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.

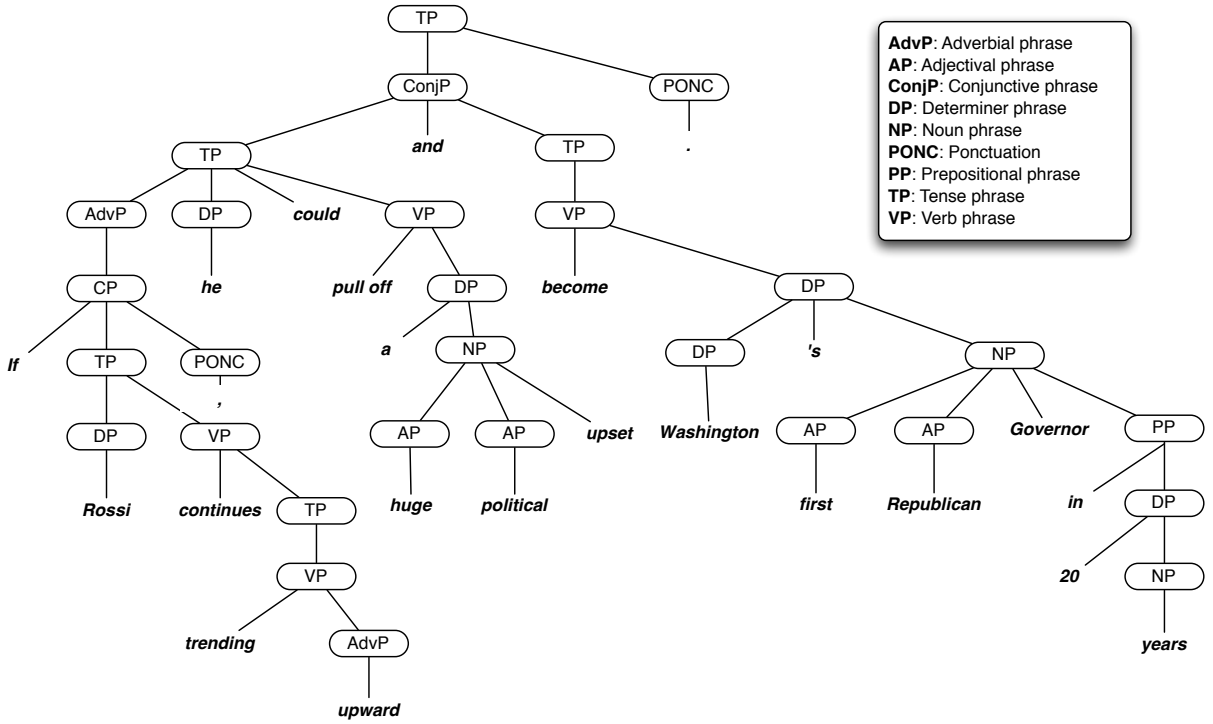

Figure 1: FIPS parse tree for the sentence *If Rossi continues trending upward, he could pull of a huge political upset and become Washington's first Republican Governor in 20 years.* The corresponding topic question was "Washington Governor Race. Follow developments concerning the 2004 election for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

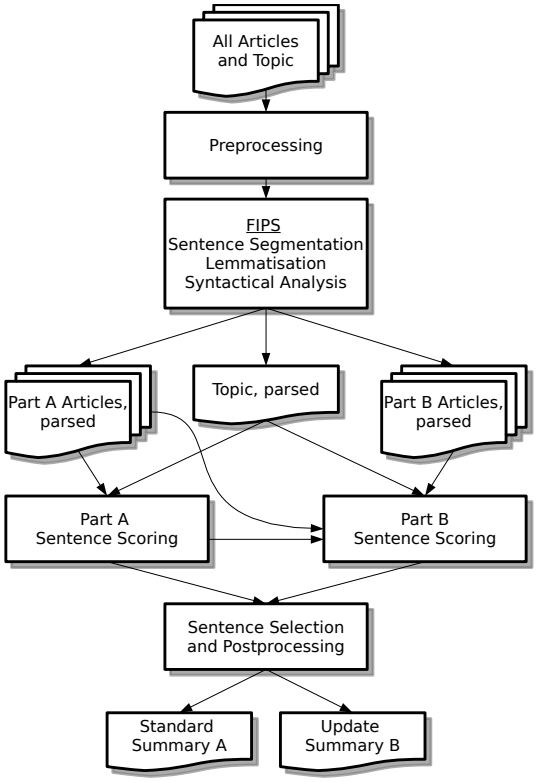

Figure 2: Workflow used by NESS on a single topic to produce two summaries. The labels refer to sections in this paper where the corresponding operations are discussed.

<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.