	27	15	-	57
A - ROUGE-2	33	16	33	71
B - Qualité globale	6	7	-	57
B - Niveau linguistique	8	11	-	57
B - Score pyramidal	8	10	-	57
B - ROUGE-2	17	24	19	71

TAB. 5.2 – Rangs obtenus par les trois versions de NESS pour les quatre méthodes d'évaluation, sur les résumés standard (A) et de mise à jour (B).

au point de compenser pour un accès plus limité aux phrases présentes dans les documents d'entrée. En revanche, pour les résumés standards, pouvoir considérer un plus grand nombre de phrases, au prix de moins bien les évaluer entre elles, semble, en se basant sur les résultats obtenus, être préférable.

La Run 1 et la Run 3 de notre système avait comme unique différence que la Run 3 n'effectuait aucune mise à jour. On note tel qu'attendu une différence, quoique faible, sur leur score de ROUGE-2, la seule méthode d'évaluation disponible pour la Run 3, qui n'a pas été évaluée manuellement.

5.3.2 Résultats relatifs

Le tableau 5.2 donne les rangs auxquels notre système s'est classé lors de la compétition pour chaque méthode d'évaluation. La colonne de droite de ce tableau indique le nombre de runs soumises qui ont été évaluées et parmi lesquelles le rang est calculé.

En comparaison avec les autres approches proposées lors de la compétition, nous observons que la Run 2 se retrouve au tout premier rang, sur 57 soumissions reçues au total, pour la qualité globale ainsi que pour le niveau linguistique dans la partie A. La Run 1 obtient aussi le deuxième rang du niveau linguistique dans

la partie A, quoique, pour des raisons énoncées précédemment, ses rangs sont plus faibles que ceux de la Run 2 dans cette partie. Nos scores dans la partie B (de mise à jour) pour nos deux premières soumissions se situent toujours parmi les dix premiers dans les évaluations manuelles. La Run 1 a mieux fait dans la partie B, où elle arrive au sixième rang pour la qualité globale et au huitième rang pour le niveau linguistique et le score pyramidal.

Dans l'ensemble, nos rangs pour les scores de ROUGE-2 sont faibles en comparaison avec nos scores dans les trois méthodes d'évaluation manuelles. Une raison qui pourrait expliquer cet écart est la façon dont de nombreuses équipes ont calibré leur programme de création de résumés. Plusieurs d'entre elles utilisent de l'apprentissage machine basé sur les scores ROUGE obtenus sur des données passées ou extérieures pour ajuster les paramètres de leur système. Or, comme nous l'avons observé nous-même, le sur-entraînement sur les scores de ROUGE est un risque important. Une approche qui n'inclut pas de vérification manuelle des résumés créés par le système risque de créer un système avec de très bons résultats de ROUGE au prix de résumés moins lisibles et moins informatifs.

Nous pouvons conclure, en nous basant sur les rangs que nous avons obtenus, que notre approche est compétitive dans la communauté scientifique. Nos excellents scores dans la partie A pour la Run 2, quoique cette partie puisse être considérée d'un enjeu moindre, démontrent la puissance de l'approche de base que nous utilisons, soit une approche statistique sans connaissances linguistiques profondes. Néanmoins, notre approche symbolique complète, soumise dans la Run 1, démontre le bien-fondé d'utiliser de telles connaissances lorsque la tâche devient plus complexe.

5.3.3 Comparaison avec les meilleures approches concurrentes

La table 5.3 contient à nouveau les rangs de nos deux premières soumissions, ainsi que ceux des trois autre programmes dont l'approche a été décrite à la section

	Run 1	**Run 2**	**Gillick**	**Chen 1**	**Chen 2**	#
A - Qualité globale	15	1	28	4	4	57
A - Niveau linguistique	2	1	47	7	3	57
A - Score pyramidal	27	15	10	11	8	57
A - ROUGE-2	33	16	1	15	12	71
B - Qualité globale	6	7	23	2	2	57
B - Niveau linguistique	8	11	44	2	1	57
B - Score pyramidal	8	10	19	4	9	57
B - ROUGE-2	17	24	3	8	5	71

TAB. 5.3 – Rangs obtenus lors de la compétition par les deux premières versions de NESS, ainsi que par la soumission de Gillick et al. et les deux de Chen et al.

3.2, soient les deux approches de Chen et al. de l'Université Tsinghua (voir sections 3.2.1 et 3.2.2) et l'approche de Gillick et al. de l'*International Computer Science Institute* (voir section 3.2.3).

C'est sans surprise que la soumission de Gillick et al. obtient les meilleurs résultats pour le score ROUGE-2 dans toute la compétition, étant donné que leur approche repose directement sur l'optimisation du nombre de bigrammes distincts présents dans leurs résumés, alors que ROUGE-2 évalue le nombre de bigrammes partagés par les résumés automatiques et les résumés de référence. Ils obtiennent cependant des résultats grandement inférieurs dans toutes les autres catégories dans les deux parties. Leur soumission a même obtenu de meilleures évaluations, en moyenne, que 3 des 8 résumeurs humains dans la partie A, pour ce qui est du score ROUGE-2 spécifiquement. Ceci permet entre autres de soulever explicitement les lacunes déjà mentionnées de l'évaluation automatique ROUGE par rapport aux méthodes d'évaluation manuelles.

Les deux soumissions de Chen et al., l'approche par distance d'information et l'approche par centralité des phrases, sont les deux plus performantes, dans l'ensemble de la compétition, obtenant des rangs parmi les tous premiers dans chacune des méthodes d'évaluation et arrivant notamment ex-æquo au deuxième

rang pour le score de qualité globale dans la partie B et au quatrième rang pour ce même score dans la partie A (alors que les deux premiers rangs sont occupés par nos propres soumissions), tout en arrivant en tête pour le niveau linguistique.

Comme nous l'avons vu au chapitre 3, leurs approches semblent pourtant bien différentes au premier coup d'œil. Trouver leurs ressemblances permettrait peut-être d'élucider ce que cette équipe accomplit de mieux que les autres et en particulier mieux que nous. Nous notons que les deux effectuent deux évaluations subséquentes, une première qui évalue la pertinence des phrases et une seconde qui évalue un ensemble de résumés potentiels créé en calculant toutes les permutations des 15 phrases possédant le meilleur score. Au niveau des critères d'évaluation des phrases, quoique la démarche soit très différente, ils utilisent dans les deux approches une évaluation de chaque mot basé sur sa fréquence-document (le nombre de documents contenant un mot parmi les dix documents d'entrée). On peut conclure que ces deux similarités expliquent en partie la performance des deux systèmes pourtant généralement dissemblables. En particulier, peu d'équipes ont effectué une évaluation de plusieurs résumés candidats et cette idée innovatrice semble être un bon moyen de maximiser la quantité d'information contenue dans le résumé et d'éviter la redondance (puisque les résumés avec des phrases redondantes seront idéalement évalués désavantageusement par rapport aux résumés peu redondants). On peut retenir de leurs techniques une attention particulière à combattre la redondance dans les résumés, ce qui n'a que des effets positifs à la fois au niveau linguistique et sur le contenu informationnel des résumés créés.

CHAPITRE 6

TRAVAUX FUTURS ET CONCLUSION

6.1 Travaux futurs

Suite à notre participation à la compétition de TAC 2008, nous pouvons, avec un certain recul, identifier plusieurs éléments à améliorer dans notre méthodologie, qui pourraient être inclus dans des travaux futurs. Nous avons choisi les éléments soulevés dans cette section en étudiant les autres approches présentées et en nous basant sur les discussions que nous avons eu la chance d'avoir avec les autres participants lors de la conférence.

D'abord, plusieurs détails de notre programme pourraient être raffinés. Un plus grand nombre de tests pourraient être effectués pour la calibration dans le but de mieux couvrir le grand espace de paramètres à explorer. Nous pourrions également utiliser une plus grande collection de tests pour calibrer le système, d'autant plus que les données de la compétition de cette année sont maintenant disponibles. Des évaluateurs automatiques de résumés autres que ROUGE sont aujourd'hui disponibles et les scores de ces alternatives à ROUGE pourraient être également utilisés. Ainsi, il serait possible d'effectuer une exploration plus détaillée de l'espace des paramètres sans risquer le sur-entraînement, puisque les différents outils d'analyse ne possèdent vraisemblablement pas la même distribution sur l'espace des paramètres. Il serait intéressant de vérifier si Fips lui-même pourrait être amélioré afin d'analyser plus de phrases et de minimiser le nombre d'erreurs dans les analyses considérées comme complètes. Un facteur de degré de confiance de Fips envers les analyses produites pourrait être utilisé avantageusement par notre système. Il serait possible, étant donné les résultats obtenus par la Run 2 lors de l'évaluation, de considérer les phrases non analysées pour l'extraction, contrairement à ce que nous

faisons maintenant. L'accès à plus de phrases augmente les chances de recueillir des phrases particulièrement saillantes pour un bon résumé d'extraction. Ainsi, pour les phrases analysées par FIPS, le calcul se ferait selon la formule de la Run 1, et pour les autres le calcul se ferait selon la formule de la Run 2. Une telle combinaison permettrait de profiter du maximum de ressources possibles.

Des modifications à notre approche sont également envisageables. Par exemple, il serait intéressant de comparer en détail la performance de notre système avec des parseurs syntaxiques autres que FIPS, de même que pour des dictionnaires sémantiques ou synonymiques autres que WordNet. Introduire de nouveaux critères et tester si leur ajout améliore la qualité des résumés produits serait aussi avantageux. Par exemple, la fréquence-document, utilisée extensivement par l'équipe ayant le mieux réussi à résoudre la tâche donnée, est un critère qu'il faudrait tester sur notre système. Des modifications à nos critères actuels devraient aussi être testées. Nous avons considéré changer la façon dont nous employons la profondeur d'arbre (essayer autre chose qu'une simple division). Similairement, il serait intéressant d'utiliser la position de la phrase dans le texte à l'aide d'un calcul différent. Il est probablement utile de considérer des alternatives à la façon dont le contenu informatif d'une phrase et les valeurs $tf \cdot idf$ sont calculés, notamment le calcul des valeurs idf qui pourrait se faire sur les 10 documents à résumer plutôt que sur tout le corpus. Varier l'utilisation de Wordnet en changeant le nombre de mots servant à l'expansion de requête ou en considérant ce nombre comme un paramètre à optimiser serait à considérer.

De nouvelles techniques pourraient être ajoutées pour améliorer les résumés créés. Notre traitement pourrait inclure davantage de mesures pour combattre la redondance, ce qui aurait pour effet d'améliorer la lisibilité et souvent la variété de contenu d'un résumé. Pour ce faire, nous introduirions un nouveau critère anti-redondance, qui ajouterait un score négatif calculé par exemple par la proximité entre la phrase considérée à extraire et toutes les phrases déjà extraites. Aussi, la

technique employée par Chen et al., qui consiste à conserver un certain nombre de phrases plus grand que nécessaire, à calculer toutes les permutations possibles de celles-ci et à effectuer une évaluation du contenu des résumés ainsi obtenus, pourrait être implémentée dans notre programme. Il serait également intéressant de tester si une utilisation des centroïdes ou d'autres techniques à base de graphes améliorerait ou non la performance de NESS.

La partie symbolique de notre approche pourrait être approfondie. En particulier, une utilisation extensive d'un parseur nous permet d'effectuer de la compression des phrases selon la méthode développée par Gagnon et Da Sylva. Cette compression représenterait possiblement une grande amélioration puisque la compression élimine les détails moins pertinents de chaque phrase pour en garder les points essentiels. Il est alors possible d'extraire plus de phrases, ce qui permet d'inclure plus d'idées et de concepts différents dans un résumé dont la taille est fixe. FIPS produit également une analyse grammaticale incluant genre, nombre et fonction grammaticale. Il serait possible d'utiliser ces informations d'une façon similaire à notre utilisation de la profondeur d'arbre, ou même en remplacement de cette information. Par exemple, un mot dont la fonction est sujet pourrait recevoir un poids plus élevé pour le calcul de similarité qu'un autre dont la fonction est complément d'objet et, à son tour, pour un autre encore qui serait complément circonstanciel. Une analyse détaillée, utilisant FIPS ou d'autres ressources, pourrait nous aider à retirer le maximum d'information de la requête elle-même, également, ce qui n'est pas fait en ce moment.

Enfin, il serait très intéressant d'essayer d'inclure notre démarche symbolique comme module à une approche différente de celle que nous avons utilisée. Par exemple, les deux approches de Chen et al. pourraient être reproduites, puis complémentées par l'usage d'informations provenant d'un traitement symbolique de l'information linguistique profonde, possible grâce à l'analyse syntaxique et grammaticale. Les critères très restreints pour identifier les "mots pertinents" de leur

première approche ou ceux pour évaluer l'importance des mots dans leur deuxième approche pourraient être remplacés par les nôtres, tout en conservant le reste de leurs démarches. Nous croyons qu'un tel emploi des techniques symboliques que nous avons proposées pour complémenter un système déjà performant peut contribuer à l'améliorer davantage. Ceci est d'autant plus probable que notre approche de base, sans la partie symbolique, est relativement peu innovatrice.

6.2 Conclusion

Nous avons réussi à développer un système symbolique de création automatique de résumés de mise à jour qui produit des résumés lisibles et de bonne qualité.

Rappelons que la problématique des résumés multi-documents et surtout des résumés de mise à jour est récente et complexe. Nous gagnons à tenter de résoudre ces problèmes et cela pourrait éventuellement mener à un programme distribué au grand public, qui "lirait" à la place de l'utilisateur un grand nombre de nouvelles sur internet, et qui lui résumerait ensuite l'information rencontrée en omettant les informations déjà connues au besoin.

Notre système proposé, NESS, a bien performé, dans le cadre d'un concours organisé par NIST et a obtenu des résultats très compétitifs. Ceci démontre l'utilité future de nos travaux et justifie la démarche employée.

Notre méthodologie s'inscrit dans le cadre des approches symboliques et d'un certain retour vers les buts initiaux de l'intelligence artificielle, c'est-à-dire de rapprocher le travail de la machine de la façon dont un être humain essaierait de résoudre le problème. L'objectif ultime de notre approche serait d'éventuellement mener à une approche de création de résumés qui s'intéresserait à la sémantique des textes plutôt qu'aux statistiques des mots individuels. Nous espérons que notre travail, qui cherche à exploiter des informations linguistiques plus riches et profondes, puisse s'insérer dans un effort partagé, déployé vers ce but.

BIBLIOGRAPHIE

[AKS05] S. Afantenos, V. Karkaletsis, and P. Stamatopoulos. Summarization from medical documents : a survey. *Artificial Intelligence in Medicine*, 33(2) :157–177, 2005.

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. In *Computer Networks and ISDN Systems*, pages 107–117, 1998.

[CYL+09] Shouyuan Chen, Yuanming Yu, Chong Long, Feng Jin, Lijing Qin, Minlie Huang, and Xiaoyan Zhu. Tsinghua university at the summarization track of tac 2008. sera plublié dans : Proceedings of the Text Analysis Conference 2008, 2009.

[Edm69] H. P. Edmundson. New methods in automatic extracting. *J. ACM*, 16(2) :264–285, 1969.

[Far05] Atefeh Farzindar. *Résumé automatique de textes juridiques*. PhD thesis, Université de Montréal, 2005.

[GFT09] David Gillick, Benoit Favre, and Dilek-Hakkani Tür. The icsi summarization system at tac 2008. sera plublié dans : Proceedings of the Text Analysis Conference 2008, 2009.

[GLNW08] Pierre-Etienne Genest, Guy Lapalme, Luka Nerima, and Eric Wehrli. A symbolic summarizer for the update task of tac 2008. In *Proceedings of the First Text Analysis Conference*, Gaithersburg, Maryland, USA, 2008. National Institute of Standards and Technology. http ://www.nist.gov/tac/publications/.

[GMCK00] Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. Multi-document summarization by sentence extraction. In *NAACL-ANLP 2000 Workshop on Automatic summarization*, pages 40–48, Morristown, NJ, USA, 2000. Association for Computational Linguistics.

[GS06] Michel Gagnon and Lyne Da Sylva. Text compression by syntactic pruning. In *Proceedings of the 19th Canadian Conference on Artificial Intelligence*, 2006.

[Hov88] Eduard H. Hovy. Planning coherent multisentential text. In *Proceedings of the 26th annual meeting on Association for Computational Linguistics*, pages 163–169, Morristown, NJ, USA, 1988. Association for Computational Linguistics.

[ITL07] National Institute of Technology Information Technology Laboratory. Document understanding conferences. http ://duc.nist.gov/, 2007.

[ITL08] National Institute of Technology Information Technology Laboratory. Text analysis conference. http ://www.nist.gov/tac, 2008.

[Jon99] Karen Sparck Jones. Advances in automatic text summarization. 1999.

[Lin04] Chin-Yew Lin. Rouge : A package for automatic evaluation of summaries. In *Proceedings of the ACL-04 Workshop : Text Summarization Branches Out*, pages 74–81, 2004.

[Luh58] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2) :159–165, 1958.

[Man01] Inderjeet Mani. *Automatic Summarization*, volume 3 of *Natural Language Processing*. John Benjamins Publishing Company, 2001.

[Mar00] Daniel Marcu. *The Theory and Practice of Discourse Parsing and Summarization*. A Bradford Book. MIT Press, Cambridge, Massachusetts, 2000.

[MBF+90] George Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. Wordnet : An on-line lexical database. *International Journal of Lexicography*, 3 :235–312, 1990.

[Mih04] Rada Mihalcea. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 20, Morristown, NJ, USA, 2004. Association for Computational Linguistics.

[MT88] W.C. Mann and S.A. Thompson. Rhetorical structure theory : Toward a functional theory of text organization. *Text*, 3(8) :243–281, 1988.

[Net09] Cable News Network. http ://www.cnn.com/, 2009.

[Pas06] Rebecca J. Passonneau. Pyramid annotation guide : Duc 2006. http ://www1.cs.columbia.edu/ becky/DUC2006/2006-pyramid-guidelines.html, 2006.

[PZ75] Joseph J. Pollock and Antonio Zamora. Automatic abstracting research at chemical abstracts service. *Journal of Chemical Information and Computer Sciences*, 15(4) :226–232, 1975.

[Rec99] W3C Recommandations. Xsl transformations. http ://www.w3.org/TR/xslt, 1999.

[RJST04] Dragomir R. Radev, Hongyan Jing, Malgorzata Stys, and Daniel Tam. Centroid-based summarization of multiple documents. *Information Processing and Management*, 40(6) :919–938, 2004.

[Weh07] Eric Wehrli. Fips, a "Deep" Linguistic Multilingual Parser. In *Proceedings of the ACL 2007 Workshop on Deep Linguistic Processing*, pages 120–127, Prague, Czech Republic, 2007.

Annexe I

Article paru dans le cadre de TAC 2008

Nous reproduisons ici notre article paru dans le cadre de la Text Analysis Conference 2008 [GLNW08].

A Symbolic Summarizer for the Update Task of TAC 2008

Pierre-Etienne Genest, Guy Lapalme
RALI-DIRO
Université de Montréal
P.O. Box 6128, Succ. Centre-Ville
Montréal, Québec
Canada, H3C 3J7

Luka Nerima, Eric Wehrli
Laboratoire d'Analyse et de
Technologie du Langage
Université de Genève
2, rue de Candolle
CH-1211 Genève 4, Switzerland

{genestpe,lapalme}@iro.umontreal.ca{Luka.Nerima,Eric.Wehrli}@lettres.unige.ch

Abstract

NESS, RALI's summarization system for the TAC 2008's update task, brings improvements and continuation to our last year's "all-symbolic" approach. The most distinctive feature of our system is to rely on the syntactical parser FIPS to extract linguistic knowledge from source documents. NESS selects sentences based on linguistic metrics, especially $tf \cdot idf$ scores that measure the relevance of the newswire article sentences to the given topic. It also measures the similarity between candidate sentences and the previous articles already read by the user. NESS ranked well in the competition, obtaining excellent scores in linguistic quality and overall responsiveness.

1 Introduction

For TAC 2008's main summarization task, we developed a multi-document, topic-driven, update summarizer. The input documents were newswire articles from the collection AQUAINT-2 and they were guaranteed to be related to their given topic. The topics themselves represent "real-world questions" that the summaries should answer. Two clusters of 10 articles, referred to as *part A* and *part B*, were assigned to each topic and a 100-word summary was created for each part. Part B articles were more recent than part A articles, and the summary of the second cluster had to provide only an update about the topic, avoiding any repetition of information from the first cluster. This was meant to simulate a user who is interested in learning about the latest developments on a specific topic, and who wishes to read a brief summary of the latest news.

RALI proposes the NEws Symbolic Summarizer (NESS) as a solution to this complex problem. We use a symbolic approach that relies on the syntactic parser FIPS to extract linguistic knowledge from the input text. $tf \cdot idf$ scores are used to measure the relevance of a sentence to the topic, as well as other linguistic metrics. NESS also incorporates specific components for update summarization and the use of WordNet. It brings several improvements to the system we developed for DUC 2007, GOFAISUM.

The tree structure that results from FIPS's analysis is well-suited to be expressed in XML format and we chose to express all the data used by our system in this format as well. We find that using XML and XSLT is well-suited to a symbolic approach to summarization like ours, because it enables the easy manipulation of structured data. Most significant parts of our system were programmed using XSLT sheets that transform and manipulate XML and text files.

This article is organized as follows. We describe FIPS and the other resources used by our system in section 2. Section 3 details the algorithm and implementation of NESS. Section 4 presents and discusses the results that we obtained in the competition. The last section provides a conclusion.

2 Resources Used

2.1 FIPS

FIPS (Wehrli, 2007) is a robust multilingual symbolic parser based on generative grammar. It is the cornerstone of a long-term project at the LATL (Language Technology Laboratory) at the Université de Genève and is used in several NLP applications: text-to-speech synthesis, automatic collocation extraction, translation of words in context, etc.[1] Although we used the English configuration of FIPS for TAC 2008, FIPS also parses French, German, Italian, Spanish and Greek.

2.1.1 The principles of FIPS

The syntactic structures built by FIPS are all of the same pattern, that is `[XP L X R]` where `XP` stands for the label of the structure, `L` stands for the possibly empty list of left constituents, `X` for the possibly empty head of phrase and `R` for the possibly empty list of right constituents. The possible categories for `X` are the usual parts of speech (noun, adjective, verb etc.). The overall resulting structure is a tree where the node labels are the `XP`s and the leaves, the `X`s. Figure 1 shows the parse tree constructed by FIPS for a sentence extracted from the TAC 2008 test data.

The parser makes use of 3 fundamental mechanisms: projection, merge and move.

The **projection** mechanism assigns a fully developed structure to each input word, based on its category and other inherent properties. Thus, a common noun is directly projected to an `NP` structure (with the noun as its head), a preposition to a `PP` structure, etc. The occurrence of a tensed verb triggers a more elaborate projection: a whole `TP`−`VP` structure is assigned. For instance, Figure 1 shows that the modal *could* occurs in the `TP` position while *pull off* is projected to a `VP`.

The **merge** mechanism combines two adjacent constituents, `A` and `B`, either by attaching constituent `A` as a left constituent of `B`, or by attaching `B` as a right constituent of any active node of `A`. In Figure 1, the determiner phrase *Washington's first Republican ...* is right attached to the `VP` *become* while the `AdvP` *If Rossi ...* is left-attached to the `TP` *could....* Merge operations are constrained by mostly language-specific conditions which can be described by means of syntactic rules. For instance, in English, a rule states that a `DP` can be left-attached to a `TP` if (1) the `DP` agrees in number and person with the `TP` and (2) the `DP` can be interpreted as the subject of the `TP`. For English, FIPS has about 30 rules for left attachment and 90 rules for right attachment.

In order to handle extrapositions, the **move** operation creates chains by linking each of the extraposed elements to an abstract (empty) element in the canonical position. For instance, when parsing the sentence "Whom did they invite?", FIPS creates the following chains: "Whom$_i$ did$_j$ they e$_j$ invite e$_i$?"

2.1.2 Lexical resources for FIPS

The lexical database used by FIPS is composed of (i) a full form lexicon, containing all the orthographical forms of the words along with their morphological descriptions, (ii) a lexicon of lexemes, containing the syntactic and semantic information of the words (corresponding roughly to the entries of a classical dictionary) and (iii) a lexicon of collocations (in fact multi-word expressions, ie. collocations and idioms). The English lexical database includes about 55,000 lexemes and 6,500 collocation entries. FIPS handles unknown words by guessing their lexical category according to their position in the sentence and the applicable syntactic rules.

2.2 XML and XSLT

XML (eXtensible Markup Language) is a formalism designed to describe and share structured information through text-based trees. It was used in this work to tag the output of FIPS as well as all other intermediate outputs, such as idf values, WordNet-obtained synonyms, and derived values and scores.

The eXtensible Stylesheet Language Transformations (XSLT) is a language for transforming the structure and content of an XML document. It is a declarative language used to build *stylesheets*, consisting of template rules each describing how a particular element of the XML document it is fed should be processed. Our implementation relies almost exclusively on XSLT stylesheets to process the input and derived data, to score and select sentences, and to create the summaries. The XSLT language makes pattern recognition and pruning of the FIPS parse trees especially easy.

[1] See `http://www.latl.unige.ch/english/latl_e.html` for a complete list of references.

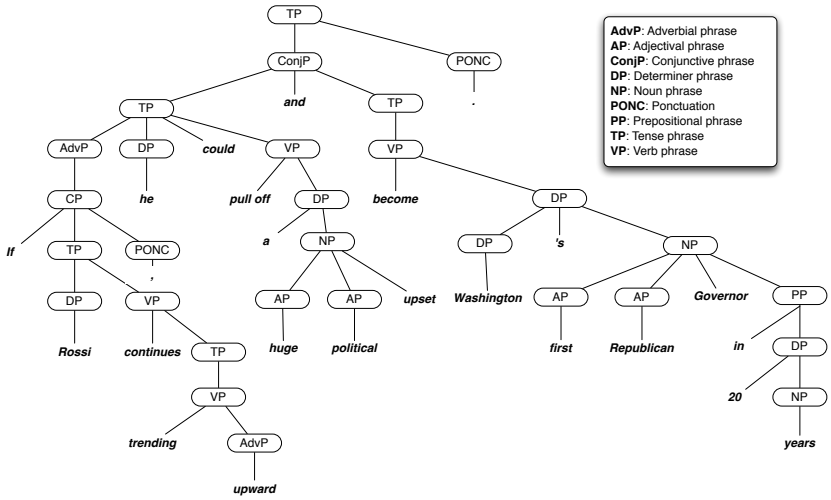

Figure 1: FIPS parse tree for the sentence *If Rossi continues trending upward, he could pull of a huge political upset and become Washington's first Republican Governor in 20 years.* The corresponding topic question was "Washington Governor Race. Follow developments concerning the 2004 election for Governor of Washington.".

2.3 WordNet

WordNet is a lexical database of the English language (Fellbaum, 1998). Nouns, verbs, adjectives and adverbs are grouped into cognitive synonym sets (synsets) that express a distinct concept. WordNet is widely used in a variety of language processing applications, notably in the domain of Information Retrieval for query-expansion (Voorhees, 1994).

As we will see in section 3, our approach uses the topic in a similar way as the query is used in IR. Therefore, we decided to use WordNet to conduct topic-expansion, our equivalent of query-expansion. For each noun of a topic (as identified by FIPS), we extract synonyms from the WordNet database to retrieve additional words to define the topic. One of our system's criteria for sentence extraction uses this broader topic to compute similarity scores between candidate sentences and the topic. This is explained in section 3.3.5.

3 Our Approach

NESS uses a symbolic approach to extract article sentences relevant to the topic. Part B summaries must also avoid any repetition of information with the part A articles. Some preprocessing is required on the input documents before FIPS is applied to produce syntactic parse trees of every sentence. A combination of linguistic metrics let us give a relevance score to each sentence, based on its estimated capability to summarize the topic. The sentences with the top scores are then selected to form 100-word summaries. This process is illustrated in figure 2.

3.1 Topic and Articles Preprocessing

Preprocessing the topic and articles is a relatively simple task for us, since they are both given to us in XML-compatible formats. The preprocessing includes extracting the information that is of use to us and making adjustments to the text to make further treatment smoother.

For the topic, we only keep the `<title>` and

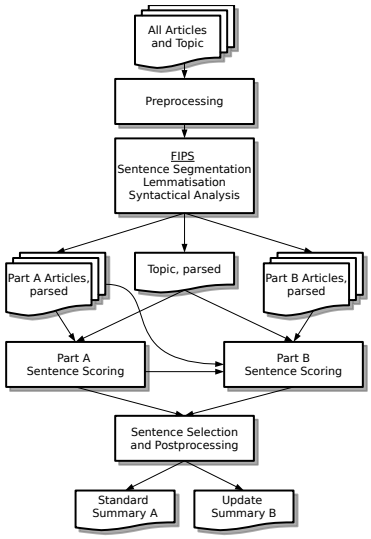

Figure 2: Workflow used by NESS on a single topic to produce two summaries. The labels refer to sections in this paper where the corresponding operations are discussed.

<narrative>; and for the articles, we keep only the text. Everything else is completely ignored, including the headline of the articles. Since many articles were missing a date, we had to ignore this information altogether.

To make sure that the files are compatible with our program's modules, we also adjust the quotation marks and remove middle initials from names, because those were not handled correctly by FIPS in the next step. At this point, we choose to repeat the title of each topic twice, to increase the relative weight of the words from the title over the ones in the narrative in all the similarity calculations based on $tf \cdot idf$ scores later on. It was observed that the titles were more essential than the narratives to describe the information need expressed by the topic statement.

3.2 FIPS Analysis

The article and topic texts resulting from preprocessing are all submitted to FIPS for sentence segmentation, lemma-extraction and syntactic analysis as described in section 2.1. All the information appears in the parse trees resulting from the execution of FIPS, on which we rely exclusively for sentence scoring. As mentioned in section 2.2, those trees are converted to XML format so that all the following treatment can be implemented in XSLT stylesheets.

3.3 Sentence scoring

Our sentence extraction scheme uses a weighted average of several criteria mixed together to compute a final score for each sentence. We selected 8 criteria; 6 general criteria used in parts A and B: `Word_Sim` , `Word_Depth_Sim` , `Lemma_Sim` , `Sent_Position` , `Sent_Weight` and `Expansion_Sim` ; and 2 update-specific criteria only used in part B: `Cluster_Sim` and `Top_Sents_Sim` .

They are all discussed in this section, as well as how we determined the weights to use for each criterion and how the final score is computed.

3.3.1 $tf \cdot idf$ Similarity Scores

Six of the criteria mentioned involve a similarity score between two "documents", which are sentences, topics or an entire cluster of articles. We compute a $tf \cdot idf$ score for each word or lemma of the two documents and construct a vector for them in the word or lemma space. tf is the term frequency in the document and idf is the inverse document frequency. These are easily computed since word segmentation and lemma extraction are a part of FIPS analyses, so all the information is readily available. idf scores are computed over the entire TAC 2008 competition corpus (960 articles and 48 topics). The actual similarity score is computed by the cosine between the two vectors v_1 and v_2, using the following equation.

$$\text{SIM}(v_1, v_2) = \cos\theta_{v_1 v_2} = \frac{\vec{v_1} \cdot \vec{v_2}}{\|\vec{v_1}\|\|\vec{v_2}\|}$$

The most common English-language words (stopwords) were ignored for computing the similarity scores. Also, because the verbs in the topics were

observed to be generally in the imperative mood and not semantically meaningful, we chose to exclude all verbs from the scores of similarity with the topics. This is possible because the FIPS analysis indicates which words are verbs, as discussed in section 2.1.

The two criteria `Word_Sim` and `Lemma_Sim` are computed directly as a similarity score between a candidate sentence and the topic. The former score is based on the words as they appear, and the latter uses the *tf* and *idf* values related to the lemmas produced by FIPS to compute the scores.

3.3.2 Word Depth in Parse Tree

The depth of a word in the FIPS parse tree is used to measure the importance of the word within the sentence. We can intuitively reason that the deeper a word appears in a sentence, the more dependant on other words it is, and therefore the less important it is. For instance, the subject of a sentence will be more important than a word used to describe it. Our heuristic to account for this is simple, we divide all the $tf \cdot idf$ scores of each word by the minimum depth at which they appear in the sentence. The criterion `Word_Depth_Sim` is the similarity score obtained by computing the cosine between the words of the topic and of the sentence, with these modified values.

3.3.3 Sentence Position

The position of a sentence within a document is usually indicative of the importance of its content. This is especially true in newswire articles, which tend to always begin with more concise, descriptive statements about the subject of the article. On the other hand, sentences that appear late in an article tend to be descriptive of only a very specific aspect of a topic. The `Sent_Position` criterion accounts for this by attributing a score equal to 1 divided by the rank of the sentence in the article (the first sentence of an article has a score of 1, the second a score of 0.5, and so on). Early sentences are highly favored by this criterion.

3.3.4 Sentence Weight

The *idf* weight of a word is indicative of how rare this word is encountered in the corpus. We can intuitively suppose that sentences containing rarer words are more likely to contain new information, whereas sentences with mostly very common words in the context of the corpus are more likely to contain little information of interest. The criterion `Sent_Weight` computes the sum of the idf weights of all the words in each sentence to estimate their total informational weight.

3.3.5 Topic-Expansion with WordNet

The criterion `Expansion_Sim` is computed by a similarity scores between a candidate sentence and an expanded topic which includes the topic words and their synonyms. The additional topic words come from WordNet synsets of the nouns of the topic as explained in section 2.3.

3.3.6 Update-specific Criteria

We included two criteria specific to the update task, done in part B of each topic. Those two criteria try to estimate whether a sentence repeats information contained in cluster A by computing $tf \cdot idf$ similarity scores as we have done before. Of course, they are weighed negatively in the computation of the final score, because a higher value of those criteria indicates a higher similarity with cluster A, which is undesirable.

`Cluster_Sim` measures the similarity between a sentence and all of cluster A by computing the similarity score as though cluster A was one entity containing all the words of each of its documents. We assume that the word-content of cluster A can serve as a rough estimate of its information-content and sentences that are too similar to that word-content are less likely to be selected in the summary for part B.

When creating the summary of part A, the 15-20 sentences with the best scores according to our system are kept for further use, as indicated by an arrow going from part A sentence scoring to part B scoring in figure 2. The criterion `Top_Sents_Sim` computes a similarity score between the candidate sentence and each of the most relevant sentences of part A, keeping the highest value computed as the score of the criterion. This tends to reject part B sentences that are very similar to one of the top most relevant sentences of part A.

3.3.7 System Tuning

For the purpose of attributing meaningful weights to the criteria, it is useful that the scores be normalized. The individual scores of each criterion are normalized so that the sentence with the highest score for a given criterion always has a score of 1, for the sentence scoring of each summary made. This way, the weights we choose describe roughly the same ratio between the criteria. This step makes the tuning process not only more intuitive but also more accurate.

To tune our system, we used last year's data for the DUC 2007 update task to determine the best weights to use by our system for each of the 8 criteria mentioned at the start of this section and described throughout. We ran our system using different values of the weights of each criterion and ran the ROUGE package on the summaries. The resulting ROUGE scores and manual observation of the summaries created served as the basis to determine which configuration of the weights was best.

However, the parameter space dimensionality was large (8 parameters that can take values from 0 to 100), the behavior of the ROUGE scores as a function of those parameters was very non-linear in that space, and each test can take up to over 40 minutes to compute. Therefore, we had to somewhat limit the range of our exploration of parameter values and settle for a local maximum that appeared qualitatively sensible.

We set the rule that the 6 criteria that are used to create both part A and part B summaries must have the same relative weights in each part. Although this rule does not lead to a perfect optimization given a certain data set, it is reasonable in the context where the part B summarization process should only differ from part A's by the added constraint of doing an update of information. This rule also takes away the likeliness of overfitting our parameters to the training data set which was relatively limited in size.

The tuning process produced a set of values which we believe to be the best one based on the tests performed for parts A and B. This choice of weights appears in table 1 in the column for run24. Notice that the criterion `Word_Sim` is not used in our preferred settings because `Word_Depth_Sim` and `Lemma_Sim` both seemed to do a better job.

Criterion name	Run24		Run50		Run68
	A	B	A	B	A / B
`Word_Sim`	.00	.00	.10	.09	.00
`Word_Depth_Sim`	.10	.09	.00	.00	.10
`Lemma_Sim`	.30	.26	.45	.39	.30
`Sent_Position`	.25	.22	.25	.22	.25
`Sent_Weight`	.20	.18	.20	.18	.20
`Expansion_Sim`	.15	.13	.00	.00	.15
`Cluster_Sim`	.00	.04	.00	.04	.00
`Top_Sents_Sim`	.00	.08	.00	.08	.00

Table 1: Weights used to produce the standard (A) and update (B) summaries of the 3 runs submitted to TAC. Run68 used the same weights for both clusters.

3.3.8 Final Score

The final score for each sentence is computed by a linear combination of the normalized scores from the criteria described above. The coefficients are weights determined after the tuning process. Table 1 shows the weights that were used in part A (no update) and part B (with update) for our 3 submissions.

3.4 Sentence Selection and Postprocessing

Sentences with the highest scores are used to create the final summary. However, many sentences have to be ignored in our evaluation or our selection process, either because we could not process them, or because such sentences usually hurt the quality of summaries which would contain them. Sentences that meet any of the following conditions were automatically dismissed.

1. Sentences that cannot be completely analyzed by FIPS, because they produce partial analyses that are difficult to manipulate, and whose content cannot be trusted entirely.

2. Identical sentences, excerpted from two different articles addressing the same topic.

3. Sentences with no verb, as they rarely convey interesting information and would hurt the overall grammaticality of the summary.

4. Sentences containing the "I" pronoun, which are usually opinions or feelings, rarely adding factual information. Naturally, FIPS's analysis

allows an easy distinction between the "I" in, say, Voyager I, and the actual pronoun.

5. Sentences ending with a colon or a question mark, which usually introduce an element of information, rather than discuss a point.

6. Sentences less than 5 word long.

For the data of TAC 2008, 28% of the sentences were dropped because FIPS could not parse them, and about 15% were dropped for the other reasons mentioned.

The algorithm used to fill the 100-word selects the best scoring sentences that do not make the summary go over 100 words.

The postprocessing of the sentences includes some minor modifications to the sentences to avoid known referential clarity problems and to compress the sentences where possible by removing fragments of little usefulness. Phrases such as "he said" appearing in a sentence, and parenthetical expressions containing uppercase acronyms, such as "(TAC)".

We would have liked to make textual replacements of relative time references such as "yesterday" or "Tuesday" with the corresponding absolute dates based on the date the article was originally published on. We were unable to do so however, because many articles lacked parts or all of the data pertaining to date of publication. This feature was present in our system last year and we had to drop it this year.

3.5 TAC Submissions

For our participation to TAC 2008, we were allowed to submit three runs, which were identified as runs 24, 50 and 68. Run24 was what we thought to be our best calibration of the system, as described in section 3.3.7. Run50 was run with a modified version of the system in which all syntactical information about word function, tree depth, etc., from FIPS were ignored and no topic expansion was used. Run68 was identical to run24 but did not try to make an update. Table 1 shows the weights that were used in each run.

4 Results and Discussion

There were four evaluation methods used to assess the quality of the summaries submitted to TAC. The *overall responsiveness* score is based on both the linguistic quality of the summary and the amount of information in the summary that helps to satisfy the information need expressed in the topic narrative, as judged by NIST evaluators. The *linguistic quality* score is based on grammaticality, non-redundancy, referential clarity, focus and structure and coherence. The *Pyramid* scores are an evaluation of summary content relying on a manual comparison of with manual models (Harnly et al., 2005). Finally, *ROUGE* scores come from an automatic comparison with reference (manual) summaries, based on repeated fragments such as n-grams (Lin, 2004).

Run	**24**	**50**	**68**	#
part A				
Overall Responsiveness	15th	1st	-	57
Linguistic Quality	2nd	1st	-	57
Pyramid Score	27th	15th	-	57
ROUGE-2	33th	16th	33th	71
part B				
Overall Responsiveness	6th	7th	-	57
Linguistic Quality	8th	11th	-	57
Pyramid Score	8th	10th	-	57
ROUGE-2	17th	24th	19th	71

Table 2: Ranks of our submitted runs for four evaluation methods, on the standard (A) and update (B) summaries.

Table 2 shows, for each of the runs described in section 3.5, how NESS performed as evaluated by those four methods, when compared to all the runs submitted in the competition.

In part B - update summaries -, we obtain good results in general, most notably in overall responsiveness (6th and 7th out of 57). In part A - summaries without update -, Run50 receives the best scores of the competition in both overall responsiveness and linguistic quality, while Run24 arrives respectively 15th and 2nd in those categories. Our Pyramid scores are relatively good, and our ROUGE scores in both parts are lower but still strong. Strangely, Run24 did better in part B while Run50 did much better in part A.

It was very interesting to have two of our runs evaluated manually instead of just one, so that we can compare different settings of our system. As

mentioned in section 3.5, Run50 made no use of other features from FIPS apart from lemmatization, and this allows us to assess the usefulness of this information when comparing with Run24. Actually, the two runs end up receiving similar results, but this is a significant piece of information. Indeed, it points to the fact that the added knowledge gained from the FIPS parse trees is offset by the loss of information due to cutting away the 28% of sentences that are not analyzed entirely (as mentioned in section 3.4). There would likely be a lot to gain from considering all the sentences, even those with a failed or incomplete analysis. FIPS usually provides a partial analysis for those sentences, and thus we can still compute scores for all our criteria for them, although it is not as accurate.

We have also noticed that there was no need to spend a lot of resources on avoiding repetitions for the update task, because the article clusters that were provided already appear to avoid repetition to some extent. Indeed, while tuning our system, we observed that it was not beneficial to spend more than 12% of our scoring toward the update task. As an additional example, we can mention that Run68, which did no updating at all but was otherwise identical to Run24, was ranked only 2 positions behind it by the ROUGE evaluation. Therefore, we believe that newswire articles - in the way they were used this year and last year - are probably not the most appropriate type of corpus to test and develop update summarizers. Maybe a different choice of article clusters or different types of document sets would be more discriminant on the quality of the updating process. We would be interested to test update summarization systems on documents with a different structure and of a different nature, such as scientific articles or legal documents.

Judging from our tests during system tuning, it also appeared that `Sent_Position` (heavily favoring the first few sentences of an article) was a very strong criterion for sentence selection, and we indeed selected it to account for 25% of our scoring (see table 1). This works so well only because of the type of the documents summarized.

5 Conclusion

We successfully developed in NESS a competitive symbolic system for update summarization. The knowledge provided by FIPS gives our system an edge for scoring sentences during the summarization process and we hope to find more and more ways to profit from it in future work. One way will be to use even the failed analysis as a tool to evaluate sentence relevance to the topic. The added features of removing verbs from $tf \cdot idf$ scoring and using topic-expansion with WordNet contributed to our better results this year, as well as our methodology for tuning the system. The update task was a new challenge and we came up with two metrics that we believe serve this purpose well, similarity with the old cluster, and the maximum similarity with the top sentences of the old cluster. NESS obtained excellent results when compared to the other systems, and we hope that this demonstrates in part the validity of our approach.

References

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Aaron Harnly, Ani Nenkova, Rebecca Passonneau, and Owen Rambow. 2005. Automation of summary evaluation by the pyramid method. In *Recent Advances in Natural Language Processing (TANLP)*, September.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the ACL-04 Workshop: Text Summarization Branches Out*, pages 74–81.

Ellen M. Voorhees. 1994. Query expansion using lexical-semantic relations. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 61–69, New York, NY, USA. Springer-Verlag New York, Inc.

Eric Wehrli. 2007. Fips, a "Deep" Linguistic Multilingual Parser. In *Proceedings of the ACL 2007 Workshop on Deep Linguistic Processing*, pages 120–127, Prague, Czech Republic.