Automatic Annotation Techniques for Supervised and Semi-supervised Query-focused Summarization

Shafiq R. Joty
rjoty@cs.ubc.ca

Abstract

In this paper, we study one semi-supervised and several supervised methods for extractive query-focused multi-document summarization. Traditional approaches to multi-document summarization are either unsupervised or supervised. The unsupervised approaches use heuristic rules to select the most important sentences, which are hard to generalize. On the other hand, huge amount of annotated data is a prerequisite for supervised training, the availability of which is very rare for a very new research problem like query-focused summarization. However, the availability of the abstract summaries from different evaluation framework allows us to experiment with the semi-supervised approach and the sentence alignment methods to annotate the document sentences automatically. We employ five different automatic annotation techniques to build the extracts from the human abstracts. We use TF*IDF[1] based cosine similarity, Extended String Subsequence Kernel (ESSK), Basic Element (BE) overlap, Syntactic Similarity, and Semantic Similarity measures as the annotation methods. Based on these annotations, we experiment with: a) two supervised multi-class classifiers; Support Vector Machines (SVM) and Logistic Regression (LR), b) three regression models; SVM, Bagging and Gaussian Processes (GP), and c) one sequence labeler; Conditional Random Fields (CRF). Our initial results of SVM classifier based on a very small subset of DUC-2006 and DUC-2007 data show the effectiveness of our approaches.

1 Introduction

Text summarization can be categorized along two dimensions: *abstract-based* and *extract-based*. An *extract-summary* consists of sentences extracted from the document while an *abstract-summary* may employ words and phrases that do not appear in the original document. Summaries can be *generic* or *query-focused*. A *query-focused* summary presents the information that is most relevant to the given queries, while a *generic* summary gives an overall sense of the document's content. In addition to *single* document summarization, which has been first studied in this field for years, researchers have started to work on *multi-document* summarization whose goal is to generate a summary from multiple documents. In this paper, we focus on *query-focused extractive multi-document* summarization which is now a hot research topic (E.Amigo et al., 2004) and can also be seen as a method to answer complex questions (Chali and Joty, 2008).

Unlike informationally simple factoid questions, complex questions often seek multiple different types of information simultaneously and do not presuppose that one single answer could meet all of its information needs. For example, with a factoid question like "How accurate are HIV tests?", it can be safely assumed that the submitter of the question is looking for a number or a range of numbers. However, with complex questions like "What are the causes of AIDS?", the wider focus of this question suggests that the submitter may not have a single or well-defined information need and therefore may be amenable to receiving additional supporting information that is relevant to some (as yet) undefined informational goal (Harabagiu et al., 2006). These questions require inferencing and synthesizing information from multiple documents. It is normally understood as an intellectually challenging human task, and perhaps the Google Answer service[2] is the best general purpose illustration of how it works (E.Amigo et al., 2004). In this service, users

[1] TF=Term Frequency, IDF=Inverse Document Frequency

[2] http://answers.google.com/–Google answer was launched

send complex queries which cannot be answered by just inspecting the first two or three documents returned by a search engine. Answers to such complex information needs are provided by experts who, usually, search the Internet, and assemble the most relevant pieces of information into a report, organizing the most important facts and providing additional web hyperlinks for further reading. Another popular answer service is Yahoo! Answers which is a community-driven knowledge market website launched by Yahoo!. It allows users to both submit questions to be answered and answer asked questions from other users. People vote on the best answer. The site gives members the chance to earn points as a way to encourage participation and is based on Korean Naver's Knowledge iN search[3]. As of December 2006, Yahoo! Answers had 60 million users and 65 million answers. In this paper, we focus on query-focused summarization as an automatic method to model this complex question answering task.

Traditional approaches to multi-document summarization are either unsupervised or supervised. *Unsupervised* approaches have the advantage that they do not rely on any manually annotated training data. The system TextRank (Mihalcea, 2005) can be effectively applied to the summarization of documents in different languages without any modifications for additional data. However, the unsupervised approaches use heuristic rules to select the most important sentences, which are hard to generalize (Shen et al., 2007). On the other hand, we need huge amount of annotated data for supervised learning methods, the availability of which is very rare for a very new research problem like query-focused summarization. When humans are employed in the process, producing such a large labeled corpora becomes time consuming and expensive. However, the availability of the abstract summaries from different evaluation framework like Document Understanding Conferences (DUC) allows us to experiment with two approaches: Firstly, the semi-supervised approach where the abstract summary-sentences (henceforth, abstracts) act as the labeled sentences and the document-sentences (henceforth, extracts) act as the unlabeled sentences and the task is to label the unlabeled sentences. Secondly, it also allows us to apply the sentence alignment methods to annotate the document sentences automatically which in turn allows us to experiment with the supervised approaches.

In this paper, we employ five different automatic annotation techniques to build the extracts from the human abstracts. We use: 1. TF*IDF based cosine similarity, 2. Extended String Subsequence Kernel (ESSK), 3. Basic Element (BE) overlap, 4. Syntactic Similarity, and 5. Semantic Similarity measures as the annotation methods. The first one is the bag-of-words (BOW) vector space model and does not consider the word order. ESSK considers the order by transforming the sentences into higher dimensional spaces and then measuring the similarity in that space. The BE captures shallow-syntactic representations in the sense that using BE we compare the head-modifier-rel (of dependency trees) triples of two sentences. The fourth one goes for deep syntactic similarity where we compare two syntactic trees based on the number of subtrees common in them. The shallow semantic similarity is based on the PropBank annotations of the sentences. Based on this annotation, we form a shallow semantic tree (Moschitti et al., 2007) and measure the similarity using tree kernel methods.

Based on the above mentioned annotations, we experiment with: a) two supervised multi-class classifiers; Support Vector Machines (SVM) and Logistic Regression (LR), b) three regression models; SVM, Bagging and Gaussian Processes (GP), and c) one sequence labeler; Conditional Random Fields (CRF) to produce the extract summary. Unlike the traditional *supervised* approaches (Kupiec et al., 1995)(Yeh et al., 2005), where summarization is seen as a two-class classification problem, in our approach we consider it as a five-class classification problem. Ulrich et al. (2009) shows that the regression models perform better than the classifiers in email summarization. Motivated by (Ulrich et al., 2009), we experiment with the regression models for this problem. The problem with the classifiers and the regression is that they treat the sentences individually (Shen et al., 2007). However, we observe that the individual treatment of the sentences cannot take the full advantage of the relationship between sentences. For example, intuitively, two neighboring sentences with similar contents should not be put into a summary together, but when treated individually, this information is lost (Shen et al., 2007). Sequential learning systems such as HMM have also been applied (Conroy and Oleary, 2001). The problem with HMM is that it cannot fully exploit the rich linguistic features mentioned above since they have to assume independence among the features for tractability. Shen et al. (2007) uses sequence labeler *CRF* (Lafferty et al., 2001), which can take full advantage of the inter-sentence relationship and rich features which may be dependent. Motivated from (Shen et al., 2007), we also experiment with CRF in our problem. Our initial results of SVM classifier based on a very small subset of DUC-2006 and DUC-2007 data show the effectiveness of our approaches.

The remainder of the paper is organized as follows: Section 2 describes the related work. In Section 3, we

in April 2002 and it was fully closed in December 2006, although its archives remain available.

[3]http://kin.naver.com/– The tool allows users to ask just about any question and get answers from other users. It is used by millions of Koreans. As of January 2008 the Knowledge Search database included more than 80 million pages of user-generated information.

give a brief description of the corpus. In Section 4, we discuss the semi-supervised approach, Section 5 presents the supervised approaches with sufficient literature review. Section 6 describes the automatic annotation schemes. Section 7 describes the feature space. In Section 8, we show the experimental settings and evaluation results with discussion. Finally, Section 9 concludes the paper with future directions.

2 Related Work

Many *unsupervised* methods have been developed for document summarization by exploiting different features and relationships of the sentences as we mentioned above, such as *rhetorical structures* (Marcu, 1997), *lexical chains* (Barzilay and Elhadad, 1997), *the hidden topics* in the documents (Gong and Liu, 2001) and *graphs based* on the similarity of sentences (Mihalcea, 2005).

Zhu et al. (2003) shows how to combine active learning and semi-supervised learning using Gaussian fields and harmonic functions. Motivated from (Zhu et al., 2003), we apply similar semi-supervised approach where the abstract sentences are considered as labeled and extract sentences are considered as unlabeled.

Most supervised extractive approaches (Kupiec et al., 1995), (Yeh et al., 2005) consider the summarization task as a two-class classification problem at the sentence level, where the summary sentences are positive samples while non-summary sentences are negative samples. One of the popular algorithms in supervised extractive summarization domain is Support Vector Machines (SVM). Hirao et al. (2003) show the effectiveness of their multiple document summarization system employing SVM for sentence extraction. For single document summaries, Conroy and Oleary (2001) utilized the Logistic Regression (LR) model to choose the best sentences of the document for the extract. In (Cortes and Vapnik, 1995), LR classifiers were compared alongside SVMs. The two classifier types yielded very similar results, with LR classifiers being much faster to train and thus expediting further development. In this paper we experiment with both SVM and LR as multi-class (i.e. five) classifiers.

Ulrich et al. (2009) shows that the regression models perform better than the classifiers in email summarization. Motivated by (Ulrich et al., 2009), we experiment with three regression models for this problem. The regression models we consider are: a) SVM, b) Bagging, and c) Gaussian Processes (GP).

Although such classifiers (or regression models) are effective, they assume that the sentences are independent and classify (or rank) each sentence individually without leveraging the relation among the sentences (Shen et al., 2007). Hidden Markov Model (HMM) based methods attempt to break this assumption (Conroy and Oleary, 2001). Although HMM can handle the positional dependence and feature dependence when the feature space is small by taking some special assumptions, they suffer from two problems (McCallum et al., 2000). Firstly, when the feature space is large and the features are not independent, the training becomes intractable as HMMs make strong independence assumptions between the observation variables. Secondly, this approach set the HMM parameters to maximize the likelihood of the observation sequence. By doing so, the approach fails to predict well in the test data (i.e. overfit).

In order to solve the above mentioned problems of HMM, Shen et al. (2007) use CRF which is a state-of-the-art sequence labeling method. The goal is to produce a label sequence corresponding to the sentence sequence with a label of 1 denoting the summary sentences and 0 denoting the non-summary sentences. While CRFs make similar assumptions on the dependencies among the class/state variables as HMMs do, no assumptions on the dependencies among the observation variables need to be made. LR can be considered as a linear chain CRF model of order zero and CRF is a discriminative version of HMM. That is CRF combines the merits of HMM and LR. In our work we also experiment with the CRF.

In automatic annotation arena, Banko et al. (1999) propose a method based on sentence similarity using bag-of-words (BOW) representation. Jing and McKeown (1999) propose a bigram-based similarity approach using the Hidden Markov Model. Barzilay (2003) combine edit distance and context information around sentences for annotation.

Some improvements on BOW are given by the use of dependency trees and syntactic parse trees (Hirao et al., 2004), (Punyakanok et al., 2004), (Zhang and Lee, 2003). Hirao et al. (2004) represent the sentences using Dependency Tree Path (DTP) to incorporate syntactic information. They apply Extended String Subsequence Kernel (ESSK) to measure the similarity between the DTPs of two sentences. Kouylekov and Magnini (2005) use the tree edit distance algorithms on the dependency trees of the text and the hypothesis to recognize the textual entailment. Punyakanok et al. (2004) represent the question and the sentence containing answer with their dependency trees. They add semantic information (i.e. named entity, synonyms and other related words) in the dependency trees. They apply the approximate tree matching in order to decide how similar any given pair of trees are. They also use the edit distance as the matching criteria in the approximate tree matching. (MacCartney et al., 2006) use typed dependency graphs (same as dependency trees) to represent the text and the hypothesis. Then they try to find a good partial alignment between the typed dependency graphs representing the hypothesis and the text in a search space of $O((m+1)n)$ where hypothesis graph

contains n nodes and a text graph contains m nodes. They use an incremental beam search, combined with a node ordering heuristic, to do approximate global search in the space of possible alignments.

In (Hovy et al., 2006), Basic Elements (BE) extracted from the dependency trees are used in summarization evaluation. Moschitti et al. (2007) use the tree kernel functions to measure the similarity between two syntactic (and shallow-semantic) trees.

3 Corpus

Document Understanding Conferences (DUC) is the major large scale evaluation framework for the text summarization systems. The main task of DUC2006 and DUC2007 was query-focused summarization. The task was:

"Given a complex question (topic description) and a collection of relevant documents, the task is to synthesize a fluent, well-organized 250-word summary of the documents that answers the question(s) in the topic."

For each topic DUC releases the topic description, the relevant document set (25 documents) and four human abstracts. For example one such topic description from DUC2007 is as follows:

```
<title>
steps toward introduction of the Euro
</title>
<narr>
Describe steps taken and worldwide reaction
prior to the introduction of the Euro on
January 1, 1999. Include predictions and
expectations reported in the press.
</narr>
```

DUC2006 and DUC2007 have 50 and 45 topics respectively. We use these data in our experiment.

4 Semi-Supervised Learning

In this section we describe briefly the semi-supervised approach following (Zhu et al., 2003). We have N feature vectors $x \in R^d$, some (abstract or summary sentences) of which have labels $y^l = 1$. The rest (i.e. extract or document sentences) have unknown labels y^u. The goal is to compute the unknown labels.

We form a graph where the sentences are nodes and the edges have weights corresponding to a similarity kernel. In our case, the weight between points x_i and x_j is:

$$w_{ij} = exp\left(-\frac{1}{\sigma}\left\|x_i - x_j\right\|^2\right)$$

In learning we want to to minimize the following error function to compute the unknown labels y^u:

$$E(y^u) = \frac{1}{2}\left(\sum_{i\in L, j\in L} w_{ij}(y_i^l - y_j^l)^2\right) + \frac{1}{2}\left(2\sum_{i\in U, j\in L} w_{ij}(y_i^u - y_j^l)^2 + \sum_{i\in U, j\in U} w_{ij}(y_i^u - y_j^u)^2\right)$$

where L is the set of labeled points and U is the set of unlabeled points. If two points are close then w will be large. Hence, the only way to minimize the error function is to make these two nearby points have the same label y. We introduce the weight matrix W with entries w_{ij} and $D = diag(d_i)$ where $d_i = \sum_j w_{ij}$. That is, D is a diagonal matrix whose $i - th$ diagonal entry is the sum of the entries of row i of W. Let the vector y_u contain all the unknown labels y^u and y_l contain all the labels y^l. Then, the error function can be written in matrix notation as follows:

$$E(y_u) = y_u^T(D_{uu} - W_{uu})y_u - 2y_l^T W_{lu}y_u + y_l^T(D_{ll} - W_{ll})y_l$$

where W_{uu} are the entries w_{ij} with $i, j \in U$. Differentiating this equation and equating to zero, gives us our solution in terms of a linear system of equations:

$$(D_{uu} - W_{uu})y_u = W_{ul}y_l$$

where, $0 \leq y_u \leq 1$. We can write this equation as the following recursive equation:

$$y_u^{t+1} = D_{uu}^{-1}\left[W_{uu}y_u^t + W_{ul}y_l\right]$$

In absence of any labeled data, the above equation gives us the *PageRank* algorithm and in absence of any unlabeled data the equation gives us the similar equation as the GP. Intuitively, the equation combines both the PageRank (for unlabeled part) and GP (for the labeled part) into a single measure.

5 Supervised Learning

In this section, we give a brief description of the supervised approaches.

5.1 Multi-class Classifiers

In this case, we treat summarization as a multi-class classification problem. We have five classes: 1. very strong, 2. strong, 3. neutral, 4. weak, and 5. very weak. SVM is a powerful methodology for solving machine learning problems introduced by Vapnik (Cortes and Vapnik, 1995) based on the Structural Risk Minimization principle. In our initial experiment we used $SVM^{multiclass}$ [4] with linear kernel and the default value of the regularizer C (0.01).

[4] http://svmlight.joachims.org/svm_multiclass.html

Logistic Regression (LR) models are proved to be useful in many applications of natural language processing. LR is a discriminative model that estimates $P(y|x)$ directly, where y is the class label and x is the given input. The main intuition of the LR model is to first map x to a real number, such that very positive number means x is likely to be positive $(y = 1)$, and very negative number means x is negative $(y = -1)$ which can be done using a linear mapping. In case of LR, this type of mapping is done via a *logistic sigmoid* function that squashes the range down to $[0, 1]$ so that one can interpret it as probability. We used the LIBLINEAR[5] package as the multiclass classifier.

5.2 Regression Models

Ulrich et al. (2009) shows that the regression models perform better than the classifiers in email summarization. Motivated by (Ulrich et al., 2009), we experiment with three regression models for this problem. The regression models we consider are: a) SVM, b) Bagging, and c) Gaussian Processes (GP). We used the WEKA (Witten et al., 1999) implementation of the different regression models.

5.3 Sequence Labeler

Shen et al. (2007) uses sequence labeler *CRF* (Lafferty et al., 2001) for single-document summarization, which can take full advantage of the inter-sentence relationship and rich features which may be dependent. Motivated from (Shen et al., 2007), we also experiment with CRF in our problem. We used the MALLET-0.4 NLP toolkit (McCallum, 2002) in our experiments.

For the classifies and sequence labeler we need categorical annotation and for the regression model we need continuous annotation. In the next section we will see how we get the required annotations automatically using the abstract summaries.

6 Automatic Annotation Techniques

In this section we describe the automatic annotation techniques in detail. As described in Section3, for each topic we have 4 abstract summaries and extract (i.e. document) sentences that come from the 25 relevant documents. We want to annotate the extract sentences using the abstract sentences. The main idea is: to annotate an extract sentence, we measure the similarity between the abstract sentences and the extract sentence. The hypothesis is that the document sentence which is very similar to the abstract sentences should be in the extract summary. In the following, Section 6.1 describes five different similarity measures and Section 6.2 describes the annotation scheme for the supervised approaches.

[5]http://www.csie.ntu.edu.tw/c̃jlin/liblinear/

6.1 Similarity Measures

In this section we describe five different similarity measures. Given two sentences, each of the measures gives us the corresponding similarity.

6.1.1 BOW model: TF*IDF based Cosine Similarity

Our first model is the simple BOW model. We represent the abstract sentence as well as the extract sentence as the vectors of their words' TF*IDF values. We measure the similarity between the two vectors by calculating the cosine of the angle between the vectors.

6.1.2 Extended String Subsequence Kernel

The ESSK is a simple extension of the Word Sequence Kernel (WSK) (Cancedda et al., 2003) and String Subsequence Kernel (SSK) (Lodhi et al., 2002). We calculate the similarity score Sim(T_i, U_j) using ESSK where T_i denotes abstract sentence and U_j stands for document sentence. Formally, ESSK is defined as follows (Hirao et al., 2004):

$$K_{essk}(T,U) = \sum_{m=1}^{d} \sum_{t_i \in T} \sum_{u_j \in U} K_m(t_i, u_j)$$

$$K_m(t_i,u_j) = \begin{cases} val(t_i,u_j) & \text{if } m = 1 \\ K^{'}_{m-1}(t_i,u_j) \cdot val(t_i,u_j) \end{cases}$$

Here, $K^{'}_{m}(t_i,u_j)$ is defined below. t_i and u_j are the nodes of T and U, respectively. Each node includes a word and its disambiguated sense found using the WSD (Word Sense Disambiguation) System of Chali and Joty (2007). The function $val(t,u)$ returns the number of attributes common to the given nodes t and u.

$$K^{'}_{m}(t_i,u_j) = \begin{cases} 0 & \text{if } j = 1 \\ \lambda K^{'}_{m}(t_i,u_{j-1}) + K^{''}_{m}(t_i,u_{j-1}) \end{cases}$$

Here λ is the decay parameter for the number of skipped words. We chose $\lambda = 0.5$ for this research. $K^{''}_{m}(t_i,u_j)$ is defined as:

$$K^{''}_{m}(t_i,u_j) = \begin{cases} 0 & \text{if } i = 1 \\ \lambda K^{''}_{m}(t_{i-1},u_j) + K_m(t_{i-1},u_j) \end{cases}$$

Finally, the similarity measure is defined after normalization as below:

$$sim_{essk}(T,U) = \frac{K_{essk}(T,U)}{\sqrt{K_{essk}(T,T)K_{essk}(U,U)}}$$

6.1.3 BE based Shallow Syntactic Similarity

We extracted BEs, the "head-modifier-relation" triples for the abstract sentences and the extract sentences using the BE package distributed by ISI [6]. The triples encode some syntactic/semantic information and one can quite easily decide whether any two units match or not- considerably more easily than with longer units (Hovy et al., 2005). We measure the similarity by counting the number of common BEs divided by the number of BEs in the extract sentence.

6.1.4 Tree Kernel based Syntactic Similarity

In order to calculate the syntactic similarity between the abstract sentence and the document sentence, we first parse the corresponding sentences into syntactic trees using Charniak parser [7] (Charniak, 1999) and then we calculate the similarity between the two trees using the *tree kernel* (Collins and Duffy, 2001). The tree kernel of two syntactic trees T_1 and T_2 is actually the inner product of the two m-dimensional vectors, $v(T_1)$ and $v(T_2)$:

$$TK(T_1, T_2) = v(T_1).v(T_2)$$

We define the indicator function $I_i(n)$ to be 1 if the sub-tree i is seen rooted at node n and 0 otherwise. It follows:

$$v_i(T_1) = \sum_{n_1 \in N_1} I_i(n_1), v_i(T_2) = \sum_{n_2 \in N_2} I_i(n_2)$$

Where N_1 and N_2 are the set of nodes in T_1 and T_2 respectively. So, we can derive:

$$\begin{aligned} TK(T_1, T_2) &= v(T_1).v(T_2) = \sum_i v_i(T_1) v_i(T_2) \\ &= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} \sum_i I_i(n_1) I_i(n_2) \\ &= \sum_{n_1 \in N_1} \sum_{n_2 \in N_2} C(n_1, n_2) \end{aligned}$$

where we define $C(n_1, n_2) = \sum_i I_i(n_1) I_i(n_2)$. Next, we note that $C(n_1, n_2)$ can be computed in polynomial time, due to the following recursive definition:

1. If the productions at n_1 and n_2 are different then $C(n_1, n_2) = 0$

2. If the productions at n_1 and n_2 are the same, and n_1 and n_2 are pre-terminals, then $C(n_1, n_2) = 1$

[6] BE website:http://www.isi.edu/ cyl/BE

[7] available at ftp://ftp.cs.brown.edu/pub/nlparser/

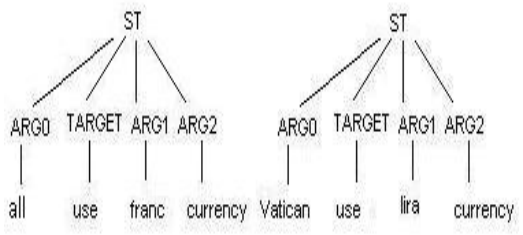

Figure 1: Example of semantic trees

3. Else if the productions at n_1 and n_2 are not pre-terminals,

$$C(n_1, n_2) = \prod_{j=1}^{nc(n_1)} (1 + C(ch(n_1, j), ch(n_2, j))) \quad (1)$$

where, $nc(n_1)$ is the number of children of n_1 in the tree; because the productions at n_1 and n_2 are the same, we have $nc(n_1) = nc(n_2)$. The i-th child-node of n_1 is $ch(n_1, i)$. Note that, the tree kernel (TK) function computes the number of common subtrees between two trees. Such subtrees are subject to the constraint that their nodes are taken with all or none of the children they have in the original tree.

6.1.5 Tree Kernel based Shallow-Semantic Similarity

Shallow semantic representations, bearing a more compact information, can prevent the sparseness of deep structural approaches and the weakness of BOW models (Moschitti et al., 2007). Initiatives such as PropBank (PB) (Kingsbury and Palmer, 2002) have made possible the design of accurate automatic Semantic Role Labeling (SRL) systems like ASSERT (Hacioglu et al., 2003). For example, consider the PB annotation:

[ARG0 all][TARGET use][ARG1 the french franc][ARG2 as their currency]

Such annotation can be used to design a shallow semantic representation that can be matched against other semantically similar sentences, e.g.

[ARG0 the Vatican][TARGET use][ARG1 the Italian lira][ARG2 as their currency]

In order to calculate the semantic similarity between the sentences, we first represent the annotated sentence using the tree structures like Figure 1 which we call Semantic Tree (ST). In the semantic tree, arguments are replaced with the most important word-often referred to as the semantic head.

The sentences may contain one or more subordinate clauses. For example the sentence, "the Vatican, located wholly within Italy uses the Italian lira as their currency."

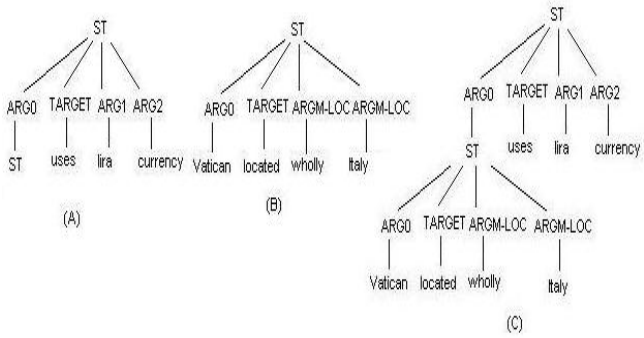

Figure 2: Two STs composing a STN

gives the STs as in Figure 2. As we can see in Figure 2(A), when an argument node corresponds to an entire subordinate clause, we label its leaf with ST, e.g. the leaf of ARG0. Such ST node is actually the root of the subordinate clause in Figure 2(B). If taken separately, such STs do not express the whole meaning of the sentence, hence it is more accurate to define a single structure encoding the dependency between the two predicates as in Figure 2(C). We refer to this kind of nested STs as STNs.

Note that, the tree kernel (TK) function defined in Section 6.1.4, computes the number of common subtrees between two trees. Such subtrees are subject to the constraint that their nodes are taken with all or none of the children they have in the original tree. Though, this definition of subtrees makes the TK function appropriate for syntactic trees but at the same time makes it not well suited for the semantic trees (ST) defined above. For instance, although the two STs of Figure 1 share most of the subtrees rooted in the *ST* node, the kernel defined above computes only one match (ST ARG0 TARGET ARG1 ARG2) which is not useful.

The critical aspect of the TK function is that the productions of two evaluated nodes have to be identical to allow the match of further descendants. This means that common substructures cannot be composed by a node with only some of its children as an effective ST representation would require. Moschitti et al. (2007) solve this problem by designing the Shallow Semantic Tree Kernel (SSTK) which allows to match portions of a ST. We followed the similar approach to compute the SSTK.

6.2 Annotation Techniques

For each topic, DUC provides 4 abstract summaries. For each extract sentence (e_k) in the document we measure the score using one of the above mentioned similarity measures by the following formula:

$score(e_k) = \forall_i Avg(\forall_j MaxSim(e_k, a_{ij}))\ \ i = 1, 2, 3, 4$

Here, e_k is the k-th extract sentence in the document. a_{ij} means j-th abstract sentence of i-th abstractor. $MaxSim$ is the maximum of the similarity measures. That means, for each extract sentence we measure the similarity with each of the abstract sentences of one human. We take the maximum of these scores as the score of the extract sentence for this human abstractor. In this way, for a extract sentence, we measure the scores for all the 4 abstractors (4 maxs). Then we take the average of these 4 max scores as the final score of this extract sentence.

Once we get the scores for all of the extract sentences, we use these scores as labels for regression. For classification or sequence labeling, we divide the sentences into 5 classes, where the top class (having highest scores) represents "very strong", second class represents "strong", third class represents "neutral", fourth class represents "weak" and fifth class represents "very weak".

7 Feature Space

The performance of the supervised approaches depends entirely on the feature set used. We divide the features into two major criteria: 1. Basic Features; the features which declare the importance of a sentence in a document and 2. Query Overlap; the features which measure the similarity between each sentence and the user query.

7.1 Basic Features

For each document sentence we extract the following basic features:

- Position of the sentence in Document (LOCAL_POS).
- Position of the sentence in the document cluster (GLOBAL_POS).
- Length of the sentence excluding the stopwords, normalized by the longest sentence in the document (LOCAL_SLEN).
- Length of the sentence excluding the stopwords, normalized by the longest sentence in the document cluster(GLOBAL_SLEN).
- Number of named entity (NE); we used the OAK (Sekine, 2002) named entity tagger to extract the named entities.
- Cue word presence (CUE); The probable relevance of a sentence is affected by the presence of pragmatic words such as "significant", "impossible", "In conclusion", "Finally" etc. We used a cue word list of 228 words. We gave the score 1 to a sentence having any of the cue words and 0 otherwise.
- Heading overlap (HEADING); The number of words common between the sentence and the heading of the news article.

- Clue Word Similarity (CLUE); Following (Murray and Carenini, 2008), we calculate this feature as a rough approximation of the ClueWordScore (CWS). For each sentence we remove stopwords and count the number of words that occur in other sentences besides the current sentence. The CWS is therefore a measure of cohesion.

Following (Murray and Carenini, 2008), for each unique word, we calculate two conditional probability. For each document, we calculate the probability of the document given the word, estimating the probability from the actual term counts, and take the maximum of these conditional probabilities as our first term score, which we call *Dprob*. In the same way, for each newspaper (for example New York Times), we calculate the probability of the newspaper given the word, which we call *NPProb*. After calculating the word level probabilities we calculate the following sentence level features according to (Murray and Carenini, 2008):

- MaxD and MaxNP; Maximum of the term scores for the words in the sentence using two different probabilities; DProb and NPProb.

- MinD and MinNP; Minimum of the term scores for the words in the sentence using two different probabilities; DProb and NPProb.

- SumD and SumNP; Sum of the term scores for the words in the sentence using two different probabilities; DProb and NPProb.

- COS1, COS2; Using a vector representation, we calculate the cosine between the preceding 3 sentences of the given sentence and the 3 sentences subsequent to the sentence, first using DProb as the vector weights (COS1) and then using NPProb as the vector weights (COS2).

- CENT1, CENT2; We similarly calculate two scores measuring the cosine between the current sentence and the rest of the sentences in the document, using each term-weight metric as vector weights (CENT1 for Dprob and CENT2 for NPProb).

7.2 Query Overlap Features

As we want to extract sentences that are not only important but also relevant to the query, we measure the following query overlap features:

- N-gram overlap (NGRAM); This is the recall between the query and the candidate sentence where n stands for the length of the $n-gram$ $(n = 1, 2, 3, 4)$.

- LCS, WLCS; Given two sequences S_1 and S_2, the longest common subsequence (LCS) of S_1 and S_2 is a common subsequence with maximum length. Weighted Longest Common Subsequence (WLCS) improves the basic LCS method to remember the length of consecutive matches encountered so far (Lin, 2004). We computed the LCS and WLCS-based F-measure between a query and a sentence following (Lin, 2004)s.

- Synonym overlap (SYN); The number of synonyms of the words of the document sentence is common with the query words.

- Hyponym/Hypernym (HYP); The number of hypernyms/hyponyms of the words of the document sentence is common with the query words.

- BE overlap (BE); The number of BEs common between the document sentence and the query.

- Syntactic overlap; We parse the document sentence and the query using the Charniak parser and measure the similarity using the tree kernel as described in Section 6.1.4.

8 Evaluations

For the class project, we performed a small experiment by annotating 20 document sets (10 from DUC'06 + 10 from DUC'07) using the ESSK and the proposed annotation scheme. We implemented 15 of the features mentioned above. We experiment with the SVM multiclass classifier by producing summaries of length 250 words for 4 topics (2 from DUC'06 and 2 from DUC'07).

We carried out a small automatic evaluation of our summaries using ROUGE (Lin, 2004) toolkit, which has been widely adopted by DUC for automatic summarization evaluation. It measures summary quality by counting overlapping units such as the n-grams (ROUGE-N), word sequences (ROUGE-L and ROUGE-W) and word pairs (ROUGE-S and ROUGE-SU) between the candidate summary and the reference/abstract summary. ROUGE parameters were set as the same as DUC 2007 evaluation setup.

Figure 3 shows different ROUGE scores of the four summaries generated by SVM multi-class classifier using ESSK annotations in the training. In this small experiment we did not perform any cross validation to learn the value of the regularizer (i.e. C is set to its default value 0.01). We have just included 15 simple features. We looked at the results of other supervised approaches in DUC'07 and our results based on this mini feature set and small experiments seems to be promising.

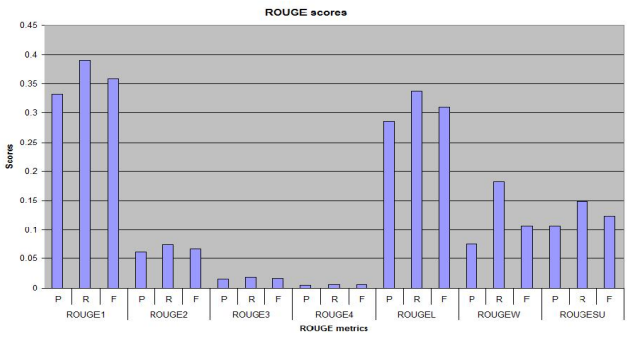

Figure 3: ROUGE Scores of four summaries generated by SVM classifier. P=Precision, R=Recall, F=F-measure

9 Conclusions and Future Directions

The main goal of this paper is to show the impact of different automatic annotation methods on the performance of different supervised machine learning techniques in confronting complex question answering problem. We propose five different automatic annotation techniques to automatically annotate the document sentences using the abstract sentences. We also propose to use one semi-supervised, 2 supervised classifiers (SVM,LR), 3 regression models (SVM, Bagging, GP), and 1 sequence labeler (CRF) to produce the summary.

However, for the class project we have just shown a small subset of the main proposed system. We annotated 20 document set using the ESSK based annotation scheme. We produced four summaries using SVM multi-class classifier including only 15 features.

In future, we would like to complete the following remaining parts:

- Implement all of the annotation techniques and annotate the document sentences using these techniques,
- Use all of the proposed supervised techniques and experiment with their performance,
- Compare the performance of different automatic annotation techniques,
- Implement the semi-supervised approach and see how it performs for this problem, and
- Implement and include all of the proposed features and compare which features are important for this problem.

References

Michele Banko, Vibhu Mittal, Mark Kantrowitz, and Jade Goldstein. 1999. Generating extraction-based summaries from hand-written summaries by aligning text spans. In *Proceedings of the Pacific Association for Computational Linguistics*. PACLING.

R. Barzilay and M. Elhadad. 1997. Using Lexical Chains for Text Summarization. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and the 8th European Chapter Meeting of the Association for Computational Linguistics, Workshop on Intelligent Scalable Test Summarization*, pages 10–17, Madrid.

Regina Barzilay. 2003. Sentence alignment for monolingual comparable corpora. In *Proceedings of EMNLP*, pages 25–32.

Nicola Cancedda, Eric Gaussier, Cyril Goutte, and Jean M. Renders. 2003. Word sequence kernels. *Journal of Machine Learning Research*, 3:1059–1082.

Yllias Chali and Shafiq R. Joty. 2007. Word Sense Disambiguation Using Lexical Cohesion. In *Proceedings of the 4th International Conference on Semantic Evaluations*, pages 476–479, Prague. ACL.

Yllias Chali and Shafiq R. Joty. 2008. Selecting sentences for answering complex questions. In *Proceedings of EMNLP*, Honolulu, Hawaii.

Eugene Charniak. 1999. A Maximum-Entropy-Inspired Parser. In *Technical Report CS-99-12*, Brown University, Computer Science Department.

Michael Collins and Nigel Duffy. 2001. Convolution Kernels for Natural Language. In *Proceedings of Neural Information Processing Systems*, pages 625–632, Vancouver, Canada.

John M. Conroy and Dianne P. Oleary. 2001. Text summarization via hidden markov models. In *SIGIR*, pages 406–407.

C. Cortes and V. N. Vapnik. 1995. Support Vector Networks. *Machine Learning*, 20:273–297.

E.Amigo, J.Gonzalo, V.Peinado, A.Peinado, and F.Verdejo. 2004. An Empirical Study of Information Synthesis Tasks. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 207–es, Barcelona, Spain.

Yihong Gong and Xin Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *SIGIR*, pages 19–25.

K. Hacioglu, S. Pradhan, W. Ward, J. H. Martin, and D. Jurafsky. 2003. Shallow Semantic Parsing Using Support Vector Machines. In *Technical Report TR-CSLR-2003-03*, University of Colorado.

S. Harabagiu, F. Lacatusu, and A. Hickl. 2006. Answering complex questions with random walk models. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 220 – 227. ACM.

Tsutomu Hirao, Jun Suzuki, Hideki Isozaki, and Eisaku Maeda. 2003. NTT's multiple document summarization system for DUC2003. In *Proceedings of the Document Understanding Conference*.

Tsutomu Hirao, Jun Suzuki, Hideki Isozaki, and Eisaku Maeda. 2004. Dependency-based sentence alignment for multiple document summarization. In *Proceedings of the 20th International Conference on Computational Linguistics*, pages 446–452.

Eduard Hovy, Chin-Yew Lin, and Liang Zhou. 2005. A BE-based Multi-document Summarizer with Query Interpretation. In *Proceedings of the Document Understanding Conference*, Vancouver, B.C. Canada.

E. Hovy, C. Y. Lin, L. Zhou, and J. Fukumoto. 2006. Automated Summarization Evaluation with Basic Elements. In *Proceedings of the Fifth Conference on Language Resources and Evaluation*, Genoa, Italy.

Hongyan Jing and Kathleen R. McKeown. 1999. The decomposition of human-written summary sentences. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 129–136, New York, NY, USA. ACM.

P. Kingsbury and M. Palmer. 2002. From Treebank to PropBank. In *Proceedings of the international conference on Language Resources and Evaluation*, Las Palmas, Spain.

M. Kouylekov and B. Magnini. 2005. Recognizing textual entailment with tree edit distance algorithms. In *Proceedings of the PASCAL Challenges Workshop: Recognising Textual Entailment Challenge*.

J. Kupiec, J. Pedersen, and F. Chen. 1995. A trainable document summarizer. pages 68–73.

John Lafferty, Andrew K. McCallum, and Fernando Pereira. 2001. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*.

Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of Association for Computational Linguistics*, pages 74–81, Barcelona, Spain.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444.

B. MacCartney, T. Grenager, M.C. de Marneffe, D. Cer, and C. D. Manning. 2006. Learning to recognize features of valid textual entailments. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*, page 4148, New York, USA.

D. Marcu. 1997. From discourse structures to text summaries. In *ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*, pages 82–88.

Andrew K. McCallum, Dayne Freitag, and Fernando Pereira. 2000. Maximum entropy markov models for information extraction and segmentation. In *ICML*, pages 591–598.

Andrew K. McCallum. 2002. MALLET: A Machine Learning for Language Toolkit.

R. Mihalcea. 2005. Language Independent extractive summarization. In *AAAI*, pages 1688–1689.

Alessandro Moschitti, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. 2007. Exploiting Syntactic and Shallow Semantic Kernels for Question/Answer Classificaion. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 776–783, Prague, Czech Republic. ACL.

Gabriel Murray and Giuseppe Carenini. 2008. Summarizing Spoken and Written Conversations. In *Proceedings of EMNLP*, Honolulu, Hawaii.

V. Punyakanok, D. Roth, and W. Yih. 2004. Mapping dependencies trees: An application to question answering. In *Proceedings of AI & Math*, Florida, USA.

Satoshi Sekine. 2002. Proteus Project OAK System (English Sentence Analyzer), http://nlp.nyu.edu/oak.

Dou Shen, Jian-Tao Sun, Hua Li, Qiang Yang, and Zheng Chen. 2007. Document Summarization Using Conditional Random Fields. In *IJCAI*, pages 2862–2867.

Jan Ulrich, Giuseppe Carenini, Gabriel Murray, and Raymond Ng. 2009. Regression Based Summarization of Email Conversations (Submitted). In *Proceedings of WWW'09*, Madrid.

I. Witten, E. Frank, L. Trigg, M. Hall, G. Holmes, and S. Cunningham, 1999. *Weka: Practical machine learning tools and techniques with java implementations*. ICONIP/ANZIIS/ANNES.

Jen-Yuan Yeh, Hao-Ren Ke, Wei-Pang Yang, and I-Heng Meng. 2005. Text summarization using a trainable summarizer and latent semantic analysis. In *IPM*, volume 41(1), pages 75–95.

D. Zhang and W. S. Lee. 2003. A Language Modeling Approach to Passage Question Answering. In *Proceedings of the Twelfth Text REtreival Conference*, pages 489–495, Gaithersburg, Maryland.

X Zhu, J Lafferty, and Z Ghahramani. 2003. Combining active learning and semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings ICML*, Washington, USA.