2-3-2014


Kai Hong
hongkai1@seas.upenn.edu

Ani Nenkova
nenkova@seas.upenn.edu

Follow this and additional works at: http://repository.upenn.edu/cis_reports

Part of the Computer Engineering Commons, and the Computer Sciences Commons

Recommended Citation


This paper is posted at ScholarlyCommons. http://repository.upenn.edu/cis_reports/989
For more information, please contact repository@pobox.upenn.edu.

Abstract
In this paper, we propose a supervised model for ranking word importance that incorporates a rich set of features. Our model is superior to prior approaches for identifying words used in human summaries. Moreover we show that an extractive summarizer which includes our estimation of word importance results in summaries comparable with the state-of-the-art by automatic evaluation.

Disciplines
Computer Engineering | Computer Sciences

Comments

This technical report is available at ScholarlyCommons: http://repository.upenn.edu/cis_reports/989
Abstract

We introduce a supervised model for predicting word importance that incorporates a rich set of features. Our model is superior to prior approaches for identifying words used in human summaries. Moreover we show that an extractive summarizer using these estimates of word importance is comparable in automatic evaluation with the state-of-the-art.

1 Introduction

In automatic extractive summarization, sentence importance is calculated by taking into account, among possibly other features, the importance of words that appear in the sentence. In this paper, we describe experiments on identifying words from the input that are also included in human summaries; we call such words summary keywords. We review several unsupervised approaches for summary keyword identification and further combine these, along with features including position, part-of-speech, subjectivity, topic categories, context and intrinsic importance, in a superior supervised model for predicting word importance.

One of the novel features we develop aims to determine the intrinsic importance of words. To this end, we analyze abstract-article pairs in the New York Times corpus (Sandhaus, 2008) to identify words that tend to be preserved in the abstracts. We demonstrate that judging word importance just based on this criterion leads to significantly higher performance than selecting sentences at random. Identifying intrinsically important words allows us to generate summaries without doing any feature computation on the input, equivalent in quality to the standard baseline of extracting the first 100 words from the latest article in the input. Finally, we integrate the schemes for assignment of word importance into a summarizer which greedily optimizes for the presence of important words. We show that our better estimation of word importance leads to better extractive summaries.

2 Prior work

The idea of identifying words that are descriptive of the input can be dated back to Luhn’s earliest work in automatic summarization (Luhn, 1958). There keywords were identified based on the number of times they appeared in the input, and words that appeared most and least often were excluded. Then the sentences in which keywords appeared near each other, presumably better conveying the relationship between the keywords, were selected to form a summary.

Many successful recent systems also estimate word importance. The simplest but competitive way to do this task is to estimate the word probability from the input (Nenkova and Vanderwende, 2005). Another powerful method is log-likelihood ratio test (Lin and Hovy, 2000), which identifies the set of words that appear in the input more often than in a background corpus (Conroy et al., 2006; Harabagiu and Lacatusu, 2005).

In contrast to selecting a set of keywords, weights are assigned to all words in the input in the majority of summarization methods. Approaches based on (approximately) optimizing the coverage of these words have become widely popular. Earliest such work relied on TF*IDF weights (Filatova and Hatzivassiloglou, 2004), later approaches included heuristics to identify summary-worthy bigrams (Riedhammer et al., 2010). Most optimization approaches, however, use TF*IDF or word probability in the input as word weights (McDonald, 2007; Shen and Li, 2010; Berg-Kirkpatrick et al., 2011).
Word weights have also been estimated by supervised approaches, with word probability and location of occurrence as typical features (Yih et al., 2007; Takamura and Okumura, 2009; Sipos et al., 2012).

A handful of investigations have productively explored the mutually reinforcing relationship between word and sentence importance, iteratively re-estimating each in either supervised or unsupervised framework (Zha, 2002; Wan et al., 2007; Wei et al., 2008; Liu et al., 2011). Most existing work directly focuses on predicting sentence importance, with emphasis on the formalization of the problem (Kupiec et al., 1995; Celikyilmaz and Hakkani-Tur, 2010; Litvak et al., 2010). There has been little work directly focused on predicting keywords from the input that will appear in human summaries. Also there has been only a few investigations of suitable features for estimating word importance and identifying keywords in summaries; we address this issue by exploring a range of possible indicators of word importance in our model.

3 Data and Planned Experiments

We carry out our experiments on two datasets from the Document Understanding Conference (DUC) (Over et al., 2007). DUC 2003 is used for training and development, DUC 2004 is used for testing. These are the last two years in which generic summarization was evaluated at DUC workshops.

There are 30 multi-document clusters in DUC 2003 and 50 in DUC 2004, each with about 10 news articles on a related topic. The task is to produce a 100-word generic summary. Four human abstractive summaries are available for each cluster.

We compare different keyword extraction methods by the F-measure\(^1\) they achieve against the gold-standard summary keywords. We do not use stemming when calculating these scores.

In our work, keywords for an input are defined as those words that appear in at least \(i\) of the human abstracts, yielding four gold-standard sets of keywords, denoted by \(G_i\). \(|G_i|\) is thus the cardinality of the set for the input. We only consider the words in the summary that also appear in the original input\(^2\), with stopwords excluded\(^3\). Table 1 shows the average number of unique content words for the respective keyword gold-standard.

<table>
<thead>
<tr>
<th>(i)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean (</td>
<td>G_i</td>
<td>)</td>
<td>102</td>
<td>32</td>
</tr>
</tbody>
</table>

Table 1: Average number of words in \(G_i\)

To better compare the performances of different keyword identification systems, we here employ a weighted approach similar to the pyramidal method (Nenkova et al., 2007). We call our approach keyword pyramid method. This approach assigns different weights for the words which appear in different number of human models, while generating a single score as its output.

For the summarization task, we compare results using ROUGE (Lin, 2004). We report ROUGE-1, -2, -4 recall, with stemming and without removing stopwords. We consider ROUGE-2 recall as the main metric for this comparison due to its effectiveness in comparing machine summaries (Owczarzak et al., 2012). All of the summaries were truncated to the first 100 words by ROUGE\(^4\).

We use two-tailed Wilcoxon signed-rank test to examine the statistical significance as advocated by Rankel et al. (2011) for both tasks, and consider differences to be significant if the p-value is less than 0.05.

4 Unsupervised Word Weighting

In this section we describe three unsupervised approaches of assigning importance weights to words. The first two are probability and log-likelihood ratio, which have been extensively used in prior work. We also apply a markov random walk model for keyword ranking, similar to (2004). We further discuss the correlations and differences between the words ranked by these three approaches. In the next section we describe a summarizer that uses these weights to form a summary and then describe our regression approach to combine these and other predictors in order to achieve more accurate predictions for the word importance in Section 7.

The task is to assign a score to each word in the input. The keywords extracted are thus the content

\(^1\)\text{F-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}

\(^2\)On average 26.3% (15.0% with stemming) of the words in the four abstracts never appear in the input.

\(^3\)We use the stopword list from the SMART system (Salton, 1971), augmented with punctuation and symbols.

\(^4\)ROUGE version 1.5.5 with parameters: -c 95 -r 1000 -n 4 -m -a -I 100 -x
words with highest scores.

4.1 Word Probability (Prob)
The frequency with which a word occurs in the input is often considered as an indicator of its importance. The weight for a word is computed as \( p(w) = \frac{c(w)}{N} \), where \( c(w) \) is the number of times word \( w \) appears in the input and \( N \) is the total number of word tokens in the input.

4.2 Log-likelihood Ratio (LLR)
The log-likelihood ratio test (Lin and Hovy, 2000) compares the distribution of a word in the input with that in a large background corpus to identify topic words, where Gigaword corpus is used here (Graff et al., 2007). The test statistic has a \( \chi^2 \) distribution, so a desired confidence level can be chosen to find a small set of topic words.

4.3 Markov Random Walk Model (MRW)
Graph methods have been successfully applied to weighting sentences for generic (Wan and Yang, 2008; Mihalcea and Tarau, 2004; Erkan and Radev, 2004) and query-focused summarization (Otterbacher et al., 2009).

Here instead of constructing a graph with sentences as nodes and edges weighted by sentence similarity, we treat the words as vertices, similar to Mihalcea and Tarau (2004). The difference in our approach is that the edges between the words are defined by syntactic dependencies rather than depending on the co-occurrence of words within a window of \( k \). We use the Stanford dependency parser (Marneffe et al., 2006). In our approach, we consider a word \( w \) more likely to be included in a human summary when it is syntactically related to other (important) words, even if \( w \) itself is not mentioned often. The edge weight between two vertices is equal to the number of syntactic dependencies of any type between two words within the same sentence in the input. The weights are then normalized by summing up the weights of edges linked to one node.

We apply the Pagerank algorithm (Lawrence et al., 1998) on the resulting graph. We set the probability of performing random jump between nodes \( \lambda = 0.15 \). The algorithm terminates when the change of node weight between iterations is smaller than \( 10^{-4} \) for all nodes. Word importance is equal to the final weight of its corresponding node in the graph.

4.4 Comparison of the Three Approaches
Here we show the correlations and differences between the three unsupervised weighting approaches: \( \text{Prob} \), LLR and MRW. For each input on the DUC 2004 dataset, we compute the Spearman correlation between the weights assigned to words by any two unsupervised methods. This results in 50 scores for each pair of comparison. In Table 2, we show the median, maximum and minimum of Spearman correlation between these methods. Among the three comparisons, \( \text{Prob} \) and MRW are the most similar (Median = 0.6745), followed by \( \text{Prob} \) and LLR (Median = 0.4859), while LLR and MRW are the least similar (Median = 0.3298). All correlations are positive and significant, however they clearly show that the ranks assigned to words differ considerably from each other. Even the maximum correlation between the two most similar lists (\( \text{Prob} \) and MRW) is only 0.751.

<table>
<thead>
<tr>
<th></th>
<th>Median</th>
<th>Max</th>
<th>Min</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{Prob vs MRW} )</td>
<td>0.6745</td>
<td>0.7505</td>
<td>0.5883</td>
</tr>
<tr>
<td>( \text{Prob vs LLR} )</td>
<td>0.4859</td>
<td>0.6867</td>
<td>0.3019</td>
</tr>
<tr>
<td>( \text{LLR vs MRW} )</td>
<td>0.3298</td>
<td>0.5300</td>
<td>0.2062</td>
</tr>
</tbody>
</table>

Table 2: Spearman correlation between word weights assigned by the three unsupervised approaches on DUC 2004

We provide an example of the top 20 keywords identified by those three approaches for two of the inputs, \( d30001t \) and \( d30049t \) in Table 3. This table gives us an intuition about the words which tend to be ranked high using each approach. We also discuss the change of ranking for the same word from the output of these three approaches. Compared with word probability, we observe a drop in rank for the words which appear more often in the background corpus using log-likelihood ratio. For example, the rank of the words \( \text{party} \) and \( \text{government} \) dropped in \( d30001t \); the rank of the words \( \text{clinton} \) and \( \text{united} \) dropped in \( d30049t \). Meanwhile, the ranks of the relatively rare words improved while using log-likelihood ratio. For instance, the rank of \( \text{CPP} \) and \( \text{Funcinpec} \) improved in \( d30001t \) and the rank of \( \text{underground} \) improved in \( d30049t \). When ranked by MRW, the words which are syntactically related with other high frequency terms have been rewarded, compared with \( \text{Prob} \). Such examples include
Table 3: Top 20 words from the input d30001t and d30049t by the three unsupervised approaches

<table>
<thead>
<tr>
<th>Metric</th>
<th>d30001t</th>
<th>d30049t</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROB</td>
<td>hun, sen, party, ranariddh, opposition government, rainy, sam, combodia, assembly comobian, sihanouk, parliament, king, prince norodom, partie, cpp, national, form</td>
<td>north, korea, nuclear, clinton, korean, kim site, south, united, underground, states weapons, koreans, president, officials, complex construction, government, program, american</td>
</tr>
<tr>
<td>TS</td>
<td>hun, sen, ranariddh, rainy, party, opposition combodia, sam, combodian, cpp, sihanouk norodom, funcipec, assembly, government parliament, prince, parties, top, election</td>
<td>north, korea, nuclear, korean, underground clinton, site, weapons, kim, koreans south, full, make, give, kartman complex, construction, iraq, yongbyon, missile</td>
</tr>
<tr>
<td>MRW</td>
<td>sen, party, government, ranriddh, rainy hun, form, opposition, assembly, parties return, make, agreed, members, sihanouk won, cpp, country, election, called</td>
<td>korea, north, site, clinton, states called, korean, nuclear, koreans, program complex, u.s., kim, officials, give weapons, pay, south, government, facility</td>
</tr>
</tbody>
</table>

5 Summary Generation Process

In this section, we outline how summaries are generated by a greedy optimization system which selects the sentence with highest weight iteratively. This is the main process we use in all our summarization systems. For comparison we also use a summarization algorithm based on KL divergence.

5.1 Greedy Optimization Approach

Our algorithm extracts sentences by weighting them based on word importance. The approach is similar to the standard word probability baseline (Nenkova et al., 2006) but we explore a range of possibilities for assigning weights to individual words. Pseudo code of the procedure is shown in Algorithm 1. We denote the set of input sentences as \( D_k \), the weighting function of words as \( f_s(w) \). For each sentence \( s_i \) in the input set \( D_k \), we compute the sentence weight \( \text{Score}(s_i) \) by summing up the weights of all words, normalized by the number of words in the sentence. Then the sentences are sorted in descending order according to their scores and put into a queue \( Q \). To create a summary, we iteratively dequeue one sentence, and append it to the current summary if the sentence is valid. Two conditions should be met for one sentence to be judged as valid, (1) the sentence should be at least 9 words in length (as in Erkan and Radev (2004)); (2) the sentence should not be redundant. A sentence is considered non-redundant if it is not similar to any sentences already in the summary, measured by cosine similarity on binary vector representations with stopwords excluded. We use the cut-off of 0.5 for cosine similarity. This value was tuned on the DUC 2003 dataset, by testing the impact of the cut-off value on the ROUGE scores for the final summary. Possible values ranged from 0.1 to 0.9
with a step of 0.1.

5.2 KL Divergence Summarizer

The KLSUM summarizer (Haghighi and Vanderwende, 2009) aims at minimizing the KL divergence between the probability distribution over words estimated from the summary and the input respectively. This summarizer is a component of the popular topic model approaches (Daumé and Marcu, 2006; Celikyilmaz and Hakkani-Tür, 2011; Mason and Charniak, 2011) and achieves competitive performance with minimal differences compared to a full-blown topic model system. Denote Q as the unigram distribution of the input, P as the unigram distribution of the summary. The objective is to minimize the following:

\[
KL(P \parallel Q) = \sum_{w \text{ in summary}} P(w) \cdot \ln \frac{P(w)}{Q(w)}
\]

Since optimizing \(KL(P \parallel Q)\) is exponential to the number of sentences in the input, we apply the Greedy-KL algorithm (Haghighi and Vanderwende, 2009). Denote \(D_k\) as the input, for each iteration we pick the sentence \(s_i \in D_k\) which would minimize the KL divergence between the current version of the summary and the probability over the input. While selecting the sentence, again we check the validity, similar to Section 5.1. Pseudo code of the summarizer is shown in Algorithm 2.

Algorithm 2 KLSUM Summarizer

1: procedure KLSUM(D_k) 
2: Compute the unigram distribution Q for \(D_k\).
3: while \(\text{Len}(\text{Sum}_k) \leq 100\) do 
4: \(j = \text{argmin}_i \ KL(P_i \parallel Q)\) \(\triangleright\)
5: \(P_i\) is the unigram distribution for \(\text{Sum}_k \cup s_i\), among all valid \(s_i \in S\).
6: \(\text{Sum}_k \leftarrow \text{Sum}_k \cup \{s_j\}\)
7: end while
8: return \(\text{Sum}_k\)

5This is different from the formula in Haghighi and Vanderwende (2009), as we minimize \(KL(P \parallel Q)\) instead of \(KL(Q \parallel P)\). The summarizer we present here is the one with better performance.

6 Global Indicators from NYT

Some words evoke topics that are of intrinsic interest to people. Here we search for global indicators of word importance regardless of particular input.

6.1 Global Indicators of Word Importance

We analyze a large corpus of original documents and corresponding summaries in order to identify words that consistently get included in or excluded from the summary. In the 2004-2007 NYT corpus, many news articles have abstracts along with the original article, which makes it an appropriate resource to do such analysis. We identified 160,001 abstract-original pairs in the corpus. From these, we generate two language models, one estimated from the text of all abstracts (\(LM_A\)), the other estimated from the corpus of original articles (\(LM_G\)). We use SRILM (Stolcke, 2002) with Ney smoothing.

We denote the probability of word \(w\) in \(LM_A\) as \(Pr_A(w)\), the probability in \(LM_G\) as \(Pr_G(w)\), and calculate the difference \(Pr_A(w) - Pr_G(w)\) and the ratio \(Pr_A(w)/Pr_G(w)\) to capture the change of probability. In addition, we calculate KL-like weighted scores for words which reflect both the change of probabilities between the two samples and the overall frequency of the word. Here we calculate both \(KL(A \parallel G)\) and \(KL(G \parallel A)\). The formulas are shown below. Words with high values for the former score are favored in the summaries because they have higher probability in the abstracts than in the originals and have relatively high probability in the abstracts. The later score is high for words that are often not included in summaries.

\[
KL(A \parallel G)(w) = Pr_A(w) \cdot \ln \frac{Pr_A(w)}{Pr_G(w)}
\]

\[
KL(G \parallel A)(w) = Pr_G(w) \cdot \ln \frac{Pr_G(w)}{Pr_A(w)}
\]

Table 4 shows examples of the global information captured from the three types of scores—\(KL(A \parallel G)\), \(KL(G \parallel A)\) and \(Pr_A(w)\)—listing the 30 content words with higher scores for each type. Here the stopwords are excluded. Words that tend to be used in the summaries, characterized by high \(KL(A \parallel G)\) scores, include locations (York, NJ, Iraq), people’s names and titles (Bush, Sen, John), some abbreviations (pres, corp, dept) and verbs of
conflict (contends, dies). On the other hand, from $KL(G \parallel A)$, we can see that it is unlikely for writers to include courtesy titles (Mr, Ms, Jr.) and relative time reference in summaries. The words with high $Pr_A(w)$ scores overlaps with those ranked highly by $KL(A \parallel G)$ to some extent, but also includes a number of generally frequent words which appeared often both in the abstracts and original texts, such as million and percent.

### 6.2 Blind Sentence Extraction

In later sections we include the measures of global word importance as a feature of our regression model for predicting word weights for summarization. Before turning to that, however, we report the results of an experiment aimed to confirm the usefulness of these features. We present a system, BLIND, which uses only weights assigned to words by $KL(A \parallel G)$ from NYT, without doing any analysis of the original input. We rank all non-stopword words from the input according to this score. The top $k$ words are given weight 1, while the others are given weight 0. The summaries are produced following the greedy procedure described in Section 5.1.

<table>
<thead>
<tr>
<th>Systems</th>
<th>R-1</th>
<th>R-2</th>
<th>R-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANDOM</td>
<td>30.32</td>
<td>4.42</td>
<td>0.36</td>
</tr>
<tr>
<td>BLIND (80 keywords)</td>
<td>30.77</td>
<td>5.18</td>
<td>0.53</td>
</tr>
<tr>
<td>BLIND (300 keywords)</td>
<td>32.91</td>
<td>5.94</td>
<td>0.61</td>
</tr>
<tr>
<td>LASTESTLEAD</td>
<td>31.39</td>
<td>6.11</td>
<td>0.63</td>
</tr>
<tr>
<td>FIRST-Sentence</td>
<td>34.26</td>
<td>7.22</td>
<td>1.21</td>
</tr>
</tbody>
</table>

Table 5: Blind sentence extraction system, compared with three baseline systems (%)

Table 5 shows that the BLIND system has R-2 recall of 0.0594 using the top 300 keywords, significantly better than picking sentences from the input randomly. It also achieves comparable performance with the baseline in DUC 2004, formed by selecting the first 100 words from the latest article in the input (LASTESTLEAD). However it is significantly worse than another baseline of selecting the first sentences from the input. Table 6 gives sample summaries generated by these three approaches. These results confirm that the information gleaned from the analysis of NYT abstract-original pairs encodes highly relevant information about important content independent of the actual text of the input.

### 7 Regression-Based Keyword Extraction

Here we introduce a logistic regression model for assigning importance weights to words in the input. Crucially, this model combines evidence from multiple indicators of importance. We have at our disposal abundant data for learning because each content word in the input can be treated as a labeled instance. There are in total 32,052 samples from the 30 inputs of DUC 2003 for training, 54,591 samples from the 50 inputs of DUC 2004 for testing. For a word in the input, we assign label 1 if the word appears in at least one of the four human summaries for this input. Otherwise we assign label 0.

In the rest of this section, we describe the rich variety of features included in our system. We also analyze and discuss the predictive power of those features by performing two-tailed Wilcoxon signed-rank test on the DUC 2003 dataset. There are in total 9,261 features used, among them 1,625 are significant (p-value < 0.05). We rank these features in increasing p-values derived from Wilcoxon test. Apart from the widely used
Random Summary
It was sunny and about 14 degrees C(57 degrees F) in Tashkent on Sunday. The president is a strong person, and he has been through far more difficult political situations, Mityukov said, according to Interfax. But Yeltsin’s aides say his first term, from 1991 to 1996, does not count because it began six months before the Soviet Union collapsed and before the current constitution took effect. He must stay in bed like any other person, Yakushkin said. The issue was controversial earlier this year when Yeltsin refused to spell out his intentions and his aides insisted he had the legal right to seek re-election.

NYT Summary from global keyword selection, \( K|L(A \parallel G), k = 300 \)
Russia’s constitutional court opened hearings Thursday on whether Boris Yeltsin can seek a third term. Yeltsin’s growing health problems would also seem to rule out another election campaign. The Russian constitution has a two-term limit for presidents. Russian president Boris Yeltsin cut short a trip to Central Asia on Monday due to a respiratory infection that revived questions about his overall health and ability to lead Russia through a sustained economic crisis. The upper house of parliament was busy voting on a motion saying he should resign. The start of the meeting was shown on Russian television.

First Sentence Generated Summary
President Boris Yeltsin has suffered minor burns on his right hand, his press office said Thursday. President Boris Yeltsin’s doctors have pronounced his health more or less normal, his wife Naina said in an interview published Wednesday. President Boris Yeltsin, on his first trip out of Russia since this spring, canceled a welcoming ceremony in Uzbekistan on Sunday because he wasn’t feeling well, his spokesman said. Doctors ordered Russian President Boris Yeltsin to cut short his Central Asian trip because of a respiratory infection and he agreed to return home Monday, a day earlier than planned, officials said.

<table>
<thead>
<tr>
<th>Table 6: Summaries generated from the Random, Blind Extraction and First Sentence systems</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Features</strong></td>
</tr>
<tr>
<td>Word Locations</td>
</tr>
<tr>
<td>Word type</td>
</tr>
<tr>
<td>Most of the NE features (6 out of 8) are significant: there are more Organizations and Locations but fewer Time and Date words in the human summaries. Of the POS tags, 11 out of 41 are significant: there are more nouns (NN, NNS, NNPS); fewer verbs (VBG, VBP, VB) and fewer cardinal numbers in the abstracts compared to the input. Capitalized words also tend to be included in human summaries.</td>
</tr>
</tbody>
</table>
| KL: Prior work has shown that having estimates of sentence importance can also help in estimating
word importance (Wan et al., 2007; Liu et al., 2011; Wei et al., 2008). The summarizer based on KL-divergence assigns importance to sentences directly, in a complex function according to the word distribution in the sentence. Therefore, we use these summaries as potential indicators of word importance. We include two features here, the first one indicates if the word appears in a KLSUM summary of the input, as well as a feature corresponding to the number of times the word appeared in that summary. Both of the features are highly significant, ranked within the top 200.

7.3 NYT-weights as Features

We include features from the relative rank of a word according to \( KL(A \| G), KL(G \| A) \), \( Pr_A(w) - Pr_G(w) \), \( Pr_A(w)/Pr_G(w) \) and \( Pr_A(w) \), derived from the NYT as described in Section 6. If the rank of a word is within top-k or bottom-k by one metric, we would label it as 1 (one feature dimension for top-k, one feature dimension for bottom-k), where \( k \) is selected from a set of pre-defined values\(^7\). We have in total 70 features in this category, of which 56 are significant, 47 having a p-value less than \( 10^{-7} \). The predictive power of those global indicators are only behind the features which indicates frequency and word positions.

7.4 Unigrams

This is a binary feature corresponding to each of the words that appeared at least twice in the training data. The idea is to learn which words from the input tend to be mentioned in the human summaries. There are in total 8,691 unigrams, among which 1,290 are significant. Despite the high number of significant unigram features, most of them are not as significant as the more general ones we described so far. It is interesting to compare the significant unigrams identified in the DUC abstract/input data with those derived from the NYT corpus. Unigrams that tend to appear in DUC summaries include president, government, political. We also find the same unigrams among the top words from NYT corpus according to \( KL(A \| G) \). As for words unlikely to appear in summaries, we see Wednesday, added, thing, etc, which again rank high according to \( KL(G \| A) \).

7.5 Dictionary Features: MPQA and LIWC

Unigram features are notoriously sparse. To mitigate the sparsity problem, we resort to more general groupings to words according to salient semantic and functional categories. We here employ two hand-crafted dictionaries, MPQA for subjectivity analysis and LIWC for topic analysis.

The MPQA dictionary (Wiebe and Cardie, 2005) contains words with different polarities (positive, neutral, negative) and intensities (strong, weak). The combinations correspond to six features. It turns out that words with strong polarity, either positive or negative, are seldomly included in the summaries. Most strikingly, the p-value from significance test for the strong negative words is less than \( 10^{-4} \)—these words are rarely included in summaries. There is no significant difference on weak polarity categories.

Another dictionary we use is LIWC (Tausczik and Pennebaker, 2007), which contains manually constructed dictionaries for multiple categories of words. The value of the feature is 1 for one word if the word appears in the particular dictionary for the category. 34 out of 64 LIWC features are significant. Interesting categories which appear at higher rate in summaries include events about death, anger, achievements, money and negative emotions. Those that appear at lower rate in the summaries include auxiliary verbs, hear, pronouns, negation, function words, social words, swear, adverbs, words related to families, etc.

7.6 Context Features

We use context features here, based on the assumption that context importance around a word affects the importance of this word. For context we consider the words before and after the target word. We extend our feature space by calculating the weighted average of the feature values of the context words. For word \( w \), we denote \( L_w \) as the set of words before \( w \), \( R_w \) as the set of words after \( w \). We denote the feature for one word as \( w.f_i \), the way of calculating the newly extended word-before feature \( w.l.f_i \) could be written as:

\[
 w.l.f_i = \sum_{i} p(w_i) \cdot w_i.f_i, \forall w_i \in L_w
\]

Here \( p(w_i) \) is the probability word \( w_i \) appears before \( w \) among all words in \( L_w \).

For context features, we calculate the weighted average of the most widely used basic features,
including frequency, location and capitalization for surrounding contexts. There are in total 220 features of this kind, among which 117 are significant, 74 having a p-value less than $10^{-4}$.

8 Experiments

The performance of our logistic regression model is evaluated on two tasks: keyword identification and extractive summarization. We name our system REGSUM.

8.1 Regression for Keyword Identification

For each input, we define the set of keywords as the top $k$ words according to the scores generated from different models. We compare our regression system with three unsupervised systems: PROB, LLR, MRW. To show the effectiveness of new features, we compare our results with a regression system trained only on word frequency and location related features described in Section 7. Those features are the ones standardly used for ranking the importance of words in recent summarization works (Yih et al., 2007; Takamura and Okumura, 2009; Sipos et al., 2012), and we name this system REGASIC.

![Figure 1: Precision, Recall and F-score of keyword identification, 100 words selected, $G_1$ as gold-standard](image)

Figure 1 shows the performance of systems when selecting the 100 words with highest weights as keywords. Each word from the input that appeared in any of the four human summaries is considered as a gold-standard keyword. Among the unsupervised approaches, word probability identifies keywords better than LLR and MRW by at least 4% on F-score. REGASIC does not give better performance at keyword identification compared with PROB, even though it includes location information. Our system gets 2.2% F-score improvement over PROB, 5.2% over REGASIC, and more improvement over the other approaches. All of these improvements are statistically significant by Wilcoxon test.

Table 7 shows the performance of keyword identification for different $G_i$ and different number of keywords selected. The regression system has no advantage over PROB when identifying keywords that appeared in all of the four human summaries. However our system achieves significant improvement for predicting words that appeared in at least one ($G_1$) or at least two ($G_2$) human summaries.

8.2 Keyword Pyramid Method

In Section 8.1, we compare the keywords extracted using different approaches with the gold standard keywords that appear in at least $i$ human abstracts ($G_i$). Doing evaluation in this way does not differentiate between the keywords which appear in different number of human models. Inspired by the pyramid method (Nenkova et al., 2007), we employ a weighted approach for evaluating the identification of summary keywords. This method takes into account the fact that words appear in more human summaries are of greater importance. Moreover, it generates a single score as its output, which makes the evaluation simpler. We name our approach as keyword pyramid method.

Consider an input with $n$ manually generated summaries. For each word $w$, we assign a weight $t(w) = i$, representing the number of summaries this word appears in. All of the summary keywords can be partitioned into tiers based on $t(w)$ in this way. We here denote the tiers as $T_i$ (1 ≤ $i$ ≤ $n$), where $T_i$ includes all words that appear in exactly $i$ human models. We name the tiers of keywords for the input here as our keyword pyramid.

Consider we are given a word list $L$ returned by a word weighting approach. Denote $w_i$ as the word ranked the $i$-th in the list. For the words which have appeared in the keyword pyramid, they are assigned weight $t(w_i)$ according to the number of human models they appear in. The words which have never appeared in human models are assigned weight zero. Denote $L_k$ as the set of top $k$ words in the list $L$, the coverage weight of these

---

8We would like to thank an anonymous reviewer for suggesting us doing this evaluation.
words can be computed as:

\[
Coverage(L_k) = \sum_{i=1}^{k} t(w_i)
\]

Similar to the pyramid method, we normalize \(Coverage(L_k)\) by the optimal coverage weight that could be achieved with \(k\) words selected. The optimal score is computed from a list where all of the words from \(T_n\) are ranked first, followed by all of the words from \(T_{n-1}\), etc. The formula of computing the optimal coverage weight is:

\[
Opt_k = \sum_{i=s+1}^{n} i \cdot |T_i| + s \cdot (k - \sum_{i=s+1}^{n} |T_i|)
\]

Here \(s = \max_i \left( \sum_{i=1}^{n} |T_i| \geq k \right)\)

The normalized keyword pyramid score is simply \(\frac{Coverage(L_k)}{Opt_k}\), ranges between 0 and 1.

We now compare the performance of the five systems using our keyword pyramid method. Figure 2 illustrates the normalized keyword pyramid scores changing with the number of keywords extracted. When \(k\) is equal to ten, our regression system performs similar to using word probability. However, our system achieves significantly better performance than all of the others when \(k \geq 20\). The improvement ranges between 2.1% to 3.2% for all of the \(k\) we test on between 20 and 100. Word probability gives the best performance among the three unsupervised approaches. This result is consistent with the ones in Figure 1 and Table 7.

<table>
<thead>
<tr>
<th>(G_i)</th>
<th>#words</th>
<th>PROB</th>
<th>LLR</th>
<th>MRW</th>
<th>REGBASIC</th>
<th>REGSUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>(G_1)</td>
<td>80</td>
<td>43.6</td>
<td>37.9</td>
<td>38.9</td>
<td>39.9</td>
<td>45.7</td>
</tr>
<tr>
<td>(G_1)</td>
<td>100</td>
<td>44.3</td>
<td>38.7</td>
<td>39.2</td>
<td>41.0</td>
<td>46.5</td>
</tr>
<tr>
<td>(G_1)</td>
<td>120</td>
<td>44.6</td>
<td>38.5</td>
<td>39.2</td>
<td>40.9</td>
<td>46.4</td>
</tr>
<tr>
<td>(G_2)</td>
<td>30</td>
<td>47.8</td>
<td>44.0</td>
<td>42.4</td>
<td>47.4</td>
<td>50.2</td>
</tr>
<tr>
<td>(G_2)</td>
<td>35</td>
<td>47.1</td>
<td>43.3</td>
<td>42.1</td>
<td>47.0</td>
<td>49.5</td>
</tr>
<tr>
<td>(G_2)</td>
<td>40</td>
<td>46.5</td>
<td>42.4</td>
<td>41.8</td>
<td>46.4</td>
<td>49.2</td>
</tr>
<tr>
<td>(G_3)</td>
<td>10</td>
<td>51.2</td>
<td>46.2</td>
<td>43.8</td>
<td>46.9</td>
<td>50.2</td>
</tr>
<tr>
<td>(G_3)</td>
<td>15</td>
<td>51.4</td>
<td>47.5</td>
<td>43.7</td>
<td>49.8</td>
<td>52.9</td>
</tr>
<tr>
<td>(G_3)</td>
<td>20</td>
<td>49.7</td>
<td>47.6</td>
<td>42.5</td>
<td>49.3</td>
<td>51.5</td>
</tr>
<tr>
<td>(G_4)</td>
<td>5</td>
<td>50.0</td>
<td>48.8</td>
<td>44.9</td>
<td>43.6</td>
<td>45.1</td>
</tr>
<tr>
<td>(G_4)</td>
<td>6</td>
<td>51.4</td>
<td>46.9</td>
<td>43.7</td>
<td>45.2</td>
<td>47.6</td>
</tr>
<tr>
<td>(G_4)</td>
<td>7</td>
<td>50.9</td>
<td>48.2</td>
<td>43.7</td>
<td>45.8</td>
<td>47.8</td>
</tr>
</tbody>
</table>

Table 7: Keyword identification F-score (%) for different \(G_i\) and different number of words selected.

8.3 Regression for Summarization

We now show that the performance of extractive summarization can be improved by better estimation of word weights. We compare our regression system with the four models introduced in Section 8.1. We also include PEER-65, the best system in DUC-2004, as well as KLSUM for comparison. Apart from these, we compare our model with two state-of-the-art systems, including the submodular approach (SUBMOD) (Lin and Bilmes, 2012) and the determinantal point process (DPP) summarizer (Kulesza and Taskar, 2012). The summaries were kindly provided by the authors of these systems (Hong et al., 2014).

As can been seen in Table 8, our system outperforms PROB, LLR, MRW, PEER-65, KLSUM and REGBASIC. These improvements are significant on ROUGE-2 recall. Interestingly, although the supervised system REGBASIC which uses only frequency and positions
low performance in keyword identification, the summaries it generates are of high quality. The inclusion of position features negatively affects the performance in summary keyword identification but boosts the weights for the words which appear close to the beginning of the documents, which is helpful for identifying informative sentences. By including other features we greatly improve over REGBASIC in keyword identification. Similarly here the richer set of features results in better quality summaries.

We also examined the ROUGE-1, -2, -4 recall compared with the SUBMOD and DPP summarizers. There is no significant difference on R-2 and R-4 recall compared with these two state-of-the-art systems. DPP performed significantly better than our system on R-1 recall, but that system is optimizing on R-1 F-score in training. Overall, our conceptually simple system is on par with the state of the art summarizers and points to the need for better models for estimating word importance.

<table>
<thead>
<tr>
<th>System</th>
<th>R-1</th>
<th>R-2</th>
<th>R-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROB</td>
<td>35.14</td>
<td>8.17</td>
<td>1.06</td>
</tr>
<tr>
<td>LLR</td>
<td>34.60</td>
<td>7.56</td>
<td>0.83</td>
</tr>
<tr>
<td>MRW</td>
<td>35.78</td>
<td>8.15</td>
<td>0.99</td>
</tr>
<tr>
<td>REGBASIC</td>
<td>37.56</td>
<td>9.28</td>
<td>1.49</td>
</tr>
<tr>
<td>KL</td>
<td>37.97</td>
<td>8.53</td>
<td>1.26</td>
</tr>
<tr>
<td>PEER-65</td>
<td>37.62</td>
<td>8.96</td>
<td>1.51</td>
</tr>
<tr>
<td>SUBMOD</td>
<td>39.18</td>
<td>9.35</td>
<td>1.39</td>
</tr>
<tr>
<td>DPP</td>
<td>39.79</td>
<td>9.62</td>
<td>1.57</td>
</tr>
<tr>
<td>REGSUM</td>
<td><strong>38.57</strong></td>
<td><strong>9.75</strong></td>
<td><strong>1.60</strong></td>
</tr>
</tbody>
</table>

Table 8: System performance comparison (%)

9 Conclusion

We presented a series of experiments which show that keyword identification can be improved in a supervised framework which incorporates a rich set of indicators of importance. We also show that the better estimation of word importance leads to better extractive summaries. Our analysis of features related to global importance, sentiment and topical categories reveals rather unexpected results and confirms that word importance estimation is a worthy research direction. Success in the task is likely to improve sophisticated summarization approaches too, as well as sentence compression systems which use only crude frequency related measures to decide which words should be deleted from a sentence.10

References


The work is partially funded by NSF CAREER award IIS 0953445.


