Unsupervised Approach for Selecting Sentences in Query-based Summarization

Yllias Chali and Shafiq R. Joty
Department of Mathematics and Computer Science
University of Lethbridge
4401 University Drive
Lethbridge, Alberta, Canada, T1K 3M4
chali,jotys@cs.uleth.ca

Abstract

When a user is served with a ranked list of relevant documents by the standard document search engines, his search task is usually not over. He has to go through the entire document contents to judge its relevance and to find the precise piece of information he was looking for. *Query-relevant summarization* tries to remove the onus on the end-user by providing more condensed and direct access to relevant information.

Query-relevant summarization is the task to synthesize a fluent, well-organized summary of the document collection that answers the user questions. We extracted several features of different types (i.e. lexical, lexical semantic, statistical and cosine similarity) for each of the sentences in the document collection in order to measure its relevancy to the user query. We experimented with two well-known unsupervised statistical machine learning techniques: K-Means and EM algorithms and evaluated their performances. For all these methods of generating summaries, we have shown the effects of different kinds of features.

Introduction

Question Answering (QA) is retrieving answers to natural language questions from a collection of documents rather than retrieving relevant documents containing the keywords of the query. After having made substantial headway in factoid and list questions (such as "Who won the nobel prize in peace in 2006?" or "Name the books written by Dr. Muhammad Yunus"), researchers have turned their attention to more complex information needs that cannot be answered by simply extracting named entities (persons, organization, locations, dates, etc.) from documents. For example, the questions : *"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."* require inferencing and synthesizing information from multiple documents, which in computation linguistics we call query-based multi-document summarization. The "definition" and "other" questions in the TREC-QA track and query-relevant summarization task of DUC [1] exemplify this shift to more complex information needs. Our paper deals with this research problem in the context of DUC 2007: "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".

In this paper, we extensively study two unsupervised learning techniques: EM and K-means for this particular problem. We extract several features of type: lexical, lexical semantic, statistical and cosine similarity for each of the sentences in the document collection to measure its relevancy to the user query. We then used the soft clustering algorithm EM and hard clustering algorithm K-means to rank the sentences and generate summaries accordingly. For each of these algorithms we have shown the effects of different kinds of features.

Although we concentrate on query relevant summarization task in this paper, the methods should also be useful in applications that share similar problems and structures. This paper is organized as follows: the following section describes how the features are extracted, in the next section we discuss the learning issues and presents our learning approach, section after that describes our experimental study. We conclude and discuss future directions in the last section.

[1] Document Understanding Conference

Feature Extraction

The sentences in the document collection and topic narration are analyzed in various levels and each of the document-sentences is represented as a vector of feature-values. We use several types of features and investigate below their contribution to generate quality summary. The features can be divided into several categories:

Lexical Features

N-gram Overlap With the view to measure the N-gram (N=1,2,3,4) overlap scores, a *Query Pool* and a *Sentence Pool* are created. In order to create the Query Pool, we took the query sentences, for each query sentence we created a set of related sentences by replacing an important word (i.e. noun, verb, adverb and adjective) by its synonym(s). We created a sentence pool for each of the document-sentences in the same way. We measure the recall based n-gram scores using the following formula:

$$N-gram(S,Q)=\frac{\sum_{gram_n\in S}Count_{match}(gram_n)}{\sum_{gram_n\in S}Count(gram_n)}$$
$$n-gramscore=argmax_i(argmax_j\ N-gram(s_i,q_j))$$

Where, n stands for the length of the $n-gram$ $(n=1,2,3,4)$ $and\ Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in the query and candidate sentence. q_j is the j^{th} sentence in the query pool and s_i is the i^{th} sentence in the sentence pool.

Longest Common Subsequence and Weighted Longest Common Subsequence A sequence $W=[w_1,w_2,...,w_n]$ is a subsequence of another sequence $X=[x_1,x_2,...,x_m]$, if there exists a strict increasing sequence $[i_1,i_2,...,i_k]$ of indices of X such that for all $j=1,2,...,k$ we have $x_{ij}=w_j$ (Cormen, Leiserson, & Rivest 1989). 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.

The longer the LCS of two sentences is, the more similar the two sentences are. Following (Lin 2004), we used LCS-based F-measure to estimate the similarity between the document sentence S of length m and the query sentence Q of length n as follows:

$$R_{lcs}(S,Q)=\frac{LCS(S,Q)}{m}$$
$$P_{lcs}(S,Q)=\frac{LCS(S,Q)}{n}$$
$$F_{lcs}(S,Q)=(1-\alpha)\times P_{lcs}+\alpha\times R_{lcs}$$

Where, LCS(S,Q) is the length of a longest common subsequence of S and Q and α is a constant that determines the importance of precision and recall. In order to measure LCS score for a sentence we took a similar approach as the previous section (i.e. sentence pool and query pool). We calculated the LCS score using the following formula:

$$LCS\ score=argmax_i(argmax_j\ F_{lcs}(s_i,q_j))$$

Where, q_j is the j^{th} sentence in the query pool and s_i is the i^{th} sentence in the sentence pool.

The basic LCS has a problem that it does not differentiate LCSes of different spatial relations within their embedding sequences (Lin 2004). To improve the basic LCS method, we can remember the length of consecutive matches encountered so far to a regular two dimensional dynamic program table computing LCS. We call this weighted LCS (WLCS) and use k to indicate the length of the current consecutive matches ending at words x_i and y_j. Given two sentences X and Y, the WLCS score of X and Y can be computed using the similar dynamic programming procedure as stated in (Lin 2004). We computed the WLCS-based F-measure in the same way as previous section using both the question pool and sentence pool.

$$WLCS\ score=argmax_i(argmax_j\ F_{wlcs}(s_i,q_j))$$

Skip-Bigram Measure Skip-bigram is any pair of words in their sentence order, allowing for arbitrary gaps. Skip-bigram measures the overlap of skip-bigrams between a candidate sentence and a query sentence. Following (Lin 2004), the skip bi-gram score between the document sentence S of length m and the query sentence Q of length n can be computed as follows:

$$R_{skip_2}(S,Q)=\frac{SKIP_2(S,Q)}{C(m,2)} \quad (1)$$
$$P_{skip_2}(S,Q)=\frac{SKIP_2(S,Q)}{C(n,2)} \quad (2)$$
$$F_{skip_2}(S,Q)=(1-\alpha)\times P_{skip_2}+\alpha\times R_{skip_2} \quad (3)$$

Where, $SKIP_2(S,Q)$ is the number of skip bi-gram matches between S and Q and α is a constant that determines the importance of precision and recall. C is the combination function. We call the Equation 3, skip bigram-based F-measure. We computed the skip bigram-based F-measure as follows:

$$Skip-bi-gramscore=argmax_i(argmax_j\ F_{skip2}(s_i,q_j))$$

Head and Head Related-words Overlap The number of heads common in between two sentences can indicate how much they are relevant to each other. In order to extract the heads from both query and sentence, the query and the sentence are parsed by Minipar [2]. From the parse trees we extract the heads and measure the overlap between them as follows:

$$Exact\ Head\ Score=\frac{\sum_{w_1\in HeadSet}Count_{match}(w_1)}{\sum_{w_1\in HeadSet}Count(w_1)}$$

Where HeadSet is the set of head words in the sentence and $Count_{match}$ is the number of matches between the HeadSet of query and sentence.

Again, we take the synonyms, hyponyms and hypernyms of both the query-head words and sentence-head words and measure the overlap similarly using the formula:

$$Head\ Related\ Score=\frac{\sum_{w_1\in HeadRelSet}Count_{match}(w_1)}{\sum_{w_1\in HeadRelSet}Count(w_1)}$$

Where, HeadRelSet is the set of synonyms, hyponyms and hypernyms of head words in the sentence and $Count_{match}$ is the number of matches between the head related set of query and sentence.

Basic Element Overlap Measure We extracted Basic Elements (BEs) for the sentences in the document collection by using BE package distributed by ISI[3]. We used the standard BE-F breaker included in the BE package.

Once we get the BEs for a sentence, we computed the likelihood ratio (LR) for each BE following (Zhou, Lin, & Hovy 2005). The LR score of each BE is an information therotic measure that represents the relative importance in the BE list from the document set that contains all the texts

[2]Available at http://www.cs.ualberta.ca/ lindek/minipar.htm
[3]BE website:http://www.isi.edu/ cyl/BE

to be summarized. Sorting BEs according to their LR scores produced a BE-ranked list. Our goal is to generate a summary that will answer the questions of a topic narrative. The ranked list of BEs in this way contains important BEs at the top which may or may not be relevant to the topic questions. We filter those BEs by checking whether they contain any word which is a *query word* or a *query related word*(i.e. synonyms, hypernyms, hyponyms and gloss words). The score of a sentence is the sum of its BE scores divided by the number of BEs in the sentence.

Lexical Semantic Features

We form a set of words which we call *query related words* by taking the important words (i.e. nouns, adjectives, verbs and adverbs) from the query, their synonyms, hypernyms/hyponyms and important words from their gloss definitions. *Synonym overlap* measure is the overlap between the list of synonyms of the important words extracted from the candidate sentence and *query related words*. Hypernym/hyponym overlap measure is the overlap between the list of hypernyms and hyponyms of the nouns extracted from the sentence in consideration and *query related words* and gloss overlap measure is the overlap between the list of important words that are extracted from gloss definition of nouns in the sentence in consideration and *query related words*.

Statistical Similarity Measures

Statistical similarity measures are based on the co-occurance of similar words in a corpus. We have used two statistical similarity measures:

Dependency-based similarity measure and Proximity-based similarity measure Dependency-based similarity measure uses the dependency relations among words in order to measure the similarity (Lin 1998b). It extracts the dependency triples then uses statistical approach to measure the similarity. Proximity-based similarity measure is computed based on the linear proximity relationship between words only (Lin 1998a). It uses the information theoretic definition of similarity to measure the similarity.

We used the thesaurus provided by Dekang Lin[4]. Using the data, one can retrieve most similar words for a given word. The similar words are grouped into clusters. Note that, for a word there can be more than one cluster. Each cluster represents the sense of the word and its similar words for that sense.

For each query-word, we extract all of its clusters from the thesaurus. Now, in order to determine the right cluster for a query word, we measure the overlap score between the *query related words* (i.e. exact words, synonyms, hypernyms/hyponyms and gloss) and the *clusters*. The hypothesis is that, the cluster that has more words common with the query related words is the right cluster. We chose the cluster for a word which has the highest overlap score.

Once we get the clusters for the query words, we measured the overlap between the cluster words and the sentence words:

$$Measure = \frac{\sum_{w_1 \in SenWords} Count_{match}(w_1)}{\sum_{w_1 \in SenWords} Count(w_1)}$$

Where, SenWords is the set of words for the sentence and $Count_{match}$ is the number of matches between the Sentence Words and the cluster of similar words.

Graph-based Similarity Measure

In (Erkan & Radev 2004), the concept of graph-based centrality is used to rank a set of sentences, in producing generic multi-document summaries. A similarity graph is produced for the sentences in the document collection. In the graph, each node represents a sentence. The edges between nodes measure the cosine similarity between the respective pair of sentences. The degree of a given node is an indication of how much important the sentence is. Once the similarity graph is constructed, the sentences are then ranked according to their eigenvector centrality. The LexRank performed well in the context of generic summarization. To apply LexRank to query-focused context, a topic-sensitive version of LexRank is proposed in (Otterbacher, Erkan, & Radev 2005). We followed a similar approach in order to calculate this feature. The score of a sentence is determined by a mixture model of the relevance of the sentence to the query and the similarity of the sentence to other high-scoring sentences. We capture this idea by the following mixture model:

$$\begin{aligned} p(s|q) &= d \times \frac{rel(s|q)}{\sum_{z \in C} rel(z|q)} + (1-d) \\ &\times \sum_{v \in C} \frac{sim(s,v)}{\sum_{z \in C} sim(z,v)} \times p(v|q) \end{aligned} \qquad (4)$$

Where C is the set of all sentences in the collection and $rel(s|q)$ is the relevance of sentence s to the query q and computed as follows:

$$\begin{aligned} rel(s|q) &= \sum_{w \in q} log\,(tf_{w,s}+1) \times log\,(tf_{w,q}+1) \times idf_w \\ idf_w &= log\left(\frac{N+1}{0.5+sf_w}\right) \end{aligned}$$

Where $tf_{w,s}\ and\ tf_{w,q}$ are the number of times w appears in s and q, respectively, idf_w is the IDF value of word w, N is the total number of sentences in the cluster and sf_w is the number of sentences that the word w appears in. The value of the parameter d in equation 4, which we call “bias”, is a trade-off between two terms in the equation and is set empirically. For higher values of d, we prefer the relevance to the question to similarity to other sentences. The denominators in both terms are for normalization. We measure the cosine similarity weighted by word IDFs as the similarity between two sentences in a cluster:

$$sim(x,y) = \frac{\sum_{w \in x,y} tf_{w,x} tf_{w,y} (idf_w)^2}{\sqrt{\sum_{x_i \in x} (tf_{x_i,x} idf_{x_i})^2} \sqrt{\sum_{y_i \in y} (tf_{y_i,y} idf_{y_i})^2}}$$

[4]http://www.cs.ualberta.ca/ lindek/downloads.htm

Ranking Sentences and Summary Generation

The method used in order to select sentences for query-based summary extraction, is of type *unsupervised learning*, that means, before the learning begins, it is not known how many subsets (clusters) there are or how they are distinguished from each other. To start with, we experimented with two unsupervised learning techniques with the features extracted in the previous section: (a) K-means learning and (b) Expectation Maximization (EM) learning.

K-means Learning

We start with a set of initial cluster centers and go through several iterations of assigning each object to the cluster whose center is closest. After all objects have been assigned, we recompute the center of each cluster as the centroid or mean (μ) of its members. The distance function we use is squared Euclidean distance instead of true Euclidean distance. Since the square root is a monotonically growing function squared Euclidean distance has the same result as the true Euclidean distance but the computation overload is smaller when the square root is dropped.

Once we have learned the means of the clusters using the K-means algorithm, our next task is to rank the sentences according to a probability model. We have used Bayesian model in order to do so. Bayes' law says:

$$\begin{aligned} P(q_k|x,\Theta) &= \frac{p(\vec{x}|q_k,\Theta)P(q_k|\Theta)}{p(\vec{x}|\Theta)} \\ &= \frac{p(\vec{x}|q_k,\Theta)P(q_k|\Theta)}{\sum_{k=1}^{K} p(\vec{x}|q_k,\Theta)p(q_k|\Theta)} \end{aligned}$$

where q_k is a class, $\vec{x}$ is a feature vector representing a sentence and Θ is the parameter set of all class models. We set the weights of the clusters as equiprobable (i.e. $P(q_k|\Theta) = 1/K$). We calculated $p(\vec{x}|q_k,\Theta)$ using the gaussian probability distribution. The gaussian probability density function (pdf) for the d-dimensional random variable $\vec{x}$ is given by:

$$p_{(\vec{\mu},\Sigma)}(\vec{x}) = \frac{e^{\frac{-1}{2}(\vec{x}-\mu)^T \Sigma^{-1}(\vec{x}-\mu)}}{\sqrt{2\pi}^d \sqrt{det(\Sigma)}} \qquad (5)$$

where $\vec{\mu}$, the mean vector and Σ, the covariance matrix are the parameters of the gaussian distribution. We get the means ($\vec{\mu}$) from the K-means algorithm and we calculate the covariance matrix using the unbiased covariance estimation procedure:

$$\hat{\Sigma} = \frac{1}{N-1}\sum_{i=1}^{N}(\vec{x_i}-\vec{\mu_j})(\vec{x_i}-\vec{\mu_j})^T \qquad (6)$$

EM Learning

EM is a "soft" version of K-means algorithm described above (Manning & Schutze 2000). As K-means, we start with a set of random cluster centers, $c_1 \cdots c_k$. In each iteration we do a soft assignment of the data-points to every cluster by calculating their membership probabilities. EM is an iterative two step procedure: 1. Expectation-step and 2. Maximization-step. In the expectation step, we compute expected values for the hidden variables $h_{i,j}$ which are cluster membership probabilities. Given the current parameters, we compute how likely an object belongs to any of the clusters. The maximization step computes the most likely parameters of the model given the cluster membership probabilities. The data-points are considered to be generated by a mixture model of k-gaussians of the form:

$$\begin{aligned} P(\vec{x}) &= \sum_{i=1}^{k} P(C=i)P(\vec{x}|C=i) \\ &= \sum_{i=1}^{k} P(C=i)P(\vec{x}|\vec{\mu_i},\Sigma_i) \end{aligned}$$

Where the total likelihood of model Θ with k components given the observed data points, $X = \vec{x_1},\cdots,\vec{x_n}$ is:

$$\begin{aligned} L(\Theta|X) &= \prod_{i=1}^{n}\sum_{j=1}^{k} P(C=j)P(\vec{x_i}|\Theta_j) \\ &= \prod_{i=1}^{n}\sum_{j=1}^{k} w_j P(\vec{x_i}|\vec{\mu_j},\Sigma_j) \\ \Leftrightarrow &\sum_{i=1}^{n} log \sum_{j=1}^{k} w_j P(\vec{x_i}|\vec{\mu_j},\Sigma_j) \end{aligned}$$

where P is the probability density function (i.e. eq 5). $\vec{\mu_j}$ and Σ_j are the mean and covariance matrix of component j, respectively. Each component contributes a proportion, w_j, of the total population, such that: $\sum_{j=1}^{K} w_j = 1$. Log likelihood can be used instead of likelihood as it turns the product into sum.

However, a significant problem with the EM algorithm is that it converges to a local maximum of the likelihood function and hence the quality of the result depends on the initialization. We experimented with one summary (for document number D0703A from DUC2007) by initializing the parameters as follows:

$$\begin{aligned} \vec{\mu_j} &= rand(1,\cdots,d) * \sqrt{\Sigma(DATA) * 10} + \mu(DATA) \\ \Sigma_j &= \Sigma(DATA) \\ w_j &= 1/K \end{aligned}$$

The highly variable nature of the results of the tests is reflected in the very inconsistent values for the total log likelihood (see figure 1) and the results of repeated experiments indicated that using random starting values for initial estimates of the means frequently gave poor results. There are two possible solutions to this problem.

In order to get good results from using random starting values, we will run the EM algorithm several times and choose the initial configuration for which we get the maximum log likelihood among all configurations. Choosing

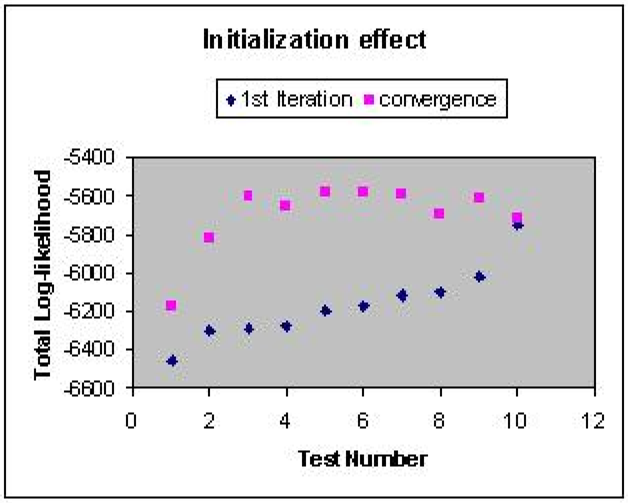

Figure 1: Plots of Log-likelihood values after one iteration (diamonds) and after convergence of EM algorithm (squares) for a set of random initial mean values

the best one among several runs is very computer intensive process. So, to improve the outcome of the EM algorithm on gaussian mixture models it is necessary to find a better method of estimating initial means for the components. To achieve this aim, we explored the widely used "K-means" algorithm as a cluster (means) finding method. That means, the means found by K-means clustering above will be utilized as the initial means for EM and we calculate the initial covariance matrices using the unbiased covariance estimation procedure (eq:6).

Once the sentences are clustered by EM algorithm, we filter out the sentences which are not question-relevant by checking their probabilities, $P(q_r|x_i, \Theta)$ where, q_r denotes the cluster "question-relevant". If for a sentence x_i, $P(q_r|x_i, \Theta) > 0.5$ then x_i is considered to be question-relevant.

Our next task is to rank the question-relevant sentences in order to include them in the summary. This can be done easily by defining a weight vector w for the features and multiplying it with the feature vector x_i representing the sentence. So, the rank of the sentence (or feature vector) $\vec{x_i}$ is given by:

$$score_i = \vec{x_i}.\vec{w}$$

Redundancy Checking and Generating Summary

Once the sentences are scored, the sentences deemed to be important are compared to each other and only those that are not too similar to other candidates are included in the final answer or summary. (Goldstein *et al.* 1999) observed this what they called "maximum-marginal-relevancy (MMR)". Following (Hovy *et al.* 2006), we modeled this by BE overlap between an intermediate summary and a to-be-added candidate summary sentence. We call this overlap ratio R, where R is between 0 and 1 inclusively. Setting $R = 0.7$ means that a candidate summary sentence s, can be added to an intermediate summary, S, if the sentence has a BE overlap ratio less than or equal to 0.7.

Experimental Evaluation

This section describes the results of experiments conducted using ROUGE automatic evaluation package (Lin 2004) on DUC 2007 dataset provided by NIST [5]. We generated summaries for the 45 topics in DUC 2007 and experimented with the performance of EM and K-means and the contributions of different types of features in generating good summaries for each of the algorithms. To assess the contribution of different features, we grouped them into four classes. They are as follows:

Lexical: N-gram (N=1,2,3,4), LCS, WLCS score, skip bi-gram, head, head synonym and BE overlap.

Lexical semantic: Synonym, hypernym/hyponym, gloss, dependency-based similarity and proximity based similarity.

Cosine similarity: Graph-based similarity.

All features: All the features above.

Discussion

Table 1 to table 3 and table 4 to table 6 show the evaluation measures for K-means and EM learning respectively. From our experiments, it is obvious that our systems achieve better results when we include lexical semantic and cosine similarity features with the lexical features and as our lexical features include the ROUGE measures so summaries based on only lexical features achieve good results.

Table 7 shows the F-scores of all the ROUGE measures for one baseline system, the best system in DUC 2007 and our two learning algorithms taking all features. The baseline system generates summaries by returning all the leading sentences (up to 250 words) in the $< TEXT >$ field of the most recent document(s).

The experiment results shows that EM outperforms the K-means algorithm and comparing with the DUC 2007 participants our systems achieve top scores.

Conclusion and Future Work

Our experiments show the following: (a) our approaches achieve promising results, (b) EM outperforms the K-means algorithm and (c) our systems achieve better results when we include lexical semantic and cosine similarity features with the lexical features. In this paper, we experimented with mainly fatures of type bag of words, later we would like to add the tree kernel based syntactic and shallow-semantic features and experiment their contribution for generating quality summary. As for this research problem, our training data were not labeled so we could not use supervised learning algorithms. Our future plan is to experiment with the supervised learning techniques (i.e. SVM, MAXENT) and see how do they perform for this problem. We have also the plan to decompose the complex questions into several simple questions before we find the similarity measures between the document sentence and the query sentence.

[5]National Institute of Standards and Technology

Scores	Lexical	Lexical Semantic	Cosine Similarity	All Features
Recall	0.074488	0.076827	0.085583	0.074625
Precision	0.080886	0.084017	0.092514	0.080753
F-measure	0.077543	0.080241	0.088891	0.077568

Table 1: ROUGE-2 measures for different feature combinations of K-means Learning

Scores	Lexical	Lexical Semantic	Cosine Similarity	All Features
Recall	0.337800	0.331936	0.346491	0.348463
Precision	0.366650	0.362165	0.373882	0.393244
F-measure	0.351575	0.346288	0.359562	0.369502

Table 2: ROUGE-L measures for different feature combinations of K-means Learning

Scores	Lexical	Lexical Semantic	Cosine Similarity	All Features
Recall	0.131197	0.127442	0.139002	0.135857
Precision	0.154859	0.152065	0.162368	0.176021
F-measure	0.141963	0.138527	0.149630	0.153353

Table 3: ROUGE-SU measures for different feature combinations of K-means Learning

Scores	Lexical	Lexical Semantic	Cosine Similarity	All Features
Recall	0.088564	0.079678	0.087056	0.084658
Precision	0.095792	0.087032	0.094192	0.091732
F-measure	0.092018	0.083171	0.090462	0.088053

Table 4: ROUGE-2 measures for different feature combinations of EM Learning

Scores	Lexical	Lexical Semantic	Cosine Similarity	All Features
Recall	0.354123	0.328675	0.344994	0.348606
Precision	0.383792	0.359347	0.373265	0.377634
F-measure	0.368281	0.343217	0.358481	0.362540

Table 5: ROUGE-L measures for different feature combinations of EM Learning

Scores	Lexical	Lexical Semantic	Cosine Similarity	All Features
Recall	0.145570	0.128454	0.138369	0.143064
Precision	0.171446	0.153288	0.162077	0.167845
F-measure	0.157339	0.139623	0.149153	0.154467

Table 6: ROUGE-SU measures for different feature combinations of EM Learning

Algrothms	ROUGE-2	ROUGE-L	ROUGE-W	ROUGE-SU
Baseline	0.064900	0.310740	0.113810	0.112780
Best System	0.122850	0.405610	0.153600	0.174700
K-means	0.090269	0.368522	0.138643	0.152057
EM	0.100875	0.373738	0.139323	0.163230

Table 7: ROUGE F-measures for the learing algorithms, baseline system and DUC-2007 best system

References

Cormen, T. R.; Leiserson, C. E.; and Rivest, R. L. 1989. *Introduction to Algorithms*. The MIT Press.

Erkan, G., and Radev, D. R. 2004. LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. In *Journal of Artificial Intelligence Research (JAIR)*, 457–479.

Goldstein, J.; Kantrowitz, M.; Mittal, V.; and Carbonell, J. 1999. Summarizing Text Documents: Sentence Selection and Evaluation Metrics. In *Proceedings of the 22nd International ACM Conference on Research and Development in Information Retrieval(SIGIR-99)*, 121–128.

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

Lin, D. 1998a. An Information-Theoretic Definition of Similarity. In *Proceedings of International Conference on Machine Learning*.

Lin, D. 1998b. Automatic Retrieval and Clustering of Similar Words. In *COLING-ACL98*.

Lin, C. Y. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004*.

Manning, C. D., and Schutze, H. 2000. *Foundations Of Statistical Natural Language Processing*. The MIT Press.

Otterbacher, J.; Erkan, G.; and Radev, D. R. 2005. Using Random Walks for Question-focused Sentence Retrieval. In *Proceedings of HLT/EMNLP (2005)*.

Zhou, L.; Lin, C. Y.; and Hovy, E. 2005. A BE-based Multi-dccument Summarizer with Query Interpretation. In *Proceedings of Document Understanding Conference (DUC-2005)*.