Enhanced web-based summary generation for search.

Brent Wenerstrom 1980-
University of Louisville

Follow this and additional works at: http://ir.library.louisville.edu/etd

Recommended Citation
https://doi.org/10.18297/etd/1547

This Doctoral Dissertation is brought to you for free and open access by ThinkIR: The University of Louisville's Institutional Repository. It has been accepted for inclusion in Electronic Theses and Dissertations by an authorized administrator of ThinkIR: The University of Louisville's Institutional Repository. This title appears here courtesy of the author, who has retained all other copyrights. For more information, please contact thinkir@louisville.edu.
ENHANCED WEB-BASED SUMMARY
GENERATION FOR SEARCH

By

Brent Wenerstrom
B.S., Brigham Young University, 2003
M.S., Brigham Young University, 2006

A Dissertation
Submitted to the Faculty of the
J.B. Speed School of Engineering at the University of Louisville
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

Department of Computer Engineering and Computer Science
University of Louisville
Louisville, Kentucky

August 2012
ENHANCED WEB-BASED SUMMARY
GENERATION FOR SEARCH

By

Brent Wenerstrom
B.S., Brigham Young University, 2003
M.S., Brigham Young University, 2006

A Dissertation Approved On

July 25, 2012
Date

by the following Dissertation Committee:

Dr. Mehmed Kantardzic, Dissertation Director

Dr. Olfa Nasraoui

Dr. Tim Hardin

Dr. Ming Ouyang

Dr. Rammohan Ragade
ACKNOWLEDGEMENTS

This dissertation would not be possible without the help, guidance and leadership of many. I would like to thank my advisor, Dr. Mehmed Kantardzic. He has been my guide through the whole PhD process answering questions on all possible topics. This dissertation owes numerous hours in editing, ideas, guidance and help to Dr. Kantardzic.

I would like to thank my wife. She has been very patient through the trials and challenges that come with getting a PhD. Thank you Katie.

My parents have been very important to this whole process setting an example of learning and working early in my life.

I would also like to thank all of the many teachers that I have had that have inspired me to learn more and ask question from elementary through graduate school.

Thank you to my PhD committee for providing feedback.
ABSTRACT

ENHANCED WEB-BASED SUMMARY
GENERATION FOR SEARCH

Brent Wenerstrom

July 23, 2012

After a user types in a search query on a major search engine, they are presented with a number of search results. Each search result is made up of a title, brief text summary and a URL. It is then the user’s job to select documents for further review. Our research aims to improve the accuracy of users selecting relevant documents by improving the way these web pages are summarized. Improvements in accuracy will lead to time improvements and user experience improvements.

We propose ReClose, a system for generating web document summaries. ReClose generates summary content through combining summarization techniques from query-biased and query-independent summary generation. Query-biased summaries generally provide query terms in context. Query-independent summaries focus on summarizing documents as a whole. Combining these summary techniques led to a 10% improvement in user decision making over Google generated summaries.

Color-coded ReClose summaries provide keyword usage depth at a glance and also alert users to topic departures. Color-coding further enhanced ReClose results and led to a 20% improvement in user decision making over Google generated summaries.

Many online documents include structure and multimedia of various forms such as tables, lists, forms and images. We propose to include this structure in web page summaries. We found that the expert user was insignificantly slowed in decision making while the majority of average users made decisions more quickly using summaries including structure without any decrease in decision accuracy.

We additionally extended ReClose for use in summarizing large numbers of tweets in tracking flu outbreaks in social media. The resulting summaries have variable length and are
effective at summarizing flu related trends. Users of the system obtained an accuracy of 0.86 labeling multi-tweet summaries. This showed that the basis of ReClose is effective outside of web documents and that variable length summaries can be more effective than fixed length.

Overall the ReClose system provides unique summaries that contain more informative content than current search engines produce, highlight the results in a more meaningful way, and add structure when meaningful. The applications of ReClose extend far beyond search and have been demonstrated in summarizing pools of tweets.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>iii</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>ix</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>xi</td>
</tr>
</tbody>
</table>

## CHAPTER

1 INTRODUCTION

1.1 Search Result Pages on Search Engines .................................. 1
1.2 Summary Text in Search Results ............................................ 2
1.3 Automatic Summarization .................................................... 5
1.4 HyperText Markup Language (HTML) ......................................... 6
1.5 Dissertation Overview ....................................................... 7
1.6 Contributions ........................................................................... 8

2 RESEARCH TRENDS IN WEB-BASED SUMMARY GENERATION 10

2.1 Automatic Summarization ....................................................... 10
2.2 Query-Biased Summarization .................................................. 16
2.3 Multi-Document Summarization ............................................... 21
2.4 Summarizing Social Media ..................................................... 27
2.5 Alternative Search Result Displays ....................................... 32
2.6 Reorganizing Search Results ................................................ 39
2.7 User Studies ............................................................................ 43
2.8 Directions and Suggestions for Further Developments .................. 46

3 RECLOSE SYSTEM 48

3.1 Motivation for ReClose Summaries .......................................... 48
3.2 ReClose Approach ................................................................. 51
  3.2.1 Query-Independent Ranking ............................................... 52
  3.2.2 Query-Biased Ranking ....................................................... 57
3.2.3 Combining Sentence Rankings ........................................... 59
3.3 ReClose Experimental Results ............................................. 61
  3.3.1 Query-Independent Module Parameter Tuning ................. 62
  3.3.2 Query-Biased Module Parameter Tuning ......................... 63
  3.3.3 Combining Modules Parameter Tuning ............................... 65
  3.3.4 Survey of User Expectations ...................................... 66
  3.3.5 Survey Participants ............................................. 68
  3.3.6 Survey Analysis ............................................. 69
  3.3.7 Discussion of Experimental Results ............................ 71

4 COLOR-CODED RECLOSE SUMMARIES .................................... 74
  4.1 Motivation for Color-Coded ReClose Summaries ............... 74
  4.2 Color-Coded ReClose Description .................................. 77
    4.2.1 Color-Coded Keywords ..................................... 77
    4.2.2 Flagged Words ............................................. 79
  4.3 Color-Coded ReClose Experimental Results ....................... 82
    4.3.1 Survey Participants and Survey Design .................... 82
    4.3.2 Summary Data ............................................. 83
    4.3.3 Results and Discussion .................................. 83

5 SUMMARIES PRESERVING DOCUMENT STRUCTURE ....................... 88
  5.1 Motivation for Summaries Preserving Document Structure .... 88
  5.2 Traditional Summaries within Search ............................. 90
    5.2.1 Preserved HTML in Summaries ................................ 91
    5.2.2 HTML Structure and Multimedia ................................ 91
    5.2.3 Proposed ReClose Extension .................................. 93
    5.2.4 HTML Entity Extraction Module .................................. 94
  5.3 User Evaluation of Summaries Preserving Document Structure .... 95
    5.3.1 Participants in the Evaluation ................................ 97
  5.4 Summaries Preserving Document Structure Experimental Results .... 97
    5.4.1 Decision Time Analysis ..................................... 98
    5.4.2 Selection Accuracy Analysis .................................. 101
    5.4.3 Participant Preferences by Summary Type .................... 102

6 VARIABLE LENGTH, MULTI-TWEET SUMMARIZATION .................... 105
  6.1 Motivation for Multi-Tweet Summaries .............................. 105
    6.1.1 Motivation for Flu Outbreak Detection through Multi-Tweet Summaries 105
6.2 Twitter ................................................................. 106
6.3 ReClose Extensions for Multi-Tweet Summarization .................. 107
  6.3.1 Tweet Gathering ................................................. 108
  6.3.2 Preprocessing Tweets using Topic Modeling ................. 108
  6.3.3 Tweet Clustering ............................................... 111
  6.3.4 Cluster Summarization ...................................... 112
  6.3.5 Presentation of Summaries .................................. 113
6.4 Multi-Tweet Summary Experimental Results ............................ 114
  6.4.1 Data Description ............................................... 114
  6.4.2 Establishing the Purity of Tweet Clusters .............. 115
  6.4.3 Closeness Centrality Distance Metric Selection ........ 117
  6.4.4 Evaluation of Variable Length Summaries ............. 120
  6.4.5 Temporal Evaluation of Tweet Topics ................. 122
  6.4.6 Human Evaluation of ReClose Multi-Tweet Summaries .... 125

7 CONCLUSION .................................................................. 128

8 DISCUSSION AND FUTURE WORK .................................. 132
  8.1 ReClose Discussion ................................................ 132
  8.2 Extensions Discussion ............................................ 135
  8.3 Future Work .......................................................... 137

REFERENCES .................................................................... 139

CURRICULUM VITAE ........................................................ 152
# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Comparison of NetSum against the baseline (first sentences of news articles) for ROUGE-1 and ROUGE-2 scores where bold indicates significance under paired tests [112].</td>
</tr>
<tr>
<td>2</td>
<td>Results from Haghighi and Vanderwende [48] of a manual pairwise evaluation of two methods. Haghighi's method is &quot;HierSum.&quot;</td>
</tr>
<tr>
<td>3</td>
<td>Comparison of three sentence ordering techniques conducted by Barzilay et al. [6].</td>
</tr>
<tr>
<td>4</td>
<td>Comparison of 9 different multi-tweet summarization methods made by Inouye and Kalita [56].</td>
</tr>
<tr>
<td>5</td>
<td>Closeness scores for sample graph.</td>
</tr>
<tr>
<td>6</td>
<td>Names and descriptions are listed of features used for query-biased sentence ranking. Minimums and maximums are calculated from the data set described in Section 3.3.2.</td>
</tr>
<tr>
<td>7</td>
<td>A comparison of LexRank, SumGraph, closeness centrality and sentence order individually and together are all shown based on ROUGE-1 scores. Results marked with α differ significantly from that of Closeness + Sentence Order using a two-sided t-test and an α level of 0.05.</td>
</tr>
<tr>
<td>8</td>
<td>R-Precision scores for linear regression (LR), REPTree, and gradient boosted decision trees (GBDT) models. In parentheses are the input features to the model which may include: query overlap (qo), meta description (m), contains verb (v), synonym overlap (syn), or any of the other features. “All” means all 8 possible features.</td>
</tr>
<tr>
<td>9</td>
<td>Example sentences and similarity scores. Punctuation and stop words are removed and stemming performed before scores are calculated.</td>
</tr>
<tr>
<td>10</td>
<td>Survey results comparing query-biased (Google), query-independent and ReClose summaries. (Percentages may not add exactly to 100 per row due to rounding.)</td>
</tr>
<tr>
<td>11</td>
<td>Summary evaluation only considering false positives, false negatives and accuracy.</td>
</tr>
<tr>
<td>12</td>
<td>The accuracy of Google summaries based on a summary’s inclusion of the top query-independent sentence.</td>
</tr>
<tr>
<td>Number</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>13</td>
<td>The accuracy of query-independent summaries based on a summary's inclusion of the top query-biased sentence.</td>
</tr>
<tr>
<td>14</td>
<td>Colors used to create the color scale.</td>
</tr>
<tr>
<td>15</td>
<td>An incomplete list of the words found in 30% or more of the search results for the query <em>algorithms</em> after stemming and the removal of stop words.</td>
</tr>
<tr>
<td>16</td>
<td>Disappointment counts and percentages for three summary techniques.</td>
</tr>
<tr>
<td>17</td>
<td>Click precision and recall comparison.</td>
</tr>
<tr>
<td>18</td>
<td>Expected and actual relevant content for web pages with a query keyword count of 60+.</td>
</tr>
<tr>
<td>19</td>
<td>Expected and actual relevant content for documents that would (Flaggable) and would not (Not Flaggable) have summaries with flagged terms.</td>
</tr>
<tr>
<td>20</td>
<td>Comparison summaries taken from a traditional search engine with and without structure.</td>
</tr>
<tr>
<td>21</td>
<td>Comparison of accuracies between summary types and evaluation groups.</td>
</tr>
<tr>
<td>22</td>
<td>Comparison of precision and recall scores between summary types and evaluation groups. Bolded values are significantly better using Fisher's test comparing to the opposing pool.</td>
</tr>
<tr>
<td>23</td>
<td>Preferences of each user evaluation group and example comments for each group. Bolded values are significantly better than 50% using a binomial test and alpha of 0.05.</td>
</tr>
<tr>
<td>24</td>
<td>Three example tweets on the left observed for Seattle, WA on Nov. 3, 2011. These three tweets are preprocessed using LDA into topic probabilities. (Rows will not exactly add to 1 due to rounding.)</td>
</tr>
<tr>
<td>25</td>
<td>Top 5 words per topic as discovered by LDA on US tweets related to flu for Nov. 3, 2011.</td>
</tr>
<tr>
<td>26</td>
<td>Measures of purity for topic sizes of 5, 7, 10 and 12 and distance measures Euclidean and cosine on Seattle tweets.</td>
</tr>
<tr>
<td>27</td>
<td>The results of varying the distant metric for ranking tweets to summarize clusters.</td>
</tr>
<tr>
<td>28</td>
<td>Comparison of cosine distance with a tfidf transform against word overlap distance on April data.</td>
</tr>
<tr>
<td>29</td>
<td>Break down of tweet level accuracy.</td>
</tr>
<tr>
<td>30</td>
<td>Break down of cluster level accuracy.</td>
</tr>
<tr>
<td>31</td>
<td>Clustering and tweet accuracy results by judge.</td>
</tr>
</tbody>
</table>
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>FIGURE</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>An example search result for the search University of Louisville on Google on May 16, 2011.</td>
</tr>
<tr>
<td>2</td>
<td>The top search result for the search computer science taken from Google on May 16, 2011.</td>
</tr>
<tr>
<td>3</td>
<td>The top of the web page found at URL <a href="http://en.wikipedia.org/wiki/Computer_science">http://en.wikipedia.org/wiki/Computer_science</a> on May 16, 2011. Dashed line box used to emphasize the first sentence of the article.</td>
</tr>
<tr>
<td>4</td>
<td>The top search result for the search z-man games found on Yahoo on May 16, 2011.</td>
</tr>
<tr>
<td>5</td>
<td>One tag taken from the HTML of the web page found at <a href="http://www.zmangames.com/">http://www.zmangames.com/</a> on May 16, 2011.</td>
</tr>
<tr>
<td>6</td>
<td>The top search result for the search university of louisville taken from Bing on May 16, 2011.</td>
</tr>
<tr>
<td>7</td>
<td>The Open Directory Project listing for the home page of the University of Louisville as seen on May 16, 2011 [1].</td>
</tr>
<tr>
<td>8</td>
<td>A rendering of an HTML table in 8(a) and the HTML tags used to create it in 8(b).</td>
</tr>
<tr>
<td>9</td>
<td>Example graph produced by Mihalcea for sentence extraction from a newspaper article [87].</td>
</tr>
<tr>
<td>10</td>
<td>A simple example of computing the DomRank ($R_D$) of text segments based on HTML node location [104].</td>
</tr>
<tr>
<td>11</td>
<td>Google News as seen on July 6, 2011.</td>
</tr>
<tr>
<td>12</td>
<td>Weighted graph of text fragments as used by Varadarajan et al. [117]</td>
</tr>
<tr>
<td>13</td>
<td>An example summary produced by the RIPTIDES information extraction summary system [125].</td>
</tr>
<tr>
<td>14</td>
<td>Characterizations of two Twitter users @w3c and @oprah by Ramage et al. [103].</td>
</tr>
<tr>
<td>15</td>
<td>Timeline of related tweets to the Marseille fire by hour as used by De Longueville et al. [25].</td>
</tr>
<tr>
<td>16</td>
<td>Twitinfo summary of an event as documented by Marcus et al. [82].</td>
</tr>
</tbody>
</table>
Example search result including a classical summary, top sentences and a thumbnail [60].
(a) Example of a thumbnail of a web page. (b) Example of an enhanced thumbnail of the same web page. [94]
Search results with both text and image snippets [132].
Three examples of visual snippets and the pages from which they were created [113].
(a) Clustered view of terms for browsing. (b) View of clusters for term browsing. [138]
(a) FooCA table display of results. (b) FooCA graph display of results. [67]
Visual widgets showing a visual representation of results and allowing for more ways to refine a search [27].
2-D particle space representing search results and their social context [64].
RB++ interface to search results [137].
WaveLens interfaces which dynamically expand search result snippets [94].
Hierarchy example originating from work by Lawrie and Croft [69].
Top ranking sentence interface used in [127].
Examples of three different summary lengths used in a survey by Cutrell and Guan [22].
Two of the top 10 search results for the query *closeness centrality* taken from Google on Mar. 23, 2011.
The web page http://www.orgnet.com/sna.html with the left half of the paragraph on closeness centrality enlarged as seen on March 23, 2011.
The web page cent-ans.htm from the website www.sscnet.ucla.edu on March 23, 2011 with the question and table relating to closeness centrality enlarged.
Diagram of ReClose methodology for producing summary text.
A sample graph with weighted undirected edges.
Google and ReClose summaries generated for the query *closeness centrality* and the web page pictured in Figure 32.
Linear regression model used for ranking query-biased sentences.
Kernel density estimates of the distribution of summary lengths in characters comparing Google, query-independent and ReClose summaries.
Survey search results ready for participant evaluations.
Survey search results after evaluations were made.
User expectations of query-biased summaries that did or did not contain the top ranked query-independent sentence for the web page.
User expectations of query-independent summaries that did or did not contain the top ranked query-biased sentence for the web page.
A search result from the top 10 search results for the query *closeness centrality* on Google as seen May 11, 2011.


Color scale of keyword frequencies.

Process of flagging terms.

An example summary using color-coding and ReClose summary text generation for the query *building a database*.

Distribution of expected relevant content according to users divided by clicked and skipped documents.

Search engine summary with (b) and without (a) structure.

Text representation of an image in HTML and corresponding image.

Two search result summaries: (a) is the plain-text version observed in Bing and (b) is the summary preserving structure version using a list.

Histogram showing distribution of click decision timings.

Box plots of the participants with the five lowest median timings and the five highest median timings.

The average decision time of users using plain-text summaries are drawn as boxes. A line is drawn from each box to the average decision time when using summaries preserving document structure. (a) experienced abstract thinkers and (b) inexperienced abstract thinkers.

Each column shows all decision times per user, per summary type. The red diamond in each group shows the average decision time.

Generative process and distribution parameters of LDA.

Example tweet cluster summary taken from tweets in Louisville, KY for April 10, 2012.

Number of words overlapping when comparing tweets in Louisville on April 10, 2012.

NMI scores plotted against the number of clusters for each data set April 10-22.

This graph shows the average NMI achievable at each fixed number of clusters across all 13 days of data. We also plot the maximum possible average.

The number of tweets assigned to each of 10 topics using LDA from 2011 week 44 through 2012 week 18. Each data point comes from a randomly selected day during the week. Due to network issues data was not collect for 2012 weeks 1, 7, 8, and 13 and is estimated using the previous and following weeks. The weighted percent of doctor visits with influenza-like illness symptoms reported to the CDC for the same week is shown as a dashed line.
Only topics 8 and 10 are shown in this graph for 2011 week 44 through 2012 week 18 to show the two sudden topic specific spikes.

An example of a summarized cluster of tweets unrelated to the flu (0/20 about flu). Taken from Louisville tweets on April 18, 2012. Twitter user names removed.

An example of a summarized cluster where the tweets are highly related to the flu (36/38 about flu). Taken from Louisville tweets on April 18, 2012. Twitter user name and swear word removed.

Plot showing total tweets from April 10-22 in Louisville compared to the number of flu related tweets.
CHAPTER 1

INTRODUCTION

The automatic generation of summaries best serve their purpose within search results by remaining short and concise. Automatic summarization is limited by the text available within a document and must take advantage of the search query and term usage within a document.

1.1 Search Result Pages on Search Engines

Internet users come to search engines for a variety of reasons. All of them want to find web pages or information that is hopefully contained within web pages. The process that a search engine user goes through when using a search engine is the following:

- A user formulates their search using one or more keywords.
- Those keywords are entered into a search engine’s search box.
- The search engine compiles a list of web pages automatically found to be relevant to the user’s search.
- The search engine orders the list of web pages before returning them to the user.
- A list of the top ranking search results are displayed to the search engine user.

Nearly all searches are performed using text search. The input to the whole process is text in the form of keywords. Keyword searches most often consists of 1, 2 or 3 keywords.

Search engines regularly crawl through the known space of web pages to compile a search index. An index allows for the search to be performed using only the computers housed at the facilities of a search engine.

Upon completion of compiling a list of web pages to return to a search engine user, the search engine will return a page of search results. A single search result consists of three parts generally (see Figure 1). At the top of a search result is the title of the web page. The title, in the case of the most common format namely HTML, is extracted from the HTML. Titles are chosen by the website creator and may provide some hint about the general topic of a web page.
Figure 1. An example search result for the search University of Louisville on Google on May 16, 2011.

The center of a search result commonly provides a brief text summary of the web page or text used to verify the usage of keywords within a web page. More will be provided in Section 1.2 about the sources of these summaries.

Lastly, an abbreviated URL is provided to the user. This provides the user with the source of the web page listed. In the cases where a user is familiar with the host of a website, the user will have a better understanding of the context within which the web page exists. For example, those familiar with the website Wikipedia, will have some idea about what a web page will look like and what it will contain if one of the search results is from Wikipedia.

Search engine users scan the list of search results linearly from top to bottom [22, 47]. Users will spend more time viewing the top two search results than any of the other search results on a web page [22, 47]. With the levels of trust held by most search engine users, users will often click on the top search result in the search results page [22].

As users view each search result they must make a decision. Either they click through the link for a search result and view more information from the source of the search result or they skip that particular result. A user will spend less than one second on average per search result [47]. Many factors may weigh upon a user's decision to click. In the case of relevance judges deciding if a particular page is relevant to a particular search, as many as 80 factors may contribute to this decision [109]. Making correct decisions as to click through or skip a result affects the overall user experience. Users that skip over a relevant search result lose the chance for that search to view a useful web page to their search needs. On the other hand a user that clicks through to a web page that is irrelevant will have wasted time and will be disappointed with that particular search result. The presentation of search results affects the accuracy of the user in their clicking accuracy, and thus the search engine user experience.

1.2 Summary Text in Search Results

Summary text within a search result is obtained from three main sources [86, 42, 44, 134]. The first main source is the web page itself. This is the most obvious source of text for most documents. Often text is used which contains one or more of the keywords searched within context in the document. This is called a query-biased summary, where part of the query or the keywords
Computer science - Wikipedia, the free encyclopedia

Computer science or computing science (abbreviated CS) is the study of the theoretical foundations of information and computation and of practical ...

Figure 2. The top search result for the search computer science taken from Google on May 16, 2011.

Figure 3. The top of the web page found at URL http://en.wikipedia.org/wiki/Computer_science on May 16, 2011. Dashed line box used to emphasize the first sentence of the article.

are considered when summarizing the document.

As an example consider the top result for the search computer science as shown in Figure 2. The summary is as follows: “Computer science or computing science (abbreviated CS) is the study of the theoretical foundations of information and computation and of practical …” In Figure 3 is pictured the Wikipedia paged linked to by the search result pictured in Figure 2. A dashed box has been added to Figure 3 to emphasize the first sentence, which is the very sentence used to summarize this web page.

A second source of summary text for search is what is called the meta description of a web page. Within the HTML of a web page a website creator may add a meta tag. Meta tags are HTML tags that allow for information about web pages to be encoded into an HTML page. Specifically one may encode a meta “description” of a web page. This generally is a human generated summary of the contents of a web page generated by the website creator. This text is not shown to visitors to the web page, but is easily extracted by crawlers or automated programs sent out by search engines to download web page content for use in the search engine index.

As an example of the use of meta descriptions consider the top search engine result
Welcome to the Z-Man Games - publisher of games. Games are our business and our pleasure.

Figure 4. The top search result for the search *z-man games* found on Yahoo on May 16, 2011.

```
<meta name="description" content="Welcome to the Z-Man Games - publisher of games. Games are our business and our pleasure."/>
```

Figure 5. One tag taken from the HTML of the web page found at http://www.zmangames.com/ on May 16, 2011.

Pictured in Figure 4 for the search *z-man games*. The summary text used is the following:

"Welcome to the Z-Man Games - publisher of games. Games are our business and our pleasure."

This text is found nowhere on a browser rendering of the web page http://www.zmangames.com. However, within the HTML you find the following bit of HTML shown in Figure 5. This meta tag provides us with the exact text used for the web page summary used by Yahoo in Figure 4.

Lastly, a major source of text used to summarize web pages comes from the Open Directory Project (ODP) found at http://www.dmoz.org. The ODP as described by themselves:

"The Open Directory Project is the largest, most comprehensive human-edited directory of the Web. It is constructed and maintained by a vast, global community of volunteer editors." Within this directory are a number of the better known websites within a number of categories. Each website entry is a short description written by a volunteer editor of the ODP. These summaries provide a more objective description and are written by hand rather than by machine. Search engines will on occasion use the short summaries found within the ODP.

Figure 6. The top search result for the search *university of louisville* taken from Bing on May 16, 2011.
• University of Louisville - A state-supported urban university in Louisville, Kentucky.

Figure 7. The Open Directory Project listing for the home page of the University of Louisville as seen on May 16, 2011 [1].

As an example consider the top search result for university of louisville on Bing as shown in Figure 6. Here we see a very short summary used to describe the official home page for the University of Louisville with the text, “A state-supported urban university in Louisville,...” No where on the University of Louisville’s home page do we find this text. Also, the HTML for this website does not contain a meta description tag. Instead we must look to the ODP to find this text. The entry on ODP for the University of Louisville can be seen in Figure 7.

In addition to these three sources, Yahoo may also extract text from the Yahoo Directory [134] to summarize web pages. The Yahoo Directory is very similar to the ODP but is maintained by Yahoo.

Understanding the sources from which current text summaries come from help us to better understand current summary techniques. Later we will take advantage of the available text to model current search engine approaches combining the summaries of Google, Yahoo and Bing (see Section 3.2.2).

1.3 Automatic Summarization

Since the invention of the printing press in 1440 by Johannes Gutenberg and its adoption in Europe, there has been more printed text than any one person can digest. This problem has only been exasperated by improvements to the printing process and eventually the ability to publish electronically. Now the information overload is so great that a single researcher in a specialized field such as data mining is not able to read all published works related to their research.

Researchers have sought to aid in the information overload problem using automatic summarization. Automatic summarization takes as input one or more documents and returns usually text that is a shortened representation of the input text. It is required that the output text is shorter and in many cases much shorter. For example online search engines will limit the number of lines of text to represent a web page to two lines of text. In the case that a single web page represents the entire content of a book then a summary containing one or two paragraphs would be equal to < 0.01% of the original volume of text.

Current approaches generally divide the contents of a document into multiple segments. This may be done at the sentence level. Then each segment of text is scored in some manner using statistics [77], machine learning [68] or some other means [30] to obtain a score of the importance of that text to the document. After text has been scored, then for the extractive approach to
summary generation, segments of text are put together into a single summary. Usually a summary is the highest ranking text segments. However, some text segments may not be added due to redundancy with previously added text [17].

The output of the summary may be used in a system for browsing large document collections or in search engine results.

1.4 HyperText Markup Language (HTML)

The HyperText Markup Language (HTML) is a way to encode web documents with structure, formatting, links to other pages and more. It was originally proposed by Tim Berners-Lee at CERN [10] as a way to share documents within CERN. Today it provides the formatting of a majority of web documents.

An HTML document is made up of a number of tags of the form `<html>`. Usually these tags are paired with the “closing” tag containing the ‘/’ character. As an example `<head>` would be the opening tag and `</head>` would be the ending tag. It is not necessary to pair tags in this manner, some tags may have no match within an HTML document. When tags are paired there may be text, comments or more tags between the tag pairs. When tags are contained between tag pairs this creates a nesting of tags enabling the complex structure found in modern day HTML.

A variety of HTML tags exist. There are several tags that provide text formatting such as making text bold `<b>` or italics `<i>`. Another tag provides links to other web documents. This “anchor” tag takes the following form: `<a href="http://louisville.edu/">`. The attribute “href” provides the value of the other web document pointed to by the anchor tag. Tags that embed images follow a form similar to the anchor tag. An image tag works as follows: `<img width=200 height=200 src="/home_img.jpg">`, where “width” provides the number of pixels wide to use displaying the image, height does the same for the “height” of the image and “src” gives the URL of the image file to display when rendering the image tag. See Figure 8 for an example of
the construction of an HTML table. Figure 8 shows the use of the `<table>` tag for starting and ending tables, the `<th>` tag for table headings, the `<tr>` tag starts a new table row, and the `<td>` tag surrounds a single cell in the table. Many more tags exist than are described here. These examples provide the main tags of concern for this dissertation.

The purpose of the HTML tags is to provide formatting to the end user to a web page. This formatting is performed when a web browser “renders” an HTML document. The rendering process starts with the browser requesting the web page to display, parsing the web page to find formatting tags, displaying the content with this formatting, fetching additional resources such as images that are embedded in the web page, and in many cases processing style guides in the form of Cascading Style Sheets (CSS) and JavaScript code. The result is what the user sees inside of their browser when accessing a URL containing an HTML document.

1.5 Dissertation Overview

In this dissertation we present our system for summary generation called ReClose. ReClose combines the rankings from a REgression model and CLOSEness centrality. The main purpose of ReClose is to generate summaries of web documents for use in search result pages. We improve on current summary generation techniques in three main ways: improved summary text, improved query highlighting and preserving structure from original documents. We then test our ReClose approach for summarizing multiple tweets with variable length summaries.

Before elaborating on the details of our system we first embark on an explanation of past research as it relates to our topic of summarization. We provide detailed explanations of a number of key papers in query-biased summarization, multi-document summarization, summarization of social media, alternative search displays, reorganizing search results and user studies in search in Chapter 2.

Current summary techniques (employed by Google, Yahoo! and Bing) are query-biased in that the summary text is generally picked mainly to provide context to the query keywords. Query-biased summaries are effective at providing the users with insight into why a particular document was selected for their query. However, these summaries do not always provide enough information that a user can understand the intent of the document. The ReClose system enhances current summary techniques by creating a multi-part summary that includes a query-biased summary portion. The query-biased summary part uses a regression model trained on summaries from Yahoo!, Google and Bing combined. It also includes a query-independent portion aimed at summarizing the main topic of a document. The query-independent module uses graph theory in the form of closeness centrality to rank sentences as though they were nodes in a graph. We believe that by combining query-biased and query-independent summary techniques users will be able to more accurately decide which web pages are relevant for a given search. Our text selection
process for ReClose is detailed in Chapter 3.

Most summaries provided by a Google or a Bing will have keywords given in bold. This allows for users to quickly see which documents include which keywords. It is possible to include more information with the highlighting of keywords. Our ReClose system uses various shades of blue to highlight keywords. In this way the shade of the keyword can provide information about the extent of relevant information may be present on a web page. We also flag terms to warn users when a web page’s topic departs significantly from the overall themes of the other search results for their query. We believe that by adding information to summaries in the form of color-coded keywords, that users will be more accurate in judging document relevance. We detail our color-coding methodology in Chapter 4.

As explained in Section 1.2, summary text is often extracted from a web page. During this extraction process any formatting, structure or images that are part of that text in the original document are removed. For example table cells in Google summaries are separated with commas. By discarding this structure information is being lost. We believe that preserving structure from the original document improves the accuracy and speed of the decision making process for the relevancy of web documents by users of search. We therefore include structured text, images, buttons and lists from the original document in our ReClose system. More information about preserving structure in summaries is explained in Chapter 5.

Lastly, we show that our ReClose system can be extended to summarizing multiple tweets. We show that our approach to summarization is effective on the extremely short Twitter documents called “tweets.” Our approach first clusters tweets by topic, where topics are discovered using Latent Dirichlet Allocation and clustering is performed by hierarchical clustering. Then we summarize each cluster using tweets ranked by ReClose’s closeness centrality approach. This produces varying length summaries. These summaries may be used for a variety of purposes. We test our approach for use by health officials to verify that when an alarming number of tweets for tracked keywords is found, that a flu outbreak is really occurring. We believe that varying lengths of summaries will provide more accurate summaries than fixed length summaries. We also believe that health officials will be able to more accurately comprehend the outbreak situation by observing summaries of multiple tweets, over simply counts based on keyword searches of tweets. We detail multi-tweet summarization in Chapter 6.

1.6 Contributions

The goals of this dissertation are to explain an improved summary generation technique, show its effectiveness and help users of search to be more successful. Users of search engines currently make a large number of decisions when faced with a long list of search results. Each search result presents a new challenge. Is a given document going to be relevant for a particular
user’s search needs? That question is answered numerous times. The user’s experience is degraded by wrong decisions. The search system needs to provide the user with enough material to make accurate decisions. We aim to improve the information presented to users. This will mean that less irrelevant links are clicked, which will mean less time wasted. The more quickly users are able to find what they are looking for the better it will reflect on the search engine used. We hope that by improving the methods used in present document summaries in search, that the whole user experience is noticeably better. We elaborate on our successes in Chapter 7.

There are many contributions of this work. This work shows the effectiveness of combining query-independent and query-biased summary techniques. It provides improved results on a data set of CNN news articles producing a ROUGE-1 score of 0.501. This work shows an effective approach to building on the work of traditional search engines such as Google, Yahoo and Bing.

The text summarization portion of this work rivals that of successful industry leading experts. In a survey we conducted users had accurate expectations 55% of the time using ReClose text and 50% of the time using Google summaries. We further improved upon these results with a unique approach to color-coding query keywords and through flagging in red terms that departed from the query topic. Color-coded ReClose summaries achieved a click precision of 80% compared to Google text summaries achieving 66% on that same summary.

We considered the preservation of HTML structure from original documents to the summary which had not fully been done before. This resulted in a significant number of individuals in one group of our survey being faster with summaries that included structure compared to plain-text summaries that could take advantage of structure and did not. In both survey groups a majority of individuals preferred the new summaries that preserved structure to those that did not.

We successfully extended ReClose to summarize Twitter tweets. We summarized potential flu outbreak alarms. We achieved this through a new approach to summarizing tweets which first clustered tweets by topic and then summarized these clusters. The cluster summaries used closeness centrality to rank the tweets just as ReClose had done in selecting text for usage in summaries.

Overall this work presents new approaches and new combinations of algorithms to present the user with summaries more representative than before this work.
CHAPTER 2

RESEARCH TRENDS IN WEB-BASED SUMMARY GENERATION

In this section we collect various research efforts that relate to web-based summary generation. We have classified the research mentioned and provide commentary on the research trends and gaps in the research where further research work could be performed.

2.1 Automatic Summarization

The first category listing for research efforts is automatic summarization. Automatic summarization has been studied since the late 1950's and includes many works that predate the search engines and the need for web-based summaries. Automatic summarization provided the groundwork for web-based summarization, though in many cases the sources for summarization and goals differ. We will not enumerate all works under the umbrella of automatic summarization, but will instead highlight those efforts that most relate to the proposed research in this proposal.

The first proposed automatic summarization research was performed by Luhn in 1958 [77]. His approach used word frequencies to determine the most important text to preserve when summarizing text. Later that same year Baxendale used the position of text to determine which text to extract for summarization [9]. The key idea is that humans often organize sentences within a document. For example news articles are often best summarized by the first couple of sentences within the article [112] due to the fact that we tend to start writing in generalities to give the overall view of a story or article and then become more specific through the body of a work. This was the beginning of automatic summarization and showed the use of word and sentence features in determining good candidate sentences for summarization.

Skipping ahead in time, in 1995 Kupiec et al. [68] were the first to use machine learning for the task of automatic summarization. Their approach used several features including word frequencies from Luhn [77] and sentence position from Baxendale [9]. Kupiec supplemented these features with sentence length and the presence of uppercase words. Kupiec's model considers the probability of each sentence $s$ being included in summary $S$ using $k$ features $F_j; j = 1 \cdots k$. Using the naive Bayes assumption of feature independence, Kupiec used the following:
\[ P(s \in S|F_1, F_2, \cdots F_k) = \frac{\prod_{j=1}^{k} P(F_j|s \in S)P(s \in S)}{\prod_{j=1}^{k} P(F_j)} \]  

(1)

Kupiec trained a naive Bayesian model on a number of human generated summaries. These professionally generated summaries referenced sources with engineering texts. Kupiec's model selected the same sentences as the experts 84% of the time. Several approaches followed which based summary generation on functions learned through training machine learning algorithms on human generated data sets. Some of these machine learning based approaches will be seen in the web-based summary generation category.

Osborne [92] in 2002 showed that the use of log-linear models with class priors could achieve perfect f2 scores of 100 when given perfect information whereas the naive Bayes classifier at best achieved an f2 score of 85. The log-linear model used was the following:

\[ \text{label}(s) = \arg \max_{c \in X} F(c) \exp\left(\sum_{i} \lambda_i f_i(c, s)\right) \]  

(2)

Here we are using a classifier which finds the label for a sentence s with the highest unnormalized maximum entropy score. The possible class labels \( c \in X \) are either to include the sentence in the summary or to not include the sentence. \( F(c) \) is a function over the class labels that increases the chance of inclusion, since this particular model without the class function tends to exclude more sentences than it should. \( F(c) \) allows for sum tuning of the output of the model. This model uses a number of features \( f_i(c, s) \) that are real-valued functions. Each of these features has a real-valued weight \( \lambda_i \) associated with it. These weights are optimized either through the use of a closed form solution or in this paper through the use of conjugate gradient descent.

ROUGE-1 is a common metric for evaluating summaries. This is an unigram co-occurrence metric for comparing ideal summaries to generated summaries. Unigram co-occurrence metrics have been shown [74] to correlate with human evaluations. ROUGE-1 is calculated as shown below adapted from [112].

\[ \text{ROUGE-1} = \frac{\sum_{\text{gram}_j \in \text{R} \cap \text{S}_i} \text{Count}(\text{gram}_j)}{\sum_{\text{gram}_j \in \text{R}} \text{Count}(\text{gram}_j)} \]  

(3)

Here \( \text{R} \) refers to the reference or ideal summary. \( \text{S}_i \) is the generated summary. In ROUGE-1, \( \text{gram}_j \) refers to unigrams or single words. Another way of stating ROUGE-1 is that it is the number of words in the ideal summary that are also contained in the generated summary. A score of 1 would be a perfect match. Stop word removal and stemming were not used when computing ROUGE-1 scores. ROUGE-2 would mean that instead of unigrams, bigrams or two consecutive words would be used in place of the unigrams.

Svore et al. [112] created an approach to automatic summarization called NetSum which used the neural network based RankNet algorithm [14]. A number of algorithms have existed for summarizing single-document news articles. Previous to this article, no algorithm was able to beat
Comparison of NetSum against the baseline (first sentences of news articles) for ROUGE-1 and ROUGE-2 scores where bold indicates significance under paired tests [112].

<table>
<thead>
<tr>
<th>System</th>
<th>Avg. ROUGE-1</th>
<th>Avg. ROUGE-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.4642 ± 0.0084</td>
<td>0.1726 ± 0.0064</td>
</tr>
<tr>
<td>NetSum</td>
<td>0.4956 ± 0.0075</td>
<td>0.1775 ± 0.0066</td>
</tr>
</tbody>
</table>

TABLE 1

to a statistically significant level the baseline of using the first few sentences from news articles. The data used consisted of a number of news stories copied from CNN.com. Each of these news articles had 3-4 bullet point sentences which are human selected summarizing sentences used to draw users into the news articles. These human generated sentences, which sometimes were extracted from the news article and sometimes were newly written sentences, were used as the gold standard for summarizing these news articles. Svore et al. measured their generated summaries using ROUGE-1 and were found to be significantly better than the baseline (see Table 1).

In addition to scoring each sentence independently, other researchers found the inter-sentence dependencies to be useful in obtaining good results. In 2001 Conroy and O'Leary [20] introduced the use of a hidden Markov model (HMM) to take advantage of inter-sentence dependencies within a document. This approach achieved F1 scores in agreement with human summaries in the range of 51-58.

Goldstein et al. [40] in 1999 show a typical approach to summarizing news stories with a weighted scoring of both statistical and linguistic features for ranking sentences. On the TIPSTER dataset the authors were able to achieve an F1 score of 0.65 on long queries and 0.53 on short topic queries.

Mihalcea [87] adapted HITS [66], Positional Power Function [55] and PageRank [95] to weighted graphs to extract sentences for summarization. Their approach brought graph theory to bear on the problem of query-independent, automatic summarization. Graphs were created using sentences as nodes and edges were established with a similarity function based on the lexical overlap of tokens normalized by sentence length. An example of the graphs produced is shown in Figure 9. Results were measured using undirected graphs, directed graphs where previous sentences point to later sentences and directed graphs with edges pointing from later sentences to previous sentences. The results of their system, called TextRank, gave results (ROUGE 0.5008) competitive with the top competitors (ROUGE 0.5011) in the DUC 2002 competition.

Erkan and Radev [30] used a measure based on PageRank [95], which they called LexRank. This was another graph theory based query-independent approach to summarization. The basic idea was that by using a similarity metric such as cosine similarity, each sentence that has a similarity above some threshold formed a link between the two sentences, in this case it was
Figure 9. Example graph produced by Mihalcea for sentence extraction from a newspaper article [87].

Undirected and unweighted. A graph was formed by continuing this process of comparing each sentence against each other sentence. A score for each node was then computed using PageRank. The formula used for PageRank is the following:

\[ p(u) = \frac{d}{N} + (1 - d) \sum_{v \in \text{adj}[u]} \frac{p(v)}{\deg(v)} \]  

(4)

where \( d \) is a "damping factor" usually chosen between 0.1 and 0.2 [95]. \( N \) is the number of nodes in the graph. \( p(u) \) and \( p(v) \) are PageRank scores for nodes \( u \) and \( v \) respectively. The \( \deg(v) \) computes the out degree of node \( v \) and \( \text{adj}[u] \) provides the set of nodes adjacent to \( u \). This provides a recursive definition that is often solved iteratively. This approach provided a single, query-independent summary for multiple documents. Their results showed that their approach outperformed centroid similarity scoring when compared on DUC 2003 and 2004 achieving ROUGE-1 scores as high as 0.4445.

Building upon the work of Erkan and Radev, Patil and Brazdil [97] created a query-independent, single document summarizer which instead of using PageRank to rank sentences in a graph, used closeness centrality. Closeness centrality found for each node the average distance to all other nodes. Their approach called SumGraph, used one minus the cosine similarity between sentences as the sentence dissimilarity. They then pruned all edges where a shorter path existed in the graph. Their summarizer used the global closeness centrality combined with the document order of sentences to create 100 word summaries. They tested SumGraph against LexRank and other familiar algorithms. Their results showed their approach achieving the highest ROUGE-1 scores for DUC 2001 (0.4543) and DUC 2002 (0.4842). For a discussion of
differences see Chapter 3.

L. Li et al. [71] in 2009 create a model for scoring text summaries based on structural SVM’s. The SVM is modified to include three additional constraints for diversity, coverage and balance. These constraints consider the relationships between the multiple sentences selected in the summary. Consider a document \( x = x_1, x_2, \ldots, x_n \in \mathcal{X} \), where \( x_i \) represents the \( i \)-th sentence of this document and \( \mathcal{X} \) represents the space of all documents. Each \( y_i \) represents a label for the \( i \)-th sentence. Summarization then is the task of selecting a subset \( y \) from the space of all possible subsets \( \mathcal{Y} \). The labeled training data is represented as follows:

\[
(x^{(i)}, y^{(i)}) | i = 1, \ldots, n
\]

Here this particular \( y^{(i)} \) is the set of labels which is the ground truth as to which sentences should be selected for summarization. The set of features in this data set is represented by \( \Psi(x, y) \). A loss function is used to measure the distance from the ground truth on a given subset of sentences \( y \):

\[
\Delta(y, \bar{y}) = \frac{2pr}{p+r}, p = \frac{< y, \bar{y} >}{< y, ar{y} >}, r = \frac{< y, \bar{y} >}{< y, y >}.
\]

The use of \( < a, b > \) denotes the number of common items between the two subsets \( a \) and \( b \).

The optimization problem can now be formulated as follows:

\[
\min_{w, \xi \geq 0} \frac{1}{2}||w||^2 + \frac{c}{n} \sum_{i=1}^{n} \xi_i,
\]

subjected to:

1. \( \forall i, \forall y \in \mathcal{Y} \setminus y^{(i)} : \xi_i \geq 0, \)
2. \( w^T \Psi(x^{(i)}, y^{(i)}) \geq w^T \Psi(x^{(i)}, y) + \Delta(y^{(i)}, y) - \xi_i, \)
3. \( w^T \Psi(x^{(i)}, y^{(i)}) \geq \sum_{y \in y^{(i)} \cap \mathcal{Y}} w^T \Psi(x^{(i)}, y) - \mu + \xi_i, \)
4. \( w^T \Psi(x^{(i)}, y^{(i)}) \geq w^T \Psi(x^{(i)}, y) + 1 - \| \sum_{y \in y} v_i \| - \xi_i, \)
5. \( w^T \Psi(x^{(i)}, y^{(i)}) \geq w^T \Psi(x^{(i)}, y) + \sum_{j=1}^{k} (u_j - \bar{u})^2 - \xi_i. \)

The optimization problem optimizes the weight vector \( w \) for the given training set \( (x^{(i)}, y^{(i)}) | i = 1, \ldots, n \). The variable \( c \) is used to adjust the trade off between the model complexity \( \frac{1}{2}||w||^2 \) and the sum of the slack variables \( \sum_{i=1}^{n} \xi_i \). The function \( \Psi(x, y) \) generates the set of features. Constraint 1 assures us that each slack variable will take on a positive value. Constraint 2 shows that each slack variable is greater than the difference of two solution subsets \( \Delta(y^{(i)}, y) \). Constraint 3 is aimed at increasing diversity. A new slack variable \( \mu \) is introduced increasing the diversity among chosen sentences with very similar attribute values. Constraint 4 quantifies the coverage of each subtopic within a document using the variable \( v_i \) to denote the
coverage of $y_i$ over all the subtopics of document $x$. Constraint 5 provides a measure of coverage of each subtopic by a possible solution $y$.

An independence graph is used to reduce the possible sentence combinations to consider as candidate summaries. The DUC 2001 data set was used. The features include a number of word frequency based features, number of words in each sentence, and the PageRank value of each sentence. The results show that the SVM with the additional constraints performs better than a number of machine learning approaches according to the ROUGE-2-R (0.574) and F1 score (0.475).

Many of the research efforts in automatic summarization have been focused on data sets provided by the Document Understanding Conference (DUC) found at http://www-nlpir.nist.gov/projects/duc/index.html and later the Text Analysis Conference (TAC). DUC was run from 2001-2007 and TAC has continued from 2008 to 2011 by the National Institute of Standards and Technology (NIST). The data sets provided by DUC and TAC for summarization tasks are based on the AQUAINT [45], TIPSTER [51], and TREC document collections. Each of these document collections are made up entirely of news articles. Each track within DUC and TAC for each year provided a set of documents and human judgments of the most relevant sentences or documents. The DUC and TAC provided data sets provide a standard against which different summarization methods may be tested using the same data and the same measures. However, this is at the cost of specifying which types of documents are to be summarized and how they are to be summarized. Using only news articles is limiting. Documents found on the Internet are often much messier and conform less in structure to the articles produced by professional news writers. This removes one difficulty of summarization. Additionally, there are many ways that documents may be summarized. Text only summaries are useful for many tasks, but some web pages may better be summarized using multimedia or tables.

Not all techniques for summarization are based on features of text entities. Some approaches require a deeper analysis of the text within a natural language processing framework [5, 84, 81]. Barzilay and Elhadad [5] in 1997 for example analyze a document to find lexical cohesion in the form of “lexical chains”. Lexical chains are formed by finding words in different sentences that are all related according to some thresholded measure. Barzilay and Elhadad used WordNet [32], an electronic thesaurus created at Princeton, to find similar words within a document. After analysis, each lexical chain is scored according to cohesion, size, etc. The importance of a sentence for summary creation is based on the chain scores.

When scoring sentences for inclusion in an automatically generated summary, often very similar sentences will score similarly. If there are similar sentences that all score highly, then it is possible that repetitious sentences will be included. To remedy this Carbonell and Goldstein [17] introduced the maximal marginal relevance (MMR) measure as a way to combine query relevance and information novelty. This approach boosted relevant content. It also penalized redundant
content when combining multiple text segments for summarization. MMR is defined as follows:

\[
MMR = \arg \max_{D_i \in R \setminus S} [\lambda Sim_1(D_i, Q)) - (1 - \lambda) \max_{D_j \in S} Sim_2(D_i, D_j)]
\]

where \( R \) is the ranked list of retrieved documents \( D_1, \ldots, D_n \) (could also be sentences within a document). \( S \) is a subset of \( R \) which have already been selected for display. \( R \setminus S \) refers to the documents retrieved but not yet selected for display. \( Sim_1 \) is the similarity measure used between documents and the query \( Q \). \( Sim_2 \) is also a similarity measure which may be the same as \( Sim_1 \) or different. It is used to compare two documents. \( \lambda \) is parameter to MMR which adjusts the preference for diversity (maximum when \( \lambda = 0 \)) and relevance (maximum when \( \lambda = 1 \)). Values of \( \lambda \) are valid when in the range \([0, 1]\).

From the cited sources we can see that automatic summarization has been around for more than 50 years with many innovations. Many of the techniques score text spans (words, sentences, etc.) based on a number features. These features may be combined automatically through a model fit to a particular data set. Other approaches may base their scoring on graph theory and the relationships among sentences or the chain words within a document to model topic discussion. Thus when documents use consistent patterns, such as using more general sentences within the first paragraph of a news story or repeating topic words throughout a document, certain algorithms will effectively find sentences or text spans that take advantage of some of these more common patterns. These citations also show that many of the efforts in automatic summarization have been news focused. News articles are professionally written, of similar length and made up of mostly text with a few pictures. News articles are not representative of all of the different types of documents that exist on the Internet or in other document collections. Many documents on the Internet contain much more image or video content than text. Many documents such as blog posts may not be well written or follow common writing patterns. Developments in the field are available through the messy content of the World Wide Web.

2.2 Query-Biased Summarization

Tombros and Sanderson [114] in 1998 conducted the first user study of query-biased summaries compared to query-independent summaries. In this study 20 users were each given a set of search tasks taken from TREC test collections and told to find as many relevant documents as they could with a time limit of 5 minutes per task. In this experiment 10 users were given query-independent summaries and 10 users were given query-biased summaries. Users were allowed to view document contents by clicking through to the document and were not limited to the text summaries. Users achieved statistically significantly higher precision (0.55 vs. 0.44) and recall (0.66 vs. 0.50) scores using query-biased summaries, viewed more search results per search task (22.6 vs. 20) and viewed far fewer documents in full (1.3% vs. 23.7%). This study was
instrumental in the movement of search engine summaries away from the first few sentences of a
document to query-biased summaries. Query-biased summaries are summaries chosen which
consider the query keywords when selecting text for the summary. This often means that sentences
are chosen which contain query keywords, but a query-biased summary does not always contain
available query keywords.

Approaches to query-biased text summary generation range from machine learning
[119, 85], to graph theory based approaches [104, 93] to matrix factorization [96]. Wang et al. [119]
in 2007 for example compare a ranking SVM against a classification SVM on TREC data for 2003,
where summaries are extracted manually for 175 documents from both the web documents and
anchor context. Additionally the authors compare performance of learners when only using
features from context or documents to using both sources of data. As hypothesized the ranking
SVM outperformed the classification approach to sentence selection and using both sources of data
outperformed using either source individually.

Metzler and Kanungo [85] compared three machine learning algorithms for the purpose of
sentence ranking for query-biased automatic summarization. They found that the gradient boosted
decision trees (GBDT) outperformed ranking support vector regression and ranking support
vectors on the TREC 2002 (0.4047), 2003 (0.7060), and 2004 (0.4806) Novelty track data when
measuring using R-Precision.

A GBDT is a combination of a basic regression trees \( f(x), x \in \mathbb{R}^N \). A tree partitions the
space made by explanatory variables into disjoint regions \( R_j : j = 1, 2, \ldots, J \). Each region is
associated with a leaf node in the regression tree. The tree assigns each region a value \( \phi_j \) such that
\( f(x) = \phi_j \) if \( x \in R_j \). A tree can then be represented as:

\[
T(x; \theta) = \sum_{j=1}^{J} \phi_j I(x \in R_j)
\]  

(9)

where \( \theta = R_j, \phi_j, I \) and \( I \) is an indicator function having a value of 1 when \( x \) is the region \( R_j \) and 0
otherwise.

A boosted tree then becomes the aggregate of each individual tree, simply expressed as:

\[
f_M(x) = \sum_{m=1}^{M} T(x; \theta_m)
\]  

(10)

Trees are fit one by one to the residuals of the boosted tree thus far. This fitting process uses a loss
function as follows:

\[
\hat{\theta}_m = \arg \min_{\theta_m} \sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + \phi_{j_m})
\]  

(11)

The loss function \( L(\cdot) \) takes two parameters the ground truth and the output of the current state
Figure 10. A simple example of computing the DomRank ($R_D$) of text segments based on HTML node location [104].

of the boosted tree. Often this function uses a learning rate to more gradually learn the function, similar to back propagation approaches [58].

Ratkiewicz and Menczer [104] proposed a method they call DomRank which creates a graph using the HTML DOM of a web page. Text spans were scored based on query overlap and a PageRank [95] score. A simple example of DomRank can be seen in Figure 10. They compared DomRank to a simple baseline which was a rule-based approximation to current search engine approaches to summary generation. The authors found through a survey that users preferred the shorter summaries created by DomRank to their baseline 63% of the time.

Otterbacher et al. [93] proposed Biased LexRank, an approach to query-biased summarization based on the previously mentioned, graph theory based approach to automatic summary generation called LexRank [30]. Biased LexRank ranked sentences based on relevance to a query and the structure of a sentence similarity graph. It significantly beat the baseline with a p-value of 0.0619 on Avg. Mean Reciprocal Rank (0.6189) and Avg. Total Reciprocal Document Rank (0.9906).

Park et al. [96] use non-negative matrix factorization, Relevance Measure and an altered similarity score to create query biased summaries of Korean news articles. This approach is compared to using only non-negative matrix factorization and to user-model based personalized summarization. On the 10 queries made on Yahoo-Korea News 200 articles were extracted and manually had summaries generated. The proposed approach scored slightly higher in terms of precision and recall than the other two methods. The personalized non-negative matrix
factorization achieved and f-measure score of approximately 0.35.

Díaz and Gervás [26] suggested a method for sentence selection that not only is query-biased but also used personalization techniques. They found that personalized summaries were more effective at helping users to identify “documents that satisfy user preferences.”

The previously mentioned approaches would be useful for general purpose query-biased summary generation. The following approaches now focus on the task of identifying text spans as fitting into a particular category, such as being a definition of a given word or phrase. These types of models are most useful in question answering systems, where the goal is not to provide relevant documents, but to answer directly questions posed by users.

Machine learning has been used to classify snippets of text as answers to specific questions. Miliaraki and Androutsopoulos [88] in 2004 built an SVM used for ranking fixed-size snippets as definitions of an input term or phrase. The authors combine two previously published techniques. First they use hypernym-hyponym relations to identify more general terms already available in WordNet [32] then sorted based on co-occurrence within a document corpus. Second they used both hand identified and automatically discovered word patterns surrounding known terms and their definitions to identify definitions for new terms and phrases. Each snippet then is compared with the word patterns to define a set of Boolean features which are used as input to an SVM to give a confidence score on a snippet being a definition to a given term or phrase. The authors showed that their approach on specific TREC data was able to achieve better accuracy than only using WordNet and co-occurrence counts. Their approach correctly answered 73% and 85% of questions from TREC-9 and TREC 2001 data sets respectively. This approach assumes that a query has already been identified as a definition question and the term to define has already been extracted from the query.

Androutsopoulos and Galanis [4] in 2005 came up with a technique to create large amounts of training data for a learner which would classify snippets as definitions for a question answering system. It uses online dictionaries and encyclopedias to obtain a large number of definitions. Then it uses a similarity measure to identify which snippets are most similar to online definitions. The user identifies the cutoff points at which a snippet is similar enough to be a positive example and where the cutoff is for negative examples. Some similarity scores will lie in between positive and negative examples and will be thrown away. The results showed that the learner was able to identify more answers to definition questions from a question answering task than previous work and compared to using smaller, less noisy data. Their technique returned the correct answer 58% of the time.

Q. Li et al. [72] use language modeling to score text spans for use as summary text. Language modeling generally uses term frequency to approximate probabilities for a language generation model. Li et al. base their scoring on $P(w|D)$ where $w$ is a word and $D$ is the
document. This calculated using a maximum likelihood estimate as follows:

\[
P(w|D) = \lambda \frac{tf(w, D)}{\sum_{v \in D} tf(v, D)} + (1 - \lambda) \frac{cf(w, C)}{\sum_{v \in C} cf(v, C)}
\]  

(12)

Here \(tf(w, D)\) is interpreted as the term frequency of the word \(w\) in document \(D\). \(\sum_{v \in D} tf(v, D)\) counts the number of tokens within a document. \(C\) is the corpus from which all documents are taken and is used to determine the background model and is based on the frequency of the word in the corpus. The \(cf(w, C)\) refers to the frequency of word \(w\) in the corpus \(C\), and \(\sum_{v \in C} cf(v, C)\) calculates the number of tokens in the entire corpus. Also, \(\lambda\) is a tuning parameter used to balance the effects of the frequency of a word in the current document and the background model. The background model provides smoothing for \(P(w|D)\) and is calculated as \(\frac{cf(w, C)}{\sum_{v \in C} cf(v, C)}\).

The previously calculated \(P(w|D)\) is used to calculate a relevance model \(P(w|M_r)\) which these authors calculate as shown below:

\[
P(w|M_r) \approx P(w|R_q) = \sum_{D \in R_q} P(w|D)P(d|q)
\]  

(13)

where \(R_q\) refers to a set of documents retrieved as relevant to a given query \(q\). The relevance model \(P(w|M_r)\) is approximated using a set of retrieved documents \(R_q\). The region of fixed length with the largest sum of scores from the language model over the words within the text span is chosen. The data source used in the experiments is not named but comes from question answering research. This approach was able to achieve an F1 score of 0.84 compared to the baseline which achieved an F1 score of 0.76. The baseline simply compared text regions by the number of query words contained in those regions.

Bhatia and Mitra [12] describe an approach to generating summaries for document-element searches. Document-elements represent figures and tables taken from scientific articles. The summaries are to provide additional insight into the meanings of the returned figures and tables beyond the caption given in the article from which the document-element was taken. The authors specify methods for ranking sentences from an article and for specifying how many of those sentences should be used with a single user-defined parameter. The generated summaries were compared with Google Desktop and just using the caption and reference sentences. Five graduate students rated the summaries produced by these three methods and gave much higher marks to the proposed method (average rating of 7.27 versus 1.94 for Google Desktop and 4.69 for caption and reference sentence).

Génèreux [38] presents an approach to summarizing blog posts for a blog search engine. This approach relied on a previous summarization system called MEAD developed in 2004 which clusters documents and compares sentences to the centroids. One of the key differences of their approach to classical summarization is that the author attempts to detect the sentiment of the
blog post and to ensure that summary sentences are aligned with the overall sentiment of the blog post. Sentiment analysis was done using SVM's. Sentences were scored based on a number of features which were manually weighted. Sentences were then filtered after ranking to encourage diversity. This algorithm was able to achieve an F1 score of 0.37 with three sentences on three groups from TAC 2008 compared to an F1 score of 0.27 only ranking sentences by query terms.

Search engines originally stored the first couple of sentences from a web page to always display to users as the web page summary. However, as shown by Tombros and Sanderson [114], users of search engines or information retrieval systems work more effectively with summaries tailored to their needs. This has led to the widespread adoption of query-biased summary text for representing web documents in search engine retrieval results. However, these approaches tend to focus too much on the query-biased portion, to the point that users are not aware of the context within which the quoted text from a web page used to summarize the web page exists. It would be wise to both provide users with the query-biased text and also query-independent text.

2.3 Multi-Document Summarization

So far only single document summarization techniques have been mentioned. There exist a number of summary techniques that combine multiple text sources into a single summary on the same topic [84, 7, 102, 31]. For example the SUMMONS system [84] combined a number of news articles on a specific topic or story to produce a single summary. It was the basis for the modern automatically generated news summary systems such as news.google.com (shown in Figure 11) or www.newsinessence.com. The SUMMONS system consisted of two steps. The first step used templates to extract specific data from various inputs on a subject. It would then use the extracted data in its linguistic generator that would select words extracted in the first step and generate new text that was grammatically correct and coherent. This is an example of a summary system not extracting and quoting exactly from documents but rather generating new summary text not contained word for word within the original documents.

Allan et al. [3] proposed a solution to summarizing news stories in time. Allan’s solution scores each sentence in a news article. News articles are scored one at a time using a probabilistic model which considers both useful and novel sentences. News articles discuss events. Sentences that relate to events within a given topic are those that have high “usefulness” scores. Usefulness is calculated as follows by Allan:

\[
P(\text{useful}) = P(s_k | \text{LM}(S)), \quad s_k \in S = \left( \prod_{w \in s_k} \frac{tf(w,S) + 0.01}{1.01 \cdot |S|} \right)^{\frac{1}{|s_k|}}
\]

where \( S = \{s_1, s_2, \ldots \} \) is the set of all sentences in all news stories. \( s_k \) represents the current sentence being scored. \( \text{LM}(S) \) is the language model for the sentences under observation. We score
usefulness as the probability of the sentence under the language model, since most sentences in a news article are on topic.  \( w \in s_k \) represents a word in the given sentence. The constants 0.01 and 1.01 are used for smoothing under Laplace's Law [80]. Also, \( tf(w, S) \) provides the term frequency of word \( w \) in the set of all sentences \( S \).

Allan combines usefulness with novelty. Novelty is calculated as follows:

\[
P(\text{novel}) = P(e(s_k) \neq e(c_i) \forall i \leq m) = \prod_{i=1}^{m} \left(1 - \left[ \prod_{w \in s_k} \frac{tf(w, c_i) + 0.01}{1.01 \cdot |c_i|} \right]^{\frac{1}{|s_k|}} \right)
\]  

The equation above used for novelty is meant to model the probability that the event described by the current sentence \( e(s_k) \) is different from all the events described in the \( m \) clusters \( e(c_i), \forall i \leq m \).

Allan combines novelty and usefulness as follows:

\[
P(\text{useful} \land \text{novel}) = P(\text{useful}) \cdot P(\text{novel})
\]  

Using the sentence scoring mechanism of \( P(\text{useful} \land \text{novel}) \) from Allan et al. [3], when including only 10% of news article summaries, nearly 80% of the 162 events were represented in the TDT-2 corpus. The TDT-2 corpus contains some 60,000 news stories from January through June of 1998.

Many of the same methods applied to single document summaries may be applied to multi-document summaries. The major difference being there are usually there is a lot more text to summarize and there are a lot more redundant text spans within the many documents. As an example consider the early multi-document summarization work done by Goldstein. Goldstein et al. [41] extends their previous work MMR [40] to a multi-document version called Maximal Marginal Relevance Multi-Document (MMR-MD). This more complicated formula takes into
Figure 12. Weighted graph of text fragments as used by Varadarajan et al. [117]

consideration again the previously added sentences as well as a comparison with other clusters of
documents and multiple terms for coverage. The new formula is as follows:

$$MMR - MD = \arg \max_{P_{ij} \in R \setminus S} \left[ \lambda (\text{Sim}_1(P_{ij}, Q, C_{ij}, D_i, D)) - (1 - \lambda) \max_{P_{nm} \in S} \text{Sim}_2(P_{ij}, P_{nm}, C, S, D_i) \right]$$

(17)

Here $\text{Sim}_1$ and $\text{Sim}_2$ expand into complex formulas of their own. $\text{Sim}_1$ represents the similarity
metric used in relevance ranking. $\text{Sim}_2$ is an anti-redundancy metric. $D$ represents the document
collection. $P_{ij}$ represents passage $j$ from document $D_i$. $C$ is a set of clusters into which the
documents have been clustered into. $R$ is the ranked list of passages according to some information
retrieval system. $S$ is the subset of passages in $R$ already selected. There are several more
summations and formulas that are involved in calculating $MMR - MD$ in whole. For brevity we have
only included the top level formula. It provides a means in $\lambda$ where one may tune the weight of
each side of the equation.

Within a list of search results, each summary represents only a single web page.
Varadarajan et al. [117] proposed the generation of composed pages or multi-document summaries
to answer queries. These composed pages contained the relevant text from a number of pages. A
single web page was summarized in a query-biased manner through creating a graph to represent
the document. An example of such a weighted graph can be seen in Figure 12. Similarity was
calculated between each node and edges were formed when a threshold was met. From these
graphs a minimum spanning tree was found which determined the sentences to include in a
composed web page. In a user survey, users gave Google results pages an average rating of 2.23 out
of 5 and Varadarajan’s approach an average rating of 3.88 out of 5.

Haghighi and Vanderwende [48] propose to model sets of documents using a hierarchical
latent Dirichlet allocation model [13]. Haghighi calls their method HierSum. Their approach ranks sentences using their probabilistic topic model. Sentences are added one by one according to the KLSum algorithm. This approach considers which sentence decreases the Kullback-Lieber divergence between the word distribution in the documents and in the summary. The purpose is to most closely match the word distributions overall in the document. This approach achieved similar ROUGE scores to current approaches at the time. Additionally, a user study was conducted in which 16 users evaluated summaries produced by HierSum and Pythy against summaries. HierSum in pairwise comparisons was significantly better by user preference than Pythy for the following four measures: overall, non-redundancy, coherence and focus, as is shown in Table 2. As you can see from this table HierSum was preferred pairwise in all categories. Most noticeable HierSum was better in coherence where it won 54 comparisons to Pythy’s 15.

<table>
<thead>
<tr>
<th>Question</th>
<th>Pythy</th>
<th>HierSum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>20</td>
<td>49</td>
</tr>
<tr>
<td>Non-Redundancy</td>
<td>21</td>
<td>48</td>
</tr>
<tr>
<td>Coherence</td>
<td>15</td>
<td>54</td>
</tr>
<tr>
<td>Focus</td>
<td>28</td>
<td>41</td>
</tr>
</tbody>
</table>

TABLE 2

Results from Haghighi and Vanderwende [48] of a manual pairwise evaluation of two methods. Haghighi’s method is “HierSum.”

Wang et al. [120] propose a multi-step process for summarizing multiple documents. In their approach, each sentence is compared against each other sentence semantically through referencing WordNet for word-level relationships. The matrix of similarities undergoes Symmetric Non-negative Matrix Factorization for the purpose of clustering sentences. Within each cluster, sentences are ranked based on scoring that considers both the average distance to all other sentences in a cluster (similar to closeness centrality) and the distance to the desired topic of the multi-document summary. Experiments show on DUC 2006 data each step of the process helps to produce the best possible ROUGE-1 score for this method of 0.3955.

Some forms of multi-document summarization use templates. The templates must be manually created and are very specific to the topics that can be summarized, but produce structured summaries that are very easily understood. As an example consider the RIPTIDES system proposed by White et al. [125]. RIPTIDES parses the text into tokens and sentences, from which a TIPSTER parse and part-of-speech annotations are produced. Information extraction is used to find domain-specific relations and relevant entities. XSLT is used to create extraction patterns under weakly supervised learning. These extraction patterns are used to fill in the template. Additionally, the most relevant text spans to “damage reports.” Often several reports are included from various sources in the final output of the system. The templates are merged between
Earthquake strikes quake-devastated villages in northern Afghanistan

A earthquake struck quake-devastated villages in northern Afghanistan Saturday. The earthquake had a magnitude of 6.9 on the Richter scale on the Richter scale.

Damage

Estimates of the death toll varied. CNN (06/02/1998) provided the highest estimate of 4,000 dead, whereas ABC (06/01/1998) gave the lowest estimate of 140 dead.

In capital: Estimates of the number injured varied.

Selected News Excerpts

CNN (06/01/98):
Thousands are dead and thousands more are still missing. Red cross officials say the first priority is the injured. Getting medicine to them is difficult due to the remoteness of the villages affected by the quake.

PRI (06/01/98):
We spoke to the head of the international red cross there, Bob McCaro on a satellite phone link. He says it's difficult to know the full extent of the damage because the region is so remote. There's very little infrastructure.

PRI (06/01/98):
Bob McCaro is the head of the international red cross in the neighboring country of Pakistan. He's been speaking to us from there on the line.

APW (06/02/98):
The United Nations, the Red Cross and other agencies have three borrowed helicopters to deliver medical aid.

Figure 13. An example summary produced by the RIPTIDES information extraction summary system [125].

documents and a final summary is produced. An example of the types of summaries produced by RIPTIDES is shown in Figure 13.

The summaries provided by RIPTIDES [125] were given grades in a manual evaluation and compared to human generated summaries and a baseline. Grades were based on content, organization and readability. The baseline system is a sentence-extraction, multi-document summarizer that relies on document position, recency and word overlap clustering. The baseline received on average a D grade. The RIPTIDES system received on average a C grade. The two manually produced summaries received grades of A- and B+.

McDonald [83] formalizes multi-document summarization leaving the measure of relevance, redundancy and summary length as generic functions. His formalization is the following: The input to such a system is a document collection $D = \{D_1, \ldots, D_k\}$. Each of the documents $D$
contains a set of textual units \( D = \{t_1, \ldots, t_m\} \). These textual units may be words, sentences, or some other textual unit. The document collection is represented as the set of all textual units \( \mathbf{D} = \{t_1, \ldots, t_n\} \) where \( t_i \in \mathbf{D} \) iff \( \exists t_i \in D_j \in \mathbf{D} \). The summary produced \( S \subseteq \mathbf{D} \) is a set of textual units from the document collection. Additionally, relevance is computed for textual unit \( t_i \) within summary \( S \) as \( \text{Rel}(i) \), and redundancy between textual units \( t_i \) and \( t_j \) is computed as \( \text{Red}(i, j) \).

The length of a textual unit \( t_i \) is computed as \( l(i) \). Summary lengths are constrained such that the total length of the summary must be less than \( K \). From this follows the inference of problem of selecting textual units for a summary:

\[
S = \arg \max_{S \subseteq \mathbf{D}} \sum_{i \in S} \text{Rel}(i) - \sum_{t_i, t_j \in S, i < j} \text{Red}(i, j)
\]

such that \( \sum_{i \in S} l(i) \leq K \) (18)

Global inference within this formalization of the multi-document summarization problem is shown by McDonald to be NP-hard. McDonald goes on to compare within this framework a greedy algorithm and dynamic programming as a way to quickly approximate a solution and Integer Linear Programming (ILP) as a method to solve for the global solution. These three approaches were compared using the DUC 2002 data set. The ILP solution did not scale well beyond 50 textual units, scaling super-linearly. The other two solutions scaled linearly.

Barzilay et al. [6] considers the ordering of sentences produced in multi-document summaries. First Barzilay shows that human comprehension of automatically generated summaries increased significantly when just the order of sentences was improved. Next, three algorithms for ordering output sentences are proposed.

The first algorithm for ordering multi-document summary sentences proposed by Barzilay et al. is called “Majority Ordering.” Majority ordering aims to adopt the same ordering of sentences as they appear within documents. Sentence orderings between documents is found through finding the ordering under which different themes takes place in a document. When theme 1 precedes theme 2 in a majority of documents, then sentences from theme 1 precede sentences from theme 2 in the output summary.

The second algorithm for ordering is called “Chronological Ordering.” This approach attaches a date to each theme from which sentences come. The date of the first mention of the theme is used as the date of the theme. Sentences from each theme are then ordered according to the dates of each theme.

The third algorithm for ordering is called “Augmented Ordering.” It uses as its basis the chronological ordering but seeks to avoid sudden theme changes. It does this by observing when themes often co-occur in documents, these themes may be paired when above a threshold. Themes are then combined into blocks using document co-occurrence of themes. Each block is then assigned a date in the same way that the chronological ordering worked to assign themes dates.
A comparison of these three ordering algorithms is given in Table 3. Here humans were asked to judge a number of summaries produced by each ordering algorithm. A Fisher exact test was used after combining the grades “Poor” and “Fair” into a single category and resulted in a p-value of 0.04. The augmented ordering was found to be best.

<table>
<thead>
<tr>
<th></th>
<th>Poor</th>
<th>Fair</th>
<th>Good</th>
</tr>
</thead>
<tbody>
<tr>
<td>Majority Ordering</td>
<td>3</td>
<td>14</td>
<td>8</td>
</tr>
<tr>
<td>Chronological Ordering</td>
<td>10</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td>Augmented Ordering</td>
<td>3</td>
<td>8</td>
<td>14</td>
</tr>
</tbody>
</table>

**TABLE 3**
Comparison of three sentence ordering techniques conducted by Barzilay et al. [6].

### 2.4 Summarizing Social Media

Automatic summarization approaches have been applied to many types of documents. This section highlights some of those efforts as they apply to social media. Social media is taking on great importance [76] and has been incorporated into search results.

Sharifi et al. [110] summarized social status updates on Twitter using phrase reinforcement and TF-IDF. The phrase reinforcement approach creates a word graph from several tweets anchored by the topic phrase that was used to find these tweets. Each node in the graph is weighted and the phrases most commonly seen before and after the topic phrase are used to summarize the tweets. The TF-IDF approach weighted terms within tweets based on frequency. Two judges created summaries. The two judges had an F-score of 0.34 comparing the manually created summaries. Both the phrase reinforcement and TF-IDF methods achieved F-scores similar to that achieved between judges (0.30 and 0.33).

Liu et al. [76] combined multiple sources to create summaries that not only showed summarizing tweets within the context of Twitter, but added clarifying information from external websites. Multi-tweet summarization is performed using concept-based optimization [39]. Then the contents of each web page linked to in the pool of tweets is analyzed to find a summarizing sentence outside of twitter. Pulling these resources together both summarizes a topic on Twitter in context and provides background to the topic being summarized.

Inouye and Kalita [56] evaluated 9 methods for summarizing multiple tweets. The results of the comparisons are shown in Table 4. The approach taken uses two judges each creating manual summaries from a number of tweets. The overlap between these two judges is seen in the entry labeled “Manual.” The rest of the entries are a comparison between the automatically create summaries and a manual summary. There appears to be no significant difference between scores. Hybrid TF-IDF and SumBasic achieve the highest F-measures. Surprisingly they achieve higher
TABLE 4

Comparison of 9 different multi-tweet summarization methods made by Inouye and Kalita [56].

F-measure scores than Manual or the human generated summaries.

Karandikar [62] applied topic modeling and multidimensional scaling to tweets as a form of preprocessing to cluster tweets and Twitter users. Karandikar explored the size of the training corpus and the number of topics. The end result found training on the news wire corpus with 300 topics performed best. Their approach achieved accuracies of 64%, 70% and 67% on data sets Hurricane Alex, China earthquake and short pieces from the medicinal domain respectively.

Ramage et al. [103] proposed a new approach to characterizing large numbers of tweets. This approach used a partially supervised learning model using Labeled LDA. Words in tweets are categorized into four categories: style, social, status, and substance. Then word clouds are generated from the words categorized into each group. Examples of this process are shown for two Twitter users: @w3c and @oprah in Figure 14.

In addition to Twitter, Cvijikj et al. [23] proposed an approach to tracking trends on Facebook1. Cvijikj collected public Facebook posts through the Graph API2. These posts were clustered using first by distribution and second by a greedy word co-occurrence clustering algorithm. Posts were collected from July 22, 2011 to July 26, 2011. During this time two major news stories broke that were used for evaluation: Amy Winehouse died and a terrorist attack occurred in Norway. Harry Potter is used to represent popular topics. The clustering approach proposed achieved F-measure scores of 0.8510, 0.1303 and 0.8115 for topics Amy Winehouse, Norway attack and Harry Potter respectively.

Several recent efforts attempt to provide analytics for spatio-temporal trends and events detected or tracked on Twitter. Sakaki et al. [108] study social, spatial and temporal characteristics of earthquake-related tweets. In their approach they devise a classifier to decipher event related tweets. They also apply Kalman filtering and particle filtering to Twitter users. Their earthquake reporting system detected earthquakes in Japan with an intensity of 3 or higher

<table>
<thead>
<tr>
<th>Method</th>
<th>F-measure</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>LexRank</td>
<td>0.2027</td>
<td>0.1894</td>
<td>0.2333</td>
</tr>
<tr>
<td>Random</td>
<td>0.2071</td>
<td>0.2283</td>
<td>0.1967</td>
</tr>
<tr>
<td>Mead</td>
<td>0.2204</td>
<td>0.3050</td>
<td>0.1771</td>
</tr>
<tr>
<td>Manual</td>
<td>0.2252</td>
<td>0.2320</td>
<td>0.2320</td>
</tr>
<tr>
<td>Cluster</td>
<td>0.2310</td>
<td>0.2554</td>
<td>0.2180</td>
</tr>
<tr>
<td>TextRank</td>
<td>0.2328</td>
<td>0.3053</td>
<td>0.1954</td>
</tr>
<tr>
<td>MostRecent</td>
<td>0.2329</td>
<td>0.2463</td>
<td>0.2253</td>
</tr>
<tr>
<td>Hybrid TF-IDF</td>
<td>0.2524</td>
<td>0.2666</td>
<td>0.2499</td>
</tr>
<tr>
<td>SumBasic</td>
<td>0.2544</td>
<td>0.3274</td>
<td>0.2127</td>
</tr>
</tbody>
</table>

1http://www.facebook.com/
2https://developers.facebook.com/docs/reference/api/
Figure 14. Characterizations of two Twitter users @w3c and @oprah by Ramage et al. [103].

with a probability of 96%.

De Longueville et al. [25] provides an in-depth analysis of all the tweets found which relate to a fire that took place near Marseille, France in July 2009. No algorithms are proposed. An example of the type of analysis performed see Figure 15. De Longueville observed the number of tweets, the sources of the tweets and the timeliness of the tweets related to this fire. This study demonstrated the effectiveness of observing local events even in countries without a high Twitter usage and where the main language used is not English.

Tumasjan et al. [115] analyzed over 100,000 tweets that mentioned either a political party or candidate in the German federal election of 2009. Tweets were automatically translated to English and analyzed based on word counts. Twitter was found to be used for discussing not just spreading political messages. The messages were dominated by a small percentage of Twitter users. However, the party and candidate counts mirrored poll results. Tumasjan’s results demonstrate that Twitter can be used as a “real-time indicator of political sentiment.”

For general event tracking, Marcus et al. [82] propose a system called Twitinfo. Twitinfo begins tracking an event after a search query is submitted to the system. Twitinfo now records tweets as they relate to the given search. Twitinfo produces a timeline of the related tweets. Peaks in this timeline are found by binning tweets per minute and labels the peaks. Twitinfo uses two classifiers to identify positive and negative sentiment as it relates to a particular event and presents sentiment in a pie chart. This system identified between 80%-100% of manually labeled peaks and
Figure 15. Timeline of related tweets to the Marseille fire by hour as used by De Longueville et al. [25].

Figure 16. Twitinfo summary of an event as documented by Marcus et al. [82].

provided a structured view of an event in Twitter as seen in Figure 16.

The usefulness of tracking text usage data to provide flu surveillance was first demonstrated by Google Flu Trends [18]. Their approach demonstrated the effectiveness of monitoring keyword searches for doing near real-time detection of flu epidemics. Quincey and Kostkova showed that
Twitter was a viable option as a source of data for Epidemic Intelligence, in following up on the work on Google Flu Trends. This work showed the existence of a large volume of tweets that conveyed messages about swine flu during the swine flu pandemic. They developed a regression model that achieved a correlation of 97% on data unseen by the model. This approach required a large number of keywords though they are automatically learned and weighted by the system.

Culotta [21] showed that one could achieve a 95% correlation with health numbers by tracking a small number of keywords. Tweets are first classified as flu related. Tweet counts are adjusted by a logistic regression model fit to classify tweets as related to flu or not. This logistic regression model achieved an accuracy of 84%. A correlation was obtained by fitting a log-odds model to the data and a 95% correlation to CDC’s influenza like illness (ILI) numbers was reported.

Formally, Culotta describes the set up in this way. \( P \) represents the true proportion of the population exhibiting ILI symptoms. Then \( W = w_1 \ldots w_k \) is a set of \( k \) keywords. \( D \) is a document collection, and \( D_W \) is the set of documents in \( D \) that contains at least one of the keywords in \( W \). Culotta defines a function \( Q(W, D) = \frac{|D_W|}{|D|} \) which is the fraction of documents in \( D \) that match the keyword \( W \).

Tweet count adjustment is calculated as follows, replacing the function \( Q(W, D) \) with \( Q_h(W, D) \):

\[
Q_h(W, D) = \sum_{d_i \in D_W} \frac{1(p(y_i = 1|x_i; \theta) > 0.5)}{|D|}
\]

(19)

Then a linear model between the log-odds of \( P \) and \( Q(W, D) \) is defined as follows:

\[
\text{logit}(P) = \beta_1 \text{logit}(Q_h(W, D)) + \beta_2 + \epsilon
\]

(20)

where \( \beta_1 \) and \( \beta_2 \) are coefficients and \( \epsilon \) is an error term. Additionally, the logit function is given as follows: \( \text{logit}(X) = \ln \left( \frac{1}{1 - X} \right) \).

Culotta’s work was able to efficiently remove spurious peaks in the data when the number of injected tweets unrelated to the flu was small. When the number of tweets collected which are unrelated to the flu, then this method is no longer effective and false positives will occur.

Achrekar et al. [2] was able to obtain a 98% correlation with CDC reported cases. One important development in preprocessing from Achrekar’s work was to remove retweets and additional tweets from the same Twitter user during a single week. Achrekar’s approach is to use auto-regression to predict future ILI numbers based on the past two weeks of Twitter data.

Achrekar’s auto-regression model is given as follows:

\[
y(t) = \sum_{i=1}^{m} a_i y(t - i) + \sum_{j=0}^{n-1} b_j u(t - j) + c + e(t)
\]

(21)

where \( t \) gives the index of the week, \( y(t) \) represents the number of reported ILI cases in week \( t \), \( u(t) \) denotes the number of unique Twitter users in the data during week \( t \), and \( e(t) \) is a sequence...
of independent random variables.” c represents a constant to account for the offset in tweet counts and ILI numbers.

Predictions are made using Achrekar’s auto-regression model using the following two relationships:

\[
\hat{y}(t) = a_i \hat{y}(t-1) + \sum_{i=2}^{m} a_i y(t-i) + \sum_{j=0}^{n-1} b_j u(t-j)
\] (22)

\[
\hat{y}(t-1) = \sum_{i=1}^{m} a_i y(t-i-1) + \sum_{j=0}^{n-1} b_j u(t-j-1)
\] (23)

2.5 Alternative Search Result Displays

In summarizing web documents in search displays, there have been many proposed improvements over traditional text based systems. Many of these systems try to provide more enticing visual displays.

White et al. [126] proposed a method they called WebDocSum which added to the titles and summaries provided by Google and AltaVista additional web page meta information: the number of links, name of the first non-text object on the web page and the size of the document. The additional summary content was preferred to the search results provided by these two search engines.

A number of papers consider in addition to text summaries, providing images to enhance each search results. Dziadosz and Chandrasekar [29] found that for informational tasks, users using search results containing both text and thumbnails took slightly longer to make relevance decisions but that users were slightly more accurate (74%) compared to the classical text only search results (72%). A thumbnail is a very small image of part or all of a web page.

Joho and Jose [60] compared four presentation styles of search results: classical (Google-like), classical with top sentences added in a list, classical with a thumbnail, and classical with a thumbnail and top sentences. An example result showing a classical summary, top sentences and a thumbnail is shown in Figure 17. Joho and Jose’s results showed that survey participants were not faster at completing search tasks, but 20 of 24 survey participants did prefer at least one of the alternatives to classic search results.

Kaugars [63] used thumbnails and zoomed views to show keywords in context for a number of documents. Initially all search results are displayed as web page thumbnails, with keyword locations highlighted. A user may zoom to a level where keywords are shown in context and other paragraphs are compressed. Users may again zoom in again to view the full, scrollable contents of a document.

The experiments conducted by Li et al. [73] showed users to be 30% faster when conducting informational searches when search results were accompanied by an image from each
web page compared to text only search results. Li et al. also show one approach for selecting dominant images within a document. Wynblatt and Benson [130] created caricature representations of web pages for display in search. They displayed link density as the background color of the listing for example and included several features of web pages such as number of large images, presence of an e-mail link, etc. Their caricatures generally include a representative image which was chosen from a document using a previously trained neural network. However, these approaches to finding representative images do not consider the case where no image should be used or where multiple images would best represent a web page.

Woodruff et al. [128, 129] propose the use of enhanced thumbnails for each search result. These enhanced thumbnails added text overlays to the thumbnail specific to the query entered. For example an unmodified thumbnail can be seen in Figure 18(a). The query enhanced version can be seen in Figure 18(b). Users were given a number of tasks to perform and were timed on each. Overall users using both the enhanced thumbnails combined with text snippets for a task had the shortest task completion time (67 s compared to 95 s for text and 86 s for thumbnails). Interestingly two of the four categories of tasks (picture and homepage) enhanced thumbnails combined with text snippets came in a close second. Users found the enhanced thumbnails approach to be intuitive.

The drawback of the enhanced thumbnails approach is that thumbnails were created on the fly adding to the overhead of generating snippets. Thumbnails would require more bandwidth at major search engines and if thumbnails are precomputed or partially precomputed, major amounts of memory would be required. As opposed to image snippets, enhanced thumbnails apply to all types of web pages.

Xue et al. [132, 133] add to search results what they call an “image snippet.” Their approach selects an image from the web page linked to by a search result and uses that image along with the summary text as can be seen in Figure 19. The authors found that users were able to more accurately select relevant documents from search results when presented with both text and image snippets (0.852), compared with both text alone (0.806) and images alone (0.778). These experiments intentionally removed documents from search results which did not contain...
One of the shortcomings of their approach is that not all web documents can be covered by this approach.

Teevan et al. [113] suggests a new approach to snippets for the purpose of providing a consistent reference to a website across bookmarks and search results that provides quick memory aids to refinding sites and uses little space. Rather than using text snippets, the proposed solution creates an image combining the URL, a logo and the most relevant image (manually selected or heuristic of largest image, depending on test). Three examples of these visual snippets can be seen in Figure 20. Users in this study suggest that they prefer text snippets to both visual snippets and thumbnails. Text snippets led to fewer clicks per search task. However, users were able to find much more quickly relevant documents they had already seen for the same task done a day earlier when using visual snippets (29 s), even when they had not before seen visual snippet for the previously seen document. Thumbnails took 35 s and text snippets took 39 s for the same tasks.

Zhang et al. [138] in 2002 proposed a technique for browsing document collections using
"concepts." The first display pictured in Figure 21(a) shows the vocabulary of the database with like terms being displayed in the same color and are clumped together. When a term is selected one can see the related terms and similar terms are linked together. This allows for browsing of a large number of documents in a way similar to hierarchical search. Another interface discussed within the same article presents instead of single term nodes, clusters described with multiple names as shown in Figure 21(b). The selection of a cluster shows the terms found in this cluster and how they relate.

Next is the FooCA system developed by Koester [67] for the visual display of "concepts." Two different interfaces were presented in this work. The first interface can be seen in Figure 22(a) which is a table display of search results. It would be one way to visualize Boolean term vectors. The rows represent documents and the columns represent terms. An 'x' is placed in cells where the given document contains the given term. The second interface present in this work can be seen in

Figure 20. Three examples of visual snippets and the pages from which they were created [113].

Figure 21. (a) Clustered view of terms for browsing. (b) View of clusters for term browsing. [138]
Figure 22. (a) FooCA table display of results. (b) FooCA graph display of results. [67]

Figure 22(b). This is a line drawing representation where the query is the root node. The query is connected by edges with terms and these terms are in turn connected with documents.

Dörk et al. [27] propose the use of visual widgets both for viewing search results in a visual form and for interacting with the results. In this paper they use the example of three separate widgets: a timeline, a map and a tag cloud. Examples of these three widgets can be seen in Figure 23. The timeline presents documents in histogram form by date and allows the selection of ranges of dates. The map widget shows a world map with indicators showing where documents originate from. Users can zoom in and out of this widget to select documents over various geographical domains. Lastly, the tag cloud presents common terms in the documents available and resizes the visualization for each term by frequency. This research presents both new views of search results and new approaches to interacting with search results.

Some approaches such as the following try to use the location on the screen of a document to provide more information about a document. For example Kelkar et al. [64] develop an alternative approach to displaying search results which takes advantage of social tools. Search results are displayed in a 2-D particle space as shown in Figure 24. The approach uses thumbnails to represent web pages and a “callout” icon representing the web conversation about said web page which allows a user to access comments made about a web page. Connections are added between associated web pages. The interface adds tag clouds and author clouds. Thus replacing textual snippets from a web page with what has been said about a web page.

Hemmje et al. [54] presented Lyberworld, which displayed documents in a three dimensional sphere with keywords shown at the edge of the sphere. Documents were presented closest to the keywords contained in those documents.

Zhang and Marchionini in 2005 [137] rather than replacing snippets add visual histograms
Figure 23. Visual widgets showing a visual representation of results and allowing for more ways to refine a search [27].

Figure 24. 2-D particle space representing search results and their social context [64].
to the display of results to provide a more intuitive browsing interface as shown in Figure 25. This desktop GUI is discussed in the context of searching a vertical database of documents. Controls include a search box that updates results after each keystroke. Additionally a number of facets are considered with histograms showing the frequency of those terms. These histograms provide additional controls to narrowing search results. Snippets are still used but the relative importance of the snippets is diminished.

The visual facets seem like a very good idea. However, they appear to not be for general use, as the set of facets was manually changed based on the document collection, whether a movie database or for the Energy Information Administration website. The users liked the interface after being trained on its use and practicing using it. Some users in surveys after using RB++ related that they did not find the interface to be intuitive.

Rather than static summaries displayed in search results, Paek et al. [94] suggest an approach similar to a fisheye lens which increases the length and/or size of the text of the snippet when it is the user’s focus. This approach is called WaveLens. In the two cases tested they found that users were able to more quickly complete search tasks and that users preferred to use one of these dynamic interfaces to the static search result lists. Examples of both dynamic interfaces can be seen in Figure 26.

Veerasamy and Belkin [118] proposed a table of bar charts to show term importance visually. Each row designated a single document, each column represented a word. The words selected included both query terms and terms used for relevance feedback. Graham [46] presented Reader’s Helper that highlighted keywords both within a single document and document lists. Each of the keywords was given a score with a matching bar showing the strength of that score visually.

Figure 25. RB++ interface to search results [137].
The approaches described in this section show a number of creative approaches to browsing search results from search systems. Creative uses of images, clouds, maps and 2-D spaces provide new ways of interacting with the results. From the studies provided it is difficult to tell if these new approaches provide something that users would consistently want to use. There were no long term studies. Generally studies were done in a lab type setting. Generally the approaches taken deal with the arrangement of documents and images within the documents. None of these methods aim to accurately describe the contents of the documents. New representations should be sought which provide users with accurate information about the contents of a web page.

2.6 Reorganizing Search Results

Even before search engines were popular, there have been researchers looking into approaches to displaying search results that differed from the now common ranked list. The approaches mentioned here refer mainly to grouping results by similarity. Most of these approaches use clustering to achieve this end [53, 135, 69, 33, 16], others use categorization [28, 24], while one includes two ranked lists one for documents and the other for snippets [127]. When looking through a large list of results, groupings allow users to quickly eliminate from their search path many irrelevant results, since relevant results tend to be more similar to each other than to irrelevant documents [53]. A majority of the techniques listed here use search engine or information retrieval results as input to their systems.

Hearst and Pedersen [53] in 1996 provide an early look at clustering in information retrieval systems. Their paper provides proof to the clustering argument that relevant documents tend to be more similar to each other than to non-relevant documents. More often than chance would have it there would be a high percentage of relevant documents in one of the clusters. This paper also presents an interactive interface for working with search results which at first returns a list of results as is most common, then allows users to cluster those results into a small number of clusters. If one of those clusters is chosen, the contents of the chosen cluster can again be clustered.
Achievements — Hubble — Hubble’s achievements

**Rank 19:** Booke with Pictures from Space (Science Library)
... of Our Creation, by Simon Goodwin. A gallery of the most significant photographs taken by the Hubble telescope explains what Hubble’s achievements can...

**Rank 34:** Amazon. COM: having info. Hubble’s Universe: A Portrait of Our...
... Ingram. A gallery of the most significant photographs of space as taken by the Hubble telescope explains what Hubble’s achievements can tell us about the...

**Rank 63:** ESA Portal: Press Release: — HST’s 10th anniversary, ESA and...
... A public conference will take place in the afternoon to celebrate Hubble’s achievements midway through its ... Notes for editors. The Hubble Space Telescope...

**Rank 100:** Presentation.com: The Hubble Decode...
... astronauts’ view of the Earth from the Moon - and the Hubble Space Telescope’s ... View from the top. On the scientific c from. Hubble’s achievements...

**Rank 111:** The Hubble. Discoverer of expanding universe...
... Hubble’s achievements were recognised during his lifetime by the many honours conferred upon him In 1948 he was elected as ... The Hubble Space Telescope...

**Rank 149:** HubbleView — Science...
... further and sharper than any optical/ultraviolet/infrared telescope ... very specific goal (like the Cosmic Background Explorer), Hubble’s achievements...

Figure 27. Hierarchy example originating from work by Lawrie and Croft [69].

into smaller clusters. It was found to be useful at eliminating quickly a large number of irrelevant results. Those results come from only four graduate students who were told to find as many relevant documents in 30 minutes as they could.

Zamir et al. [135] in 1997 suggests one approach to clustering document titles and snippets. The author’s approach improves the efficiency of word-intersection clustering to gain a time complexity of $O(n \log n)$, with other clustering approaches best known to the author at the time achieving at best $O(n^2)$. The author then develops a measure for testing the quality of clusters which uses a hand derived set of clusters for comparison. According to his measure and timing experiments word-intersection clustering was both faster and created better clusters than using cosine similarity and group-average inter-cluster distance. Additionally, the use of phrases rather than word overlap produced the best clusterings but dramatically increased the clustering time.

Lawrie and Croft [69] in 2003 provides an approach to generating hierarchies over Google search results composed of a title and a snippet. An example of the resulting hierarchies and interface can be seen in Figure 27. Phrases were ranked for inclusion in the hierarchy based on topicality (measured by the Kullback-Leibler divergence) and predictiveness (using a co-occurrence language model). Hierarchies were judged based on the reachability of documents given simple heuristics for search, such as only viewing clusters with 10 or fewer documents.
Ferragina and Gulli [33] in 2005 have created a hierarchical snippet clustering engine which combines results from a number of search engines. Cluster labels are created from “gapped sentences” or from phrases in sentences which are not necessarily contiguous. These labels are compared to two databases of good labels to find more human recognizable labels. The two sources are anchor texts extracted from over 200 million web pages and the categories used in DMOZ, the manual categorization of useful pages online. Their results suggest that their engine outperforms the other available clustering algorithms in terms of speed in some cases and user preference in others, except in the case of Vivisamo. Users either preferred Vivisamo or liked both the authors’ solution and Vivisamo clustering interfaces equally well.

Campos and Dias [16] in 2005 create a hierarchical clustering of Google results. Their approach takes several steps building on the work of others. The first step taken is to filter the results based on their own measure of relevance, which assumes that more relevant documents will have their URL’s repeated multiple times in the results returned. Documents with more oft repeated URL’s are preserved. Then the algorithm uses SENTA software to extract phrases from the documents, detects relevant terms using the WEBSPY software and lastly clusters the results using an algorithm called PoBoc. No empirical comparison is attempted with other result clustering algorithms.

Categorization methods often use supervised learning algorithms to identify from a known list how documents should be labeled and grouped. Dumais et al. [28] in 2001 compare a number of different interfaces for showing search results. Mainly their comparison was between list results and categorized results. Here categorized results means that a preset of categorization has been specified and documents are then assigned categorizes using text classification. Each experiment involved between 18 and 20 individuals who were given 30 search tasks to complete and were measured by their time to completion. Users completed the first 15 tasks using one interface then 15 different tasks using a different interface. The authors conclude that for the given tasks, the category interface resulted in significantly faster task completion times. Additionally, the tests displaying snippets in a pop-up shown when hovering over a document title resulted in slower task completion times for both list and categorized interfaces. Removing category labels or requiring a user to expand categories before seeing results increased task completion time.

Dakka et al. [24] in 2005 proposed a method for extracting and creating multiple hierarchies of facets for browsing large databases of text or annotated objects. The top level facets are first manually defined. An SVM then categorizes each term in each document to each of the known facets. Each data point is made up of the noun, hypernyms discovered through WordNet, and other keywords found in the document. Each of the tested data sets had manually identified hierarchies that could be used for training. Hierarchies for each facet were created using term co-occurrence counts. Where terms found more often were close to the root and terms found less
often but co-occurring with other terms became the children of those terms. This approach was able to achieve an F1 score of 80% for the Corbis data set, which is an annotated corpus of images, compared to the manually created facets. No user studies were done.

White et al. [127] compare three different search result interfaces. The first result, their baseline, lists titles to web documents in the order given by AltaVista with a mouse hover revealing more information about the document, such as snippets. The second interface included two lists, the first was the list of titles of web documents and the second is a list of relevant sentences, ranked independently from the documents. This second interface can be seen in Figure 28. Selecting a sentence highlighted the matching document title. The third interface added relevance feedback to the system so that interaction with the document titles and document summaries led to automatic reranking of relevant sentences. Experienced users did not appreciate the occasional deletion of a relevant sentence based on their interactions in the third system. Users overall were able to search more quickly with the aid of the relevant sentences, looked into more search results beyond the top 10, found the new system easy to use, and overall liked the relevant sentence addition compared to the ranked list of document titles.

Zeng et al. [136] in 2004 discusses finding names for clusters. Their approach was to transform the seeming unsupervised task of clustering search results with the associated task of finding labels and transforms that problem into a supervised learning task. The training data consisted of a set of 100 to 200 candidate phrases for each of 30 queries extracted from one day of queries on MSN search. Then three judges marked 10 “good phrases” and 10 “medium phrases” out of the 100 to 200 for each query. Phrases which belonged to the “good phrases” list of one judge and were either good or medium for the other judges, or which were on all three judges medium
list, were given a score of one and all other phrases were given a score of zero. Various classification algorithms including support vector regression with various kernels, linear regression and logistic regression were used for this task. The phrases were ranked and precision scores were given to the top 5, 10 and 20 phrases. All algorithms achieve just above 0.7 precision for the first five results.

Wang and Zhai [121] in 2007 propose a cluster labeling solution which takes advantage of search engine query logs. The algorithm proposed in this paper first searches for similar queries to the current query and ranks similar queries using the OKAPI ranking function. For each query there are a set of documents which were clicked on from that query. The top ranking queries and documents are clustered using star clustering. Star clustering specifies for each cluster a central query. Additionally a representative pseudo-document is created as a centroid. All documents returned by the search engine for the given query are then compared to each centroid and are listed under the cluster that they are most similar to, labeled by the central query for that cluster.

One interesting result from this paper found that when ranking query's difficulty by MAP score, it was found that the most difficult queries often have a better precision score at 5 in the best cluster compared to the listed search results. The easiest queries on the other hand by MAP score most often have better precision at 5 compared to the clustered results. The better precision scores could simply be a function of the MAP scores, as both MAP and precision use relevance calculations on the list results, rather than an argument in favor of using clustering on difficult queries.

2.7 User Studies

Several research papers rather than propose new approaches to summarization, study how users currently interact with search engine search results. Studies look into the various characteristics of each web page surrogate or the combined title, summary and URL representing a single web page. Many of these studies were conducted on major search engines and provide results on a large scale, in the thousands of users range. These studies provide insight into how users currently interact with search engines, giving us understanding how we can improve search engines to better aid the users.

Lin et al. [75] in 2003 conducted a study of the length of summary to use in a question answering systems. This study was done in two phases and was administered to 32 graduate and undergraduate students. In the first phase students were given a factoid question and a single answer returned by a question answering system. The user could mark “accept,” “reject” or “more information.” The more information option would increase the context. At first only the answer was given. A user could increase the context to a sentence, then a paragraph and finally the entire document. From the first phase it appeared that paragraph length was most preferred, being preferred in 53% of cases and exact answer was least preferred at 3%. The second phase involved
the students being given a specific information task with one or more associated questions. For each task the student would be assigned the length of context given, being the same possibilities as the previous phase, with students not allowed to change the amount of context given during the second phase. Users were timed and the number of clicks and queries were monitored. There was little variation in time for each of the context lengths, since only a single answer was returned per query. However, users tended to pose more queries the shorter the context was. From this study it appears that for users to trust the information given by a question answering system to a factoid question, a paragraph length response was ideal, even though most question answering systems have been developed to return only the answer to the question.

Cutrell and Guan [22] in 2007 conducted a user study involving the lengths of text summaries and task type. 22 participants were given 12 tasks each. Half of the tasks were informational and the other half were navigational tasks. Users were exposed randomly for each task to one of three possible summary lengths. The lengths were small (one line of text), medium (two to three lines of text) and long (six to seven lines of text). Examples of each different length summary is shown in Figure 29. Snippets were generated using the same algorithm for picking sentences as was used by MSN Search at the time. Additionally, the target document for the search was randomly inserted into the results at positions ranging from first to eighth. Navigational search tasks benefited most from short summaries. The time to complete the navigational search tasks was lengthened when long summaries were shown. Also, click accuracy was highest for navigational search tasks when using short summaries. On the other hand for informational tasks the time to complete a task decreased significantly when using long summaries while at the same time improving the accuracy of user clicks. Using eye tracking equipment users were also found to fixate on far fewer results when using longer summaries on the informational tasks.

Rose et al. [106] in 2007 described experiments performed at Yahoo on how users respond consciously to changes of the attributes of text summaries. 2500 users from Yahoo were chosen randomly to participate in these surveys. A user was presented with a hypothetical situation and a query to go along with it. They were then presented with a result summary which was hand-generated and varied along certain attributes then asked to answer at most 12 survey questions about the summary. It appears that users saw multiple results per query, though the exact amount is not clear. Each participant answered questions for two scenarios. Experiments tested a number of attributes of summaries such as choppiness, length and query term density. Results showed in some cases the direct expected result occurred, in other cases “halo” effects took place or indirect effects such as improved trust. In the case of summary truncation, it was found best to chop the last part of the sentence off. In the case of length when varying between 2, 3 and 4 lines, no significant differences were found.

Clarke et al. [19] in 2007 analyze click through patterns of MSN Search users according to
Welcome to the Oklahoma City Zoo
http://www.cpb.ouhsc.edu/OKC/OKCZoo/

<table>
<thead>
<tr>
<th>Short</th>
<th>Medium</th>
<th>Long</th>
</tr>
</thead>
<tbody>
<tr>
<td>The oldest zoo in the Southwest and one of the top in the nation, the Oklahoma ...</td>
<td>The oldest zoo in the Southwest and one of the top in the nation, the Oklahoma City Zoo's 110 acres are home to more than 2,800 of the world's most exotic animals.</td>
<td>The oldest zoo in the Southwest and one of the top in the nation, the Oklahoma City Zoo's 110 acres are home to more than 2,800 of the world's most exotic animals. The Cat Forest/Lion Overlook was completed in 1997. New in 1993 was the Great Escape, a simulated tropical forest with gorillas, orangutans and chimpanzees. Also found at the zoo are the Noble Aquatic Center: Aquaticus, a Children's Zoo and Discovery Area, Herpetarium, Island Life Exhibit, Dan Moran Aviary and the Safari Tram. Open 9-5 (Oct-March), 9-6 (April-Sept). Rides additional (weather permitting and seasonal).</td>
</tr>
</tbody>
</table>

Figure 29. Examples of three different summary lengths used in a survey by Cutrell and Guan [22].

Various features of search results. They compared click through numbers of consecutive results. An inversion occurs when the lower ranking document of the two receives more clicks. Inversions signal a preference by the user in the lower ranking documents. A number of features were found to cause inversions in a significant number of cases due to certain easily defined features, most notably when the higher ranked result has no text summary or when it is much shorter than the lower ranking result.

Kaisser et al. [61] in 2008 present a study in which 12,790 queries are judged by three judges as to their category among 17 possible categories and a guess as to the best summary length for that query. They then took a sample of 170 queries from the 1099 queries where all three judges agreed on both query type (where query type can be one of 17 question answering categories such as person, organization, time, advice, etc. including "unjudgable") and best summary length. Judges chose the text for the various summary lengths, namely phrase, sentence, paragraph, section and article from Wikipedia for each of the 170 queries. For each query and each summary length ten judges rated the summary on a scale of 0-10. A regression model was fit to the judges' average rating and the predicted rating by the previous set of judges, and the slope was found to differ significantly from zero. The results suggest that ideal summary lengths differ by task and humans are able to predict ideal summary lengths.

Khan et al. [65] study the affect of removing abstracts from search results and compare that to the removal of the top 5 results. First they asked for 200 users to tell them of difficult search tasks that they came across and were able to solve. 168 tasks were selected from the 200
received for various reasons. Each of the 168 tasks were performed by 20 participants for each of the three experimental groups, totaling 10,080 observations. Search results were limited to 10 having the next page button disabled. From these observations, the removal of abstracts significantly increases search task time. However, search task time was increased to a much larger degree when the top 5 results were removed from the search results.

From these studies it appears that search results missing text summaries are less often clicked. Also, the ideal length of the text summary varies by the type of search conducted. When the search is more informational, then longer text summaries are more helpful, while navigational searches aimed at finding a known website benefited more from shortened text summaries. Additional studies may be performed to better understand which summary techniques provide the better summaries on a large scale and how often the text provided matched what the user was looking for or if better text could have been selected. Very little has been done in terms of obtaining feedback about the text selection process of major search engines.

2.8 Directions and Suggestions for Further Developments

From the works on document summarization explained in this section, we can see that a number of directions have been pursued. Many techniques have been outlined for summarizing single or multiple documents, as well as summarizing web-documents in a query biased manner. These approaches have included statistical techniques, machine learning techniques, graph theory based techniques and more. Yet there has not been a satisfactory combination of the query-biased and the query-independent summary techniques. Our hypothesis is that combining both query-biased and query-independent summary techniques would lead users to better understand documents and be more accurate in judging document relevancy in search. This is the basis for the text selection process in our ReClose system.

We have also seen alternative displays and alternative organization of search results. Some of these approaches have used visual placement of results. Others have used images or thumbnails to enhance the displays of these results. In each of these cases the consideration for accurately conveying the information about the document is not always considered. Not all documents are the same. Some documents should be summarized with an image or multiple images and some not. Additionally, none of these approaches have explored using summarized tables to summarize a document or the HTML structure of the document to summarize a web-document. We believe that including document structure will lead users to more accurately and more quickly make judgments on the relevance of documents linked in search results. ReClose preserves structure from original documents as described in Chapter 5.

Lastly, we saw a number of user studies. These research efforts provide useful insight into the interaction between users and the displays used by search engines. Some useful insights have
been provided about characteristics of document surrogates skipped while other surveys have given insight into ideal lengths of text summaries used in these search result pages. We believe that varying the length of summaries will lead to more efficient use of summaries. Short summaries are appropriate in some circumstances meaning less text must be read. In other circumstances longer summaries lead to a more accurate understanding and the less wrong decisions. We vary summary length in the extended ReClose system in Chapter 6.

The automatic summarization of web documents is a field rich in possibilities with a great opportunity to help users of search engines better interact with search results.
CHAPTER 3

RECLOSE SYSTEM

In this section we take the first step in creating our ReClose system. Here we define how we generate text in the ReClose system. Several automatic summarization approaches for query-independent summarization and for query-independent summarization, but we have seen no good combination of both. We propose to combine both into a single method to give users a view of both the query keywords in context and also an understanding of the major topics of a document. This will help users to more accurately identify which documents to select when reviewing search results from a search engine. Individuals using ReClose summaries showed a 10% increase in the expectation accuracy over individuals using Google summaries, and therefore better user satisfaction.

3.1 Motivation for ReClose Summaries

When a user enters search terms on a search engine, the search engine returns a number of web pages. Each web page is summarized using the title of the web page, a brief text summary and a URL. Each of these linked web pages is called a search result. A user will follow a link to the linked web page to find out more information after reading or scanning a search result.

Search engine users face a unique challenge in determining which of the search results to click on. Each result is represented by a title, brief summary and URL. Search engine users proceed one by one from the top of the search results [22, 47] deciding upon which results to click through to see more information. With so little text to describe a specific web pages, wrong assumptions are often made about the content of a web page.

Inaccurate expectations about web pages in the most extreme cases lead users to click on completely irrelevant web pages or to skip web pages perfect for the needs of a user. More often users find that their expectations of the relevant content of a web page do not match reality. Disappointment occurs when expectations are not met [59].

Consider the two example search results shown in Figure 30. These search results were displayed in response to the query closeness centrality. Closeness centrality is a measure from graph theory used to rank nodes in a graph.
Social Network Analysis
Closeness Centrality. Fernando and Garth have fewer connections than Diane, yet the pattern of their direct and indirect ties allow them to access all the...
www.orgnet.com/sna.html - Cached - Similar

Degree centrality of a node refers to the number of edges attached to the node. In order to know the standardized score, you need to divide each score by...
www.sscnet.ucla.edu/soc/faculty/mcfarland/cent-ans.htm - Cached - Similar

(a) High expectations
(b) Low expectations

Figure 30. Two of the top 10 search results for the query closeness centrality taken from Google on Mar. 23, 2011.

The first of the two search results in Figure 30(a) is titled “Social Network Analysis”. The description shows the use of the keywords “closeness” and “centrality” while giving an example seemingly related to closeness centrality from a social network. One may expect that a website entitle “Social Network Analysis” would have a great deal of information about analyzing social networks, with perhaps a number useful links. One may also expect a lengthy description of closeness centrality is given as well as an intuitive example.

If we were to click through to the top result in Figure 30(a) we would find the web page pictured in Figure 31. Figure 31 shows only a single paragraph is given under the heading “Closeness Centrality”. This single paragraph only completes the thought of the summary text with another two sentences. This is far short of the expectations previously expressed about this web page. It is very likely that a number of users would click through to this page, scan the page for closeness centrality, find very little information and immediately leave the page disappointed.

The second search result shown in Figure 30(b) mentions degree centrality but not closeness centrality. Most individuals would expect that this web page contains no useful information related to closeness centrality. In reality this web page contains answers to a number of exercises related to different centrality measures for a class at UCLA. One of the questions listed on this web page was about closeness centrality, and as shown in Figure 32 this web page includes a table giving a number of closeness centrality calculations for an example graph. For some individuals trying to figure out this measure, actual calculations would provide workable examples to test your understanding of the concept. From the search result for this web page, likely very few users would click through to this web page missing out on the valuable example calculations.

The two example search results provided show there is room for improvement in the representation of search results. Summaries created for search engine results mainly aim to create indicative summaries meant to help a user decide if a search result is relevant [114]. If too much emphasis is placed on providing relevance evidences, summaries may become misleading. Ideal summaries also provide users a view of the objectives, scope and findings of a web page [78].
It is in this light that we propose our novel ReClose algorithm. ReClose combines two types of summaries into a single summary which will enable users to estimate more accurately the content of summarized web pages. More accurate user expectations will lead to fewer disappointments, and in general better user satisfaction in search engine user experiences.

Each time that a user views a summary of a web page, that user develops some set of expectations about a given web page. Those expectations could be that the web page summarized includes no relevant content. A user may expect to find a relevant image or video on a web page. Users may also review a web page summary and expect to find a detailed explanation of an answer to a specific user question. We measure the accuracy of user expectation as it relates to the length of relevant content. In our study we have asked users the extent to relevant content on a web page given only the summary and then the extent of relevant content after viewing the summarized web page. Accuracy is judged by matching relevant content expected and relevant content found.

ReClose improves upon previous query-biased and query-independent approaches to summary generation. It then combines an improved query-biased summary generation technique with an improved query-independent summary technique. Query-biased summaries are focused on
the keywords of a search query [114]. Usually this leads to a view of keywords used in a web page within the context of one or two sentences. Query-independent summaries provide a global view of a web page [77]. When these two summary types are combined, users will see both the local and global contexts of the user’s keywords. Adding a global view to the local view of keywords in a summary, we prove leads to more accurate expectations. More accurate expectations will lead to better decision making in deciding which search results to click through and will lead to better search engine experiences.

3.2 ReClose Approach

The ReClose approach to summarization aims to improve the searching process by providing summaries which give search users both a local and global context of their keywords. This is done through the combination of two approaches to summarization (see Figure 33). First the sentences of a document are sorted using a query-independent approach to summarization. Next a second ordering of sentences is created using a query-biased method. Lastly, the two orderings are combined into a labeled, single summary which generally expands the view of a user.
beyond just the keywords from current approaches to summarization. This section explains both the query-biased and query-independent approaches and how these approaches are combined.

3.2.1 Query-Independent Ranking

Query-independent summarization summarizes whole documents while ignoring query keywords. These approaches aim to give readers the gist of very long documents in a very short text summary. Several approaches have been previously mentioned for query-independent summarization in Section 2.1 of Chapter 2. Approaches range from unsupervised graph based approaches [30, 97] and statistical approaches [77] to supervised approaches [68, 112]. ReClose explores the use of sentence characteristics in an unsupervised environment. Specifically we explore the use of closeness centrality and sentence order to rank sentences for use in summarization.

An explanation of our techniques requires some formalization. We start with ranked set of documents $D_1, D_2, \ldots, D_m \in R$ where $R$ contains all search results returned by some search engine. Each document $D_j$ contains an ordered set of sentences $s_1, s_2, \ldots, s_n$. Rather than working directly with the sentences we will often compute a number of attributes of each sentence, of which we will use the notation $\phi_a(s_i)$ denotes the $a^{th}$ attribute on the $i^{th}$ sentence of some document.

**Closeness Centrality**

We use closeness centrality to rank all sentences in a document based on how similar they are to the rest of the document. It is assumed that the most similar sentence to the rest of the document will be the best sentence to represent the whole document.

To use a graph based approach like closeness centrality one must create a graph from the document being analyzed. This is done by treating each sentence ($s_i \in D$) as a node. We then
create undirected edges between sentences based on similarity. Graph-based approaches to summarization are effective at using self-similarity within a document to find representative sentences. This self-similarity can be seen in the edges.

Similarity is measured using the Jaccard index [57] on the bag of words representation of each sentence as follows:

$$WordOverlapSimilarity(w_1, w_2) = \frac{|w_1 \cap w_2|}{|w_1 \cup w_2|}$$  \hspace{1cm} (24)

where $w_1$ and $w_2$ are sets of words representing two different sentences. It may be observed that word overlap as defined in the previous equation has a maximum value of one and a minimum of zero. All edges with a similarity score of zero were removed from the graph.

Closeness centrality [35] is one approach to globally ranking nodes within a graph. It ranks highest the node with the shortest average distance to the rest of the nodes in the graph.

In 1947 a number of graph theory problems were explored by Bavelas [8] among those was the passing of messages. It was concluded that messages are passed through a graph in the least amount of time if they originate from the center. This idea of passing of messages forms the basis for the idea of finding nodes with the shortest average distance to all other nodes. A measure of a node being central to a graph was proposed in [70]. Since that time, closeness centrality has its applications in the social sciences, and is used quite frequently in describing social networks. [90] demonstrated that closeness centrality could be used to rank authors in physics, biomedical research and computer science. The ranking using closeness centrality showed the connectedness of authors based on the co-authoring of papers.

Closeness centrality was also used in ranking sentences for summarization in [97] and in [131]. Patil and Brazdil is the most similar to our method. However, they disallow the addition of distances when traversing edges. Instead on long paths, the total path length is calculated as the longest edge. We allow for the accumulation of edge lengths through simple addition. Both [97] and [131] combine the output of closeness centrality with other inputs. For example Patil and Brazdil combine closeness centrality with sentence order to more heavily weight early sentences. Xie combines closeness centrality with several other graph theory based centrality measures and other sentence statistics such as paragraph location in the document. We expect to achieve a competitive query-independent sentence ranker.

The use of closeness centrality is justified by assuming that the most representative sentence in a document is also the sentence that is on average most similar to the rest of the document. Closeness centrality assumes that edge values represent distances between nodes and that distances may be added through traversing multiple nodes. Word overlap similarity is transformed into word overlap distance in the following manner: Closeness centrality assumes that edge values represent distances between nodes. Word overlap similarity is transformed into word
overlap distance in the following manner:

\[ \text{Distance}(w_1, w_2) = 1 - \frac{|w_1 \cap w_2|}{|w_1 \cup w_2|} \]  

(25)

It may be observed that two identical sentences have a distance of zero. This will have no affect on the overall solution, as each other node will have identical distances to each of the two identical sentences.

The closeness centrality score is calculated as the average shortest path distance for each node to all other nodes in the graph. This means that the highest ranking sentence in a document will be the sentence most similar to each other sentence, or the sentence with the most in common with the whole document. The JUNG Java framework\(^1\) is used to calculate closeness centrality.

Closeness centrality as used in this dissertation is calculated in the following formula, similar to that used in [107] and in [35].

\[ \text{Closeness}(s_i) = \frac{n - 1}{\sum_{s_j \in D \setminus s_i} \text{ShortestPathDist}(s_i, s_j)} \]  

(26)

where \(s_i\) represents any node (sentence) in the graph, \(n\) is a count of the number nodes in the graph (document), \(D\) represents the set of all nodes or the document, and the function “ShortestPathDist” refers to the shortest path distance between two nodes. This value will be the inverse of the average distance of each node to all other nodes and will rank the nodes from most central to least central based on distance.

![A sample graph with weighted undirected edges.](http://jung.sourceforge.net/index.html)

As an example consider Figure 34. Let us begin by computing the closeness score for node A. The shortest path from A to B has a distance of 0.7. The shortest path from node A to node D requires the traversal first of the path from nodes A to B then from nodes B to D, for a total distance of 0.9. The shortest path from node A to node C traverses the path from nodes A to B and B to C for a total of 1.1. Then the closeness score for node A would be computed in the following equation.

\[ \text{closeness}(A) = \frac{4 - 1}{0.7B + 0.9D + 1.1C} \approx 1.11 \]  

(27)

\(^1\)http://jung.sourceforge.net/index.html

54
Table 5

Closeness scores for sample graph.

<table>
<thead>
<tr>
<th>Node</th>
<th>Closeness Score</th>
<th>Avg. Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.11</td>
<td>0.90</td>
</tr>
<tr>
<td>B</td>
<td>2.31</td>
<td>0.43</td>
</tr>
<tr>
<td>C</td>
<td>1.67</td>
<td>0.60</td>
</tr>
<tr>
<td>D</td>
<td>2.14</td>
<td>0.47</td>
</tr>
</tbody>
</table>

The closeness scores would be computed in the same fashion for all other nodes. The closeness scores for the graph in Figure 34 are shown in Table 5. Closeness centrality is the inverse of the average distance from a node to all other nodes. In Table 5 average distance is also displayed as an intuitive measure for understanding the graph and closeness centrality. From this table one can see that using closeness centrality, node B is the most central node with a closeness centrality score of 2.31 and the smallest average distance to all other nodes of 0.43. The centrality of this node is also apparent when viewing the graph. Node D is a close second with a closeness centrality of 2.14 and having a very similar average shortest distance to node B in the graph.

Closeness centrality as has been described so far requires that all nodes are directly or indirectly connected. Often in documents there are one or more sentences that would have no edges connecting them to the rest of the graph. Closeness centrality computes the shortest path between each pair of nodes. If no path exists, this presents a problem for the algorithm. To overcome this, unconnected nodes and small components that are unconnected are found and removed. If multiple components remain in the graph, closeness centrality scores would need to be computed separately for each component, and then rankings would be combined after the fact.

Among the documents tested on, there were never multiple large components found.

Closeness centrality scales as a factor of the number of nodes and edges. Currently the time complexity to compute the closeness centrality scores for all nodes in a graph is $O(np + n^2 \log n)$ [34], where $n$ represents the number of nodes in the graph and $p$ represents the number of edges in the graph. In the case of a fully connected graph the complexity becomes $O(n^3)$ and in the case where there are a minimum number of edges, or $p \approx n$, then the complexity becomes $O(n^2 \log n)$.

Okamoto et al. [91] achieved a better time complexity with their algorithm TOPRANK(k), by producing a ranking that with high probability ranks correctly the top $k$ nodes. Their algorithm achieves a time complexity of $O((k + n^{\frac{2}{3}} \cdot (\log n)^{\frac{1}{3}})(n \log n + p))$. In the case of a fully connected graph where $k$ is a small constant, this becomes $O(n^{\frac{2}{3}} \cdot (\log n)^{\frac{1}{3}})$, which is smaller than $O(n^3)$ to compute the exact rank for all nodes.
Sentence Order

A very simple approach to ranking sentences for inclusion in summaries is to rank the sentences by the order the sentences were written in a particular document. Often times well-written documents will begin with more general statements appropriate for summarization. Up until Svore’s RankNet [112], no other news article summarization technique had achieved significantly better summaries, than simply using the first couple of sentences for the summaries.

Generally no score is necessary when using sentence ordering. However, we will consider using sentence order in combination with closeness centrality, and we will consider two formulations of sentence order scoring. [97] uses the following equation for weighting sentence order:

\[ \text{SentenceOrder}_{\text{Nonlinear}}(s_i) = \frac{1}{\sqrt{i}} \]  

where \( s_i \) represents a single sentence and \( i \) is the index of a sentence. This formulation provides a nonlinear weighting giving most weight to the earliest sentences.

We additionally consider a normalized version of the sentence order as follows:

\[ \text{SentenceOrder}_{\text{Linear}}(s_i) = \frac{|D| - i + 1}{|D|} \]  

\( D \) is the ordered set of sentences that make up a document. The first sentence in a document is given an index of 1 and the last index will be equal to \( |S| \). The sentence order parameter is in the range \([1,1]\), where the maximum value of 1 is for the first sentence.

Combining Query-Independent and Sentence Order Rankings

Closeness centrality is effective at finding sentences that are most similar to the rest of the document. However, it does not consider the placement of these sentences. Sentence placement is usually intentional within a document providing hints about the purpose of each sentence. Closeness centrality and sentence ranking are very different in their individual approaches. Closeness centrality and sentence order have no overlap in the factors each considers in rankings. These two approaches would seem to be complimentary to each other. We suspected that a combination of the two approaches would provide better results than either approach individually.

When combining closeness centrality and sentence order we consider an equal weighting of the scores of each approach. Additionally, we normalize the closeness centrality scores. This combination is shown below, where an equal weighting would occur when \( \lambda = 0.5 \).

\[ \text{score}(s_i) = \lambda \text{SentenceOrder}(s_i) + (1 - \lambda) \frac{\text{Closeness}(s_i)}{\max_k \text{Closeness}(s_k)} \]  

56
3.2.2 Query-Biased Ranking

A second module in the ReClose methodology is the query-biased ranking. This is the approach taken by major search engines in producing search result summaries. The goal is to find content which most helps users find relevant web pages. Most often this means content that directly contains query keywords.

Our technique to ranking sentences in a query-biased fashion models the current summary techniques of Google, Yahoo and Bing together. It does this by first obtaining large sets of data where the query, web page content and search engine summaries are known. It then converts this raw text data into numerical data for use with regression modeling. We model the sentence selection process of all three search engines combined. The result is a regression model. The output of the regression model is used to score each sentence within a document. This produces a sentence ranking that we can use to generate query-biased summaries.

Sentence Selection Training Data

This process begins with a large number of queries. For example we used 185 queries. Each query is submitted one at a time to Google, Yahoo and Bing. Our system records for each search result the summary text returned by the search engine and the URL. After obtaining URLs from each of the three search engines, we obtain the intersection (Google ∩ Yahoo ∩ Bing) of URLs within search results for each query. We are modeling the text selection process for all three search engines combined, and therefore only consider URLs of which we have summaries from all three search engines.

Next we download the documents located at each of the URLs obtained previously. For each of these documents, we split the visible content on sentence boundaries. In the case of HTML documents, we also consider the meta description. The meta description is a human generated summary of a web page hidden within the source code of many HTML pages. The meta description content is also divided on sentence boundaries and added to the available sentence content.

Data Preprocessing

Our modeling process requires as input numeric features. As a next step, the large pile of text must be converted to a number of numeric inputs using a number of features ($\phi_a$). In our case each data record in the data set represents a single sentence ($s_i$).

The real-valued input features used to represent a sentence are the following: query overlap, query synonym overlap, meta description, contains verb, sentence order, word count, exact query match, and language modeling. These features are described in Table 6. This is an extended set of input features defined by Metzler and Kanungo [85], enhanced with additional inputs: meta description and “contains verb.” It was our belief that the added features would provide
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Min.</th>
<th>Max.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query Overlap</td>
<td>The percent of query keywords in the current sentence.</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Query Synonym</td>
<td>This is the same as query overlap, but also allows for a synonym of a keyword</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Overlap</td>
<td>to be in the current sentence.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Meta Description</td>
<td>The value is 1 in the case that the current sentence comes from the description in a meta tag (not displayed in the browser) and 0 otherwise.</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Contains Verb</td>
<td>A Brill tagger is used on the current sentence. If the sentence contains at least one verb, a value of 1 is given, 0 otherwise.</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Sentence Order</td>
<td>This attribute provides a sense for the location within the document of the current sentence. (sentence order / number of sentences)</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Word Count</td>
<td>A count of the number of words in the current sentence is given.</td>
<td>0</td>
<td>1,134</td>
</tr>
<tr>
<td>Exact Query</td>
<td>1 is returned when all keywords are found in the same order as the query, without terms in between, 0 otherwise.</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Match Language</td>
<td>This feature is based on a language model approach to information retrieval found in [111]. More details on this feature are given by [85].</td>
<td>-103</td>
<td>-2</td>
</tr>
</tbody>
</table>

TABLE 6

Names and descriptions are listed of features used for query-biased sentence ranking. Minimums and maximums are calculated from the data set described in Section 3.3.2.

information critical to modeling the summary techniques of Google, Yahoo and Bing as a combined entity. Different subsets of features are tested to determine the most successful set in Section 3.3.2.

The target output modeled is a measure of the importance of a sentence. Our training data approximates the importance of a sentence using the summaries produced by Google, Yahoo and Bing. More exactly it will be the number of search engines that use the current sentence divided by the number of search engines used, or three in our case. For example if Google and Yahoo both use sentence3 in their search summaries from a particular web page, then the resulting output target for that sentence would be 2/3. In this approach all three search engines are weighted equally. The target output $t_i$ for sentence $s_i$ is calculated as follows:

$$ t_i = \frac{\sum_{j=1}^{E} I(s_i \in S_j)}{|E|} $$  \hspace{1cm} (31)

where $E$ is the set of search engines considered which in this case is Google, Yahoo and Bing. $S_j$ is the set of sentences used to generate the summary for the $j^{th}$ search engine, and $I(s_i \in S_j)$ is an indicator function which is 1 when the sentence $s_i$ is contained in the set of sentences $S_j$ used for the generation of the summary for search engine $j$ and 0 otherwise.
Predictive Model of Sentence Scoring

Now that we have defined the inputs and target output of a learning process, we can apply standard modeling techniques to approximate the query-biased approach of the ensemble of search engines. A good model of the summaries produced by Google, Yahoo and Bing will create a competitive query-biased summarizer [123]. Models are produced which are of the form:

\[ f : D \rightarrow \mathbb{R} \]  \hspace{1cm} (32)

where a function is created which maps elements of any document \( D \) to a real number \( \mathbb{R} \). Training data is created with target values in the range of \([0, 1]\) and in most cases the output of the learned function is in this range, but it is not strictly limited to this range of output.

We use a data set with approximately 180,000 sentences. We evaluated a number of learning approaches with different parameter settings using 5-fold cross validation on this data set.

Once a model has been built for query-biased summarization, it produces a partial ordering on the sentences within a document. Each sentence will first be preprocessed and values given to each input feature. The regression model will then for each sentence map the input values to a real-valued output. The sentences in the document will be sorted according to the regression output in descending order. The highest ranking sentences will be those which would most likely be selected by the majority of Google, Yahoo and Bing.

3.2.3 Combining Sentence Rankings

Our main research hypothesis is that combining query-biased and query-independent summarization approaches will provide users with more accurate expectations for each search result.

In this new approach the sentence rankings from a query-biased approach and a query-independent approach are combined. The summary generation methodology using both approaches is called ReClose. The ReClose approach takes the following steps to produce combined summaries:

1. Create query-biased and query-independent sentence rankings.

2. Select the top ranked query-independent sentence and add it to the summary set.

3. View sentences starting with the top query-biased ranked sentence and proceed down the rankings.

   • Add the current sentence to the summary set if it is not “too similar” to any previously added sentence.
Stop adding when the sentences added from the query-biased ranking reach or surpass half of the maximum allotted characters.

4. Added sentences from the query-independent ranking in the same fashion as the query-biased additions.

5. Cut excess characters from the last added sentence from each ranking so that the summary length falls below the required threshold.

Step 1 in the ReClose approach requires an ordering of sentences from each of the various approaches to summary generation. Each sentence is scored by the query-biased regression model. Ties are broken by sentence order in the original document. All sentences are also ordered using our query-independent approach.

Always the top query-independent ranked sentence is used in a ReClose summary as seen in step 2. There are two labeled halves to a ReClose summary as shown in the example in Figure 35. Each of these halves are labeled to provide information about the sources of sentences used for each portion of the summary. When the query-independent portion of a ReClose summary contains keywords, then the user is informed that the main intent of the linked web page is likely the subject of the query keywords. Much less often would this subtle message be relayed to the summary user if priority was first given to the query-biased portion of the summary. Then less frequently would keywords be in the query-independent portion of the summary.

After selecting the top query-independent sentence, then all sentences required to fill the query-biased portion of the summary are selected in step 3. It is important to show the relevance of a web page to the user through selecting good query-biased sentences. If no evidence of relevance is shown to the user, a user will likely skip that search result. As sentences are chosen for use in a summary, sentences that are “too similar” to currently selected sentences are passed over. This avoids providing redundant summary information. Redundant sentences provide no new insights or information to a user. Similarity is measured using word overlap similarity as defined in Section 3.2.1. The threshold for this similarity measure is a user-defined parameter. This process is stopped once the maximum allowed characters are reached for the query-biased portion of the summary. Each portion of the summary is allowed half of the total possible characters.

At this point the summary is filled beyond the maximum possible characters with additional query-independent ranked sentences as necessary in step 4.

The sentences that will be used in the summary have been chosen at this point in the combination process. The final step, step 5 is to remove characters in excess of the maximum allotted characters. This is done by first taking the last added query-biased based sentence. Then finding the positions in the sentences of any query keywords. If so it is best to preserve these words in the shortened version of the sentence. We attempt to chop off the beginning and end of the
Google summary:
Degree centrality of a node refers to the number of edges attached to the node. In order to know the standardized score, you need to divide each score by ...

ReClose summary:
Page: a) Calculate Degree centrality scores of each node in the network above ...
Keyword: Thus, closeness centrality is the most appropriate. ...

Figure 35. Google and ReClose summaries generated for the query closeness centrality and the web page pictured in Figure 32.

sentence around the query keywords. When it is impossible to prevent the loss of some keywords or no keywords are present, it is preferable to remove the end of the sentence [106].

When a word boundary is selected for character removal, that portion of the sentence is removed. If the word boundary causes the total character count for the query-biased portion of the summary to fall below the maximum allowed, then the maximum size of the query-independent portion is extended by these extra characters. This process is then repeated for the last chosen query-independent summary sentence.

As an example of the kind of summaries produced by ReClose consider the summary comparison shown in Figure 35. This figure shows both the summary generated by Google and by ReClose for the query closeness centrality for the web page shown in Figure 32. The original summary for this web page given by Google gives a user the impression that there is no content related specifically to closeness centrality, but that other centrality methods were discussed. The ReClose summary shows in the query-biased portion of the summary that closeness centrality is discussed on this web page. The query-independent (page) portion of the summary hints at the nature of this web page, that it contains a number of problems and answers from a college assignment on centrality measures. Thus we can see the two summary types working together to provide users more accurate expectations in a limited space.

3.3 ReClose Experimental Results

Our experimental results take a two step process. First we must optimize the parameters of the system in order to produce the best summaries for the user. Each of the modules within ReClose is tuned with data sets that we have collected from online resources. Once the system has been properly prepared, we generated a number of summaries that were evaluated by humans. We have devised a survey to obtain human feedback both for ReClose summaries and how ReClose summaries compare with traditional summarization techniques in Google.

http://www.sscnet.ucla.edu/soc/faculty/mcfarland/soc112/cent-ans.htm
3.3.1 Query-Independent Module Parameter Tuning

To ensure the success of our ReClose methodology, we must fine tune each of the modules. We have devised a query-independent approach which takes advantage of self-similarity within a document using closeness centrality. This approach may benefit from the sentence order within a document as was found by [97].

We measure the various algorithms possible for query-independent sentence ranking using an approach similar to that found in [112]. We created a data set composed of CNN news stories downloaded from CNN.com\(^3\). Many of the news articles listed on CNN, contain a section titled “Story Highlights.” The “highlights” section generally contains 3-4 bullet points with key insights on the news articles. These bullet points are human generated. Sometimes they contain direct quotations from the article, and other times are completely new combining ideas from multiple sentences or paragraphs. These bullet points we take as the gold standard for the summarization process. All summaries will be compared to this gold standard on a word-by-word basis.

More specifically we gathered some 229 articles spanning a large number of topics from world news to sports to politics. These articles came from 4 different news days during January and February 2010.

The ROUGE-1 metric was used in the evaluation of these approaches. This is an unigram co-occurrence metric for comparing ideal summaries to generated summaries. Unigram co-occurrence metrics have been shown [74] to correlate with human evaluations. ROUGE-1 is calculated as shown below adapted from [112].

\[
\text{ROUGE-1} = \frac{\sum_{\text{gram}_j \in R \cap S_i} \text{Count}(\text{gram}_j)}{\sum_{\text{gram}_j \in R} \text{Count}(\text{gram}_j)}
\]

(33)

Here \(R\) refers to the reference or ideal summary. \(S_i\) is the generated summary. In ROUGE-1, \(\text{gram}_j\) refers to unigrams or single words. Another way of stating ROUGE-1 is that it is the number of words in the ideal summary that are also contained in the generated summary. A score of 1 would be a perfect match. Stop word removal and stemming were not used when computing ROUGE-1 scores.

The ROUGE-1 scores obtained using our data set of CNN news articles is shown in Table 7. The best ROUGE-1 score of 0.501 was obtained using a simple, unweighted combination of closeness centrality and sentence order. Second highest was the SumGraph [97] approach. The combination of closeness centrality outperformed either closeness centrality or sentence order individually. Surprisingly, our closeness centrality and sentence order combination significantly outperformed LexRank.

Our methodology is similar to SumGraph in that both use closeness centrality and sentence order but each differs in a number of particulars. Both approaches do not consider

\(^3\text{http://www.cnn.com}\)
A comparison of LexRank, SumGraph, closeness centrality and sentence order individually and together are all shown based on ROUGE-1 scores. Results marked with $\alpha$ differ significantly from that of Closeness + Sentence Order using a two-sided t-test and an $\alpha$ level of 0.05.

sentences with less than 10 words and both use Porter stemming [99]. SumGraph however uses a different sentence distance measure, the nonlinear rather than our linear sentence order scoring, non-cumulative graph distances, and does not remove stop words. These differentiations produce an improved ROUGE-1 score for our combined approach to query-independent sentence ranking.

The goal of this phase of our research was to obtain a competitive query-independent methodology which will be combined with a query-biased technique. This goal has been achieved as shown in Table 7. The integration of our approaches to query-independent and query-biased summarization should support our main hypothesis about the superiority of an integrated approach.

### 3.3.2 Query-Biased Module Parameter Tuning

Our approach to query-biased sentence ranking models the summary approaches of Google, Yahoo and Bing combined. In tuning our system we need to select a set of features to use, an algorithm to model the summary approaches and parameter settings of the chosen algorithm. We have tested in this section several sets of input parameters. We also have tested the following algorithms: gradient boosted decision trees (GBDT), linear regression (LR) and REPTree.

The three algorithms chosen for testing give us very different approaches to feature combination. Gradient boosted decision trees [36, 37] were found to be effective by Metzler and Kanungo [85]. The GBDT algorithm is a large ensemble of regression trees able to represent very complex mappings of features to output. Specifically we used the R implementation of GBDTs [105]. The REPTree [101] algorithm is a regression tree approach. It is possible that summaries for search engines are created using a number of simple rules. If this is the case, and we use the same input features as the rules used by these search engines, than a regression tree representation should be similar to the rule sets used. Linear regression provides a simple linear combination of inputs which adds greater variety to the available modeling techniques. We used the REPTree and linear regression implementations found in Weka [49].

Our data set was created by observing the text summaries and URLs for the top 28 results.

<table>
<thead>
<tr>
<th>Sentence Ranker</th>
<th>ROUGE-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Closeness + Sentence Order</td>
<td>0.501</td>
</tr>
<tr>
<td>SumGraph</td>
<td>0.492</td>
</tr>
<tr>
<td>Sentence Order</td>
<td>0.480</td>
</tr>
<tr>
<td>Closeness</td>
<td>0.461$\alpha$</td>
</tr>
<tr>
<td>LexRank</td>
<td>0.423$\alpha$</td>
</tr>
</tbody>
</table>

TABLE 7
TABLE 8

R-Precision scores for linear regression (LR), REPTree, and gradient boosted decision trees (GBDT) models. In parentheses are the input features to the model which may include: query overlap (qo), meta description (m), contains verb (v), synonym overlap (syn), or any of the other features. “All” means all 8 possible features.

<table>
<thead>
<tr>
<th>Model Description</th>
<th>R-Prec.</th>
</tr>
</thead>
<tbody>
<tr>
<td>LR(qo+m+v+syn)</td>
<td>0.645</td>
</tr>
<tr>
<td>LR(qo+m+v+syn+sent. order)</td>
<td>0.642</td>
</tr>
<tr>
<td>LR(qo+m+v+syn+exact match)</td>
<td>0.632</td>
</tr>
<tr>
<td>REPTree(qo+m+v+syn)</td>
<td>0.629</td>
</tr>
<tr>
<td>GBDT(all)</td>
<td>0.611</td>
</tr>
<tr>
<td>REPTree(all)</td>
<td>0.606</td>
</tr>
<tr>
<td>LR(qo+m+v+syn+language model)</td>
<td>0.582</td>
</tr>
</tbody>
</table>

R-Precision scores for each of 185 queries on Google, Yahoo and Bing. The 185 queries were gathered over two days from the SearchSpy tool available on Dogpile. This tool allows a user to view live queries being entered on Dogpile search engine. For each query, we took an intersection of the 28 URLs from Google, Yahoo and Bing and downloaded each available web page. Web pages were split on sentence boundaries. After removing white space and punctuation each web page sentence and each summary sentence fragment were compared for overlap. Sentences found to overlap the search result summary were marked so that an appropriate output could be learned. This whole process resulted in 181,516 sentences that could be used for the training and testing of supervised learning algorithms. Each algorithm and feature combination were tested using 5-fold cross validation on this data set.

The metric used to determine the best model was R-Precision [79]. R-Precision is a combination of precision and recall. Often precision and recall are defined in terms of relevant and irrelevant documents. R-Precision is computed as the precision score over the top $n$ documents where $n$ is equal to the number of relevant documents for that specific query. In our usage of R-Precision instead of considering the number of relevant documents, we consider the number of relevant sentences. For the computing of precision, sentences that appeared in 2/3 of the search engine summaries were considered as “relevant” or as sentences that should be highly ranked. R-Precision is defined as follows:

$$\text{R-Prec.} = \frac{|\{\text{relevant sentences}\} \cap \{\text{top } n \text{ sentences}\}|}{|\{\text{relevant sentences}\}|}$$

where $n = |\{\text{relevant sentences}\}|$.

The highlights of our testing can be seen in Table 8. Here we see that our best R-Precision score of 0.645 was achieved by linear regression using query overlap, meta description, contains verb and synonym overlap features. These four features were found to work best. The addition of

4http://www.dogpile.com/dogpile/ws/searchspy
5http://dogpile.com
Figure 36. Linear regression model used for ranking query-biased sentences.

\[
0.0189 \times \text{query overlap} + 0.0173 \times \\
\text{output} - \text{synonym} + 0.6569 \times \text{meta desc.} + \\
0.0045 \times \text{contains verb} + 0.0004
\]

<table>
<thead>
<tr>
<th>sim.</th>
<th>Sentences</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9</td>
<td>big-o-notation 1 point 2 points 3 points 1 day ago</td>
</tr>
<tr>
<td>0.8</td>
<td>Returns the value to which the specified key is mapped in this hashtable. Returns: the value to which the key had been mapped in this hashtable, or</td>
</tr>
<tr>
<td>0.7</td>
<td>Written by Mike Macauley on November 14, 2010 View Comments</td>
</tr>
<tr>
<td>0.6</td>
<td>Submitted by Martin Kucej; posted on Monday, November 08, 2010</td>
</tr>
<tr>
<td>0.5</td>
<td>ToLowerCase Returns a copy of this string converted to lowercase. ToUpperCase Returns a copy of this string converted to uppercase.</td>
</tr>
<tr>
<td>0.4</td>
<td>An array can hold all your variable values under a single name. And you can access the values by referring to the array name.</td>
</tr>
<tr>
<td>0.3</td>
<td>This class implements a hashtable, which maps keys to values. the previous value of the specified key in this hashtable, or</td>
</tr>
</tbody>
</table>

TABLE 9

Example sentences and similarity scores. Punctuation and stop words are removed and stemming performed before scores are calculated.

any other feature to our linear regression model decreased the R-Precision score. Examples include adding sentence order for a very small decrease, the exact match feature for a mild drop and adding the language model feature brought the R-Precision score below all other models displayed.

Both REPTree and GBDT automatically select which features are used. In both cases all features available were supplied. Neither REPTree with a R-Precision score of 0.629 or GBDT with a score of 0.611 were able to match the success of our highest achieving linear regression model.

The model used in ReClose is shown in Figure 36. The largest weight is given to the meta description feature, showing how often sentences from the meta description of a web page are used by Google, Yahoo and Bing. It should be noted that when ordering sentences by query-biased scoring, ties are broken by sentence order within the original document.

We have implemented two competitive sentence ranking techniques to achieve our main research goal, which is to combine query-independent and query-biased sentence ranking techniques to achieve better summaries than currently produced.

3.3.3 Combining Modules Parameter Tuning

There are two user defined parameters in the combination module of ReClose: a) maximum sentence similarity and b) maximum summary character count.
Similarity between sentences is defined as word overlap similarity as shown in Section 3.2.1. When deciding upon a threshold value we had two considerations in mind. First we considered the distribution of similarity scores for a number of examples. A majority of non-zero similarity scores fall between the range of 0.0-0.3. Only about 0.7% of the similarity scores fall above 0.4 in our data set. A threshold of 0.4 would affect very few summaries. Second we come to understand the range of possible similarity values by viewing a sample of sentence comparisons for a number of similarity bins. Examples of sentences and their similarities in the range of 0.3 to 0.9 are shown in Table 9. Real differences in the meanings of sentences can be seen around the value 0.4 leading again to the decision to use 0.4 as the maximum similarity threshold value.

The next parameter to adjust is the maximum character lengths of summaries. We selected a maximum character length similar to the length of Google summaries. This better enables a comparison of ReClose summaries with Google summaries. In viewing Figure 37 we see a large amount of variability in the lengths of Google summaries. We do not attempt to model this high variability. Instead we take the most frequent lengths which are in the range 150-165 characters. This high frequency range makes up 60% of the Google summaries and includes the median value of 154 characters. The other 40% includes summaries from length 52 through 149 characters. A view of the narrow range of summary lengths used by ReClose and query-independent summaries compared with the much larger range of Google summary lengths can be seen in Figure 37.

3.3.4 Survey of User Expectations

The hypothesis that combining summary techniques will improve the accuracy of user expectations cannot be measured in an automated fashion. There is no gold standard available for these types of measures. It is therefore imperative to obtain human feedback on summary
perceptions. Our survey tracks expectations based on summaries and again after a user sees the final destination page. In this way users provide us information on summaries that provided accurate impressions and those that did not. We analyzed our hypothesis that with more information available in our combined ReClose summaries that these summaries would provide more accurate impressions to survey users. This is tested through a comparison of the accuracy measurements comparing ReClose summaries with the traditional query-biased summaries of Google.

Survey participants were shown a number of summaries. For each summary participants were asked to explicitly mark down their expectations. Participants would then view the web page summarized and make a mental comparison of their expectations and the actual content of the web page. Participants would then mark down the difference. The smaller the difference between the expectations held by each participant when compared to reality, the more accurately a summary conveys the right message about a particular web page.

### Estimate Relevant Content Quantity

<table>
<thead>
<tr>
<th>Rank</th>
<th>Summary</th>
<th>Expected Relevant Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Data mining is the process of extracting patterns from data ... The related terms data dredging, data fishing and data snooping refer to the use of data ...</td>
<td>None Sentences Paragraphs Pages Book</td>
</tr>
<tr>
<td>2</td>
<td>Overview Generally, data mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it ...</td>
<td>None Sentences Paragraphs Pages Book</td>
</tr>
<tr>
<td>3</td>
<td>Most local search applications are atomic ... WalkScore is a site that provides aggregate search results - a computed value for the walkability of a location ...</td>
<td>None Sentences Paragraphs Pages Book</td>
</tr>
<tr>
<td>4</td>
<td>Data mining is a powerful new technology with great potential to help companies focus on the most important information in the data they have collected about ...</td>
<td>None Sentences Paragraphs Pages Book</td>
</tr>
<tr>
<td>5</td>
<td>... data mining, including the foundations of probability, the foundations of statistical data analysis, and most of the classic machine learning and data ...</td>
<td>None Sentences Paragraphs Pages Book</td>
</tr>
</tbody>
</table>

Figure 38. Survey search results ready for participant evaluations.

Each survey participant was randomly assigned to one summary approach. Survey participants were asked to view summaries for at least two queries. Participants chose queries from a batch of 25 preselected queries. For each query, participants viewed exactly five summaries. Figure 38 shows the query-independent summaries for the query Data mining. There were five options available when marking down one’s expectations. These options are “None”, “Sentences”, “Paragraphs”, “Pages” and “Book”. Each of these options represent that amount of content relevant to the query shown. Choosing the option “None” signifies that the survey participant expects the web page to be irrelevant. Selecting the option “Pages” shows that the user expects that the web
page is not only relevant, but contains large amounts of information related directly to the current query.

After a participant has marked their expectations for all five summaries for a single query, the participant will be presented with links to each web page and options for comparisons, see Figure 39. Participants were given five options again for the comparison: “Much Less”, “Less than Expected”, “Same”, “More than Expected” and “Much More”. The summaries giving the most accurate impressions will obtain the mark “Same” meaning an individual’s impressions match the real content of the web page. If either the “Much Less” or “Less than Expected” options are marked, then the survey participant has experienced disappointment at finding less relevant information than expected. In the case that an individual finds more relevant content than expected, then that individual is positively surprised by what they find. However, if a user underestimates a web page’s usefulness, that same user may not click on a relevant web page. In that case a user has missed out on relevant information.

Figure 39. Survey search results after evaluations were made.

### 3.3.5 Survey Participants

Voluntary participation in the survey consists of two pools of people. The first pool is made up of graduate and undergraduate students in the Computer Engineering and Computer Science Department at the University of Louisville. The second group was made up of members of the online board game community hosted by BoardGameGeek.com.

For each of the two pools of people 25 queries were selected which related to their area of expertise. For the CECS students 25 Wikipedia article titles listed under Computer Science were selected. Examples of queries used by the CECS graduate students are: associative array,
Survey results comparing query-biased (Google), query-independent and ReClose summaries. (Percentages may not add exactly to 100 per row due to rounding.)

<table>
<thead>
<tr>
<th>Summary Source</th>
<th>Disap - pointment</th>
<th>Matched (Accuracy)</th>
<th>Positive Surprise</th>
<th>Total Partic. Summaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query-Biased</td>
<td>25 (21%)</td>
<td>60 (50%)</td>
<td>35 (29%)</td>
<td>13 120</td>
</tr>
<tr>
<td>ReClose</td>
<td>29 (19%)</td>
<td>83 (55%)</td>
<td>38 (25%)</td>
<td>13 150</td>
</tr>
<tr>
<td>Query-Ind.</td>
<td>29 (26%)</td>
<td>48 (44%)</td>
<td>33 (30%)</td>
<td>8 110</td>
</tr>
</tbody>
</table>

**TABLE 10**

**bioinformatics, software tester, and cryptography.** Queries for the board gaming community came from the glossary found on BoardGameGeek.com. Examples of vocabulary found on BoardGameGeek.com are as follows: *euromage*, *analysis paralysis* and *goznard*. Members of this gaming community are generally familiar with these terms, using them often in online forums.

The experiment was conducted on the Internet. Graduate and undergraduate students in the CECS department were asked in person to visit the URL at their earliest convenience. A thread was also posted on BoardGameGeek.com which asked willing participants to visit a listed URL. The pool from which a survey participant was from was identified in the URL.

In these experiments there were 34 participants, 23 students and 11 board game community members (after removing outliers). Participation in this survey was completely voluntary. This meant that there were occasional participants that were not as engaged during the survey as they should have been. For example one individual completed their second 5 search results in 16 seconds. During this second set all 5 search results had the same options chosen. Therefore, this second set of search results were removed as an outlier. Participants on average completed a set of 5 search results in 2 and a half minutes, thus showing 16 seconds to be an outlier. Survey responses were removed from the database when a user did not complete the survey or marked all of the same options for one of the two questions. Survey responses were removed for the individual that completed the entire survey in 2 min., being an extremely short amount of time to complete this survey. In all there were survey results for 6 survey participants removed from the original 40 respondents, which resulted in the final count of 34 survey participants.

### 3.3.6 Survey Analysis

The main results of this survey are summarized in Table 10. In this table we can see that the ReClose summaries resulted in the most accurate user impressions. In 83 out of 150 (or 55%) ReClose summaries survey participants found exactly what they expected when clicking summary links. ReClose summaries also led to the fewest number of disappointments at 19% of summaries. Disappointments occur when users find less relevant content than expected, especially in the extreme case that a seemingly relevant document is not actually relevant. Query-biased (Google)
<table>
<thead>
<tr>
<th>Summary Source</th>
<th>False Pos.</th>
<th>False Neg.</th>
<th>Accurate</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Query-Biased</td>
<td>12 (10.0%)</td>
<td>7 (5.8%)</td>
<td>84.2%</td>
<td>120</td>
</tr>
<tr>
<td>ReClose</td>
<td>10 (6.7%)</td>
<td>10 (6.7%)</td>
<td>86.7%</td>
<td>150</td>
</tr>
<tr>
<td>Query-Independent</td>
<td>7 (6.4%)</td>
<td>8 (7.3%)</td>
<td>86.3%</td>
<td>110</td>
</tr>
</tbody>
</table>

**TABLE 11**

Summary evaluation only considering false positives, false negatives and accuracy.

summaries took a close second in fewest disappointments by percentage with only 21% of summaries resulting in disappointment.

As was expected query-independent summaries led to the least accurate user impressions. We would expect users to have less accurate impressions in a search environment, when the query keywords are not directly considered when summarizing a document.

Overall for the survey size used, there is no statistically significant difference found among all three summary approaches. Using the $\chi^2$ test on the count data shown in Table 10 resulted in a p-value of 0.427, larger than the usual 0.05 threshold. The results are not statistically significant for the number of summaries observed by participants. We would expect that with a much larger sample size, ReClose would maintain its advantage and would achieve a significant edge over current summary methods.

Table 10 contains all observed summaries in our survey. Table 11 contains only false positives and false negatives from the same survey results. A false positives occurs when a user clicks on a web page only to find the web page irrelevant. In this case a user expectations far exceed the actual relevant content. The second case we call a false negative. False negatives occur when a user would have skipped over a relevant web page. This means that the user underestimated a web page to the point that they did not even click through to see that web page.

In light of these two extremes (false positives and false negatives), we see in Table 11 that again ReClose summaries are most accurate (86.7%). Users were most likely to click through to irrelevant web pages (false positive) when using query-biased summaries (10%). ReClose and query-independent summaries were nearly identical in helping users to avoid false positives at 6.7% for ReClose summaries and 6.4% for query-independent summaries. This shows that query-independent and ReClose summaries do much better at helping users to avoid skipping over relevant information not apparent in Google summaries. On the other hand, the case of false negatives (clicking through to irrelevant web page), query-biased summaries did best at a low 5.8%. ReClose summaries were close behind at 6.7%. Query-independent summaries were slightly worse in this category achieving a percentage of 7.3%.

When we consider overall how well each summary type helped users to avoid extreme circumstances, we see that ReClose summaries have an advantage over query-biased summaries.
<table>
<thead>
<tr>
<th>Top QI Sent.</th>
<th>Accurate</th>
<th>Inaccurate</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included</td>
<td>33 (64.7%)</td>
<td>18 (35.3%)</td>
<td>51</td>
</tr>
<tr>
<td>Not Included</td>
<td>20 (37.0%)</td>
<td>34 (63.0%)</td>
<td>54</td>
</tr>
<tr>
<td>Unknown</td>
<td>7 (46.7%)</td>
<td>8 (53.3%)</td>
<td>15</td>
</tr>
</tbody>
</table>

**TABLE 12**

The accuracy of Google summaries based on a summary’s inclusion of the top query-independent sentence.

ReClose and query-independent summaries were accurate in avoiding extreme cases in 86.7% and 86.3% respectively. Query-biased summaries achieved an accuracy of 84.2%. Although extreme results in expectation discrepancies are somewhat rare, ReClose summaries performed best at helping users to avoid these major disappointments.

### 3.3.7 Discussion of Experimental Results

In the previous section, the results of the performance of ReClose summaries were shown in comparison with Google and query-independent summaries. The section provides further evidence that users are more accurate when using summaries that combine both query-biased and query-independent summarization approaches.

We first start by considering query-biased summaries, in this case Google summaries. Google summaries will on occasion use sentences in a summary that rank highly both by our query-biased sentence ranker and our query-independent sentence ranker. In Table 12 we take the “Query-Biased” row from Table 10. We then connect the sentence scores produced by the query-independent module with the sentences selected by Google. We then establish two groups, those summaries that contained the top ranking query-independent sentence and those that did not. There were 15 summaries for which we were not able to trace the origins of the text spans used by Google. Within these two main groups we consider summaries that “matched” or in which users accurately estimated relevant content and those that did not. Interestingly Google summaries that contained the top ranked query-independent sentence produced accurate user estimates 65% of the time. Google summaries that did not contain the top ranked sentence using our query-independent ranking were accurate only 37% of the time. Applying a $\chi^2$ test to this count data (ignoring unknowns) reveals a significant difference among the two groups. The p-value obtained was 0.008.

Google summaries that were able to combine an element of query-independent summarization to the traditional query-biased summaries produced more accurate user impressions.

In Figure 40 for the same groupings used in Table 12 we see that overall the expectations are higher for summaries that contain top ranking query-independent sentences. These types of summaries both provide higher expectations of relevant content and provide more accurate user impressions.
Figure 40. User expectations of query-biased summaries that did or did not contain the top ranked query-independent sentence for the web page.

<table>
<thead>
<tr>
<th>Top QB Sent.</th>
<th>Accurate</th>
<th>Inaccurate</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Included</td>
<td>29 (49.2%)</td>
<td>30 (50.8%)</td>
<td>59</td>
</tr>
<tr>
<td>Not Included</td>
<td>19 (37.3%)</td>
<td>32 (62.7%)</td>
<td>51</td>
</tr>
</tbody>
</table>

TABLE 13

The accuracy of query-independent summaries based on a summary’s inclusion of the top query-biased sentence.

We now divide the query-independent summaries into those that contain the top ranked sentence in the query-biased ranking and those that do not. This division is shown in Table 13. Again we see that the summaries that included sentences ranked highly by both rankings were more accurate. In this case query-independent summaries including top ranked query-biased sentences provided accurate impressions 49.2% of the time. While the other set of summaries were only accurate in 37.3% of cases. Again we see that when including both a view of documents as a whole and documents on a keyword basis, the user impressions generated are more accurate.

It now comes as no surprise that query-independent summaries including sentences of top ranked query-biased sentences, lead to higher expectations (see Figure 41). In the case of survey participants seeing only query-independent summaries, the difference is even more drastic. There are a high number of cases where expectations were zero ("None") when not including top ranked query-biased sentences. While at the same time the summaries that did include these top ranked sentences, led to high expectations many times ("Pages").

Overall this last piece of analysis shows that summaries that provide both local and global context to keywords lead to more accurate user impressions. Sometimes query-biased or query-independent summaries may inadvertently include elements of both summary approaches. However, better summaries can be created by intentionally including some text meant as a
Figure 41. User expectations of query-independent summaries that did or did not contain the top ranked query-biased sentence for the web page.

query-independent summary and other text meant as a query-biased summary. This will lead to more accurate summaries, better user decision making, and overall better user experiences with search engines.
CHAPTER 4

COLOR-CODED RECLOSE SUMMARIES

Color-coded ReClose summaries use the improved text selection system proposed as ReClose and combine that with a color-coding scheme. The purpose is to provide users of search systems more information about each search result without requiring additional space. Two color-codings exist. The first uses color depth to show usage information about the query keywords. The second color-coding flags terms departing from the topics of the query to warn the user that a particular document may not be as relevant as may be expected from the summary text. Overall these two color-coding techniques along with the text selection of ReClose combined for a 20% increase in click precision comparing the color-coded ReClose summaries to Google generated summaries.

4.1 Motivation for Color-Coded ReClose Summaries

Search engine usage has become a part of every day life for internet users. Every time a search is conducted on Google or Bing a list of search results is presented to the user. One of the major challenges that users face as they search for that needle of information in the Internet haystack is deciding which of the search results presented is relevant to their search needs and which are not. When conducting searches for facts and information the choices are not always obvious.

Each search result is composed of a title, a short text summary and an abbreviated URL. The title usually is revealing about the overall message of a web page. However, it is written by the web content creator and may be a slogan of a company or an advertising pitch, which can be misleading. The URL can be very helpful when one is familiar with the host contained in the URL, but many URLs encountered are not familiar to us.

The text summary is extracted from three possible locations [86, 42, 134]. 1) Spans of text may be taken directly from the content of a web page. 2) It may come from the HTML meta description. The meta description is embedded in the HTML of a web page. It is not displayed to users visiting a web site, but is usually a general description of a web page or web site hand
written by the content creator. 3) Lastly the text could come from the Open Directory Project. The Open Directory Project is a community built directory of websites with a number of short, human-written website summaries.

When search results are presented to users, the user has the task of deciding which results are relevant to their search and which are not. Within information science it has been found that as many as 80 factors contribute to the decision of judge deciding which documents are relevant to a particular search [109]. Users typically make this decision in a matter of seconds. When a user decides to click on a search result there are two possible outcomes that depend on a user's expectations for that web page:

1. The user's expectations were not met leading to disappointment.

2. The user's expectations were met or exceeded resulting in satisfaction.

Users may incorrectly skip relevant content missing out on potentially important information, but it is the feeling of disappointment (possibility 1) that will most negatively affect a search experience. We aim to improve the user's accuracy in click decisions for the purpose of decreasing occurrences of disappointment.

![Figure 42. A search result from the top 10 search results for the query closeness centrality on Google as seen May 11, 2011.](http://www.dmoz.org)

As an example of the kinds of disappointment that may be realized consider the search result to the query closeness centrality pictured in Figure 42. Closeness centrality is a graph theory measure used for ordering nodes. The search result shown in Figure 42 has a title of "Social Network Analysis". This page is dedicated to the analysis of social networks. Closeness centrality as is shown in the summary is clearly mentioned. One also finds an example description of closeness centrality in a social network. One may expect that this page contains a lengthy description of closeness centrality followed by this example. However, clicking through to the result page leads to Figure 43. The web page does discuss social network analysis as would be expected by the title, but there is only a single paragraph on closeness centrality. This single paragraph only describes a brief example barely longer than the text summary given by the search result. This web page did not meet the previously detailed expectations and would lead to disappointment on the part of the searcher.

1http://www.dmoz.org

75
Closeness Centrality

Fernando and Garth have fewer connections than Diane, yet the nodes in the network more quickly than anyone else. They are in an excellent position to monitor the information happening in the network.

Figure 43. Web page at http://www.orgnet.com/sna.html as seen on May 11, 2011.

The user in the previous search example would be aided by the two main features of color-coded ReClose summaries. First, keywords are highlighted with color depth to provide global context rather than just the local context of one or two sentences surrounding a keyword. This "global context" refers to the extent of discussion on a web page containing the query topic. In the previous example, the user would have been aware before clicking that there were very few occurrences of the terms "closeness" and "centrality" by visual clues of color enhanced query keyword highlighting.

Secondly, major departures from the main topics of a web search are flagged. If the main subject of a web page is different from the intent of the search user, then a topic term is shown in red. This warns the user that the keywords may be peripheral to the main subject of the web page. Both color depth and topic word flagging are shown in this research to effectively improve user click precision and decrease user disappointment. This in turn will improve the efficiency of the user and lead to better user experiences with the search engine.
4.2 Color-Coded ReClose Description

The goal of color-coded ReClose summaries is to increase the accuracy (precision) with which users click on search results to find relevant documents. Increasing accuracy will in turn lead to fewer disappointments and a better user experience. Color-coded ReClose summaries aim to improve upon current search result summaries using two main parts. First, we highlight query keywords using variable shades of blue to show the depth of usage of those query keywords on a web page. Second, we display in red terms central to the web page's topic which potentially differ from the topic of the keywords searched for.

4.2.1 Color-Coded Keywords

We color-code keywords to provide additional context about the usage of keywords. The query-biased summaries of say Google or Bing will provide one or two text spans generally that show one or two usages of the keywords searched. In this way the context on a scale of say plus or minus ten words from the keywords are shown. Our color-coding of the keywords adds depth to each keyword just as colors can provide terrain depth on a topographical map. Many topographical maps will provide a key that shows the elevation range of the map and provide different colors for each subdivision of elevation. This "color-coding" provides users of these maps a more intuitive view than simply a set of contour lines to understand depth. Our depth refers to the frequency of query keywords on a web page. This gives a user a greater appreciation for how long discussions involving the keywords may be compared to other search results.

The color coding can be thought of as a function of the following form:

\[ f: \mathbb{Z}^+ \rightarrow C \]  \hspace{1cm} (35)

where the function takes a positive integer \( \mathbb{Z}^+ \), representing the frequency of a query term within a document and maps that term count to a particular color in the set of all colors \( C \). The color is calculated as follows:

\[ c_t = \text{Color}(C^i + \Delta C \times \min(count(t), threshold)) \]  \hspace{1cm} (36)

where \( C^i \) represents the color of the smallest value as a vector of RGB integer values ranging from 0-255. The \( \Delta C \) is the difference between the color of the smallest and largest frequencies. \( count(t) \) produces a count of the number of times term \( t \) appears in document \( D \). The \( threshold \) value is the upper value of color mappings. All counts above this threshold will be mapped to the color of the highest allowable color representation. The \( \text{Color}() \) function takes a vector of RGB values and produces a color. \( \Delta C \) is calculated in the following manner:
Here $C^f$ refers to the final color used in the scale. The subscripts $R$, $G$, and $B$ represent the red, green and blue entries in each color vector respectively.

The key used in our surveys is shown in Figure 44. We count the frequency of each keyword on a web page after the removal of stop words and use of Porter stemming. Then for each possible frequency between zero and 63 a different shade of blue is used. (A keyword may be contained in a summary and not on a web page if it is contained in the meta description but not the web page’s content). A diagram of color-coding query keywords is shown in Figure 45. Now summaries of web pages that talk at great lengths about say “canines” will be distinguishable from a web page that has very little text which mentions “canines”.

<table>
<thead>
<tr>
<th>Color Names</th>
<th>Frequencies</th>
<th>RGB Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duke blue</td>
<td>63</td>
<td>0 26 87</td>
</tr>
<tr>
<td>Egyptian blue</td>
<td>30</td>
<td>16 52 166</td>
</tr>
<tr>
<td>deep sky blue</td>
<td>0</td>
<td>0 191 255</td>
</tr>
</tbody>
</table>

**TABLE 14**

Colors used to create the color scale.

The exact colors used are in Table 14. We chose to use a light blue (deep sky blue) for the smallest frequency value of zero. Then to make the range between 0 and 30 more pronounced we chose an intermediate, but fairly dark blue (Egyptian blue) at a frequency of 30. A dark blue (Duke blue) was used for a frequency of 63+ which was still distinguishable from regular text in black. Highlighted words in blue can be quickly searched even with a limited number of red terms [50].

For the colors chosen in Table 14, we present a formula that produces the color for each possible term count.

$$\Delta C = (C^f_R - C^f_R, C^f_G - C^f_G, C^f_B - C^f_B)/\text{threshold}$$

(37)
Extract Query Keywords
Extract Web Page Terms
Count Terms
Select Color
Highlight Terms

Figure 45. Process of color-coding query keywords.

\[ c_t = \begin{cases} 
    \text{Color}(0 + 0.5 \times f, 191 - 4.6 \times f), \\
    255 - 3 \times f) & \text{if } f < 30, \\
    \text{Color}(16 - 0.5 \times f_a, 52 - 0.8 \times f_a), \\
    166 - 2.4 \times f_a & \text{if } f \geq 30. 
\]  

(38)

where \( f \) represents \( \text{count}(t) \) and \( f_a = \min(f - 30, 63) \).

It is unlikely that most users will be able to know exactly what color represents which frequency, but it will be obvious which summaries contain more frequent keywords. For example in the summary in Figure 45 the keyword "database" is more frequent in the document than the keyword "building". It will also be obvious which end of the scale each keyword belongs to, whether the tail end 0-20 or the top end of 60+, which is where the real value is had.

4.2.2 Flagged Words

The goal of the flagging module is to visually differentiate web pages in which the search keywords are the main topic from those web pages where the search keywords are peripheral to the main topic of the page.

We assume that the most frequent term(s) in a document is central to the main topic of a document. We are not concerned with presenting to the user the exact topic of a document, but instead are intent upon finding the departures of document topics from the searched topic. Generally only a single term is considered for flagging to limit the information overload of the user. A single term should allow a user to discern the potential topic of a document in addition to the summary text.

We have designed an algorithm to determine if we should flag any terms within a
document summary. Often due to the nature of search the most frequent term in a document is one of the keywords. These terms should not be flagged. Additionally, many terms belong to the same topic as the query keywords and should not be flagged. Our algorithm does not flag terms highly related to the queried topic. The steps in our algorithm are diagrammed in Figure 46 and are outlined below:

1. Determine the most frequent term in a document.

2. Obtain a count of the top ranking documents also including this top term.

3. Threshold the percentage of documents containing the top term.

The algorithm begins by first determining the most frequent term in a document (step 1). This involves counting term usage within a document after the removal of stop words.

Once we have determined the most frequent term in a document, we then consider all other top ranking documents returned for the search (step 2). In our case we used the top 28 documents (not including the current document), since this is the maximum number of documents returned through Google’s Web Search API.

The percentage of top ranking documents for the current search containing the most frequent term is then thresholded (step 3). We used a threshold of 60%. Terms that occur in more than half of the top documents for a search generally are highly related to the search terms. Examples of terms that you would find for the query **algorithms** are shown in Table 15. At the very top of the list you have the term “algorithms” which is also what we searched on. This term should appear in every document retrieved. As you decrease the threshold, the number of terms available rapidly increases after 60% to include words such as “can” and “books” which are less

---

Figure 46. Process of flagging terms.

---

2[http://code.google.com/apis/websearch/]
<table>
<thead>
<tr>
<th>Percent</th>
<th>Words</th>
</tr>
</thead>
<tbody>
<tr>
<td>90-100%</td>
<td>algorithms</td>
</tr>
<tr>
<td>80-89%</td>
<td>computer</td>
</tr>
<tr>
<td>70-79%</td>
<td>programming, use</td>
</tr>
<tr>
<td>60-69%</td>
<td>home, information, search, page, number</td>
</tr>
<tr>
<td>50-59%</td>
<td>new, sort, problem, science, data, graph, can, C, set, books</td>
</tr>
<tr>
<td>40-49%</td>
<td>get, privacy, mathematical, random, related, software, site, last, structures, generate, complexity, including, online, tool, link</td>
</tr>
<tr>
<td>30-39%</td>
<td>solution, course, another, adds, time, need, counting, common, download, languages</td>
</tr>
<tr>
<td>20-29%</td>
<td>test, approximation, copy, function, heap, assignment, appearing, stony, maintained, arithmetic, 2007, understanding, lower, instructor, NP</td>
</tr>
</tbody>
</table>

**TABLE 15**

An incomplete list of the words found in 30% or more of the search results for the query *algorithms* after stemming and the removal of stop words.

related to the topic of “algorithms”. However, towards the bottom of this list you find both highly related terms like “NP” as in “NP-complete” or “NP-hard”. You also find terms like “2007” which have nothing to do with the search. Terms that are found in 60% of search documents are both rare, and highly related to the search at hand. There still are some very common terms on web pages like “home” that sneak into that ranking.

Terms that do not meet the threshold will be displayed in the summary colored red. For example see the summary in Figure 46 where the term “JDBC” is flagged. JDBC refers to one method in Java for connecting to databases. It is distantly related to the query *building a database*, but clearly shows that this particular document is less focused on the building of the database, and more focused on Java related issues.

More formally, $R$ represents a set of documents which are the top $m$ documents returned by a given search engine $(D_1, \ldots, D_m)$. Each document $D$ contains a set of terms $t_1, \ldots, t_n$. Using this set up we can obtain the most frequent term $t_j$:

$$t_j = \arg \max_i \text{count}(t)$$

(39)

where $\text{count}(t)$ provides us with the count of the number of times that term $t$ is found in a given document. To discover if the term should be flagged or not we must calculate the percentage of other documents in $R$ which also contain this term. This we can obtain by calculating $\text{percent}(t)$ as follows:
Figure 47. An example summary using color-coding and ReClose summary text generation for the query building a database.

\[
\text{percent}(t_j) = \frac{\sum_{D_i \in R \setminus D_x} I(t \in D_i)}{|R \setminus D_x|}
\]  

(40)

Here \(D_x\) represents the current document in which we have found \(t_j\) the most frequent term of that document. \(I(t \in D_i)\) is an indicator function which produces 1 when the term \(t\) is found in document \(D_i\) and 0 otherwise. When this fraction exceeds some user defined threshold, for which we have used 60%, then we should flag term \(t\).

After we have determined that a term should be flagged for a particular summary, we must ensure that the flagged term is included in the summary. To accomplish this we filter the query-independent sentence ranking to only include sentences including the flagged terms. This ensures that the flagged term will appear in at least one sentence included in the summary.

4.3 Color-Coded ReClose Experimental Results

We hypothesize that color-coding ReClose generated summaries that users will have more accurate expectations of the web pages summarized. To test this we created a survey that allow us to compare the accuracy of user expectations based on summaries. We mainly compare color-coded ReClose summaries against Google summaries. We additionally compare ReClose summaries with and without color-coding to ensure that the color-coding made a difference, and that text selection alone was not the main cause for improvement.

4.3.1 Survey Participants and Survey Design

For our survey we recruited 21 volunteers among undergraduate and graduate students in the Computer Engineering and Computer Science department at the University of Louisville. Surveys were conducted exclusively online.

The summary analysis was broken down into two parts and repeated for each of the three summary techniques under comparison. First a user would be shown 5 summaries for a randomly selected query. For each summary a user would mark if they would click on that summary. Then they would mark the amount of relevant content expected. The choices available were “None”, “Sentences”, “Paragraphs”, “Pages” or “Book”. Rather than just obtaining which results a user would click on, we obtain a finer grained understanding of the process through how much relevant
content a user expected.

Second, users were provided links to each destination page and viewed these pages one at a time. A user marked down the actual amount of relevant content using the same options presented for expectations. In this way rather than finding out if a user believes a page is relevant or not to their search, we can also monitor lesser disappointments, such as a user expecting to find pages and pages of relevant content but in actuality only finding a couple of sentences. In this case the document is still relevant, but the user is likely not satisfied with the results.

Survey participants were shown 5 summaries per summary type for a total of 15 summaries.

4.3.2 Summary Data

Survey participants were randomly assigned three queries out of a pool of 15 queries. These queries were chapter titles and project titles from an introductory course in computer science so that all query topics were familiar to the survey participants. Some example queries were logic gates and creating a web page.

For each of the 15 queries, 28 search results were obtained from Google. We downloaded each linked web page in the search results resulting in 400 successfully downloaded and parsed web pages out of 420 possible. We only used 5 search results per query. To decide which search results to use, we randomly selected web pages from two pools. The first pool was likely to have search results with flagged summaries because when the frequencies of terms in a document was ranked the query keywords had a low rank. The second pool contained the top 5 search results as ranked by Google.

After determining the pool of search results most likely to be flagged and the top Google search results, randomly we select 2-4 results from the pool of results likely to be flagged. Then the remaining results are taken starting starting with the top ranked Google result from the second pool.

4.3.3 Results and Discussion

User Relevance Ratings

First we verify the relationship between user click behavior and the relevance markings. Figure 48 shows the distribution of expected relevance obtained from users for search results clicked and skipped. This figure shows that no user would click on a result if they expected no relevant content. If a user expected only a sentence or two of relevant data, users were unlikely to click (72% or 64/89). A natural division emerges from the expectation results. Users expecting "Sentences" or "~one" would skip the result 82% (116/141) of the time, leading us to call this section "irrelevant". The other half of the relevant spectrum we labeled "relevant". Users clicked through 84% (146/174) of the time when expecting "Paragraphs" or more of relevant information.
Performing a \( \chi^2 \) test on the count data revealed by this dividing line resulted in \( \chi^2 \) value of 134.8 and a p-value < 0.001, clearly showing a significant difference between these two groups. Click through and expectation have a lot in common, but expectations provide more insight into the mental process of the search users.

The expectations of survey participants was fairly inaccurate. Only 34% (108/315) of expectations matched exactly the actual relevant content of web pages. In another 34% (108/315) of expectations resulted in actual content being opposite of expectations in terms of the relevant/irrelevant split mentioned earlier. For example there were 16 occurrences where a survey participant marked a relevant expectation of “Paragraphs” or higher only to find no relevant content.

The discrepancy between expectations and actual content surprised some of our survey participants. For example one participant said, “It also showed me how some summaries can be very misleading. They can seem like there is a lot of information on your topic but don’t really have much or they can look as though they wouldn’t have any relevant information but end up being the best site for you. The survey really made me look at how search results are displayed differently.”

In our survey color-coded ReClose summaries achieved a much lower percentage of disappointment at 23% than Google summaries achieved at 34% as shown in Table 16. Disappointment was recorded when the relevant content was lower than what their expectations. When we conduct a \( \chi^2 \) test on the count data comparing Google and color-coded ReClose we obtain a \( \chi^2 \) value of 2.8 and a p-value of 0.09. This p-value does not fall below the usual threshold value of 0.05. However, there still remains an obvious difference between the results of Google summaries and color-coded ReClose summaries that would become more pronounced with the additional survey participants.
TABLE 16

Disappointment counts and percentages for three summary techniques.

<table>
<thead>
<tr>
<th>Summary Source</th>
<th>Disap - pointment</th>
<th>Satisfied or Surprised</th>
<th>Total Summaries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>36 (34%)</td>
<td>69 (66%)</td>
<td>105</td>
</tr>
<tr>
<td>ReClose</td>
<td>28 (27%)</td>
<td>77 (73%)</td>
<td>105</td>
</tr>
<tr>
<td>Color-Coded</td>
<td>24 (23%)</td>
<td>81 (77%)</td>
<td>105</td>
</tr>
</tbody>
</table>

TABLE 17

Click precision and recall comparison.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Click Precision</th>
<th>Click Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>66% (38/58)</td>
<td>60% (38/63)</td>
</tr>
<tr>
<td>ReClose</td>
<td>75% (39/52)</td>
<td>64% (39/61)</td>
</tr>
<tr>
<td>Color-Coded</td>
<td>80% (49/61)</td>
<td>70% (49/70)</td>
</tr>
</tbody>
</table>

We now look at the precision with which users chose to click on a result. Considering that a majority of users did not click when expectations were a couple sentences or less, we label all web page views with a few sentences or less of relevant content as “irrelevant.” Survey participant marking more than a few sentences worth of relevant content are labeled as “relevant.” Dividing clicks into relevant and irrelevant allows for us to calculate click precision. We define click precision as the percentage of summary views with clicks that led to relevant web pages. Click recall is the percentage of relevant documents that were clicked. The results of these calculations for each summary technique can be seen in Table 17.

Table 17 shows that users clicked more often (61 times) and had a higher click precision (80%) when using color-coded ReClose summaries than either Google (66%) or ReClose summaries highlighting with bold (75%). When users used Google summaries they clicked through to relevant web pages only about 2/3 of the time that they clicked. With more precise clicks, users using color-coded ReClose summaries also clicked on more of the relevant content having a click recall score of 70%. Individuals using Google and bolded ReClose summaries skipped more relevant content having recall scores of 60% and 64% respectively.

In practice a higher click precision will be more noticeable to users. Users are aware of clicks to irrelevant content, experiencing disappointment. However, there is no form of feedback for click recall. Users are not aware that they have skipped over a relevant document. One of the main objectives of color-coded ReClose summaries was to improve the click precision for users. From the numbers in Table 17 it is clear that color-coded ReClose summaries improve the precision of users, both over Google summaries and ReClose summaries without color-coding. This leads to fewer disappointments in practice.
TABLE 18

Expected and actual relevant content for web pages with a query keyword count of 60+.

Color-Coding Analysis

We now consider the effectiveness of the two color-coding features: color-coded keywords and flagged words. In this section comparisons are only made between bolded and color-coded ReClose summaries. You can be assured both outperformed Google summaries, but here the focus is just on the added color-coding features. We first consider the color-coded keywords. The scale we used allowed for usage count differentiation from 0-63. Summaries were not evenly distributed across this range. Nearly half (49% or 37/75) of the summaries used had at most a keyword with 0-9 usages on the web page summarized. We would expect that users would have low expectations for summaries that at most contained keywords on the low end of the scale. Looking at the results, there was no perceived change in behavior for summaries containing low count query keywords (0-9) to medium count (10-59). Only in the case of high count query keywords (60+) was there a noticeable change in behavior. There were 13 summaries (17%) with at least one query keyword with a usage count of 60+. For these 13 summaries, participants found the actual relevant content to be high. For example no matter the summary type, more than 50% of views led to actual relevant content in the “Pages” level. This was rarely expected when using bolded ReClose summaries, see Table 18. Bolded ReClose summaries led to 23% of pages views in the “Pages” level expectations. The color-coded ReClose summaries more often led to higher expectations in line with the actual content. In 44% of views, color-coded users identified an expectation in the “Pages” range. Color-coded ReClose summaries also led to the highest actual relevant content as well at 67%. In the case of high usage count keywords, color-coded ReClose summaries led to justifiably higher expectations.

First we compare the effect that flagging had on expectations which can be seen in Table 19 in the column marked “Expected Relevant”. In this table documents were broken into two groups, documents that had terms flagged by color-coded ReClose (rows marked “Flaggable”) and documents that did not (rows marked “Not Flaggable”). When color-coded ReClose summaries had flagged terms, the expectations were much lower (29% expected to be relevant) than those same summaries without color-coding (40% expected to be relevant). A similar pattern was found for
color-coded summaries without flagged terms having higher expectations. This shows that the flagging of terms directly affected the expectations of the user.

<table>
<thead>
<tr>
<th></th>
<th>Expected Relevant</th>
<th>Actual Relevant</th>
<th>Click Prec.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReClose</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flaggable</td>
<td>21/53 (40%)</td>
<td>24/53 (45%)</td>
<td>70%</td>
</tr>
<tr>
<td>Not Flaggable</td>
<td>30/52 (58%)</td>
<td>37/52 (71%)</td>
<td>78%</td>
</tr>
<tr>
<td>Color-Coded</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flaggable</td>
<td>15/52 (29%)</td>
<td>25/52 (48%)</td>
<td>57%</td>
</tr>
<tr>
<td>Not Flaggable</td>
<td>44/53 (83%)</td>
<td>45/53 (85%)</td>
<td>87%</td>
</tr>
</tbody>
</table>

TABLE 19

Expected and actual relevant content for documents that would (Flaggable) and would not (Not Flaggable) have summaries with flagged terms.

There is a much lower percentage of documents found to be relevant that had flagged terms. Even in the case where flagged terms were not shown to users (bolded ReClose summaries), 45% of documents that could have been flagged were found to be relevant compared to 71% of documents that would not have had flagged terms. What is interesting is how flagging affects the click precision of users. Those that saw the flagged terms had a click precision of 57% on flagged summaries compared to 70% that did not see the flagging for these same summaries. However, users expected more and were more precise when color-coding was available and no flagged terms appeared in a summary achieving a click precision of 87% compared to 78% without color-coding. Overall with far fewer clicks among flagged summaries, the overall click precision was higher for the color-coded version of ReClose (see Table 17).
CHAPTER 5

SUMMARIES PRESERVING DOCUMENT STRUCTURE

Summaries preserving document structure seek to take advantage of the structure and multimedia present in many HTML documents. In this way summaries that include portions of tables, images, etc. will use present this content as originally presented in the summarized documents. This helps users more quickly process the information and better understand the summarized document.

5.1 Motivation for Summaries Preserving Document Structure

We explore the enhancement of ReClose summary generation with structured text and multimedia. The inspiration for this study comes from the jumbled mess of text that is shown to a user when summary text is extracted from a table. For example consider Figure 49(a) in which the summary text shows a table of football players invited to the NFL combine. A search engine user may correctly guess that the text comes from a table, but one may just as easily guess that how one sees the text now is how it is displayed on the originating web page. Compare now the impression given by viewing the same text as a table as the original author of this text meant it to be viewed as in Figure 49(b). In the summary preserving structure there is no mistaking the authors intent of the summary text. This web page contains a table of player names.

In addition to tables we also explore the usage of lists, images, buttons, text fields and hyperlinks. Table 20 shows three examples of the benefits of adding structure to document summaries. The first summary shows a number of Italian locations in an illogical sequence. The structured version shows that this text comes from a list and that each of the locations forms a hyperlink to another document. The second summary is clarified by adding the context of buttons and text fields. This context shows the true purpose the sentence “iPhone iPad Android” to be three radio buttons so that one may search based on different devices.

From Figures 49(a) and 49(b) and Table 20 we see that structured text and multimedia adds additional insight not present in current search result summaries. We propose the addition of structured text and multimedia to current search results which will have the following benefits for users:
Figure 49. Search engine summary with (b) and without (a) structure.

<table>
<thead>
<tr>
<th>Type</th>
<th>Traditional Summary</th>
<th>Summary with Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>List</td>
<td>Villa Ponte 17435 Villa Rental in Tuscany, Betolle Italy Tuscany Siena South Siena</td>
<td>Villa Ponte 17435: Villa Rental in Tuscany, Betolle</td>
</tr>
<tr>
<td></td>
<td>near Betolle</td>
<td>» Italy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>» Tuscany</td>
</tr>
<tr>
<td></td>
<td></td>
<td>» Siena</td>
</tr>
<tr>
<td></td>
<td></td>
<td>» South Siena</td>
</tr>
<tr>
<td></td>
<td></td>
<td>» near Betolle</td>
</tr>
<tr>
<td>Buttons and</td>
<td>Chomp. Download Chomp iPhone Android</td>
<td>Chomp Download Chomp iPhone Android</td>
</tr>
<tr>
<td>Text Field</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>iPhone ○ iPad ○ Android</td>
</tr>
<tr>
<td>Images</td>
<td>tenerife property sw1. exclusive land for sale; exclusive land for sale; exclusive land for sale; exclusive land for sale; exclusive land for sale; exclusive land for ...</td>
<td>Tenerife Property SW1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• exclusive land for sale</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• exclusive land for sale</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• exclusive land for ...</td>
</tr>
</tbody>
</table>

TABLE 20

Comparison summaries taken from a traditional search engine with and without structure.

- Additional semantics are added to the search results.
- Users will have a better understanding of the summarized web pages through structured text and multimedia.
- Users will be better satisfied with the search results.
This work offers two contributions. First it provides an alternative basic unit to automatic summarization in search. Rather than all forms of summarization being based only on text, this work expands that view to include HTML entities such as tables and lists that contain text. Second, this work provides experimental validation that automatic summarization systems which output summaries with structured text and multimedia in addition to unstructured text outperform unstructured text alone.

5.2 Traditional Summaries within Search

When using a traditional search engine, such as Google, Yahoo or Bing, a majority of search results look like the example shown in Figure 49(a). As with this example, a majority of summaries used in search engines use plain text. These summaries come from three main sources [86]. They can come from the text of a web page, hidden summaries embedded in web pages and from third parties which use web page summaries. The text of the summary may change depending on the search query used to find a given web page. How exactly any one search engine determines what text to use is not well known. However, we do know some characteristics common to many of the summaries used [123, 124]. Summaries used in search engines tend to contain query keywords, which leads us to refer to these summaries as “query-biased”. These summaries may contain a different form of a keyword or synonyms of the keyword. These summaries more often use text from summaries of web pages than web page content, when such summaries are present. Additionally, sentence-like content containing a verb is more often used than say several menu links. The result is a one or more text spans that is meant to both provide context to the query keywords and give insight into the purpose of the provided web page [124].

In addition to these common plain text summaries, traditional search engines including Google, Yahoo and Bing will occasional provide additional multimedia content. Some video summaries (such as YouTube videos) may include an image from the video as part of the summary. Results from image, video, product, etc. type searches when integrated into the results often use the type of content searched as the summary. For example, the integration of image search contains several images. Occasionally a top search result when linking to a web site home page, may display commonly used hyperlinks within that page shown below the search result. Certain types of searches and search results will contain a single image, such as a recipe search like waffles and a result from a common recipe web site like allrecipes.com. Such traditional search engines also now offer a glimpse at the content of a web page when one mouses over a web page, such as Google’s Instant Previews [43]. Other enhancements to summaries used in search engines include number of reviews on some web pages, date of indexing of a web page, and news images. The limited use of structure such as lists, tables and buttons within search result summaries shows that there is still a lot of room for improvement in the usage of structure.
5.2.1 Preserved HTML in Summaries

For the purpose of structure we will mainly focus on HTML documents, though the ideas extend to all types of web documents. An HTML document is made up of HTML tags and text content. These HTML tags provide text formatting, text structure, hyperlinks, meta data about the document and more. With a few notable exceptions, summary text is drawn from the plain text content. There are three main exceptions we wish to discuss. 1) Sometimes web page summaries are provided by webmasters. These summaries are embedded within the HTML and are not visible to users viewing the web page through a browser. These summaries are stored in the page header. 2) Images attached to web pages through HTML tags may optionally include text representations. The text representation of an image may be used within a search engine summary. 3) Summary text may also be taken from the Open Directory Project (DMOZ\(^1\)). Open Directory Project summaries are human generated and are used to explain links within the project. The text of these summaries is freely available and is used by search engines.

5.2.2 HTML Structure and Multimedia

We propose to include in summaries used in search engines text structure and multimedia. Specifically we will be using structure common to HTML documents, which includes the following:

- Images
- Tables
- Lists
- Buttons and text fields
- Hyperlinks

Images

Images have already been shown to be effective at improving search results [73, 132, 133]. Images are more quickly comprehended than text. The difficulty lies in the selection of an image that will both effectively represent the content of a web page and at the same time avoid using images that may be offensive to the audience of text-based search engines.

Traditional search engines use summary text of images in search summaries. For the purpose of testing we replace the image summary text with the original image, resized to fit within the search results. As an example of the types of summary text and images seen, see Figure 50. In Figure 50(a) we see that the text used as part of a search summary, which is “exclusive land for sale.” This text is quite vague and could refer to any of a large number of plots of land that could

\(^1\)http://www.dmoz.org/
be for sale. The image shown in Figure 50(b) is much more specific and provides much greater insight into the relevance of the returned web page.

Tables

Tables within HTML are used for two main purposes: for the display of tabular data and to position other HTML elements within the web page. When HTML tables are being used to display tabular data, then the tables provide additional meaning that should be present in the search engine summary. These help users to understand the data contained in them through structure. In the second case where the table is used only to position text and HTML elements on a web page, the table provides no new information and the structure should not included in a summary. Each type of table must be identified before summaries are generated. The usage of HTML to provide tabular data and placement appear nearly the same when parsing the HTML. We will not explore the automatic classification of these two uses of tables as has already been done elsewhere [15].

The usefulness of a table's structure can be seen in the previously mentioned Figures 49(a) and 49(b). The meaning of the text is more readily available by viewing the potential NFL players in a table as shown in Figure 49(b), rather than a seeming run-on, nonsensical text as in Figure 49(a).

Lists

Currently search engines combine the bulleted or numbered lists of web pages into a single line. This can lead to unclear meaning in a summary. Consider the example shown in Figures 51(a) and 51(b). If you look at the last part of the original text summary provided by a traditional search engine, Bing you will find “work order management preventative maintenance”. These five words together do not form a logical idea. However, when we put this text in the context of the original formatting we see that “work order management” is separate from “preventative maintenance” and as separate entities represent different concepts. Different meanings are produced as you separate textual entities as they were meant to be presented.
ideal for sites who wants to start on a CMMS. a truly free unrestricted software. asset management. work order management preventive maintenance ...

(a) Search result summary observed in Bing.

- ideal for sites who wants to start on a CMMS
- a truly free unrestricted software
- asset management
- work order management
- preventive maintenance

(b) Summary preserving a list.

Figure 51. Two search result summaries: (a) is the plain-text version observed in Bing and (b) is the summary preserving structure version using a list.

Buttons and Text Fields

Text may appear on buttons or be part of an HTML form. In these cases the intent of the text is unclear. Adding the context that text comes from a button make evident the intention of that text. For example “Sign Up” has more meaning if it shown as being part of a button. Additionally, HTML inputs such as radio buttons, check boxes, etc. provide context to entities that may clump together. For example in Table 20 the entry that contains “iPhone iPad Android” as part of the summary shows that these are all separate entities meant to be selected when the radio buttons are inserted between each unique concept.

Hyperlinks

Lastly, HTML links can be very useful in certain situations. Some web pages are useful to searchers, because they contain links to very relevant documents. We show users which terms are hyperlinks. Users of the search engine are then more certain of the intention of the linked text and where a web page may lead in one’s search.

Traditional search engines (such as Google, Yahoo and Bing), on occasion add to a search result hyperlinks for common menu items. This allows users to skip a step by moving directly to a subpage within a web site. However, this does not help to clarify summaries used in search engines.

5.2.3 Proposed ReClose Extension

Currently our ReClose system for summary generation takes as its basic unit sentences [124]. The ReClose system would break a document into sentences. We then used to rankings to generate summaries. For the first ranking, sentences were represented by a number of numeric attributes such as the percentage of query words present in the sentence. These attributes were fed into a trained regression model to provide a score. The scores were ranked and the top sentences chosen to represent the query-biased portion of the summary. The second ranking created a graph.
structure where each node was a sentence and the edges were measured as the dissimilarity between sentences. Closeness centrality was then used to rank sentences based on which sentence was most similar or least dissimilar to the rest of the sentences in a document. These sentences represented a summary of the document as a whole.

We are proposing that we change the basic unit in our ReClose system or in any summary generation system. Rather than a sentence being the basic unit, instead we could have sentences, table cells, multimedia, list items and buttons. This would require that additional attributes be created to represent structure of one of these basic units. Rankings would be evaluated across all basic types and the top scoring structured unit, multimedia or text would be chosen for the summary. New ranking algorithms are in the works as well as new data structures for storing these basic types.

The output of this system would also look very different. No longer would summaries be made up of only highlighted text, but it would also include on occasion structure and multimedia. To evaluate summaries preserving document structure compared to plain text summaries, we ran a user evaluation. Users were shown a query, a text summary and given the option to select the relevance of that web page. Users were evaluated on the time taken to make a selection and the accuracy of their selection. Lastly, users then marked their preference either for a system that outputs plain text summaries, as they are used to or the new types of summaries that include structure and multimedia.

5.2.4 HTML Entity Extraction Module

For the purpose of experimentation we take previously selected text from a web page and injects structure and multimedia elements found in the original HTML web page. We do this by parsing the original HTML web page into a convenient data structure which contains a string representation of the available text content. The chosen summary text is matched against the string, then our data structure injects the HTML that was found between and surrounding the sequence of characters found to match.

Parsing an HTML web page involves using regular expressions as found in Java to loop through each HTML tag and process the text content followed by the HTML tag. Each HTML tag is processed to remove unnecessary formatting. The text content is also preprocessed. The string representation of the text content removes all non-letter, non-number characters and changes all letters to a uniform lower case. Search engines may use a different letter case than the original web page, add or remove space characters and change punctuation. One reason punctuation is changed by the search engines is to simulate table or list boundaries found in the HTML.

We also save an array of character information objects along with the string representation. The character information objects save what additional characters beyond letter
and number characters surround each character that would need to be included to reproduce the summary text. Each character also contains a reference to a linked list of tag objects. If HTML tags are found between characters than the preceding character stores the list of tags.

When a substring of text content matches the chosen summary text, then a new summary is produced by going character by character and adding to a buffer the characters referenced by that character information object. If the character is the first character, then preceding non-letter, non-number characters are added first, like a starting quotation mark. Then each middle character adds what characters and tags may lie between each of them, such as space characters and HTML tags that we intend to reproduce. When the last character is reached, than ending context is also added like the ending punctuation. This may be repeated if multiple text spans are found, in which case ellipses (...) are added between text spans.

When we process the HTML tags we do the following: We remove all attributes except for vertical and horizontal alignment for table cells, image width, height and source attributes, and we add the text in image “alt” attributes and meta tag description values to the string. Within the tag objects we make references to the corresponding starting or ending tag, such as the start and end of a table row using the \(<tr>\) tag. When the summary includes only the start or end tag, then we can reference the missing close or open of a tag and insert that missing tag into the appropriate location in the summary. We do the same thing for parent tags, which are necessary for list items and table cells. Also, if image text is used, then the starting and ending portions of the image tag are sure to be added to replace the original text with an image tag.

This whole process produces near identical text to the text submitted with the addition of HTML tags that provide text structure and multimedia.

5.3 User Evaluation of Summaries Preserving Document Structure

We tested the effectiveness of structured text and multimedia on a number of search engine users. The evaluation process was made up of two major steps. The first step involved users reviewing 30 search results one at a time. There were three parts to a search result. Each search result displayed a unique search query, search question and search engine summary. Within a single evaluation 15 of the summaries for search included structured text and multimedia (test group) and another 15 excluded these elements (control group). Users were then asked whether they “should” click or not. Some users click without thinking when using search engines. We wanted to be sure that users were clicking only when the result appeared to be relevant. The user had three options to use when answering if they “should” click on a result: yes, no or unsure. The response to this question was saved for each search result as well as the time it took for each individual search result.

The second step of the user evaluation asked the user of two systems which would they
prefer. In this step there were four summaries per "system". One of the systems represented structured text and multimedia summaries. The other system presented original Bing summaries. Which system appeared on the right was randomly chosen. Users would select which of the two systems they preferred. A participant could then optionally comment on why they made their selection.

The search queries used were taken from two main sources. First 150 queries were obtained from the website SearchHippo\(^2\) on Feb. 14 and 22. SearchHippo is a search engine which gives users the opportunity to see the most recent 15 queries submitted to the website. All queries which appeared to be URL's were removed. Second we obtained the top 20 trending queries on Google Hot Trends\(^3\) on Feb. 22, 23, 24 and 28. From these queries, duplicates and most non-English queries were removed. Our final count was 185 queries.

All 185 queries were submitted to a traditional search engine, Bing, through Bing's Search API\(^4\). We observed the URLs and summaries for the top 40 search results for each of the 185 queries and downloaded the linked web pages. We were able to download 7,214 web pages. We then attempted matching the Bing summary to portions of each web page for the purpose of finding summaries drawn from HTML formatted regions of web pages. We were able to match Bing results with portions of web pages or ODP summaries for 6,244 of these results. Of these results only 440 summaries currently contained one of the target HTML formatting tags. From these 440 we removed those where the formatting would not be noticeable. We removed results with all white or clear images. We removed summaries which crossed table boundaries where the table was only used for paragraph placement. We removed very small images and tables with less than 4 cells. We then randomly chose 30 summaries from each of four groups: forms, tables, lists and images. There were only 10 usable form summaries. Some of the summaries used HTML from multiple groups. The final pool of potential summaries included 94 summaries.

Each user within our evaluation was shown 30 summaries randomly selected from our pool of 94. 15 of these summaries were only made up of text and came directly from Bing. 15 of the summaries were shown to the users with structured text and/or multimedia present from the original web page. Among these 30 summaries no summary was repeated to the same user with and without structure and multimedia. The 30 search results were randomly ordered on a single page mixing the ordering of Bing summaries with summaries preserving document structure.

The time at which a choice was recorded or altered for each summary was used to compute timings for each summary. The first search result timing was recorded from the time the page opened to the time when the first selection was made. The second timing was obtained from the time of the first selection to when a choice was made on behalf of the second search result. If a

\(^{2}\)http://www.searchhippo.com  
\(^{3}\)http://www.google.com/trends/hottrends  
\(^{4}\)http://www.bing.com/toolbox/bingdeveloper/
user altered a choice any time during the evaluation, then the time from the last click to the current choice was added upon the previous timing. Out of order decisions were permitted as timings were taken from the last click to the current choice.

Lastly, to aid the search engine users with their timing, the search result page was automatically scrolled for the user. When the last entry on a page was recorded a 1 second, animated scroll occurred. The animation made it obvious to the user what was occurring. The automatic scrolling made it harder to accidentally skip a search result, scrolling to the right position each time. Automatic scrolling in most cases will be faster than user scrolling.

5.3.1 Participants in the Evaluation

We used two pools of people expecting potentially different responses from each group. Our first group was made up of those who effectively think in the abstract. These practiced “abstract thinkers” making up the first pool are 22 students from the Computer Engineering and Computer Science department at the University of Louisville. CECS students can be effective at viewing source code and placing that source code in mental models of a system. This form of abstract thinking we do not expect to be common in the general population. We therefore use a second pool of people who represent the more general population or the non-expert abstract thinkers. The second group obtained by sending out approximately 70 invites mainly through Facebook to acquaintances and friends of friends. None of those sent messages do any form of programming on a regular basis. Our invites resulted in 34 participants for our user evaluation.

The basic demographics of the two groups are as follows: A majority of the CECS students were male while the majority of Facebook acquaintances were female. A majority of participants in both groups are in their 20’s. Many of the Facebook acquaintances are homemakers and none of them work in the computer industry. Most of the Facebook acquaintances have Bachelor degrees while nearly all of the CECS students were graduate students either working on a Masters or PhD.

The user evaluation was posted online. Users completed the evaluation using their own computers outside of a lab setting to better recreate more natural search engine usage among our subjects.

5.4 Summaries Preserving Document Structure Experimental Results

There are three dimensions upon which we evaluated the effectiveness of structured text and multimedia within search summaries. We analyzed how quickly users decide upon the relevance of a result. We tested the accuracy with which the relevance decisions are made. Lastly, we reviewed which types of summaries users prefer. The values in each of these dimensions have provided evidence for the usage of structured text and multimedia.
It took users between 3 minute and 3 hours to complete the evaluation. Certainly some of the longer times were due to interruptions. After the 3 minute time the next closest time was 6 minute and there were about 4 participants with times just over 6 minute. It appeared that the 3 minute evaluation time was an outlier. Additionally, this user’s comments were in Spanish, which is surprising since every other comment was in English. We removed the user taking only 3 minute to complete the evaluation.

5.4.1 Decision Time Analysis

Generally users of search engines do not visit search engines to review search results. Instead users of search engines are looking for answers or resources for questions that they have. The more quickly a user can obtain their desire the more successful the search engine has been and the more satisfied search engine users will be. It is therefore important that users quickly understand the search results as they make decisions as to which to click upon. The more quickly the decision can be made without losing accuracy, the better quality the results are.

From the user evaluation conducted, we have obtained 1650 click decision. By click decision we mean choosing whether a user should click on a search result, skip it or mark "unsure" meaning there is not enough information to decide. The timing of each decision has been recorded.

From the results we see a broad range of timings for each click decision from 1.3 seconds to 2.8 hours and a median time of 13.3 seconds. The 2.8 hour click decision time is without a doubt an outlier and was removed from our time analysis data. It came from one of the CECS students, who likely realized mid-evaluation that he/she was late for their next class. We also removed from our time analysis the other 15 timings with values higher than 78 seconds, which were rare (16/1650), seem to be extreme values for the decisions made and are more than 3 standard deviations from the mean. The distribution of click decision timings is shown in Figure 52.

We also see large variations in the timings of each individual. Consider Figure 53 where we show box plots of the users with the five smallest medians and the five largest median timings. Here we see that there are individuals where the majority of their timings fall below 20 seconds. While there are others that carefully consider their options and have a majority of their timings over 20 seconds. This shows that it is important to consider the variation in time due to individual speed. This is countered by having all users recorded making decisions for both types of summaries.

When we separate out the two groups of experienced (computer science group) and inexperienced (Facebook group) abstract thinkers, we see very different timings. In both groups we compared the timings when using summaries preserving document structure to text only summaries. We did this by comparing averages. To avoid the long right tail heavily impacting averages, we first took the logarithm.

First we compare decision times in the computer science group. This is shown in Figure
Figure 52. Histogram showing distribution of click decision timings.

Figure 53. Box plots of the participants with the five lowest median timings and the five highest median timings.

Figure 54. The average decision time of users using plain-text summaries are drawn as boxes. A line is drawn from each box to the average decision time when using summaries preserving document structure. (a) experienced abstract thinkers and (b) inexperienced abstract thinkers.
Figure 55. Each column shows all decision times per user, per summary type. The red diamond in each group shows the average decision time.

54(a). The square in this chart represents the average decision time when users are using plain text summaries. A line is drawn from each square to the average decision time for summaries preserving document structure. If the line goes up, then plain text summaries were faster for that user. If the line goes down then summaries preserving document structure led to faster decision times. In the computer science group 9 users were faster using summaries preserving document structure, while 13 were faster using plain text summaries. From Figure 54(a) we can see that for most users there was not a big difference between the two. However, there is a noticeable difference for the slowest two and the fastest 7. The fastest group or the 7 with the lowest average decision time, every one of them was faster with plain text summaries. For the power users who make blink decisions using abstract thinking, they were much faster using plain text summaries. Except for a couple extreme cases little other observed differences were present. A paired t-test was performed to check for a statistical difference, and with a p-value of 0.11, none was found.

We see a very different effect among the Facebook pool, see Figure 54(b). A large majority of the Facebook pool were faster using summaries preserving document structure or 23 of 33. Many of the differences were very small, but there are more noticeable lines pointing down in Figure 54(b). First we compared the difference in values using a paired t-test and there was not a significant difference at the alpha—0.05 range with a p-value of 0.096. You can visualize this comparison of averages in Figure 55. Here we see the averages overall are very similar between the two structures, despite having a large range of decision times. However, if you look at the number of people who were faster, there is a significant difference using a Wilcoxon signed rank test pairing
the averages of users. The result was a p-value of 0.041. Visually, it appears that slower users were most positively impacted using summaries preserving document structure among the inexperienced abstract thinkers of the Facebook pool.

Perhaps those that are experienced thinking abstractly have a more difficult time viewing summary types they are unfamiliar with. They have gained experience quickly assimilating text only summaries and drawing conclusions on the destination page from their abstract models, that to actually see the structure slows them down. While the inexperienced in thinking abstractly take the summaries at face-value. Seeing the structural context does help them understand the coming page. This explanation jives perfectly with what one of experienced users state: “tables require me to decipher another layout. It’s slower than just reading the information without extra images, lists, etc.”

5.4.2 Selection Accuracy Analysis

Users take time analyzing which search result to click on. We have shown the effects of summaries preserving document structure on analysis time. Users are also affected by the accuracy of their decision. When a user clicks a link that leads to an irrelevant web page, that user has wasted time and this wasted time may negatively reflect on a search engine’s results.

When judging the impact of summaries preserving document structure on decision making we needed to judge the accuracy of the click selections. We had a judge go through and create first the context of each query such as for the query katherine jenkins the query question was “Who is Katherine Jenkins (singer)?” Decisions were made based mainly on how well the web page answered the question posed in the search question. In the pool of used summaries, nearly 50% are relevant and 50% are not relevant, meaning that guessing all relevant would result in a low accuracy.

Accuracy was little affected by the introduction of structure and multimedia in the summary. Accuracies by summary type and by group are shown in Table 21. In this table we see that the more common search engine user, those inexperienced in abstract thinking were more accurate using summaries preserving document structure, while the abstract thinkers were less accurate. The inexperienced abstract thinkers were significantly more accurate than was the experienced group using Fisher’s exact test with a p-value of 0.038. Perhaps the experienced abstract thinkers were overly confident in their mental models, and took time to ignore the introduced structure, leading to lower accuracies.

Table 22 displays both precision and recall of the click decision made. Precision is most important as it is more important to have users clicking more frequently on relevant web pages, than it is for a user to select most of the relevant web pages. We see from Table 22 that the inexperienced abstract thinkers had higher precision scores, while the experienced abstract thinkers
obtained higher recall scores. The inexperienced pool had a statistically significantly higher precision score than the experienced pool using the summaries preserving document structure. It is interesting to notice that the experienced pool had a higher precision using text only summaries (68.8% vs. 63.0%) while the inexperienced pool had a higher precision using summaries preserving document structure (73.2% vs. 70.4%). The structure helps those that do not think abstractly. While the more experienced abstract thinkers could not let go of their mental model and the structure distracted them from making more right choices.

### Table 21

<table>
<thead>
<tr>
<th>Pool</th>
<th>Type</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experienced</td>
<td>Structured</td>
<td>65.3% (188/288)</td>
</tr>
<tr>
<td></td>
<td>Plain Text</td>
<td>70.0% (205/293)</td>
</tr>
<tr>
<td>Inexperienced</td>
<td>Structured</td>
<td>72.8% (313/430)</td>
</tr>
<tr>
<td></td>
<td>Plain Text</td>
<td>71.7% (307/428)</td>
</tr>
</tbody>
</table>

Comparison of accuracies between summary types and evaluation groups.

### Table 22

<table>
<thead>
<tr>
<th>Pool</th>
<th>Type</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experienced</td>
<td>Structured</td>
<td>63.0%</td>
<td>69.9%</td>
</tr>
<tr>
<td></td>
<td>Text Only</td>
<td>68.8%</td>
<td>71.3%</td>
</tr>
<tr>
<td>Inexperienced</td>
<td>Structured</td>
<td>73.2%</td>
<td>64.8%</td>
</tr>
<tr>
<td></td>
<td>Text Only</td>
<td>70.4%</td>
<td>62.2%</td>
</tr>
</tbody>
</table>

Comparison of precision and recall scores between summary types and evaluation groups. Bolded values are significantly better using Fisher’s test comparing to the opposing pool.

### 5.4.3 Participant Preferences by Summary Type

In addition to the speed an accuracy with which users select relevant search results, we would also like to know which types of summaries users prefer. Overall 67% (37/55) prefer the summaries preserving document structure to text only summaries. This means about 2 in every 3 people that participated in the user evaluation preferred the summaries preserving document structure. The per group preferences are shown in Table 23. Table 23 shows that 17 out of 22 (77.3%) experienced abstract thinkers preferred the summaries preserving document structure. At the same time 20 of 33 (60.6%) of the inexperienced abstract thinkers preferred the summaries preserving document structure. This is a much higher, but not significant, percentage of experienced abstract users preferring summaries preserving document structure. Perhaps, experienced abstract thinkers see the value in adding structure, though their habits do not yet reflect it. Part of the inexperienced pool does not like change and would prefer the summaries to
remain familiar, though they comprehend them more quickly.

<table>
<thead>
<tr>
<th>Pool</th>
<th>Summary Type</th>
<th>Example Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experienced</td>
<td>Structured</td>
<td>[Structure] makes the summary info easier to read. It doesn’t require scanning a</td>
</tr>
<tr>
<td></td>
<td>77.3%</td>
<td>single block of text to find relevant info.</td>
</tr>
<tr>
<td></td>
<td>Plain Text</td>
<td>It seems to be much easier to decide on the websites usefulness.</td>
</tr>
<tr>
<td>Inexperienced</td>
<td>Structured</td>
<td>Glancing at the two it is a lot quicker to mentally access the information and</td>
</tr>
<tr>
<td></td>
<td>60.6%</td>
<td>decide if it is valuable for your search.</td>
</tr>
<tr>
<td></td>
<td>Plain Text</td>
<td>It’s just easier and faster to read. The other seems cluttered.</td>
</tr>
</tbody>
</table>

**TABLE 23**

Preferences of each user evaluation group and example comments for each group. Bolded values are significantly better than 50% using a binomial test and alpha of 0.05.

Comments on a user’s preferences were optional. However, we did receive 38 comments providing a lot of feedback about users’ thoughts and reactions. Tables and images received the most attention from users. Some users really liked table formatted data such as in the comment, “Tabulated data is ALWAYS easier to read in **table** format…” While another participant wrote, “I feel like I get more information out of the text than the **tables**.” One reason for liking or disliking tables is the speed at which tables are comprehended. One participant wrote, “Because the information is organized in a **table** or has illustrations, you can determine more quickly how relevant the info is.” However, one participant felt that tables added additional mental work as was mentioned before, “**tables** require me to decipher another layout. It’s slower than just reading the information without extra images, lists, etc.” Lastly we received this mixed comment, “Having the search results structured (rather than a big odd-sounding sentence) was useful in gleaning information quickly. Depends on the context though - sometimes the structure takes up some space and tells me a lot. Other times it takes up a lot of space and tells me little.” The use of tables depends a lot on the content of that table and how the user perceives tables.

Images received a similar reception. Some users really liked the images, “Also, the embedded **images** were useful in determining the content of the web page.” Another user felt that images were a big distraction drawing attention away from current summaries, “… however I noticed that the instant I saw an **image** my eye jumped immediately to the URL with the **image**, bypassing the one I was viewing. I needed words to be sure it was what I was looking for.”

Overall there were several individuals that preferred some or all of the changes made through the addition of structure and multimedia to search results:

- “More intuitive and easier to read and understand.”
- “I preferred the first one (summaries preserving document structure) because I thought
having the spreadsheet format and the links implemented in the body of the search was very
  catchy and easier for me to see what I was actually looking for."

• "Using system 1 (summaries preserving document structure) puts the data in an easier
  context to deal with. The information is more readily processed."

Overall users preferred summaries when possible to include tables, images, lists, buttons
  and hyperlinks. There will always be those that are familiar with the current approach and would
  prefer no change take place, but these users queried in our evaluation overall would benefit from
  the addition of structured text and multimedia to summaries used in search engines.
CHAPTER 6

VARIABLE LENGTH, MULTI-TWEET SUMMARIZATION

Now we consider one application of ReClose that is outside of search, summarizing multiple flu related tweets. This gives us an opportunity to expand the use of ReClose summarization to new domains and test the idea with multiple documents that are each extremely short. Additionally, we create tweets of varied length.

6.1 Motivation for Multi-Tweet Summaries

Thus far the central focus has been summarizing web pages within search. We believe that ReClose is more general than that. For this purpose we propose extending ReClose for the purpose of summarizing multiple tweets. We believe that this application will demonstrate the usefulness of ReClose to extend beyond web pages and HTML, to social media.

Twitter\textsuperscript{1} tweets (short status messages of about 140 characters) are very different from web pages. Web pages are generally very long containing at least several sentences. Tweets are made up of a maximum of 140 characters and often contain only a single sentence or multiple very short sentences. Summarization must now consider multiple authors with different vocabularies, poor spelling and abbreviations, and very sparse text data. We tackle this through transforming our data using topic modeling and clustering tweets before summarization. The end result is a variable length summary used for summarizing pools of tweets as they relate to flu outbreaks.

6.1.1 Motivation for Flu Outbreak Detection through Multi-Tweet Summaries

Monitoring text data available online for disease outbreaks has been a valued commodity for quite some time. For example The Global Public Health Intelligence Network (GPHIN) was established in 1997 by the Public Health Agency of Canada's Centre for Emergency. GPHIN is the central component of the World Health Organization's (WHO) Alert and Response Operations. GPHIN follows a number of news organizations in 9 languages using an automated system that then outputs articles related to a number of health concerns. Articles are translated and reviewed by analysts and output to GPHIN's user community, which includes WHO [52].

\textsuperscript{1}https://twitter.com/
Many of the sources of data used by organizations similar to GPHIN also use news sources. For a local news source to report a high level of disease activity it must be brought to the attention of the news organization. This is a much slower process than if one could directly monitor individuals becoming sick. This would allow us to observe in real-time the increase in sickness within a given area which would be much quicker than local news sources. Event detection in near real-time of local phenomenon is possible through the streaming text data available on Twitter [89].

Several systems [2, 21, 100] have been proposed for the purpose of detecting flu outbreaks. Generally these systems automatically select keywords to follow on Twitter, then count the number of tweets at each time step per keyword. Generally baselines are created for the detection of outlying count data. Additionally, the system may include a regression model for the purpose of predicting future values based on the streaming counts. Alarms may be triggered automatically to alert health officials that an outbreak may be occurring. When the right set of keywords are chosen these types of system can be highly accurate, reaching as high as a 98% correlation with actual reported flu cases. However, Twitter tweets are highly noisy [11].

For example consider Wiz Khalifa, a famous rapper, who mentioned a symptom of the flu, a headache, in a tweet. His tweet was retweeted 4500 times. These 4500 retweets represented a single case of the keyword “headache” causing an anomalous increase of counted tweets, and signaling a possible outbreak! Most of the proposed outbreak monitoring methodologies do not concentrate on the problem of false alarms. Frequent false alarms are certain to decrease the confidence the medical community would place in an automated system for early detection of disease outbreaks.

We propose a semi-automatic outbreak detection system. Health officials will be much more confident in the output of a system that incorporates both precise algorithms and the understanding of a human expert. When our system detects a rise in activity which could predict an outbreak, our system provides a summarized view of the tweets for that particular area. This allows for a health expert to verify that the increased volume of tweets are related to sickness. Human verification will provide a higher level of trust in the system. It will be instrumental in the potential uptake of disease outbreak monitoring within organizations such as GPHIN. This work provides an approach to multi-tweet summarization with an emphasis on detecting flu outbreaks.

6.2 Twitter

Twitter is an online, social networking website. Users of the site have the ability to share short (140 character) messages called “tweets.” These message will be observed by other users that follow the given user. To follow one must find the user name of another Twitter user and click a button labeled “Follow.” When a Twitter user views their home page they are presented with a large number of tweets written or forwarded by the Twitter users that this user follows. Twitter is most commonly used by friends communicating with friends, fans following the tweets of popular
individuals and companies and for near-real time news. Twitter has become a very large service. As of March 2012, Twitter had more than 140 million active users and sees more than 340 million tweets per day [116]. This means that there are a large amount of status updates that relate to current events and about every other topic being actively pursued online.

Messages and following relationships are available publicly. When you search another user, all of the users that follow and are followed by that user are available. Tweets may be searched from the Twitter website. Additionally, one may obtain large numbers of tweets if one uses one of the provided APIs. Twitter provides a rate limited search API\(^2\) which allows one to programmatically run a number of searches on recent tweets up to 9 days old. Twitter also provides a streaming API\(^3\) in which one may obtain tweets as they happen. Free streaming API usage is limited to a 1% sample of tweets be default. One may also provide limiting factors that will allow one to get up to 100% of a specific subset of tweets. Such limiting factors could include a geographic bounding box, search keywords and phrases or user names. After the limiting factors are provided, one may receive up to 1% of the tweets during that second.

To better understand tweets, there are a couple of standards that one must be aware of. The first is that when other users are mentioned in a tweet, the user name is prefixed by the character '@', such as if my user name were "student" then in a tweet that refers to me one might see a mention of "@student." There is also special use of the character '#'. This is used to create what people call "hash tags." A hash tag uses the '#' followed by one or more words concatenated together. Hash tags create an easy way for messages on the same topic to be used. It means that a search for the hash tag will result in all tweets being found with that exact character sequence making up the hash tag. Additionally, when hash tags become popular, others are alerted to this fact and may join in the discussion, reusing the same hash tag in their message. These "trending" hash tags are listed on a users home page in Twitter along with some of the most common words at the moment on Twitter.

6.3 ReClose Extensions for Multi-Tweet Summarization

Within this section we outline our extensions to ReClose to summarize multiple tweets for the purpose of detecting flu outbreaks. Our end goal is to summarize topics within a large number of tweets. To do this we first gather tweets as our data source. We estimate the location of the tweet so that it may be assigned to a local. We cluster the tweets of a given local to obtain topic groupings within the tweets. We select one or more tweets to represent a cluster of tweets. We create a presentation of these summarized clusters to the user for further analysis. This means that when an automatic system signals an alarm for a specific region, our summaries provide a

\(^2\)https://dev.twitter.com/docs/using-search
\(^3\)https://dev.twitter.com/docs/streaming-apis
means to understand trends in the data as verification.

6.3.1 Tweet Gathering

We gathered tweets using Twitter's Streaming API. Twitter provides an online API which once connected will continually supply tweets. The number of tweets available through free accounts is limited. Several options exist for choosing a subset of tweets to download. We limit the tweets obtained through keyword search. We download nearly all tweets which contain any one of these keyword phrases: flu, sore throat, cough, runny nose and headache. With each tweet we save the ID of the tweet, the time obtained, the text of the tweet, the user supplying the tweet to Twitter, and the location the user provided in their account. The location provided by the Twitter user is in a free form text field. This means that users may enter any text for their location.

An outbreak detection system will be of most use if approximate locations can be obtained for a large percentage of tweets. We submit every unique user location to Yahoo!'s PlaceFinder4. From this online resource we obtain Yahoo!'s best guess for the city, state and country of a given Twitter user. Of the 19 million tweets we have obtained, Yahoo! is able to identify a city level location for 54% of the tweets or about 10.4 million tweets. Many of the location fields do not provide enough information to identify any location. For example 25% of the tweets have an empty location field.

6.3.2 Preprocessing Tweets using Topic Modeling

One of the major challenges of working with tweets is feature sparsity. If you represent tweets with a word vector, the matrix of tweets by words will be more than 96% empty if you have 350 words in a group of tweets and average 11.6 words per tweet. We therefore consider using topic modeling to reduce the number of features to represent a tweet. Then a distance metric will provide more variety in the distances for use in clustering than a majority of distances being maximum distance. The output of topic modeling in our case is a topic probability vector over a fixed size set of topics per tweet. The first major extension to ReClose is to preprocess tweets with topic modeling.

Topic modeling was the final approach chosen for preprocessing tweets for clustering. We also considered Latent Semantic Analysis (LSA) and Porter Stemming with word vectors. LSA certainly would have a similar result to topic modeling, however the number of features produces was far larger than topic modeling and produced inferior clusters. Stemming and word vectors had many more features than LSA and resulted in tweets being clustered with only a single word in common, though the usage of the term between the two tweets may be at odds.

4http://developer.yahoo.com/geo/placefinder/
The specific form of topic modeling used is Latent Dirichlet Allocation (LDA) [13]. LDA is a model that assumes that tweets are formed in a generative process. LDA represents tweets as being made of a mixture of topics. For each word in a tweet, first a topic is randomly obtained over the distribution of topics for a tweet. Then each topic provides a probability distribution over words. A word is randomly obtained from the topic distribution. This process is pictured in Figure 56 and more formally described below.

Each tweet $w$ in our corpus $D$:

1. Choose $N \sim \text{Poisson}(\xi)$ (number of words in a tweet).
2. Choose $\theta \sim \text{Dir}(\alpha)$ (distribution of topics in tweets).
3. For each of the $N$ words $w_n$:
   
   (a) Choose a topic $z_n \sim \text{Multinomial}(\theta)$.
   (b) Choose a word $w_n$ from $p(w_n|z_n, \beta)$, a multinomial probability conditioned on the topic $z_n$.

The topic model is obtained through either the EM algorithm [13] or Gibbs sampling [98]. In either case an iterative process searches for models with improved likelihoods of representing the text corpora. LDA needs a large text corpora to find models which consider co-occurrences for most of the words within a given tweet group. We generally use around 10,000 tweets for training an LDA model.

Before we ran LDA over our training data all letters were made lower case, English stop words were removed and all user names (words starting with the '@' character) were removed. Once we have trained a topic model, we use the topic model to preprocess the tweets to be clustered. Each tweet is transformed from a bag of words to a vector of topic probabilities. This can greatly reduce the dimensionality of the data set. It also reduces the distances of topically similar tweets while preserving high distances between topically different tweets.

As an example of preprocessing tweets with LDA consider the three tweets shown in Table 24. Tweets 1 and 2 contain no words in common and yet both are very similar topically. If we only consider word vector based similarities then tweets 1 and 2 would have a similarity score of 0.
Three example tweets on the left observed for Seattle, WA on Nov. 3, 2011. These three tweets are preprocessed using LDA into topic probabilities. (Rows will not exactly add to 1 due to rounding.)

<table>
<thead>
<tr>
<th>ID</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Get Zicam RT @[removed]: Woke up with a sore throat and a runny nose :)</td>
</tr>
<tr>
<td>2</td>
<td>I need a rest and theru flu #FML</td>
</tr>
<tr>
<td>3</td>
<td>Immune Boosting Flu Shot Alternatives <a href="http://t.co/bljISLUp">http://t.co/bljISLUp</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>Topic 1</th>
<th>Topic 2</th>
<th>Topic 3</th>
<th>Topic 4</th>
<th>Topic 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.01</td>
<td>0.02</td>
<td>0.00</td>
<td>0.96</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td>0.02</td>
<td>0.04</td>
<td>0.01</td>
<td>0.94</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>0.01</td>
<td>0.02</td>
<td>0.95</td>
<td>0.01</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**TABLE 24**

Top 5 words per topic as discovered by LDA on US tweets related to flu for Nov. 3, 2011.

<table>
<thead>
<tr>
<th>Topic 1</th>
<th>Topic 2</th>
<th>Topic 3</th>
<th>Topic 4</th>
<th>Topic 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>cough</td>
<td>headache</td>
<td>flu</td>
<td>flu</td>
<td>flu</td>
</tr>
<tr>
<td>lol</td>
<td>i'm</td>
<td>http</td>
<td>shot</td>
<td>stomach</td>
</tr>
<tr>
<td>drops</td>
<td>worst</td>
<td>sore</td>
<td>season</td>
<td>mccoyny</td>
</tr>
<tr>
<td>i'm</td>
<td>bad</td>
<td>sick</td>
<td>shots</td>
<td>http</td>
</tr>
<tr>
<td>syrup</td>
<td>day</td>
<td>i'm</td>
<td>practice</td>
<td>practice</td>
</tr>
</tbody>
</table>

**TABLE 25**

Tweets 2 and 3 on the other hand share a single word in common “flu,” yet these two tweets have less in common than tweets 1 and 2. When we train LDA on one day’s worth of tweets for the entire USA (45,716 tweets) with only 5 topics then the resulting set of 5 features are also shown in Table 24.

After the transform we see that both tweets 1 and 2 share a lot of words from Topic 4, while tweet 3 contains mainly terms from Topic 3. After the transform tweets 1 and 2 are close in terms of distance and tweet 3 is distant from tweets 1 and 2. The 5 most probable terms within each topic are shown in Table 25. Of interest is the fact that the term “flu” is highly probable and common in Topics 3, 4 and 5, but the usage of the term differs greatly. When flu is used with the term “shot” then it is more likely to be from Topic 3, whereas when flu is used with terms related to being sick, it is more likely to come from Topic 4. Thus we are able to separate different usages of the same term effectively and bring more similar tweets together despite a lack of similar words between the two tweets.
6.3.3 Tweet Clustering

ReClose was initially used for summarizing single web pages. Summarizing multiple tweets is more similar to multi-document summarization. We therefore, first cluster the tweets before summarizing the multiple tweets. Tweets are clustered as a means to find common themes and trends within a set of clusters. Clusters provide topical groupings of tweets that we can later use to create more refined summaries. Rather than providing a single summary of all tweets in a day, we present the user with several groupings that are summarized. In this way major trends are more evident by observing the size of the cluster being summarized, and how similar the observed tweets are.

We did not want to determine ahead of time the number of clusters that would be shown for each set of tweets. We would rather that the data determine the number of clusters. We therefore used agglomerative, hierarchical clustering.

Agglomerative, hierarchical clustering begins by putting each tweet into its own cluster. Then one by one clusters are combined based on which two clusters increase some objective function the least. This process continues until we have combined all clusters into a single cluster, while preserving the hierarchy and the objective function values.

After we have obtained a hierarchy of tweets, we then follow the change in the objective function used for combining clusters. We start at the top of the hierarchy where only a single cluster exists. Then we observe each combination from this point and the change in the objective function previous to that addition. The point in the hierarchy at which the highest percentage change in the objective function takes place is the point at which clusters are obtained for further use. To avoid situations where a large number of tweets are not contained in a cluster, we limit the number of clusters to a maximum of 10.

One approach to hierarchical clustering is to use Ward’s minimum variance method [122] which originally used the error sum of squares (ESS) to determine the loss of information that results from representing clusters by their centroids. Then each cluster combination calculates the increase on ESS by joining two clusters. This is done as follows for the generic clusters $c_1$ and $c_2$:

$$MinVar(c_1, c_2) = ESS(c_1 \cup c_2) - ESS(c_1) - ESS(c_2)$$

(41)

In our hierarchical clustering we use an adapted version of Ward’s minimum variance that removes the squares on each term as is done in Weka [49]. When calculating either the ESS or error sum (ES) (without squared terms) one must first obtain the centroid $a$ of the cluster as follows, where the cluster contains $k$ vectors of the form $x_i$:

$$a = \frac{x_1 + x_2 + \cdots + x_k}{k}$$

(42)
ES is calculated as follows when using Euclidean distance for a cluster of $k$ tweets:

$$ES_{\text{Euclidean}} = \sum_{i=1}^{k} ||a - x_i||$$

(43)

ES can also be calculated for the cosine distance which is shown below:

$$ES_{\text{cosine}} = \sum_{i=1}^{k} \left( 1 - \frac{a \cdot x_i}{||a|| ||x_i||} \right)$$

(44)

We can substitute either $ES_{\text{Euclidean}}$ or $ES_{\text{cosine}}$ for $ESS$ in equation 41. At each step in our hierarchical clustering we join the clusters with the smallest increase in ES. Generally the clusters with the closest centroid points are selected for joining as this causes the least increase in the $\text{MinVar}$ function.

We also use the $\text{MinVar}$ function as a basis for our heuristic which selects the number of clusters to use. At some point in the hierarchy we will necessarily begin joining clusters which are not very similar. We would like our algorithm to find this point in time where the largest increase in variance is introduced. We do this by comparing consecutive values for $\text{MinVar}$, when the greatest percentage decrease occurs when tracing these values from the last join to the first then we stop. To avoid the number of clusters to be similar to the number of points being clustered, we have a maximum number of clusters that we will consider. We do not consider the $\text{MinVar}$ values beyond the maximum number of clusters. Additionally, when we find the greatest percentage decrease in $\text{MinVar}$ if any clusters are found which are too small according to some preset threshold, the small clusters are added into the closest cluster according to the same metrics used for joining clusters in our hierarchical clustering.

The result of the preprocessing and clustering tweets, is that now we have our tweets separated into like topics. Representative tweets used for summarizing a number of tweets will be much more meaningful having grouped similar tweets.

### 6.3.4 Cluster Summarization

Once clusters have been found, we can now apply ReClose query-independent summarization to each cluster separately to rank the tweets. This provides the basis from which each cluster is summarized. For our summarization problem we create a graph where each node represents a tweet in our cluster. We create undirected edges from each node to each other node. The edge is weighted with the distance between each tweet.

For the purpose of summarizing tweets we calculate closeness centrality as follows:

$$\text{closeness}(t) = \frac{|G| - 1}{\sum_{n \in G \setminus t} \text{dist}(t, n)}$$

(45)

where $t$ and $n$ are both tweets in our graph $G$ and $|G|$ counts the number of tweets in the graph. In this way average distance is calculated per node in the graph to all other nodes. We then rank
the nodes by the inverse of average distance so that those nodes with the lowest average distance are ranked first.

The observant reader will notice that we do not acquire the shortest path between nodes for closeness centrality, but instead simply take their direct distance. We believe that indirect paths within the graph of tweets are invalid assessments of the distances between tweets, and instead only focus on direct distances.

Word overlap distance is calculated based on representing each tweet as a set of words. Word overlap is calculated as follows:

$$WordOverlap = 1 - \frac{t_1 \cap t_2}{t_1 \cup t_2}$$

where $t_1$ and $t_2$ represent two tweets as the set of words contained in those tweets. Using word overlap common words within the cluster are certain to be present in the summarizing tweet(s). Additionally, there is more differentiation in the closeness centrality measure than when using only topic vectors.

When summarizing a cluster of tweets the first selection is the highest ranked tweet according to closeness centrality. After that the next highest tweet is chosen that is not too similar to a previously selected tweet. In our case we avoid selecting tweets which have identical bags of words.

We vary the number of summarizing tweets per cluster depending on the size of the cluster. Smaller clusters receive fewer summarizing tweets, as there are less tweets to represent. Large tweets may have as many as three summarizing tweets. Through experimentation we determined the following formula to work best:

$$n = \begin{cases} 3 & \text{if } \log_3(|c|) \geq 4 \\ \lfloor\log_3(|c|)\rfloor & \text{if } \log_3(|c|) < 4 \end{cases}$$

The $|c|$ represents the size of the cluster in tweets. This formula assumes that each cluster has at least 3 tweets. With the above formula clusters of size 3-8 will be represented by 1 tweet, 9-26 by 2 tweets and all larger sizes by 3 tweets. Having multiple tweets provides confirmation of the message given in the first tweet.

### 6.3.5 Presentation of Summaries

In addition to summarizing tweets of clusters there are two additional pieces of information provided for each cluster. We provide the number of tweets in the cluster and highlight common words in the cluster. An example of the format used can be seen in Figure 57. In this example there are 27 tweets within the cluster. Each additional summarizing tweet beyond the first is prefixed by ‘+’ sign. Common terms are bolded. The term “headache” was seen in 25 of the 27
tweets and “giving” was seen in 3 of the 27 tweets. Both are highlighted in this case being in at least 3 tweets.

<table>
<thead>
<tr>
<th>(27) Headache ??</th>
<th>+ When a headache pops up out of no where &lt;</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+ Im giving myself a headache... Its unlike me</td>
</tr>
</tbody>
</table>

Figure 57. Example tweet cluster summary taken from tweets in Louisville, KY for April 10, 2012.

Giving the users the size of the cluster and highlighting terms gives a user more of a sense of the cluster they are dealing with. It is important when judging several clusters to know the relative sizes of the clusters. Highlighted terms remind the user what terms are common within a cluster. This allows a user to both identify common terms in the cluster and to see those terms in context.

6.4 Multi-Tweet Summary Experimental Results

The output of ReClose for Tweets is a display showing summarized clusters of tweets. The effectiveness of this system is judged by how well users are able to distinguish between false alarms and true outbreaks. In our experiments we provide three judges with a number of summaries. The judges label each cluster as related to the flu or unrelated. We then compare the accuracy of these cluster labels against a gold standard in which another judge went tweet by tweet in the dataset and labeled individual tweets as relating to the flu or not.

6.4.1 Data Description

We have been gathering tweets with flu related keywords from Oct. 24, 2011 through the present May 15, 2012 for a total of 19.3 million tweets. We remove retweets from our data before clustering. Retweets are similar to forwarding in email. A tweet is copied and resent in your name with a reference to the previous poster.

For the purpose of experimentation we have chosen to limit our summaries to a single day within a single city. This limits the number of tweets that must be hand labeled. When performing LDA we often need much larger amounts of unlabeled data for training. As input we use tweets for the same day but limit the source as coming only from the US.

Before we process the text of a tweet we perform a number of preprocessing steps on the text. We remove all user names referenced in the tweets which are all prefixed by ‘@’ which will not be useful in analyzing the text of the tweet. We transform all URLs into the token “url,” since the exact URL path is not important to our text analysis but the presence of a URL provides some information. We do not consider tokens of less than 3 letters. Occasionally a word repeats a single character a large number of times such as the following “sorryyyy.” We replace repititions of a
single character more than 3 times with two occurrences of a character.

A number of challenges exist when performing text analysis on Twitter data. Some of these challenges are a large number of misspellings and abbreviations. The other challenge is the sparseness of the word vector representation of text. For example consider the tweets we obtained for April 10, 2012 for the city of Louisville, KY. There are 103 tweets. These tweets include 327 unique words after removing stop words. Of these 327 unique words, 268 (about 80%) of these words occur only once in all tweets for that day.

In Figure 58 we consider every comparison between all 103 tweets (5,253 comparisons). In 60% of comparisons there are no common words between the two tweets. Of the remaining 40%, 39% of comparisons have only one word in common. When many tweets have only 1 word in common with other tweets, this is not enough often to establish a common topic. The lack of common words between tweets was of most concern with clustering, with trying to find several tweets that were very similar.

6.4.2 Establishing the Purity of Tweet Clusters

Before showing human users the summaries, there were a few parameters that needed adjustment. These decision included the number of topics to use in LDA, the distance metric to use for clustering, the distance metric to use in closeness centrality and if any data transform needed to take place for closeness centrality calculations. We manually labeled two days worths of tweets from Seattle, WA for Nov. 3, 2011 and Nov. 30, 2011, since each day had around 100 tweets. These two days were used to test the various parameters of the proposed methodology. Each tweet was labeled as being about someone non-famous being sick with the flu or not, since the main goal is to understand the trends in tweets relating to flu outbreaks.

We attempted as many as 100 topics when preprocessing the tweets with LDA, the result
was the curse of dimensionality. With 100 topics, a single cluster was slowly grown adding one tweet to the same cluster time after time. There was no point in time at which a good set of clusters was produced. After further exploration we settled on testing the following possible number of topics: 5, 7, 10, and 12. We also had two main distance measures to compare: Euclidean and cosine.

We measured the effectiveness of clustering based on purity [79]. Purity normally sums the number of a given class from the cluster with the most of the same class. However, we allow for multiple clusters to have a majority of class “yes” (related to flu) and multiple clusters to have a majority of class “no.” More formally we calculate purity as follows:

\[
purity(C, \Psi) = \frac{1}{N} \sum_{i=1}^{k} \max_{j} |c_i \cap \psi_j|
\]

where \(N\) is the number of tweets, \(C = \{c_1, c_2, \ldots, c_k\}\) represents the set of \(k\) clusters and \(\Psi = \{\psi_1, \psi_2\} = \{\text{“yes”}, \text{“no”}\}\) represents the two possible classes. In the best scenario where each cluster only contains tweets of one class, the purity of the clustering would be 1. With only two classes the lowest possible purity score would be 0.5 where each cluster contains an equal number of tweets from each class.

<table>
<thead>
<tr>
<th>date</th>
<th>topics</th>
<th>cosine purity</th>
<th></th>
<th>Euclidean purity</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11/3</td>
<td>5</td>
<td>0.73</td>
<td>4</td>
<td>0.70</td>
<td>4</td>
</tr>
<tr>
<td>11/30</td>
<td>5</td>
<td>0.68</td>
<td>5</td>
<td>0.66</td>
<td>6</td>
</tr>
<tr>
<td>11/3</td>
<td>7</td>
<td>0.74</td>
<td>5</td>
<td>0.75</td>
<td>5</td>
</tr>
<tr>
<td>11/30</td>
<td>7</td>
<td>0.70</td>
<td>6</td>
<td>0.71</td>
<td>6</td>
</tr>
<tr>
<td>11/3</td>
<td>10</td>
<td>0.71</td>
<td>6</td>
<td>0.65</td>
<td>4</td>
</tr>
<tr>
<td>11/30</td>
<td>10</td>
<td>0.73</td>
<td>7</td>
<td>0.59</td>
<td>2</td>
</tr>
<tr>
<td>11/3</td>
<td>12</td>
<td>0.71</td>
<td>8</td>
<td>0.73</td>
<td>10</td>
</tr>
<tr>
<td>11/30</td>
<td>12</td>
<td>0.77</td>
<td>8</td>
<td>0.59</td>
<td>2</td>
</tr>
<tr>
<td>averages</td>
<td></td>
<td>0.72</td>
<td>6</td>
<td>0.67</td>
<td>4.9</td>
</tr>
</tbody>
</table>

TABLE 26

Measures of purity for topic sizes of 5, 7, 10 and 12 and distance measures Euclidean and cosine on Seattle tweets.

In Table 26 we show the results for the hand labeled Seattle data set. When comparing distance metrics in clustering there is one big difference between the two, cosine distance leads to more clusters on average than Euclidean distance. On average cosine distance produced 6 clusters compared to 5 clusters by Euclidean distance. The purity measures are very similar except for the few cases where Euclidean distance produced only 2 clusters. In both of those cases the purity suffered. On 11/30 for LDA topic sizes of 10 and 12 Euclidean distance produced 2 clusters and in both cases had a purity score of 0.59. It appears that using cosine distance will be more dependable from the two days worth of test data.
We also attempted to use clustering based on word vectors and preprocessing using latent semantic analysis. In both cases the purity scores ranged from 0.55 to 0.62. These other approaches were much worse than applying LDA with a small number of topics.

We are using unsupervised clustering to produce clusters with a purity scores that is mostly between 0.65 and 0.75 depending on the number of clusters and number of topics used for LDA. For comparison we also ran several classifiers in Weka [49] on the transformed data produced by LDA performing 10-fold cross-validation. For the date 11/3 the best classification accuracy achieved was 0.66 produced by both logistic regression and J48 (similar to C4.5). For the date 11/3 the best classification accuracy was 0.73 achieved by an SVM. Classifiers that used the labeled data did not achieve accuracy scores better than the purity scores achieved by hierarchical clustering of topic vectors.

When selecting the number of topics for LDA to produce it appears that more topics produces more clusters when using cosine distance. This pattern follows for every setting of topics from 5 up to 12. We would prefer to limit the number of clusters so that we are not always producing the maximum number of clusters. As a compromise between purity and the number of clusters, we chose to use 10 topics which also is the maximum number of clusters that we allow our clusterer to produce. It is also interesting to note that like our example tweets in Table 24 most tweets have nearly all of their words fall into a single topic, yet the actual clustering and heuristic used produce a different number of clusters than the number of topics used in LDA.

6.4.3 Closeness Centrality Distance Metric Selection

Next we need to decide how to measure distance between tweets within a cluster when summarizing. For this purpose we consider the following approaches: word overlap, cosine distance on topic vectors, and cosine distance on tf-idf transforms of word vectors. Closeness centrality produces a ranking on the tweets within a cluster. We will measure how closely the ranking is to ranking all tweets of the majority class first and tweets from the minority class last using Kendall’s tau (τ) coefficient considering ties. We will also use a greedy algorithm which will use the labels of the tweets to estimate how a human would perceive the clusters and label the clusters. We will then estimate the accuracy we should expect from human users on the tweets.

Kendall’s tau coefficient will be calculated per cluster and then averaged across all of the clusters. When we compute the Kendall’s tau we compare each tweet at each rank with each other tweet tweet below it in the ranking. Kendall’s tau is computed as follows:

$$\tau = \frac{n_c - n_d}{\sqrt{(N-n_1)(N-n_2)}}$$  \hspace{1cm} (49)

where $n_c$ represents the number of concordant pairs or when a tweet with the majority label has a higher closeness centrality than a tweet with the minority label. $n_d$ represents the number of
discordant pairs or a pair of tweets where the minority label in a cluster has a higher closeness centrality than a tweet with the majority label. \( N \) represents the total number of pairs or \( \frac{1}{2} n(n-1) \) where \( n \) is the number of tweets in a cluster, \( n_{t1} \) represents the number of pairs that have tying closeness centrality scores and \( n_{t2} \) is the number of pairs of tweets that have the same label. The values of \( \tau \) range from 1 where the two rankings agree perfectly to -1 where the two rankings are just opposite.

When we talk about accuracy we are talking about the accuracy of labels applied to clusters across all tweets. Accuracy helps us look at how accurately we understand the summarized tweets as a whole. This same measure will be used when users assess the quality of the summaries. First to pick the label of the cluster we use a simple heuristic as follows: When a cluster is represented by a single tweet, we use the label of that tweet to label the whole cluster. When a cluster is represented by two tweets, we use the label of the first or tweet with the higher closeness centrality ranking. When a cluster is represented by three tweets, we take the majority label of these three tweets. Finally accuracy is judged by the number of tweets where the cluster label and human label match. This is calculated as follows:

\[
\text{accuracy}(\mathcal{T}, \Lambda_h, \Lambda_c) = \frac{\sum_{i=1}^{n} I(\lambda_h(t_i) = \lambda_c(t_i))}{n}
\]

Here \( \mathcal{T} = \{t_1, t_2, \cdots, t_n\} \) is the set of all tweets for a given day. \( \Lambda_h = \{\lambda_h(t_1), \lambda_h(t_2), \cdots, \lambda_h(t_n)\} \) represents the labels given by a human one-by-one to each tweet. \( \Lambda_c = \{\lambda_c(t_1), \lambda_c(t_2), \cdots, \lambda_c(t_n)\} \) are the tweet labels obtained from the containing cluster. A cluster’s label applies to all tweets within that cluster. \( I(\lambda_h(t_i) = \lambda_c(t_i)) \) is an indicator function where 1 occurs when both labels are equal. Lastly, \( n \) is the total number of tweets on a given day.

<table>
<thead>
<tr>
<th>distance</th>
<th>date</th>
<th>( \tau )</th>
<th>acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>word</td>
<td>11/3</td>
<td>0.18</td>
<td>0.67</td>
</tr>
<tr>
<td>overlap</td>
<td>11/30</td>
<td>0.17</td>
<td>0.73</td>
</tr>
<tr>
<td>cosine -</td>
<td>11/3</td>
<td>0.23</td>
<td>0.67</td>
</tr>
<tr>
<td>tfidf</td>
<td>11/30</td>
<td>0.22</td>
<td>0.73</td>
</tr>
<tr>
<td>cosine -</td>
<td>11/3</td>
<td>-0.03</td>
<td>0.57</td>
</tr>
<tr>
<td>topic vector</td>
<td>11/30</td>
<td>-0.02</td>
<td>0.71</td>
</tr>
</tbody>
</table>

TABLE 27

The results of varying the distant metric for ranking tweets to summarize clusters.

Table 27 shows the results of applying different settings for closeness centrality. Word overlap computes distance based on the words present in the two tweets. “Cosine - tfidf” uses a word vector approach but applies term frequency - inverse document frequency to the word vector so that less common words that overlap have a higher weight in the distance metric. Lastly, “cosine - topic vector” refers to using cosine distance across topic vectors produced by LDA and used in clustering. It appears that Kendall’s \( \tau \) provides the most separation between each ranking, since it
uses the entire ranking rather than just the top 1-3 tweets in the ranking. Using cosine distance on the tfidf vector produces the highest Kendall’s $\tau$ coefficient for both days tested. The cosine distance over topic vectors appears to produce almost random rankings. The Kendall’s $\tau$ coefficient when using the topic vector has values just slightly below 0 meaning that the ranking is nearly random compared to the majority labels in the clusters. The accuracies for word overlap and cosine - tfidf favor cosine - tfidf. We would have used cosine - tfidf except for some additional testing we did.

We had three judges go tweet by tweet and label each tweet during the 2 week time period, April 10, 2012 - April 22, 2012 in Louisville, KY. Each judge was assigned a number of days to label. Judges were shown approximately 10 tweets from the testing data and provided with example labels. Judges were told not to make judgment calls between a cold or a flu, but to mark all sickness related tweets as having the flu. In the case that a tweet contained symptoms, unless another cause was evident, was to be marked as being flu related. Many of the tweets have obvious labels. Some of the tweets do not provide enough information to make an accurate decision. For example the tweet, “@[removed]; Cough Cough . .” has an ambiguous meaning. Without more context it is difficult to ascertain if these coughs reflect symptoms of the author or take on the slang meaning of the term “cough” often seen on Twitter.

<table>
<thead>
<tr>
<th>date</th>
<th>cosine - tfidf $\tau$</th>
<th>cosine - tfidf acc.</th>
<th>word overlap $\tau$</th>
<th>word overlap acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/10</td>
<td>0.11</td>
<td>0.72</td>
<td>0.20</td>
<td>0.72</td>
</tr>
<tr>
<td>4/11</td>
<td>0.20</td>
<td>0.65</td>
<td>0.07</td>
<td>0.66</td>
</tr>
<tr>
<td>4/13</td>
<td>0.11</td>
<td>0.60</td>
<td>0.07</td>
<td>0.67</td>
</tr>
<tr>
<td>4/14</td>
<td>-0.27</td>
<td>0.74</td>
<td>0.11</td>
<td>0.77</td>
</tr>
<tr>
<td>4/15</td>
<td>0.07</td>
<td>0.86</td>
<td>0.18</td>
<td>0.86</td>
</tr>
<tr>
<td>4/16</td>
<td>-0.05</td>
<td>0.68</td>
<td>-0.16</td>
<td>0.60</td>
</tr>
<tr>
<td>4/17</td>
<td>0.12</td>
<td>0.73</td>
<td>0.20</td>
<td>0.73</td>
</tr>
<tr>
<td>4/18</td>
<td>0.20</td>
<td>0.82</td>
<td>0.18</td>
<td>0.80</td>
</tr>
<tr>
<td>4/19</td>
<td>-0.10</td>
<td>0.51</td>
<td>0.13</td>
<td>0.68</td>
</tr>
<tr>
<td>4/20</td>
<td>0.23</td>
<td>0.63</td>
<td>0.25</td>
<td>0.63</td>
</tr>
<tr>
<td>4/21</td>
<td>0.36</td>
<td>0.73</td>
<td>0.28</td>
<td>0.73</td>
</tr>
<tr>
<td>4/22</td>
<td>0.42</td>
<td>0.73</td>
<td>0.38</td>
<td>0.73</td>
</tr>
<tr>
<td>avg.</td>
<td>0.12</td>
<td>0.70</td>
<td>0.16</td>
<td>0.72</td>
</tr>
</tbody>
</table>

TABLE 28
Comparison of cosine distance with a tfidf transform against word overlap distance on April data.

Once we had two weeks of labeled data we ran our test statistics one more time on the new data. It allowed for us to test for significance, before providing the users with summaries. More specifically, the tests we ran the summaries had only 13 clusters to work with, this is not enough data to accurately judge distance metrics for cluster. We ran again cosine - tfidf against word overlap as is shown in Table 28. Now that we have several more clusters to work with (66 in total),
we see different results emerge.

Table 28 shows that using word overlap with closeness centrality results in a higher average accuracy (0.72 compared to 0.70) and a higher Kendall’s τ coefficient. Using word overlap seems to be more reliable, there are a few extreme values when using cosine distance over tfidf values. Cosine distance achieved the lowest accuracy on any day of 0.51 on 4/19. Cosine distance had three negative Kendall’s τ coefficient on 4/14 (-0.27), 4/16 (-0.05) and 4/19 (-0.10) compared to one day, 4/16 (-0.16) when using word overlap.

After determining to use word overlap with closeness centrality to select summarizing tweets, we compare our clustering and tweet selection to random selection using statistical permutation tests. On the April data hierarchical clustering averaged a purity score of 0.74 using 5.5 clusters. To perform the permutation test we repeatedly observe random clusterings achieved through randomly selecting a number of clusters and partition points in the data. Each time a random clustering is performed we assess the purity score of the random cluster. In 1,000,000 samples not one of our random partitions achieved a purity score as high as 0.74 showing our hierarchical clustering to be significantly better than random with a p-value of approximately 0.0.

Using our previously mentioned heuristic for labeling clusters, word overlap plus closeness centrality achieved an accuracy of 0.71 on our April data set. We again performed permutation testing on our results. We took each cluster obtained from hierarchical clustering and randomly selected tweets as summarizing tweets. We then used our simple heuristic for selecting cluster labels as before for the randomly selected summaries. Repeating our permutation test 10,000 times only 4% of randomly selected summaries achieved an accuracy of 0.71 or better. The accuracy achieved using closeness centrality and our heuristic for cluster labeling achieved a significantly higher accuracy than random with a p-value of 0.04.

6.4.4 Evaluation of Variable Length Summaries

We here evaluate the use of a variable number of clusters, rather than sticking with a fixed number of clusters. A fixed number of clusters is much easier in practice. However, a fixed size will not accurately capture the nature of a set of tweets. One set of tweets may contain three large, distinct topics. While another set of tweets may contain eight topics with a lot of overlap. ~o two sets of tweets will be identical. Our approach to clustering has used Ward’s minimum variance method [122], which aims to create compact, spherical clusters in the topic space.

Thus far we have evaluated clusters based on purity. Purity will never decrease as the number of clusters increases. At worst, the two new clusters formed by dividing clusters will have the same proportions of our two classes. In this case purity will stay the same. We therefore introduce normalized mutual information (NMI) [79]. NMI uses entropy to adjust for the number of clusters produced. In this way one is not guaranteed to have the highest purity score with the
largest number of clusters. NMI is calculated as follows:

\[
NMI(\Omega; C) = \frac{I(\Omega; C)}{H(\Omega) + H(C)}/2
\]

(51)

where \( \Omega = \{ \omega_1, \omega_2, \ldots, \omega_K \} \) represents the set of clusters and \( C = \{ c_1, c_2, \ldots, c_J \} \) represents the set of classes, which in our case are only “flu” and “not”. Each \( \omega_k \) is the set of all tweets inside cluster \( k \) and each \( c_j \) is a set of all tweets contained in class \( j \). Information \( I(\Omega; C) \) is calculated in the following manner:

\[
I(\Omega; C) = \sum_k \sum_j \frac{|\omega_k \cap c_j|}{N} \log \frac{N|\omega_k \cap c_j|}{|\omega_k||c_j|}
\]

(52)

where \( N \) is the total number of tweets in all clusters combined. Entropy in the case of \( H(\Omega) \) is defined as follows:

\[
H(\Omega) = -\sum_k \frac{|\omega_k|}{N} \log \frac{|\omega_k|}{N}
\]

(53)

And \( H(C) \) is calculated as:

\[
H(C) = -\sum_j \frac{|c_j|}{N} \log \frac{|c_j|}{N}
\]

(54)

We took our Louisville dataset using April data and calculated NMI for each number of clusters from 1 to 10. We plotted the calculated scores per number of cluster for each day from April 10-22 and show at what point the maximum NMI score is reached. These plots can be seen in Figures 59(a) and 59(b). In Figure 59(a) are the six days of data in which the optimal number of clusters according to NMI was 6 or less. The highest NMI score is shown with a circle at that point. The clusters for April 16 and April 22 achieved by far their highest points with only 2 clusters. According to NMI, there is no additional informational gain for April 16 and April 22 by dividing the first two clusters produced.

Figure 59(b) shows that about half of the days (7 of 13) in our data set reached their highest point at 7 or beyond. Only April 11 reached its highest NMI value at the maximum number of allowed clusters of 10. Thus we see that at a various numbers of clusters did different data sets reach their highest NMI values.

Figure 60 shows the average NMI score for each possible number of clusters. The highest average NMI is at 8 clusters with an average NMI of 0.137, which again shows that the maximum number of clusters does not achieve the best NMI score. Additionally we have plotted the average NMI score of the highest points for each day of data. The maximum possible average NMI score is 0.152. This is much higher than any fixed value. Optimizing clustering for NMI with our data will not produce a fixed number of clusters for summarization. Without optimizing for NMI, our heuristic achieves an average NMI score of 0.121 since we are optimizing based on the greatest increase in variance with respect to the topic space.

Evaluating the clustering through NMI has shown that the ideal number of clusters with respect to NMI is different for each data set. There is no ideal number of clusters for normalized
mutual information. We also see that picking a fixed size number of clusters does not come close to achieving the highest values of NMI. In principle one would expect different cluster sizes per set of tweets. Through analysis with NMI we see that this is the case, one should use a variable number of clusters. What this does not take into account is the added amount of time it takes an individual to read additional sentences. If the same amount of information can be presented with one fewer cluster, that will take up less user time. There is an additional push for the number of clusters to be small while still remaining representative of the summarized data.

6.4.5 Temporal Evaluation of Tweet Topics

Now we analyze our tweet data in the time domain. The Centers for Disease Control and Prevention (CDC) provide weekly reports on influenza through the United States. For comparison we have obtained their influenza-like illness (ILI) data. The CDC data provides statistics on a per
Figure 61. The number of tweets assigned to each of 10 topics using LDA from 2011 week 44 through 2012 week 18. Each data point comes from a randomly selected day during the week. Due to network issues data was not collected for 2012 weeks 1, 7, 8, and 13 and is estimated using the previous and following weeks. The weighted percent of doctor visits with influenza-like illness symptoms reported to the CDC for the same week is shown as a dashed line.

Each week contains a count of the number of patients seen at one of the reporting out-patient facilities where the doctor identified the patient having an influenza-like illness. The counts are broken down into 4 age categories as well as a total and as a percent of the total number of patients visiting the reporting facilities.

The CDC reports statistics only on a weekly basis. Rather than summing tweets per week we randomly sampled one day per week from the data. We then ran LDA over the 27 sampled days with the number of topics equal to 10. There were in total 1.4 million tweets during this period. The results of this analysis can be seen in Figure 61.

From Figure 61 we see that as a percent of the total number of tweets the percent of tweets to any given topic has a low variance. We see two major deviations from a constant percentage of total tweets for topic 8 at week 4 and topic 10 at week 16. If we only plot topics 8 and 10 then we come up with Figure 62. Figure 62 shows both of these topics average 1,000 or less tweets in a given day. However, on the days of the two spikes, the total number of tweets for one of these topics was around 10,000. By taking a closer look we come to see the phenomenon in which this research was designed for.

Topic 8 contains a high percentage of tweets with the word “love”. This seems to be an unusual association with flu related terms. On closer look we see that when this topic spiked nearly all messages during the spike were of the form: “RT @[removed]: #oomf is MINE, my baby, my headache, my love, my smile, my frown, my wrong, my right, my sweetheart, my pain, ...”
..." The user account named in the tweet varied from tweet to tweet. This thought went viral with more than 8,000 retweets of this message in a single day.

Topic 10 appears to be sports related with the second most common word among topic 10 tweets being “LeBron,” who is a famous basketball player currently playing for the Miami Heat. On Jan. 19, 2012 it was reported, “LeBron James missed shoot around with flu like symptoms. He is a game time decision.” Also, on that same day it appears that Eli Manning, a famous football quarter back who plays for the New York Giants, left practice early with flu like symptoms. Both of these events were reported in high volume with the sports and flu related topic receiving some 11,000 tweets on this day.

The spike for topic 8 (Figure 62) is not related to the flu. All 8,000+ tweets generated for this topic could cause a false alarm. This topic would easily have been captured and reported as a single cluster. Further human analysis would immediately reject this cluster. The spike for topic 10 (Figure 62) is truly related to the flu. However, of the more than 11,000 tweets, there were only two people that were actually sick. Again these are false positives that would easily be spotted in review of potential alarms that these spikes could have triggered.

Returning to Figure 61, we plotted the reported number of ILI cases across the entire USA. At first glance it does not appear to be correlated with the tweet counts we obtained. First you must realize that the ILI counts lag behind the real-time reporting of Twitter by approximately two weeks. The high point in the ILI counts will be closer to the reported highs around week 5 in the Twitter data. Additionally, it is obvious that those reporting high correlations like 98% [2] required a lot of fine tuning of the collected terms. Our system will require a lot of additional fine
tuning to increase the accuracy with which alarms are reported. This does not change the
accuracy of users of our summaries. Even with the fine tuning of tracking keywords there will be
large unexpected spikes in Twitter traffic as observed in Figure 62 showing the relevance of our
summarization approach.

6.4.6 Human Evaluation of ReClose Multi-Tweet Summaries

We have three judges which perform the summary evaluations. One of the three judges has
both a degree in computer science and a medical degree. The other two are native or near native
in English. These three judges were assigned previously to label individually at least four days.
The data sets for each day was labeled one by one to create labeling ground truths for each day.
After summaries were created, we presented each judge with summaries for the eight days for
which they did not label tweets one by one. The idea is that we would like for users of our system
to quickly realize the trends in the data as they relate to flu outbreaks. The first step in that
process is realizing which clusters relate to Twitter users expressing their own sicknesses. We asked
our judges to identify those clusters.

Figure 63. An example of a summarized cluster of tweets unrelated to the flu (0/20 about flu).
Taken from Louisville tweets on April 18, 2012. Twitter user names removed.

Figure 64. An example of a summarized cluster where the tweets are highly related to the flu (36/38
about flu). Taken from Louisville tweets on April 18, 2012. Twitter user name and swear word
removed.

In Figures 63 and 64 we have two example clusters from those reviewed by the judges.
Figure 63 shows an example of a cluster that is made up of all tweets unrelated to the flu. The
summarizing tweets show in two very short tweets a common usage of the word “cough” found
throughout our Twitter data. Users of Twitter often use the word cough in the slang meaning to
give attention to something, or show that the opposite is true from what might be expected.
Figure 64 shows an example cluster where nearly every tweet (36/38) was related to the flu.
Headache is the most common term among the tweets that we have and this cluster contains a
large number of tweets about having a headache.

The results of our judges labeling clusters one by one from April 10 - April 22 is plotted
over time in Figure 65. For the most part the noise levels by day are constant, except for on April
11 and April 16. Most telling is that April 16 was the highest peak in the data set provided, yet the
Figure 65. Plot showing total tweets from April 10-22 in Louisville compared to the number of flu related tweets.

actual levels of flu related tweets were actually lower than neighboring days with fewer total tweets. This is the type of information that is missing from count based flu outbreak detection systems.

<table>
<thead>
<tr>
<th>Cluster Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>flu</td>
</tr>
<tr>
<td>Ground Truth</td>
</tr>
<tr>
<td>not</td>
</tr>
<tr>
<td>Accuracy</td>
</tr>
</tbody>
</table>

TABLE 29

Break down of tweet level accuracy.

Overall our judges achieved an accuracy of 0.71 as seen in Table 29. This table provides the confusion matrix of tweet classifications. In total 60% of tweets have a ground truth label of “flu.” This means that if we were observing the keywords for outbreak alarms chosen in this dissertation, then only 60% of the actual tweets relate to flu outbreaks. Through the cluster labels 72% of tweets are given a label of flu. Overall the judges’ labels obtained an accuracy (0.71) very near what we predicted (0.72) in the previous subsection for word overlap plus closeness centrality.

<table>
<thead>
<tr>
<th>Cluster Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>flu</td>
</tr>
<tr>
<td>Majority</td>
</tr>
<tr>
<td>not</td>
</tr>
<tr>
<td>Accuracy</td>
</tr>
</tbody>
</table>

TABLE 30

Break down of cluster level accuracy.

These numbers are directly affected by the purity of the clusters obtained. Improvements
to the purity measure will lead to higher accuracies. For example if we every cluster had a purity score of 0.7, the best accuracy that could be achieved would be 0.7 if the majority label was selected every time. We would also like to assess the cluster labels without consideration for purity. To do this we compute cluster level accuracy, by assigning the majority label to each cluster and comparing cluster labels against the majority label for that cluster. When computing cluster level accuracy, clusters with an equal number of labels for both classes were ignored. The results of the cluster level accuracy is shown in Table 30. Our judges were very reliable at the cluster level and obtained a cluster level accuracy of 0.86. Of the 124 clusters only 17 of these clusters were mislabeled.

<table>
<thead>
<tr>
<th>Judge</th>
<th>Accuracy</th>
<th>Cluster Acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.69</td>
<td>0.93</td>
</tr>
<tr>
<td>2</td>
<td>0.74</td>
<td>0.84</td>
</tr>
<tr>
<td>Medical Exp.</td>
<td>0.70</td>
<td>0.82</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>0.71</strong></td>
<td><strong>0.86</strong></td>
</tr>
</tbody>
</table>

TABLE 31

Clustering and tweet accuracy results by judge.

Accuracies are shown per judge in Table 31. From this table we see that our medical expert did not achieve an accuracy higher than other judges nor did he achieve cluster level accuracies higher than other judges. This suggests that perhaps the review of tweet summaries does not have to be handled by medical professionals. Familiarity with Twitter is a big help though. Hash tags and user names are not easily deciphered by those unfamiliar with Twitter.

Thus far we have only shown results of performing clustering and summarization on sets of tweets with around 100 tweets. The number of tweets one must read scales as a function of the number of clusters not the size of the data set. For example on data set of 1 million tweets, only 30 tweets would be required in the case that 10 clusters were found in the 1 million tweets.
CHAPTER 7

CONCLUSION

In this work we present our ReClose system. ReClose improves search summary generation in three main ways. First, it improves the text selection process by incorporating both query-biased and query-independent summaries. Second, we added color-coding to the summaries. This provides additional information to the user about keyword usage within documents and warns users of topic departures. Third, we incorporate structure from the original document into the summaries. This structure may include tables, lists, buttons as well as multimedia such as images. Additionally, we demonstrated the effectiveness of ReClose’s query-independent summary generation techniques in the realm of Twitter for the purpose of flu outbreak detection while also demonstrating the usefulness of variable length summaries.

The results of the ReClose text selection process show first that individual models work well and than that the overall summary generation process outperforms Google. On our CNN news article summarization task we found that we could obtain best results by combining closeness centrality with sentence order. This combination produced a ROUGE-1 score of 0.501, which was better than more traditional query-independent summarizers on the same task. SumGraph achieved a ROUGE-1 of 0.492 and LexRank a score of 0.423. We additionally tuned our query-biased module by selecting the learning algorithm and features which produced the best R-Precision score which was linear regression with an R-Precision score of 0.645. These high scoring modules led to high quality summaries when combined.

We compared the accuracy with which users selected relevant documents in search results and found that the text produced by ReClose without highlighting led users to be 10% more accurate than with Google's text summaries or 55% accurate compared to Google's 50% accuracy. Thus the combined summary approaches of query-biased and query-independent helped users to be more accurate than the highly tuned query-biased summaries of Google. Thus we proved our hypothesis correct that combining summary techniques leads users to be more accurate.

ReClose is the only functional system combining query-biased and query-independent summary approaches. We have demonstrated the effectiveness of ReClose’s text selection process. Our contribution to science is the combination of summary approaches into a multipart summary.
We also provided repeatable steps to achieve higher query-independent results on query-independent news summarization than SumGraph or LexRank. We found that when you look at Google summaries that our query-independent summarizer ranked high, that Google summaries had an accuracy with our users of 65% while summaries not including sentences ranked highly by our query-independent module had an accuracy of only 37%. Clearly we show that our query-independent ranker could improve Google summary text.

We saw even further improvement in the results with the addition of color-coding to ReClose summaries. The accuracy of relevance decisions by users jumped from 66% with Google summaries to 77% with color-coded ReClose summaries. More importantly we looked at click precision or the accuracy of a user when they have clicked on a link. In this case the click precision score went from 66% for Google summaries to 80% with color-coded ReClose summaries, more than a 20% improvement over the already refined Google summaries. We have shown that the text generated by ReClose is an improvement over Google text summaries and the results are further improved with the addition of color.

We also verified that the different color attributes were useful to users. We found that when summaries showed the darkest blue or reflect the highest level of keyword usage on a web page that the expectations of a user increased. Not only that but in a majority of cases there were high levels of relevant content on those web pages summaries using the darkest shade of blue. Not only did user expectations increase, but users were rewarded with higher quality content with high expectations. We found when users were presented with color-coded ReClose summaries, that summaries that had no flagged terms were clicked higher and that users had a higher click precision (87%) on summaries without flags than when using plain-text summaries (click precision of 78%).

Our color-coding scheme for search result summaries is unique and independent of any particular summary techniques. However, we have shown that when coupled with ReClose summaries greatly improves the user decision making process. The color-coding techniques of ReClose provide another contribution to the scientific community and to industry.

When we compare plain-text summaries to summaries which preserve document structure and multimedia we see faster decision time and a strong preference for structure. We split our evaluation into two groups of people: those experienced in abstract thinking (computer science students) and those not experienced (acquaintances). The inexperienced group was significantly faster at decision making using summaries that preserved document structure (23 of 33 people). While a few more experienced abstract thinkers were faster using plain-text summaries (9 of 22 people). The inexperienced group is more representative of the general population. We would expect that if a major search engine such as Google or Bing incorporated a lot more structured text and multimedia that the average user would make decisions more quickly.

Interestingly, there was not statistically significant difference in the accuracies of user
decisions with and without structure. Experienced users were slightly more accurate with plain-text summaries (65% vs. 70%) and inexperienced users were slightly more accurate with summaries that preserved document structure (73% vs. 72%).

We did find that in the case of both groups users preferred summaries that retain structure to plain-text summaries. 77% of experienced users preferred structure and 61% of inexperienced users preferred structure. We received many comments with more of them leaning to being more positive about structure and multimedia. One such comment was the following: “More intuitive and easier to read and understand.”

From the results obtained through testing structured text and multimedia in summaries, we infer that abstract thinkers or those working in the computer industry are more adept at inferring web page properties based on pure text. It appears that average search engine users will benefit most in terms of decision time. We therefore contribute validation that the addition of structured text and multimedia enhance search engine summaries. We also contribute the use of HTML entities as the basic unit for generating summary text in place of sentences.

The extension of ReClose to summarizing groups of tweets achieved high-quality, variable number of clusters, accurate user assessments of cluster summaries, validation of variable length summaries and validation of unforeseen tweet spikes relating to flu. We evaluated the clusters produced by clustering tweets in the topic space produced by an Latent Dirichlet Allocation model using our stopping heuristic for hierarchical clustering. The result was that we achieved purity scores in the 0.70+ range which rivaled any of the supervised learning algorithms that we tried on this same data. High purity scores led to high quality cluster summaries.

We evaluated the use of closeness centrality, word overlap distance metric and the number of topics to use for ranking tweets to summarize tweet clusters and found it that this combination along with 10 topics produced higher Kendall’s tau correlation coefficient with the ideal ranking than other approaches.

We evaluated variable length summaries using normalized mutual information (NMI). From this measure we observe that when we take each day as a separate data set, that some days (3 of 13) the best achievable NMI is for 8 to 10 clusters which is at or near the maximum number of clusters we allow. There were also three data sets were the best achievable NMI above 10 clusters was at either 2 or 3 clusters. From this we see that to achieve our best clustering we cannot rely on a fixed number of clusters. We must vary the number of clusters and in turn the size of the resultant summary.

We explored the variance in the topics available in the tweets collected based on flu related keywords. We found that as expected there were two topics which on one particular day would grow to as much as 10 times their normal size. For example when LeBron James was diagnosed with flu-like symptoms and was a game time decision, the number of sports related flu tweets
jumped from 237 the previous week to 8,223 that day which is a jump of more than 3,400%. These
types of jumps are typical of Twitter and make for very noisy data to follow. It is necessary that if
one is following the flu on Twitter, one must have confirmation before acting on the Twitter stream.

Users of our system were asked to classify each cluster as relating to the flu or not. When
we compared cluster labels to each individual tweet we achieved an accuracy of 0.71. Now the best
that we can do is to cluster each cluster according to its majority label. If we compare human
cluster labels to majority labels for clusters than we are 86% correct or have a cluster level
accuracy of 0.86. This is quite good considering that users need to only see a small number of
tweets regardless of the size of the tweet group being summarized. This means that humans can
get as good as 86% accurate on very large tweet groups by viewing less than 0.1% of the data.

We contribute an extension to ReClose showing its usefulness for summarization outside of
web pages and search result pages. We provide the first flu outbreak Twitter summarization
system. We contribute a new approach for both clustering tweets and for ranking tweets for
multi-tweet summaries.

We expect that our flu outbreak summarization system will contribute to the adoption of
Twitter social media for following flu outbreaks due to the more reliable information of health
official reviews of tweets rather than simply an automated alarm system based on tweet counts.
CHAPTER 8

DISCUSSION AND FUTURE WORK

8.1 ReClose Discussion

In our ReClose system text selection has taken a different route than has previously been considered. There are many approaches to summarization. We considered two types of summaries: query-biased and query-independent. Query-biased summaries bias the selection of text by the query keywords used to search for the document. Traditional search engines such as Google, Yahoo! and Bing all use query-biased summaries. Query-biased summaries have the benefits of showing to users why a particular document was retrieved and how those keywords are used within the document. Query-independent summaries on the other hand generate summaries that may or may not include query keywords. The main purpose of the query-independent summaries are to preserve the essence of the document as a whole in a very short span of text. Query-independent summaries are generally good at providing insight into the main topic of a document.

The ReClose system uses both query-independent and query-biased approaches to summarization. Both approaches are considered in retrieving text from a document for a summary. When constructing the text summary our ReClose system will create a two part summary. One part is focused on providing a query-biased summary to aid a user of a search engine in finding relevant documents. The second part is a query-independent summary which aids users in understanding the main ideas of a particular document to further the understanding of the user about the document summarized. By combining query-biased and query-independent summary techniques we hoped to improve the accuracy with which users could select documents from search engine results.

The ReClose system highlights text in an untraditional manner. Instead of simply bolding query keywords in a summary, we provide a color-enhanced highlighting mechanism. Query keywords in the summary text are highlighted in various shades of blue. If the summary appears to depart severely from the main topic of the presented documents then additional terms are flagged in red to warn the user of the topic departure.

As we present a web page summary, we analyze the usage of query keywords on that web page. Depending upon the frequency of those keywords the darkness of the shade of blue used
varies. A darker color alerts the user to higher usage, while a very light shade will warn the user of the infrequent use of that particular query keyword. We do present the user with a key which shows the user what each shade of blue means. However, we do not expect that a user will be able to identify very subtle differences in shade. Instead when a user sees that one summary contains a darker shade of blue for a particular query keyword, they will know that the summary with the darker keyword points to a document with a higher usage of that keyword. In this way we have added very useful information to the summary without expanding the size of the web page summary.

In addition to highlighting query keywords we also analyze the main topic words of each web page. If we find that the topic word of a particular page is very rare among the other documents returned by a particular search, then that could be a sign that this document with a rare topic word is an outlier. We flag the offending term with a bright shade of red. Users will immediately be aware that the highlighted word is a topic word, central to the topic of that particular document, and that the flagged term is unusual to the search results. In this way a user is warned about this document and may choose to avoid flagged summaries, depending on the topic flagged. Again, we are providing additional information without the cost of extending the summary size.

The ReClose system incorporates document structure and multimedia whenever possible. There are many forms of web documents that exist on the Internet. One of the most common is HTML documents. HTML enables a web page author to incorporate images, videos, tables, lists, buttons, etc. into their web page. The ability to add structure and multimedia to web documents makes the documents much richer in terms of content and allows for messages to be presented more dynamically and more clearly. We believe that this structured text and multimedia is important to the messages and topics of many web pages. Traditional search engine summaries rarely contribute structure or multimedia from the original web page into the final summary presented to users.

Our ReClose system does not discard the HTML entities present in many web pages. Instead when summaries are presented where the text is extracted from a part of the web page, any structure or multimedia present in that region of the web page is preserved and provided in the generated summaries. This means that tabular data is presented as tabular data rather than comma separated values. Text extracted from HTML image tags is replaced by the original image. Text found in lists is presented as lists, and form fields and buttons are presented as they originally are in the web page. This complicates the extraction process of text for use in our summary system, but provides users with enhanced, clearer, more easily understood summaries.

Additionally, our current approach to storing structure from web pages attaches the structure to individual characters. This will increase the size of the index stored by a search engine. Not all HTML will need to be stored. One will not need to store header HTML, paragraph
tags and many of the tags that would not add structure to the resulting summary. It is difficult to estimate the amount of additional space required to store table, list, form and image HTML because it is difficult to gauge how frequently such structure is used on the Internet. Search engines such as Bing and Google already store many of the images stored online in their image searches, so that image size would not need to be considered. We would estimate that the size of the index would increase by somewhere between a fourth to a half depending on how and which HTML elements were preserved. This would require that only the minimal HTML be stored such as table tags not including unnecessary attributes such as table width.

When various HTML structure is included in summaries in search, using the current methods of summary generation, the size of the summaries will increase. Means will need to be sought to change how the size of the summaries are monitored with structure. Likely including tables and lists will increase the size of summaries. This will mean more web-page real estate occupied by summaries and possible fewer search results that can be shown in a single screen. Not every summary will include structure. Most of the time including structure improves a users understanding of the page and is worth the additional real estate. There are will be many cases that the additional space is not worth it. Additional heuristics will need to be implemented to work with these expanded summaries and when to use text from structure and when not to.

In addition to summarizing web pages, we have also extended ReClose to summarizing groups of tweets on Twitter. Tweets provide a new challenge. Each tweet is very short, 140 characters at most, and each are written by different authors. This means that there is a large amount of diversity in the vocabulary used in tweets. It also means that there are a large number of misspellings and abbreviations used. This poses a big challenge to traditional automatic summarization techniques. There is very little overlap among many tweets making it hard to determine which tweets fall under which topics.

We have extended ReClose to Twitter for one particular usage case. We would like to present health officials with summaries of trends occurring on Twitter as they relate to influenza outbreaks. We believe that a semi-automatic flu outbreak detection system is an improvement over many of the proposed automatic flu outbreak detection systems. Automatic systems depend on a predetermined set of keywords and the frequency of tweets containing these keywords to raise alarms on influenza outbreaks. We proposed to include a human in the loop. Automatic flags would still be raised, but we would provide an interface through summarization, in which a user of the system could view trends shortly after they occur to determine if a flu outbreak is truly happening or some other type of event has taken place which uses overlapping keywords.

ReClose for Twitter first clusters the set of tweets to find common topics within the group of tweets. We then apply the same query-independent text selection used for web pages to tweets. Instead of using sentences as a basic unit of summarization, we use tweets. The result is that our
system produces a variable number of clusters, depending on the topics present. Each cluster is summarized through ReClose. Users are then able to determine which clusters relate to individuals messaging about their sicknesses and which clusters are about some other topic.

### 8.2 Extensions Discussion

The ReClose system was designed for the express purpose of using it for improving search results. As the results show, it would do a fine job of this. We would expect that if ReClose were incorporated into search results to improve the interaction between users. In our study of adding structured text and multimedia it was found to increase the speed of decision making. We would expect that to hold true when implemented on a larger scale. Also, from our studies of improved text selection and color-coding that users would make more accurate decisions about which search result documents to click through to.

If the decision making process was both faster and more accurate, users would waste less time clicking through to irrelevant results. They would also spend less time on search result pages. The result would be a higher rate of satisfaction of search users. This would in turn lead to more usage and to users liking to use the search engine more. There may be additional indirect benefits to such a system. Users of all walks of life of search systems would be benefited in many ways if they are more likely to find what they are looking for and find it more quickly. Travelers would be more likely to find hotels and flights. Doctors would more likely find necessary information for their job, benefiting patients. And researchers would more likely be able to find answers to the questions that affect their work.

We would expect that if a search engine were to implement our changes, that there would also be a number of users that would not initially be happy with the changes made. With any changes made there will always be those that resist change. It would be necessary and expect that my results would need to be verified before such revolutionary changes were to be made in search result summaries.

There would also need to be some additional testing to figure out how the ranking of text entities would change as one swaps from sentences as the basic unit to HTML entities. The structured text and multimedia extraction and preservation in summaries would need to be extended to additional file formats. The PDF format for example does not encode tables as tables, but only text and lines. This can present challenges to the discovery of tables in the text and can result in occasional mistakes where text and lines not meant as a table would be extract. Additionally, there would be some tables that would be obvious to human eyes, but would not be extracted automatically.

In addition to helping in search results, we have also shown that our system is effective at summarizing social media. Social media can provide near real-time access to the events going on
all over the world. We can get more immediate information about an earthquake in Japan. We can obtain political results as they happen. We also have a window into public opinion and what is happening in the lives of those that tweet. ReClose could effectively help politicians understand the trends in political opinion online. As shown earlier, we can observe when people are getting sick in a specific region or as a whole in the region. We can understand some information about what the sickness is and what we can do about it.

We believe that our ReClose system for Twitter could have a large benefit to the tracking of influenza outbreaks. Many of the current proposed systems automatically generate alarms and seek to combat false positives in an automatic manner. It is impossible to foresee all causes of false positives in such a system. We cannot guess now all possible usages of keywords that one may be tracking now. It is therefore necessary to verify that a flu outbreak is truly possible before any actions are performed. We believe that our system would greatly assist the health professionals tracking flu outbreaks to determine if an alarming spike in tweets with keywords relating to flu outbreaks really contains a large number of tweets from sick people. As we observed the sickness of a famous person can cause an increase of 8,000 tweets in the United States.

If many false alarms are reported by a flu outbreak system and one is not able to verify them, the trust that any individual or organization can have in such a system will be low. It can be very expensive to react to a potential outbreak to try to contain it. One cannot expect that health officials will be happy to act on false alarms. One false alarm can be enough to lose trust. We must provide ways to verify the potential of a flu outbreak before actions are taken. If we can create trust in the system by involving a human in the loop, then we will be able to gain the trust of the health professionals. Without trust, no system will be maintained and a potentially useful source of information would be lost.

As we track more and more complex questions about what people are saying on say Twitter, this will require additional extensions to ReClose. For example the current summaries shown are for a static view of a single time period. Instead, one could expand this work to show summaries over long periods of time to understand what is happening over a whole month or year. Some time related analysis was performed in Chapter 6. The graphs shown for topic size through out time could be adapted and enhanced through summarizing text.

Now that we have summarized the “multi-documents” of Twitter or the multiple tweets, we can extend the same techniques to multi-document summarization. This presents many possible directions. One can look to summarize trends across multiple blogs, news sites, medical research, etc. Multi-document summarization can give high level insight to a large number of articles on a specific medical research on a single topic. One can also foresee benefits to automatic summarization to automatically summarizing legal proceedings as they relate to the specifics a current case.
8.3 Future Work

There are several directions that future work can take building on this dissertation. There are improvements to how summary content is selected for summary generation, data sources used in text generation and summary formatting.

Through the exploration of preserving structure, it was realized that various structure has different affects on the comprehension of summaries. First we shall explore in more detail the effect of each possible HTML structure has on comprehension. For example if we have two sentences that in principle say approximately the same thing, but one is contained in a table and the other is in a list, which would be best to display. Second, we would like to consider the ranking of HTML entities. Are their entities that should have greater weight like images? Perhaps tabular data is generally found to be more useful even when there is a lack of keywords in the table contents. There will need to be some analysis which entities should be weighted highest and how that ranking should consider the HTML entity type and the content contained in those entities.

We will need to explore structured text and multimedia as it exists in other file formats that exist on the Internet. Certainly one could foresee that summaries for PowerPoint slides would be very different from that of an Excel file. Acrobat’s PDF file is very common on the Internet. Further analysis would need to be conducted to understand how to extract the various text structures that are possible.

Further testing and revisions are necessary to establish the ideal colors for the color-coding. It would be necessary in this testing to establish the effects of different color choices have on those with visual impairments and on the color-blind. Perhaps it would be very difficult for a percentage of the population to distinguish the various shades of blue currently used in the color-coding. There were no color blind reported in our studies.

When one has indexed the Internet as Google has, then there are times that descriptions of a website from other websites could be used to more effectively describe the content of a web page. Currently search engines consider text from the Open Directory Project\(^1\). Perhaps descriptions from Wikipedia and other websites could be used to more effectively summarize web pages. Currently anchor text or text used to link to a web page is used in establishing the ranking of a particular web page in search results. When we know that a large number of web pages refer to a particular page, it may be possible to use information from all of those linking pages to establish better summaries of a web page. Perhaps stitching together discussion from a number of pages would provide a more accurate view of a web page than the content of a web page would have. For really long web pages extracting a couple sentences does not give a very good overview of a web page while some sentences external to that web page may simply and effectively establish the overall purpose of a web page.

\(^1\)http://www.dmoz.org/
Information about the type of query being performed could be taken in consideration to give more direct feedback to users. Currently, Google and Bing vary the presentation of search results slightly for a few categories of search queries. For example, recipe searches on Google result in specially formatted search results from certain websites (such as allrecipes.com), where summaries are enhanced with the number of stars given by user ratings and a list of ingredients are added to the summaries. However, this direction could be pushed further. Rather than simply a query-biased summary, answer-biased summaries could be pursued in which the answers to questions are surfaced in summaries, rather than just the query keywords. When a user searches for flights, summaries for each web page listed should include example flight schedules. When a user searches for symptoms of the flu, then symptoms should be directly listed in the summaries of web pages rather than the search query flu symptoms.

The ReClose system could be extended to summarize people on Facebook, events in social media, documents used by software engineers, news articles as they relate to stocks, trends in medical records, academic conference proceedings, medical journals, world history events, biographies, legal cases by topic, user logs, student reports, and blogs. There are still many untapped opportunities to use automatic summarization that we believe ReClose could make a big impact to.

We could further use fuzzy sets to represent unclear meaning in words in the case of extending our summary methods of Twitter. Often times terms have unclear meanings. Rather than putting a term under a specific topic, instead a fuzzy set could be used to represent this unclear meaning and put different proportions on different meanings of the word.

We have begun the process of extending ReClose to work with multiple documents through our work on Twitter tweet group summaries. However, large documents present new challenges that are not present in short “documents” like tweets. One may consider how to summarize multiple search result documents simultaneously so that either their summaries take into account what the other says, or so that there is a single summary across multiple documents. There are many possible ways that search result pages could be presented to users.

ReClose opens the door to several extensions and enhancements to the way that search result summaries are presented to users and how summarization is performed.
REFERENCES

[1] Open directory project listing.  


145


CURRICULUM VITAE

NAME: Brent Kevin Wenerstrom

ADDRESS: Department of Computer Engineering and Computer Science
University of Louisville
Louisville, KY 40292

EDUCATION:

Ph.D. Computer Engineering and Computer Science
University of Louisville
est. 2012

M.S. Computer Science
Brigham Young University
2006

B.S. Computer Science
Brigham Young University
2003

PROFESSIONAL EXPERIENCE:

2010-2012 Graduate Student Instructor, University of Louisville, Louisville, KY
2008-2010 Teaching Assistant, University of Louisville, Louisville, KY
2006-2007 Software Engineer, iCrossing, Salt Lake City, UT
2005-2006 Research Assistant, BYU Data Mining Lab, Provo, UT
2004-2005 Software Engineer, Screen-scraper.com, Provo, UT
2002-2003 Teaching Assistant, BYU, Provo, UT

PUBLICATIONS:

Journal Articles:


Book Chapters:


Conference Papers:

5. B. Wenerstrom, M. Kantardzic (2012), Search Result Summaries Improved by Structure and Multimedia, Accepted for publication in the 4th International Conference on Knowledge Discovery and Information Retrieval.


Workshop Papers:


Thesis:

13. B. Wenerstrom (2006), Temporal Data Mining in Dynamic Feature Spaces (Masters thesis), Brigham Young University, Provo, Utah.

VOLUNTEER WORK:

- 2009 Volunteer at Midwest SAS Users Group Conference
- 2003, 2007-2010 Cub Scout Leader
- 1999-2001 Service and Religious Mission to Tacloban, Philippines
- 1998-1999 Chess Club President, BYU, Provo, UT

HONORS & AWARDS:

- 2012 Nominated for Professor of the Year in CECS Department
- 2011 CSE Doctoral Award from CECS Dept. at the U. of Louisville
- 2010 Completion of Graduate Teaching Academy
- 2009 International Association for Development of the Information Society Outstanding Paper Award
- 2009 Student Scholarship to Midwest SAS Users Group Conference
- 2007-2008 University Fellowship at the U. of Louisville
- 1993 Eagle Scout