Support Vector Machines for Query-focused Summarization trained and evaluated on Pyramid data

Maria Fuentes
TALP Research Center
Universitat Politècnica de Catalunya
mfuenteslsi.upc.edu

Enrique Alfonseca
Computer Science Departament
Universidad Autónoma de Madrid
Enrique.Alfonseca@uam.es

Horacio Rodríguez
TALP Research Center
Universitat Politècnica de Catalunya
horaciolsi.upc.edu

Abstract

This paper presents the use of Support Vector Machines (SVM) to detect relevant information to be included in a query-focused summary. Several classifiers are trained using pyramids of summary content units information. The Mapping-Convergence algorithm is used with positive, unlabeled data, and a small set of negative seeds.

The SVMs are tested on two Document Understanding Conference (DUC) 2006 systems. The performance of the new approaches is compared with the original systems using the DUC 2005 corpus as test data. For evaluation purposes, we also present an automatic method based on pyramid data with good correlation with other human or automatic procedures.

1 Introduction

Multi-Document Summarization (MDS) is the task of condensing the most relevant information from several documents in a single one. In the particular case of query-focused summarization, the summary has to provide, in terms of DUC-2005 and DUC-2006 contests[1], a "brief, well-organized, fluent answer to a need for information", described by a short query (two or three sentences). In both contests the participant systems had to synthesize 250-word sized summaries for fifty sets of 25-50 documents in answer to some queries. The main difference between DUC-2005 and DUC-2006 was that in 2006 there was not indication about the granularity (specific, generic) of the desired answer, and the questions try to ask for less information.

In previous DUC contests, from 2001 to 2004, the manual evaluation was based on a comparison with a single human-written model. Much information in the evaluated summaries (both human and automatic) was marked as "related to the topic, but not directly expressed in the model summary". Ideally, this relevant information should be scored during the evaluation. The pyramid method (Nenkova and Passonneau, 2004) addresses the problem by using multiple human summaries to create a gold-standard, and by exploiting the frequency of information in the human summaries in order to assign importance to different facts. However, the pyramid method requires to manually match fragments of automatic summaries to the Semantic Content Units (SCUs) in the pyramids. A proposal to automate this part of the process was presented in Fuentes et al. (2005).

As proposed by Copeck and Szpakowicz (2005), the availability of human-annotated pyramids constitutes a gold-standard that can be exploited in order to train extraction models for the summary automatic construction. This paper describes several models trained from the information in the DUC-2006 pyramid annotations using Support Vector Machines (SVM). The evaluation, performed on the DUC-2005 data, has allowed us to discover the best configuration for training the SVMs, and indicates that, for both systems a gain can be obtained when the automatic pyramid scores are compared.

[1] http://www-nlpir.nist.gov/projects/duc/

2 Related work

An important step in the generation of the summaries is the extraction and ranking of candidate sentences. The procedures taken by most of the systems participating in the two previous DUC competitions combine two different kinds of metrics: one that identifies the saliency of a sentence inside a document, and other that identifies the similarity to the query. Some common techniques for both metrics are the following:

- To identify salient sentences, it is possible to estimate the probability that a term appears in the summary by studying the term frequency in the document cluster (Nenkova and Vanderwende, 2005), with an approximate oracle score (Conroy et al., 2006) or using Bayesian approaches (Daumé III and Marcu, 2005). Other techniques are using centrality metrics and graph-based algorithms.
- To identify the similarity between the query and each sentence, common procedures are using tree similarity (Schilder and McInnes, 2006), a Question-Answering system (Fuentes et al., 2006; Lacatusu et al., 2006) or the vector space model, possible extended with query expansion (Alfonseca et al., 2006), syntactic and semantic relations (Lacatusu et al., 2006) or Latent Semantic Analysis (Hachey et al., 2006).

Apart from sentence selection, these systems usually implement sentence trimming (Dorr et al., 2003) and strategies for sentence reordering and redundancy elimination.

Concerning the use of supervised Machine Learning techniques in summarization, one of the first applications was in Single-Document Summarization (Ishikawa et al., 2002). A similar approach was applied to Multi-Document Summarization by training a SVM on single-document summarization data and using it to rank all the sentences from a document collection (Hirao et al., 2003). Fisher and Roark (2006) train a perceptron on DUC-2001 and DUC-2002 multidocument summarization data (using ROUGE as the scoring function) to rank sentences, and a second perceptron trained on DUC-2005 re-ranks the sentences to take into consideration the query.

3 Approach

Following the work of Hirao et al. (2003), we envision the extraction of important sentences as a binary classification problem, where a sentence is either apt or not suitable for inclusion in a summary. Not only SVMs can be trained for this classification problem, but also they can rank the candidate sentences in order of relevance (Kazawa et al., 2002).

To train the SVM, it is necessary to first build a training corpus. To do this, we have used the DUC-2006 dataset, including topic descriptions, document clusters, peer and manual summaries, and pyramid evaluations as annotated during the DUC-2006 manual evaluation. From all these data, the training set is generated in the following way: first of all, the sentences in the original documents are matched with the sentences in the summaries (Copeck and Szpakowicz, 2005). Next, all document sentences that matched a summary sentence containing at least one SCU are extracted. Built in this way, the training set contains only possitive examples. The lack of negative ones is addressed in section 3.3. Finally, an SVM is trained using the previous annotations. The following subsections further elaborate each of these steps.

3.1 Linguistic preprocessing

The documents from each cluster are preprocessed using a pipe of general purpose processors performing tokenization, POS tagging, lemmatization, fine grained Named Entities Recognition and Classification, anaphora resolution, syntactic parsing, semantic labeling (using WordNet synsets, Magnini's domain markers, and EuroWordNet Top Concept Ontology labels), discourse marker annotation, and semantic analysis. The same tools are used for the linguistic processing of the query.

As a result, sentences are enriched with lexical and syntactic language-dependent representations. For each sentence, its syntactic constituent structure (including head specification) and the syntactic relations between its constituents (subject, direct and indirect object, modifiers) are obtained. Using these data, a semantic representation of the sentence is produced, that we call *environment*. It is a semantic-network-like representation computed using a process that extracts the semantic units (nodes) and the

Romano_Prodi$_1$ is$_2$ the$_3$ prime$_4$ minister$_5$ of$_6$ Italy$_7$		
i_en_proper_person(1)	entity_has_quality(2)	quality(4)
entity(5)	i_en_country(7)	mod(5,7)
which_entity(2,1)	which_quality(2,5)	mod(5,4)

Figure 1: Environment representation of a sentence.

semantic relations holding between the different tokens. Unit and relation types belong to an ontology of about 100 semantic classes (e.g. person, city, action or magnitude), and 25 relations between them (mostly binary, e.g. *time_of_event*, *actor_of_action*, *location_of_event*). Both classes and relations are related by taxonomic links allowing for inheritance. Figure 1 shows a sentence environment example.

3.2 Collection of positive instances

As indicated before, every sentence from the original documents matching a summary sentence that contains at least one SCU is considered a positive example. We have used a set of features that can be classified into three groups: those extracted from the sentences, those that capture a similarity metric between the sentence and the topic description, and those that try to relate the cohesion between a sentence and all the other sentences in the same document or collection.

The features calculated from attributes of the sentences themselves are the following:

- The position of the sentence inside its document.
- $\frac{1}{N_d}$, where N_d is the number of sentences in the document.
- $\frac{1}{N_c}$, where N_c is the number of sentences in the cluster.
- Three binary attributes indicating whether the sentence contains positive, negative and neutral discourse markers, respectively. For instance, *what's more* or *above all* are positive discourse markers, indicating relevance, while *for example* or *incidentally* indicate lack of relevance.
- Two binary attributes indicating whether the sentence contains *right-directed* discourse markers (those that affect the relevance of the sentence fragment after the marker, such as *first of all*), or discourse markers affecting both the fragment at the right-hand side and at the left-hand side, such as *that's why* (Alonso, 2005).
- Several boolean features that are evaluated to true if the sentence starts with or contains a particular word (e.g. a quote or the verb *to say*) or part-of-speech (e.g. personal pronouns, conjunctions, demonstrative pronouns...)
- The total number of named entities included in the sentence, and the number of Named Entities of each kind considered (people, organizations, locations and miscellaneous entities).
- *SumBasic score* (Nenkova and Vanderwende, 2005). This metric is based on the observation that high-frequency words in the document clusters tend to occur as well in human summaries. Therefore, each sentence receives a weight equal to the average probability of its words in the cluster. The original SumBasic algorithm provides an iterative procedure that updates word probabilities as sentences are selected for the summary. However, as we are evaluating separate sentences but not selecting them for the summary yet, weights are not updated in our case.

  Two different scores are calculated, by estimating word probabilities using only the set of words in the current document, and using all the words in the cluster.

The following features try to capture the similarity between the sentence and the query:

- The percentage of word-stem overlapping between each sentence and the query.
- Three boolean features indicating whether the sentence contains a subject, object or indirect object dependency in common with the query.
- The overlapping between the environment predicates in the sentence and those in the query.
- Two similarity metrics calculated by expanding the query words using the Google search engine, as described by Alfonseca et al. (2006).
- *Modified SumFocus score* (Vanderwende et al., 2006): this score extends the SumBasic score (mentioned above) in order to capture the similarity of a sentence to the query. Because SumBasic is already a feature, only the score obtained by estimating word frequencies from the topic description is included in this feature.

The following features try to capture the relation-

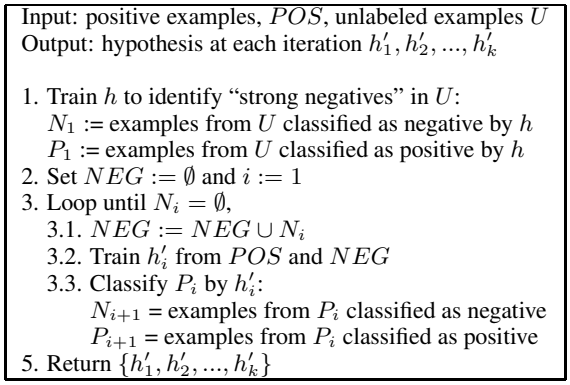

Figure 2: Mapping-Convergence algorithm, from Yu et al. (2002).

ships between this sentence and the remaining sentences in the document:

- Word-stem overlapping between this sentence and the other sentences in the same document. The mean, median, standard deviation and histogram of the distribution of overlappings are all calculated and included as features.
- Word-stem overlapping between this sentence and the other sentences in the same cluster.
- Synset overlapping between this sentence and the other sentences in the same document.
- Synset overlapping with other sentences in the same collection.

3.3 Model training

In order to train a traditional SVM, both positive and negative examples are necessary. A possible procedure to train on just positive instances is a One-Class Support Vector Machine (OSVM) (Manevitz and Yousef, 2001), that calculates a boundary around positive instances. However, according to Yu et al. (2002), OSVMs are prone to underfitting and overfitting when data is scant (which happens in this case), and a simple iterative procedure called Mapping-Convergence (MC) algorithm can greatly outperform OSVM (see pseudocode in Figure 2). It starts by identifying a small set of instances that are very dissimilar to the positive examples, called "strong negatives". Next, at each iteration, a new SVM h'_i is trained using the original positive examples, and the negative examples found so far. The set of negative instances is then extended with the unlabeled instances classified as negative by h'_i.

The following settings can be varied during the experiments:

- Concerning the positive examples, two different sets have been considered: those obtained by applying Copeck and Szpakowicz (2005)'s proposal, and those obtained by matching document sentences to manual summaries.
- Concerning the kernel function of the SVM, we have experimented both with polynomial kernels (lineal, quadratic and cubic) and with Radial Basis Function kernels (RBF).
- Concerning the MC algorithm, two procedures to choose the initial set of "strong negative" examples have been considered: a modification of the algorithm by Yu et al. (2002) to be able to handle non-binary features, and choosing a small set of negative examples manually from the unlabeled instances. In this last case, for each cluster, the two or three automatic summaries with lowest manual pyramid scores were considered to manually select negative sentences. In average, 11.9 sentences were selected for each document cluster. In a similar direction as with positive example, negative example selection could also be done automatically.

3.4 Summary generation

Using the trained SVMs, they can be used to rank the sentences given a new document collection and a topic description. The sentences ranked at the top can be next reordered and checked for redundancy in order to generate the final summary.

4 Evaluation Framework

To evaluate the performance of the SVM in detecting relevant information in query-focused multidocument summarization task the DUC 2005 corpus has been used. The performance of each system is evaluated automatically using the ROUGE and autoPan metrics (described in section 4.3).

4.1 Test Corpus

The 20 clusters manually evaluated in DUC 2005 with the pyramid method were used as test. Features of each sentence from each cluster were computed as described in section 3.1. For complex questions a sentence splitting process was applied. This process

creates new sentences when a conjunction that joins two words with the same POS is found.

4.2 Systems

The DUC 2006 systems evaluated are (Alfonseca et al., 2006), and one of the variants presented in (Fuentes et al., 2006). The performance of each system is contrasted with the use of several SVM in the relevant information detection step.

4.2.1 UAM-Titech06

The UAM-Titech06 system (Alfonseca et al., 2006) is a summarization system focused on producing coherent summaries. To that purpose, questions are divided into subquestions from which aims are identified, and separate mini-summaries are produced for each of the aims. These are later merged together in a final summary.

The main steps in processing a document collection are the following:

1. Linguistic processing of the query and the original documents.
2. Identification of the aims of the questions, and background knowledge. For example, possible aims are *advantages, disadvantages, problems, legal privileges*, etc. of a certain status, specified by a set of background terms extracted from the topic description.
3. Automatic collection of separate corpora for each of the aims, and sentence ranking according to similarity to each corpus.
4. Multi-summaries generation, by choosing the top-ranked sentences from each of the ranks (with no repetitions).
5. Multi-summaries merging in a single summary, generating a small introduction to each to indicate its focus.

4.2.2 SEMsum

With the aim of obtaining non-redundant and cohesioned summaries, the SEMsum system (Fuentes et al., 2006) is organized in the following steps:

1. The query and the set of documents to be summarized are linguistically preprocessed as described in section 3.1.
2. The Relevant Information component take into account the query to detected relevant passages.
3. With the preprocessed documents the Candidates Similarity Matrix Generator component is in charge of computing the similarity matrix among candidates. For that purpose, it uses the environment of each sentence. Environments are transformed into a labeled directed graph representation, where nodes are assigned to positions in the sentence and labeled with the corresponding token, and edges are assigned to predicates (a dummy node, 0, is used for representing unary predicates). Only unary and binary predicates are used. On top of this representation, a rich panoply of lexico-semantic proximity measures between sentences have been built.
   Each measure combines two components:
   A lexical component which includes the set of common tokens, i.e. those occurring in both sentences. The size of this set and the strength of the compatibility links between its members are used for defining the measure.
   A semantic component, computed over the subgraphs corresponding to the set of lexically compatible nodes.
   Four different measures of overlapping have been defined: strict unary predicates, strict binary predicates, loose unary predicates, and loose binary predicates. The loose measures allow a relaxed matching of predicates by climbing up in the ontology of predicates, e.g. provided that A and B are lexically compatible, *i_en_city(A)* can match *i_en_proper_place(B)*, *location(B)* or *entity(B)*. Obviously, loose overlapping implies a penalty on the score.
4. In order to select the candidates, three criteria have been taken into account: Relevance with respect to the query; Density and cohesion; and Anti-redundancy. A graph-based representation of the sentences is used to select the candidates sentences, by using the semantic similarity matrix previously computed. The score used is based on PageRank, as used by Mihalcea and Tarau (2005), but without making the distinction between input and output links.
5. In the Summary Composer sentences are selected by relevance until the desired summary size is achieved. For each selected sentence, it is checked whether the previous sentence in

the original document is also a candidate. If positive, both are added to the Summary in the order they appear in the original document.

4.3 Evaluation measures

4.3.1 ROUGE

ROUGE (Lin and Och, 2004) is an automatic procedure for evaluating summaries, based on n-gram co-occurrences. In the last DUC competitions, both ROUGE-2 (henceforward R-2) and ROUGE-SU4 (R-SU4) were used to rank automatic summaries.

4.3.2 Pyramid-based metric (autoPan)

AutoPan[2] is a procedure for automatically matching fragments of text summaries to SCUs in pyramids, in the following way:

- The text in the SCU label and all its contributors is stemmed and stop words are removed, obtaining a set of stem vectors for each SCU. The system summary text is also stemmed and freed from stop words.
- A search for non-overlapping windows of text which can match SCUs is carried. A window and an SCU can match if a fraction higher than a threshold (experimentally set to 0.90) of the stems in the label or some of the contributors of the SCU are present in the window, without regarding order. Each match is scored taking into account the score of the SCU as well as the number of matching stems. The solution which globally maximizes the sum of scores of all matches is found using dynamic programming techniques.

In what follows, we show that autoPan scores are highly correlated to the manual pyramid scores, a fact that supports the use of autoPan for automatically evaluating the different experiments reported in this paper. We will refer hereforth to the *pyramid score* obtained from the automatic pyramids as **autoPan** and to the scores obtained from manual pyramids as **manPan**.

We use the modified pyramid score computed from the peer annotation in DUC 2005. This score is a ratio of the sum of weights of the SCUs found in the peer (OBServed) to the sum for ideal summary (MAXimum). In the score used, MAX is computed using the average number of SCUs that were found in the seven human model summaries in the corresponding pyramid. Like recall, it indicates the proportion of the target highly weighted SCUs that were found in the peer.

The constituent annotations automatically produced are scored using the same metrics as for manual annotations, and statistical evidence supports the hypothesis that the scores obtained by automatic annotations are correlated to the ones obtained by manual ones for the same system and summary. We apply a Spearman test to the scores obtained by every summary, including the human ones. In total, the data set consists of 540 sampes. The test reports values $r = 0.58$ between **autoPan** and **manPan**, which exceed the critical value for a confidence of 99%. If we repeat the Spearman test with only the automatic systems (500 samples) the value is $r = 0.52$, but it remains inside the 99% confidence level. Nevertheless, it should be noted that the scores of automatic annotations tend to be quite lower than those of manual ones.

If instead of considering the results summary by summary, we take the averages of the scores for each system we observe there seems to be linear dependency between the two variables. If we apply linear regression, we obtain a Pearson coefficient of 0.98 for **autoPan** and **manPan**, with a data set size of 28 samples.

Once we found that the scores from the automatically constructed pyramids correlates with those from the manually constructed ones, we consider the correlation of these scores to the pyramid scores manually assigned, to the responsiveness measure (RESP) used in DUC05, and to R-2 and R-SU4 measures. We apply Spearman and Pearson tests to the average of the scores obtained in all clusters by every non-human system. The results is summarized in Table 1. In the table MANPAN1 refer to the original pyramid score, and AUTOPAN and MANPAN2 refer to the modified score, as described above.

The **autoPan** metric correlates well with all other metrics in both tests: the Pearson correlation test and the Spearman rank correlation. All values exceed the confidence level of 99%, so its use in the evaluation is therefore justified.

[2] Software is available at http://www.lsi.upc.edu/∼egonzalez/autopan.html

	TEST	R2	RSU4	MANPAN1	MANPAN2	RESP
AUTOPAN	Spearman	0.683	0.665	0.802	0.802	0.649
	Pearson	0.755	0.725	0.820	0.851	0.699

Table 1: Correlation values for several metrics, evaluating average scores of non-human systems.

				UAM-Titech06			SEMsum		
System	**Positive examples**	**Seed negatives**	**Kernel**	**R-2**	**R-SU4**	**autoPan**	**R-2**	**R-SU4**	**autoPan**
Original				0.048	0.105	0.052	**0.077**	**0.136**	0.066
Extended with SVM	Obtained from peer summaries	Annotated	RBF	**0.071**	**0.131**	**0.072**	0.066	0.126	**0.069**
			Polynomial	0.062	0.119	0.064	0.061	0.118	0.052
		Automatic	RBF	0.036	0.089	0.024	0.052	0.106	0.035
			Polynomial	0.055	0.113	0.058	0.058	0.117	0.056
	Obtained from manual summaries	Annotated	RBF	0.025	0.075	0.024	0.046	0.101	0.020
			Polynomial	0.046	0.102	0.053	0.043	0.098	0.024
		Automatic	RBF	0.018	0.063	0.009	0.045	0.106	0,018
			Polynomial	0.038	0.087	0.021	0.044	0.099	0.028

Table 2: ROUGE and autoPan results on UAM-Titech06 and SEMsum using or not using the SVM.

4.4 Results

Table 2 shows the results obtained by the two original systems, and by their various combinations with Support Vector Machines during the initial sentence ranking step. Some observations on the results are the following: firstly, concerning the different configurations of the SVM, some trends can be found:

- SVMs trained using the set of positive examples obtained from the pyramid data consistently outperform SVMs trained using the examples obtained from the manual summaries. This may be due to the fact that the number of positive examples obtained from manual summaries (on average 12,75 per cluster) is smaller than from SCUs (on average 48,9).
- Generating automatically a set with seed negative examples for the M-C algorithm, as indicated by Yu et al. (2002), usually performs worse than using a set of seed negative examples selected manually from the SCU annotation. This may be due to the fact that its quality is better, even though the amount of seed negative examples is one order of magnitude smaller in this case (11.9 examples in average).
- The best results are obtained when using a RBF kernel, while previous summarization work (Hirao et al., 2003) uses polynomial kernels.

Concerning the two systems tested, UAM-Titech06 would be ranked in the middle zone among the participants in DUC-2005 in terms both of autoPan and ROUGE. As can be seen in Table 2, a large gain can be obtained when combined with SVM, reaching very good ROUGE results, and attaining the best autoPan result of all the systems evaluated in this paper (0.072). 0,081 is the score of the top autoPan system (Daumé III and Marcu, 2005), it also scored highest among DUC-2005 participant systems for responsiveness. UAM-Titech06's performance varies largely depending on the particular SVM used, probably due to the fact that the system just chooses the top-ranked sentences from the SVM output, so its output completely depends on the sentence rank received.

On the other hand, while SEMsum autoPan score is better when using SVM (0.069) than in the original (0.066), in terms of ROUGE, the original SEMsum (0.077, 0.136) obtains better scores. It is ranked among the best DUC-2005 participant systems. The best participant (Ye et al., 2005) has an R-2 score of 0.078 (confidence interval [0.07388 – 0.08075]) and an R-SU4 score of 0.139 (confidence interval [0.13534 – 0.14264]), when evaluated on the 20 clusters used here, which is not statistically significant with respect to SEMsum's scores. Using ROUGE measures it can not be said that the substitution of its Passage Retrieval for the SVM ranks constitute an increase in performance.

5 Conclusions and future work

The annotations created during the previous two DUC conferences provide a valuable source of information for training automatically text summarization systems using Machine Learning techniques. In this paper, we explore different possibilities for applying them in training SVMs to be used as Passage Retrieval modules in two existing systems.

The experiments have provided some insights on which can be the best way to exploit the annotations. On the one hand, the positive examples obtained from the annotations of the peer summaries are more useful than those obtained from the annotations of model summaries. That is probably due to the fact that most of the peer systems are extract-based, while the manual ones are abstract-based.

On the other hand, using a very small set of negative example seeds seems to perform better than choosing automatically the negative examples as the ones that are more different to the positive instances.

The best SVM obtained has been able to produce a very large improvement on the UAM-Titech06 system, and a slight improvement, in terms of the autoPan evaluation metric, when applied to the SEMsum summarization system.

Some open lines for future work are: a) extending the feature set that characterizes a sentence, such as including new features relating sentences with environment overlapping measures or including features from the adjacent sentences (Fisher and Roark, 2006); and b) automating the selection of the seed negatives so it is not necessary any human involvement in training the SVMs.

References

E. Alfonseca, M. Okumura, A. Moreno-Sandoval, and J. M. Guirao. 2006. Googling answers' models in question-focused summarisation. In *Proc. DUC-2006*, New York.

L. Alonso. 2005. *Representing discourse for automatic text summarization via shallow NLP techniques, PhD thesis*. Barcelona University.

J. M. Conroy, J. D. Schlesinger, D. P. O'Leary, and J. Goldstein. 2006. Back to basics: CLASSY 2006. In *Proc. DUC-2006*.

T. Copeck and S. Szpakowicz. 2005. Leveraging pyramids. In *Proc. DUC-2005*, Vancouver, Canada.

Hal Daumé III and Daniel Marcu. 2005. Bayesian summarization at DUC and a suggestion for extrinsic evaluation. In *Proc. DUC-2005*, Vancouver, Canada.

B. Dorr, D. Zajic, and R. Schwartz. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proc. HLT-NAACL 2003 Workshop on Text Summarization*.

S. Fisher and B. Roark. 2006. Query-focused summarization by supervised sentence ranking and skewed word distributions. In *Proc. DUC-2006*, New York, USA.

M. Fuentes, E. Gonzàlez, D. Ferrés, and H. Rodríguez. 2005. QASUM-TALP at DUC 2005 automatically evaluated with the pyramid based metric autopan. In *Proc. DUC-2005*.

M. Fuentes, H. Rodríguez, J. Turmo, and D. Ferrés. 2006. FEMsum at DUC 2006: Semantic-based approach integrated in a flexible eclectic multitask summarizer architecture. In *Proc. DUC-2006*, New York, USA.

B. Hachey, G. Murray, and D. Reitter. 2006. The embra system at DUC 2005: Query-oriented multi-document summarization with a very large latent semantic space. In *Proc. DUC-2005*, Vancouver, Canada.

T. Hirao, J. Suzuki, H. Isozaki, and E. Maeda. 2003. Ntt's multiple document summarization system for DUC2003. In *Proc. DUC-2003*.

K. Ishikawa, S. Ando, S. Doi, and A. Okumura. 2002. Trainable automatic text summarization using segmentation of sentence. In *Proc. 2002 NTCIR 3 TSC workshop*.

H. Kazawa, T. Hirao, and E. Maeda. 2002. Ranking SVM and its application to sentence selection. In *Proc. 2002 Workshop on Information-Based Induction Science (IBIS-2002)*.

F. Lacatusu, A. Hickl, K. Roberts, Y. Shi, J. Bensley, B. Rink, P. Wang, and L. Taylor. 2006. LCC's GISTexter at DUC 2006: Multi-strategy multi-document summarization. In *Proc. DUC-2006*, New York, USA.

C.-Y. Lin and F. J. Och. 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proc. ACL 2004*, Barcelona, Spain.

L.M. Manevitz and M. Yousef. 2001. One-class SVM for document classification. *Journal of Machine Learning Research*.

R. Mihalcea and P. Tarau. 2005. An algorithm for language independent single and multiple document summarization. In *Proc. IJCNLP, 2005*, Korea.

A. Nenkova and R. Passonneau. 2004. Evaluating content selection in summarization: The pyramid method. In *Proc. HLT/NAACL 2004*, Boston, USA.

A. Nenkova and L. Vanderwende. 2005. The impact of frequency on summarization. Technical Report MSR-TR-2005-101, Microsoft Research.

F. Schilder and B. T. McInnes. 2006. TLR at DUC 2006: approximate tree similarity and a new evaluation regime. In *Proc. DUC-2006*, New York, USA.

L. Vanderwende, H. Suzuki, and C. Brockett. 2006. Microsoft research at DUC 2006: Task-focused summarization with sentence simplification and lexical expansion. In *Proc. DUC-2006*, New York, USA.

S. Ye, L. Qiu, and T.S. Chua. 2005. NUS at DUC 2005: Understanding documents via concept links. In *Proc. DUC-2005*.

H. Yu, J. Han, and K. C-C. Chang. 2002. PEBL: Positive example-based learning for web page classification using SVM. In *Proc. ACM SIGKDD International Conference on Knowledge Discovery in Databases (KDD02)*, pages 239–248, New York. ACM Press.