Semantic Annotation and Summarization of Biomedical Text

A Dissertation

Submitted to the Faculty

of

Drexel University

by

Lawrence H. Reeve

in partial fulfillment of the

requirements for the degree

of

Doctor of Philosophy

July 2007

Acknowledgements

I would like to thank the members of my dissertation committee for their participation in my dissertation proposal and final defenses, for reviewing my papers, and for the many comments and suggestions which improved this work. I would particularly like to thank Dr. Han for the many hundreds of selfless hours spent molding me into a doctoral candidate. The members of my dissertation committee are:

Dr. Hyoil Han, Advisor and Chairperson
Dr. Il-Yeol Song
Dr. Xia Lin
Drexel University, College of Information Science and Technology
Philadelphia, PA 19104

Dr. Ari D. Brooks
Drexel University, College of Medicine
Philadelphia, PA 19104

Dr. Suk-Chung Yoon
Chairman and William R. Bailey Endowed Chair Professor
Widener University, Department of Computer Science
Chester, PA 19013

I would also like to thank my employers, CIMS Lab, Inc. and IBM Corporation, for their financial support of my program. Most importantly, I would like to thank my immediate family for allowing me to time to pursue this path: my wife Christine and my children Lawrence III, Lisa, and Laura; as well as my parents, Lawrence Sr. and Rae, for their never-ending support and for instilling the value of education in me. Finally, I would like to recognize those in my family who have suffered or are suffering from cancer and who made me realize on a daily basis that in some small way my interest in information might help those in medicine treat the disease better: my father (multiple myeloma), my father-in-law (colon cancer), and my sister-in-law (breast cancer).

Table of Contents

List of Tables

List of Figures

Abstract

Semantic Annotation and Summarization of Biomedical Text

Lawrence H. Reeve
Hyoil Han, Ph.D.

Advancements in the biomedical community are largely documented and published in text format in scientific forums such as conference papers and journals. To address the scalability of utilizing the large volume of text-based information generated by continuing advances in the biomedical field, two complementary areas are studied. The first area is Semantic Annotation, which is a method for providing machine-understandable information based on domain-specific resources. A novel semantic annotator, CONANN, is implemented for online matching of concepts defined by a biomedical metathesaurus. CONANN uses a multi-level filter based on both information retrieval and shallow natural language processing techniques. CONANN is evaluated against a state-of-the-art biomedical annotator using the performance measures of time (e.g. number of milliseconds per noun phrase) and precision/recall of the resulting concept matches. CONANN shows that annotation can be performed online, rather than offline, without a significant loss of precision and recall as compared to current offline systems. The second area of study is Text Summarization which is used as a way to perform data reduction of clinical trial texts while still describing the main themes of a biomedical document. The text summarization work is unique in that it focuses exclusively on summarizing biomedical full-text sources as opposed to abstracts, and also exclusively uses domain-specific concepts, rather than terms, to identify important information within a biomedical text. Two novel text summarization algorithms are implemented: one using a concept chaining method based on existing work in lexical chaining (BioChain), and the other using concept distribution to match important sentences between a source text and a generated summary (FreqDist). The BioChain and FreqDist summarizers are evaluated using the publicly-available ROUGE summary evaluation tool. ROUGE compares n-gram co-occurrences between a system summary and one or more model summaries. The text summarization evaluation shows that the two approaches outperform nearly all of the existing term-based approaches.

1. INTRODUCTION

The output of biomedical research is largely documented as findings in the form of literature written in free-form text format (Nenadic, Mima, Spasic, Ananiadou, & Tsujii, 2002). The written texts are then accumulated in large online databases made readily accessible due to recent advances in software and communications. For example, the PUBMED database provided by the United States National Library of Medicine contains over 16 million publications from over 4,800 journals (United States National Library of Medicine, 2006a). The United States National Institutes of Health clinical trials database contains information on over 13,500 clinical trials (United States National Library of Medicine, 2005a). To use such resources, practicing physicians and biomedical researchers are faced with the task of locating, reading and evaluating relevant biomedical literature. For example, oncologists must find the clinical trial information related to their cancer specialty, evaluate the study for its strength, and then possibly incorporate the new study information into their patient treatment efforts (Brooks & Sulimanoff, 2002), (Jaques, 2002). The large number of clinical trials conducted and the data produced by them make the information assimilation process time consuming. This research builds several novel approaches for semantic annotation and text summarization, and is an effort to help reduce the time to assimilate the textual data resulting from large collections of literature in the biomedical domain by reducing the amount of data that must be manually read and processed. In this research, document and text are used interchangeably.

There are two complementary components to this research: Semantic Annotation and Text Summarization. Semantic Annotation, sometimes called concept matching in the biomedical literature, is the process of mapping phrases within a source text to distinct concepts defined by domain experts. Text Summarization is a method for reducing the amount of text data which must be read while retaining the key ideas of the source text. A system for producing summaries of biomedical text using biomedical concepts as the unit for identifying key information is constructed. This system is expected to allow physicians and researchers to quickly review biomedical documents without requiring a reading of the full source text. The corpus of biomedical text used in this research is from randomized controlled trials in oncology.

The first component, Semantic Annotation, takes biomedical source text as input and then identifies domain concepts within the source text. In semantic annotation, a domain-specific thesaurus can be used to find concepts within free-form texts. A thesaurus is organized by concept, with one or more synonymous words and/or phrases expressing the concept. Using the thesaurus concepts, words take on meaning rather than being a surface feature. A benefit of semantic annotation in the biomedical domain is merging different ways of expressing the same concept, possibly using different words, into a single concept.

A key issue with semantic annotation is the variability of human language, which makes the concept mapping process non-trivial. For example, the biomedical concept *Lung Cancer* has many possible expressions, such as *Cancer of the Lung* and *Pulmonary Carcinoma*. While much research has been done in biomedical semantic annotation, its use is largely designed for indexing documents based on the concepts identified in the text. Such systems are designed to be used in an offline environment, where speed is not critical. In some systems, such as MetaMap (Aronson, 2001a), efforts are made to find a best-matching concept, while in other systems, such as IndexFinder (Zou, Chu, Morioka, Leazer, & Kangarloo, 2003), all possible concepts are found. The difference is usually determined by the application in which the annotations will be used. For example, finding all concepts within a source text is useful in search and retrieval indexing, while best-matching annotations are useful in applications such as text summarization. Existing concept annotators are slow performing, precluding their use in online applications, where the text is annotated dynamically, rather than statically. In typical search and retrieval applications, static annotation is fine since neither the text nor the concept resource is expected to change. However, in some applications, dynamic annotation is needed to allow for changing concept resources or unseen texts. For example, a user may want to find concept annotations from multiple concept resources, such as UMLS and NCI Thesaurus. A text's concept annotations in the presence of changing concept resources, requiring new or modified concept annotations, can lead to concept annotation maintenance issues (Dingli, Ciravegna, & Wilks, 2003). An annotation system designed for online use can avoid concept annotation maintenance issues by providing annotations dynamically (at runtime). However, in order to provide dynamic annotations, the system must perform at a level of acceptable end-user response time. Our research focuses on constructing and evaluating a biomedical annotator which can be used in an online environment with accuracy competitive with state-of-the-art offline annotators.

The second component of the research is Text Summarization, where the use of domain-specific concepts to find important information within a text is examined. The use of domain-specific concepts is hypothesized to provide a better method for identifying important textual information than the use of terms, which have not been annotated for meaning. Text summarization expresses key ideas using fewer sentences than the source text. The benefit of having fewer sentences while retaining key ideas is potentially faster assimilation of content. The goal of text summarization is to present a subset of the source text which contains the most important points within a source text with minimal redundancy. The summary can then be used by the reader to (a) make a determination if the source text should be read in its entirety, or (b) act as a surrogate for the source text to obtain information without reading the entire source text. The use of text summarization allows a user to get a sense of the content of a source text, or to know its information content, without reading all sentences within the source text. The reduction of data afforded by text summarization increases scale by (a) allowing users to find relevant source texts more quickly, and (b) assimilating only essential information from many texts with reduced effort. Much work has already been done by the text summarization community, largely in the general domain with genres such as news. While some research work has been done for domain-specific summarization in restricted domains such as legal and medical, the work is incomplete. For example, the use of domain-specific resources in text summarization, such as vocabularies, ontologies and so forth, is largely ignored.

It is sometimes the case an author's abstract is available as a summary of the paper. For example, in the biomedical domain, published clinical trial results usually have an abstract to supplement the full-text. In cases where an abstract is available, any system-generated summary competes with the author's abstract. While system-generated summaries may seem like a duplication of effort, each user is likely to have a unique information need, different background, different motivations, and so forth, and a single summary, even the author’s own summary, does not address this issue.

Our research implements an end-to-end system where a biomedical source is first annotated with biomedical concepts, the discovered concept output is then directed into a summarizer stage, and a finally a summary is generated based on a user-defined size. Both the semantic annotation and resulting summary are envisioned to be used in an online environment, where expected response times are lower than in a state-of-the-art offline system. Although our research is focused on the biomedical domain, the resulting system is expected to be useful within other domains which have domain-specific resources available.

The remainder of this chapter identifies the contributions of our research, presents some background on semantic annotation and text summarization, and describes the organization of this dissertation.

1.1 Research Questions

The goal of this research is to construct a system to perform semantic annotation and summarization of biomedical texts which has performance competitive with, or better than, existing systems. The identification and use of domain-specific concepts is used to produce summaries. The methods developed, while requiring input from domain resources, are domain-independent, and could be applied to other domains, such as summarizing legal texts. The long-term result of this research is to reduce the amount of work required by practicing physicians and researchers to assimilate new knowledge from continuing advancements in biomedicine. This research may also be helpful for future researchers performing multiple-document summarization where concepts, rather than terms, can reduce the variability of language among documents.

For the Semantic Annotation component, methods for performing faster annotation of biomedical texts are examined. The idea is to improve the performance of annotation so that it can be used in an online environment. For the Text Summarization component, concepts are used in place of terms to identify important areas within the source text which should be extracted into a summary. The idea is to determine if the use of domain-specific concepts improves the performance of summarization. In addition, since the research on the characteristics of biomedical texts is largely absent from existing literature, two important characteristics of biomedical texts are examined: (a) finding an optimal size of a biomedical summary (at least for randomized controlled trials), and (b) finding the locations within texts where sentences are extracted from when constructing model summaries. Biomedical texts are usually written with a definite structure, such as Introduction → Methods → Results → Discussion → Appendix. Knowing which sections human summarizers are likely to extract sentences from is expected to be useful in improving summarization performance by weighting sections differently, instead of giving equal weight to all sections.

The following research questions are addressed in this research:

1. Does the use of language modeling methods improve biomedical semantic annotation performance over existing methods which use simple word metrics?

2. How well does a concept-based approach for biomedical text summarization perform as opposed to term-based summarization approaches?

   2.1. Does adapting a lexical chaining approach for use with domain-specific concepts improve text summarization performance over existing approaches?

3. Does the use of frequency distribution of terms and/or concepts in a source text improve text summarization performance?

1.2 Contributions

The primary contributions of the research are as follows:

1. An ontology-based biomedical annotator for annotating biomedical texts which can be used in an online environment is designed, implemented and evaluated.

2. Single document text summarizers which use domain concepts to identify salient sentences within a source to produce an extractive summary are designed, implemented and evaluated.

3. Several characteristics of biomedical texts, specifically concept distribution and summary size, are studied.

1.3 Dissertation Organization

This dissertation is organized into six chapters. The first chapter, this chapter, introduces annotation and text summarization, describes the contributions of our research, and presents background information. Chapter 2 provides a literature review, broken down into two major areas. The first area is semantic annotation, and it is reviewed in the large (i.e., not domain specific) as well as in the biomedical domain. The second area is text summarization, which has a rich history extending back approximately 60 years. Chapter 3 describes the approaches for producing a semantic annotation and text summarization system for the biomedical domain. We describe the design of our phrase-unit concept annotator called CONANN. CONANN iteratively filters out potential matches of a source phrase with ontology concept instances. Filtering is accomplished by measuring the words in common between a source phrase and ontology concept instances using coverage (word overlap) and coherence (word order). The final selection of a concept instance is done using language modeling or phrase counting. The annotated phrases of a source text are passed to a text summarizer. We described two new methods for using concepts to find the most important sentence with a text. The BioChain method links related concepts together and then uses the strongest linkages to identify important sentences. The FreqDist method matches the frequency distribution of a summary with the frequency distribution of the source text. A hybrid summarizer which combines both approaches is also described. Chapter 4 discusses the evaluation methodology for annotation and text summarization. The CONANN annotator is evaluated using both intrinsic and extrinsic evaluation approaches. An intrinsic evaluation compares CONANN's concept output to the concept output generated by a state-of-the-art concept annotator. The extrinsic evaluation measures the use of the generated concept annotations in a text-summarization task. Chapter 5 presents the results of the evaluation. We first present the concept distribution and ideal summary size characteristics of the biomedical texts within our corpus. The performance of CONANN using different filtering combination and mapping methods is also presented. The results of the performance of BioChain, FreqDist, and the hybrid summarizers are presented. In addition, the performance of these summarizers is compared to several other publicly-available summarizers. Significance testing of ROUGE scores is also described. Chapter 6 concludes and presents some areas for future work.

1.4 Background of Semantic Annotation

Semantic annotation is the process of identifying pre-defined concepts and entities within a text. One or more domain-specific resources are used to annotate the text with the concepts and entities found. For example, Figure 1 shows the annotation of a biomedical sentence using the Unified Medical Language System (UMLS) (United States National Library of Medicine, 2005c). The sentence phrases are shown as text within rectangles, while the corresponding UMLS concept which the text maps to is shown as text contained in ovals. Directed lines indicate a relationship between concepts. Undirected lines indicate a link between a concept instance and a concept. In Figure 1, concept {*Myeloma*} is related to concepts {*Progressive*, *Cancer*, and *Haematologic Disease*}. Further, the concept {*Haematologic Disease*} is related to {*Plasma Cell*} in addition to {*Myeloma*}. From this relationship graph, it can be seen that the concept {*Myeloma*} is indirectly related to the concept {*Plasma Cell*}. The use of Semantic

Annotation allows for such relationships to be discovered in text, and so is more powerful than the use of surface terms alone.

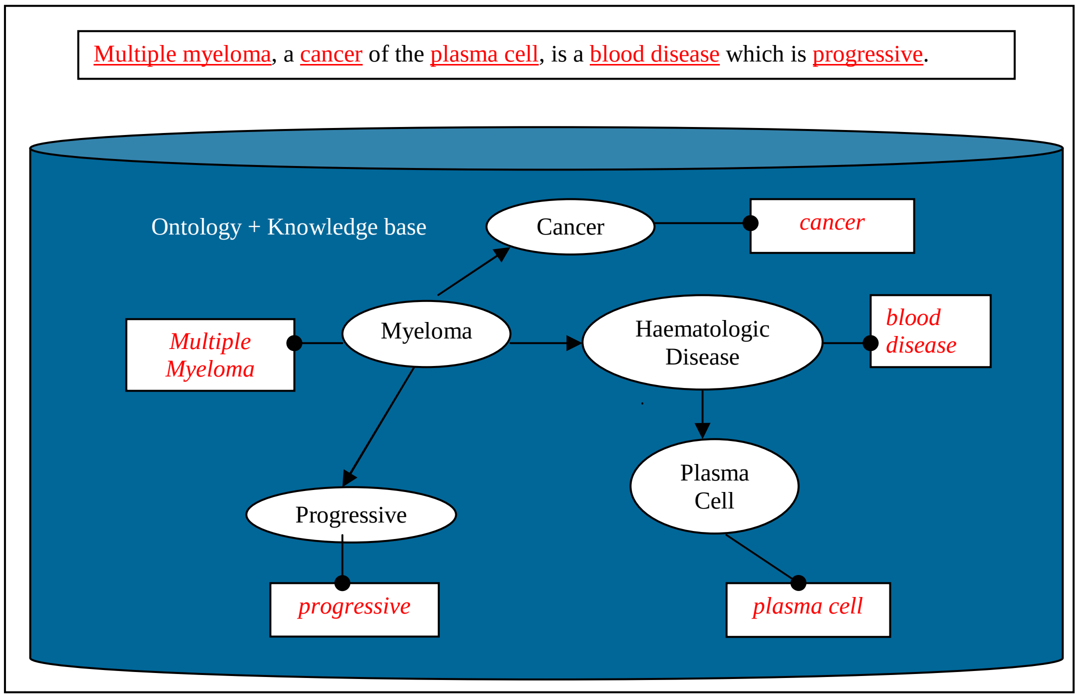

Multiple myeloma, a cancer of the plasma cell, is a blood disease which is progressive.
Ontology + Knowledge base
Cancer
cancer
Multiple Myeloma
Myeloma
Haematologic Disease
blood disease
Plasma Cell
Progressive
progressive
plasma cell

Figure 1: A graphical view of semantic annotation of a sentence. Ovals represent biomedical concepts and rectangles represent instances of the biomedical concepts from the sentence. Directed lines indicate relationships between the concepts, while undirected lines link a concept instance to a concept.

Once the text has been annotated with concepts, the sentence phrases have meaning which is more easily processed by a machine than raw text alone. There are two expected advantages of using biomedical concept annotations, rather than raw biomedical text, in this research: (a) synonym merging, and (b) semantic filtering. Synonym merging is the process of using synonyms identified by various medical vocabularies and collapsing them into a single concept. This is important for identifying salient information for text summarization. If only raw text were used, different expressions of the same concept would be identified as distinct entities. For example, the phrase *heart attack* can be expressed in several ways, such as {*heart attack*, *coronary attack*, *myocardial infarction*}. If raw text is used and the author used multiple expressions, these three phrases would be considered distinct. Since reiteration is a strong component of identifying important information within text (Sparck Jones, 1999), it is better to collapse these three synonymous phrases to a single concept, such as {*Myocardial Infarction*}, to help identify reiteration and thus the most important parts of the text as expressed by an author.

The second major use of annotations in this research is to use them to filter out concepts which are not important to a user of a summarization system. Semantic filtering allows customizing a summary for a user's information need. For example, an experienced physician may not need much background information on a clinical trial, and instead want to focus more heavily on the results and methodology of the clinical trial. In this case, the physician can customize the summary to filter out qualitative concepts, which express primarily opinion and background information. The user can be presented a list of concepts which are not important, as well as a list of concepts which are important. One example filtering method is concept weighting. In concept weighting, the unimportant concepts selected by the user are negatively weighted, while the important concepts selected by the user are given increased weight over neutral or unimportant concepts.

1.5 Background of Text Summarization

While the result of the research work could be used to summarize abstracts, the goal is not to summarize abstracts (summarizing the summary), but instead to use the full source text to produce a summary. There are several reasons for wanting to generate text summaries from a full-text source even in the presence of the author's abstract:

1. There exists no ideal summary. An ideal summary is dependent on each user, including factors such as information need and domain background. An author's abstract is one view of an ideal summary, but users may want alternative summaries.

2. The abstract may be missing content from the full-text (Cohen & Hersh, 2005), (National Institute of Health, 2005).

3. Customized summaries can be useful in question-answering systems where they provide personalized information. Such summaries have been described in the literature as *user-focused* (Mani & Bloedorn, 1998) and *query-relevant* (Carbonell & Goldstein, 1998) summaries.

4. The use of automatic or semi-automatic summary generation by commercial abstract services may allow them to scale the number of published texts they can evaluate.

5. The generation and evaluation of summaries allows for evaluation of sentence selection methods that may be useful for use in multi-document summarization. The idea is that if sentence selection methods do not work well for single-document summarization, it is unlikely they will identify important data across multiple documents.

Figures 2 and 3 below motivate the need for text summarization even in the presence of the author’s summary (the paper’s abstract). Figure 2 shows an abstract from a biomedical text. While the text itself is relatively short (as compared to the original source text), if the reader wishes to quickly know the outcome of the research described in the abstract, the abstract must be read partially or completely, or skimmed at the very least, to find the information. In contrast, Figure 3 shows a summary which automatically identified the result information and displayed it.

Adjuvant Chemotherapy for Adult Soft Tissue Sarcomas of the Extremities and Girdles: Results of the Italian Randomized Cooperative Trial.

Adjuvant chemotherapy for soft tissue sarcoma is controversial because previous trials reported conflicting results. The present study was designed with restricted selection criteria and high dose-intensities of the two most active chemotherapeutic agents.

Patients and Methods: Patients between 18 and 65 years of age with grade 3 to 4 spindle-cell sarcomas (primary diameter >= 5 cm or any size recurrent tumor) in extremities or girdles were eligible. Stratification was by primary versus recurrent tumors and by tumor diameter greater than or equal to 10 cm versus less than 10 cm. One hundred four patients were randomized, 51 to the control group and 53 to the treatment group (five cycles of 4'-epidoxorubicin 60 mg/m2 days 1 and 2 and ifosfamide 1.8 g/m2 days 1 through 5, with hydration, mesna, and granulocyte colony-stimulating factor).

Results: After a median follow-up of 59 months, 60 patients had relapsed and 48 died (28 and 20 in the treatment arm and 32 and 28 in the control arm, respectively). ***The median disease-free survival (DFS) was 48 months in the treatment group and 16 months in the control group (P = .04); and the median overall survival (OS) was 75 months for treated and 46 months for untreated patients (P = .03).*** For OS, the absolute benefit deriving from chemotherapy was 13% at 2 years and increased to 19% at 4 years (P = .04).

Conclusion: Intensified adjuvant chemotherapy had a positive impact on the DFS and OS of patients with high risk extremity soft tissue sarcomas at a median follow-up of 59 months. Therefore, our data favor an intensified treatment in similar cases. Although cure is still difficult to achieve, a significant delay in death is worthwhile, also considering the short duration of treatment and the absence of toxic deaths.

Figure 2: Sample biomedical document abstract (Frustaci et al., 2001)

The median disease-free survival (DFS) was 48 months in the treatment group and 16 months in the control group (P = .04); and the median overall survival (OS) was 75 months for treated and 46 months for untreated patients (P = .03).

Figure 3: Summarized version of the abstract shown in Figure 2
(Frustaci et al., 2001)

There are two different approaches for generating summaries from text: extractive and abstractive (Afantenos, Karkaletsis, & Stamatopoulos, 2005). The *extractive* approach extracts sentences or parts of sentences verbatim from text and uses them to generate a summary. The extractive approach is the most common way to perform summarization, and is the method used in this research. Extractive approaches for text summarization usually follow a model of scoring sentences based on a set of features, such as term frequency, keyphrase identification, and sentence location. A set of highest-scoring sentences from the source text is used to form a final summary. The task of sentence selection can be considered an information retrieval task, where the set of all sentences within a text are evaluated (scored), and the highest scoring sentences are selected as being the most relevant to a user. The top-*n* highest-scoring sentences in a text are extracted, using *n* as an upper bound on the number of sentences to select. The extraction summarization task, then, is to identify a minimal subset of sentences from the source text which are relevant to the user and which minimize redundancy. This research uses the extractive approach.

The second and significantly more difficult approach is called *abstractive summary generation*, and involves generating summary text using natural language processing and generation techniques. For example, the template method of abstractive summary generation uses a predefined template where the fields in the template are filled-in from information contained in the source text. A different method for abstraction uses a syntactical analysis of the source text to identify key components of each candidate sentence, and this analysis is used to form new sentences from existing sentences.

1.6 Background of Biomedical Concepts

One way to provide meaning to biomedical documents is by creating ontologies, and then linking information within each document to specifications contained in the ontology using a markup language (Berners-Lee, Hendler, & Lassila, 2001). Ontologies are conceptualizations of a domain that typically are represented using domain vocabulary (Chandrasekaran, Josephson, & Benjamins, 1999). Automated semantic annotation is the process of mapping instance data to an ontology (S. Handschuh, Staab, & Volz, 2003), (Reeve & Han, 2006). The resulting annotations from the semantic annotation processing are what provide the link between information stored within a document and the ontology (Berners-Lee et al., 2001). In this work, the annotations are then used to identify important areas of a text, such as phrases, sentences, paragraphs, and sections useful for generating a text summary. In the biomedical domain, the National Library of Medicine (http://www.nlm.nih.gov/) provides resources for identifying concepts and their relationships under the framework of the Unified Medical Language System (UMLS) (United States National Library of Medicine, 2005c). UMLS contains many sub-components, but three are used for this research: Metathesaurus, Semantic Network and MetaMap Transfer.

The UMLS Metathesaurus contains concepts and real-world instances of the concepts, including a concept name and its synonyms, lexical variants, and translations

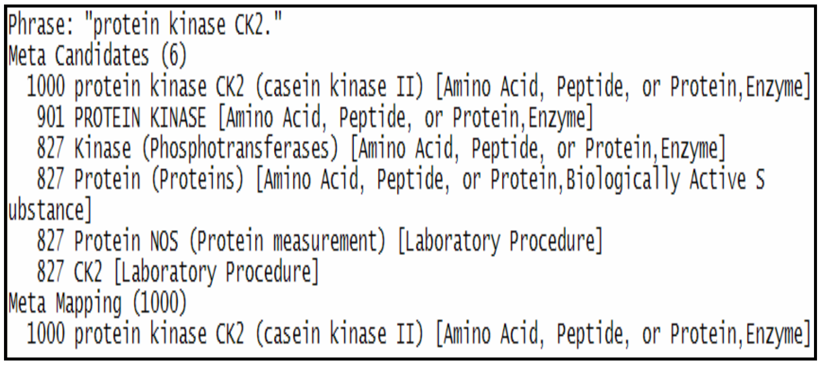
(United States National Library of Medicine, 2006b). The Metathesaurus is derived from over 100 different vocabulary sources resulting in over one million biomedical concepts. Table 1 shows the example concept *Multiple Myeloma* taken from the Metathesaurus, and displays several of the concept instances (i.e., synonymous words and phrases) associated with the concept. The instances are derived from the vocabulary sources. The key idea is that a single concept may have multiple ways of being expressed (instances). The Metathesaurus organizes the concept instances.

Table 1: UMLS concept and its concept instances

Concept Name	Concept Instances
Multiple Myeloma	Multiple Myeloma
	Myeloma
	Plasma Cell Myeloma
	Myelomatosis
	Plasmacytic myeloma

The UMLS Semantic Network organizes the Metathesaurus concepts into categories called semantic types (United States National Library of Medicine, 2004). There are currently 135 semantic types.

The MetaMap Transfer (MetaMap) application (United States National Library of Medicine, 2005b) maps biomedical text to concepts stored in the Metathesaurus as follows. The text-to-concept mapping in the MetaMap application is done through a natural language processing approach. Sentences are first identified, and then noun phrases are extracted from each sentence. MetaMap proceeds through several stages to map a noun phrase to one or more concepts. Term variants of the phrase are generated, candidate concepts are generated, and a scoring process is done for each candidate concept. The highest scoring concept is then selected as the concept for the phrase. It is possible a noun phrase can map to more than one concept. In this case, no disambiguation step is performed, and MetaMap returns multiple concepts. Figure 4 shows an example of MetaMap mapping of the phrase *protein kinase CK2*. The output shows the phrase, the concept candidates preceded by their score ("Meta Candidates"), and the final mapping of the phrase ("Meta Mapping"). In the example, there are six candidate mappings, shown in descending score order. The final mapping takes the highest scoring candidate, shown as "Meta Candidate (1000)" in Figure 4. In cases where a phrase cannot be successfully disambiguated, it is possible for MetaMap to generate a final mapping consisting of more than one concept. Finally, MetaMap indicates the semantic type for the selected concept, shown as a text description in square brackets next to the concept text description in Figure 4.



```
Phrase: "protein kinase CK2."
Meta Candidates (6)
  1000 protein kinase CK2 (casein kinase II) [Amino Acid, Peptide, or Protein,Enzyme]
   901 PROTEIN KINASE [Amino Acid, Peptide, or Protein,Enzyme]
   827 Kinase (Phosphotransferases) [Amino Acid, Peptide, or Protein,Enzyme]
   827 Protein (Proteins) [Amino Acid, Peptide, or Protein,Biologically Active S
ubstance]
   827 Protein NOS (Protein measurement) [Laboratory Procedure]
   827 CK2 [Laboratory Procedure]
Meta Mapping (1000)
  1000 protein kinase CK2 (casein kinase II) [Amino Acid, Peptide, or Protein,Enzyme]
```

Figure 4: MetaMap Transfer mapping of the phrase
protein kinase CK2

2. LITERATURE REVIEW

The review of literature is divided into four sections. The first two sections describe semantic annotation. Domain-independent semantic annotation is first described followed by semantic annotation for biomedical texts. The last two sections describe two recent text summarization methods: lexical chaining of terms and term frequency. These two methods are adapted in this research to use concepts rather than terms. In addition, recent work in biomedical text summarization and question-answering systems is presented.

2.1 Semantic Annotation

Semantic Annotation is a method for providing machine-understandable information based on meaning. One way to provide meaning by creating ontologies, and then linking information within a document to specifications contained in the ontology using a markup language (Berners-Lee et al., 2001). Ontologies are conceptualizations of a domain that typically are represented using domain vocabulary (Chandrasekaran et al., 1999). Semantic annotation is the process of mapping instance data to an ontology. Benefits of adding meaning to the Web include: query processing using concept-searching rather than keyword-searching (Berners-Lee et al., 2001); custom Web page generation for the visually-impaired (Yesilada, Harper, Goble, & Stevens, 2004); using information in different contexts, depending on the needs and viewpoint of the user (Dill et al., 2003); and question-answering (Kogut & Holmes, 2001).

2.1.1 Semantic Annotation for General Text-based Documents

Manual annotation can be done using tools such as Semantic Word (Tallis, 2003), which provides an environment for authoring as well as marking up documents from within a single interface. However, manual approaches suffer from several drawbacks. Human annotators can provide unreliable annotation for many reasons: complex ontology schemas, unfamiliarity with subject material, and motivation, to name a few (Bayerl, Lüngen, Gut, & Paul, 2003). It is expensive to have human annotators markup documents (Cimiano, Handschuh, & Staab, 2004). A human annotator may not consider using multiple ontologies (Dingli et al., 2003). Documents and ontologies can change, requiring new or modified markup, leading to document markup maintenance issues (Dingli et al., 2003). Finally, the volume of existing of existing documents on the Web can lead to an overwhelming task for humans to manually complete (Kosala & Blockeel, 2000). For all these reasons, manual efforts have been identified as a "knowledge acquisition bottleneck" (Maedche & Staab, 2001).

Semantic Annotation Platforms (SAPs) are systems for performing semi-automatic semantic annotation. Semi-automatic systems, rather than completely automatic systems, are used because it is not yet possible to automatically identify and classify all entities within source documents with complete accuracy (Popov et al., 2003). There are many advantages of semi-automatic annotation, such as providing document volume scalability by reducing or potentially eliminating the human workload (Dill et al., 2003), and providing annotation services where the source document is stored separately from its corresponding annotations (Dill et al., 2003). SAPs vary in their architecture, information extraction tools and methods, initial ontology, amount of manual work required to perform annotation, performance and other supporting features, such as storage management of ontologies, knowledge bases, and annotations. Some SAPs were designed for a specific domain, but usually can be adapted to fit new domains.

Figure 5 shows the general abstraction layer of a SAP. The Application layer is responsible for providing an end-user interface to the annotation services provided by a SAP. Examples include facilities for (a) annotating a document or document set and then potentially confirming the annotations before committing them; (b) providing a query interface for searching annotations; and (c) providing a user interface for configuring the information extraction component. The Upper Interfaces layer is primarily the application programming interface (API) layer. A set of programmatic interfaces are described in this layer. Applications call the defined APIs in order to perform actions on behalf of an application. The APIs can be quite numerous, covering annotation, information extraction, search, storage management, and many other provided services. The Upper Interface APIs are designed to shield the applications from changes in the Lower Interface. The Lower Interface contains the actual components that perform work for an application. The Upper Interface will remain consistent to an application, but the Lower Interface is expected to change based on the various components used. For example, the Information Extraction component may switch from a pattern-based tool to a statistical tool, and it is unlikely the programmatic interface is the same for both. The Upper Interface implementation will need to change to accommodate the various Lower Interface components. Finally, the Storage Layer is designed to provide storage and storage management facilities for storing long-term data such as annotations and knowledge bases.

Examples of existing annotation platforms include AeroDAML (Kogut & Holmes, 2001), Armadillo (Dingli et al., 2003), MnM (Vargas-Vera et al., 2002), MUSE (Maynard, 2003), Ont-O-Mat (S. Handschuh, Staab, & Ciravegna, 2002), and SemTag/Seeker (Dill et al., 2003). The platforms are primarily distinguished by (a) the features offered, (b) the information extraction method used to find entities within documents, and (c) whether or not they are extensible. Features offered by SAPs include ontology and knowledgebase management (storage, editors), access APIs, annotation storage to allow multiple ontologies/annotations per document, and information extraction methods.

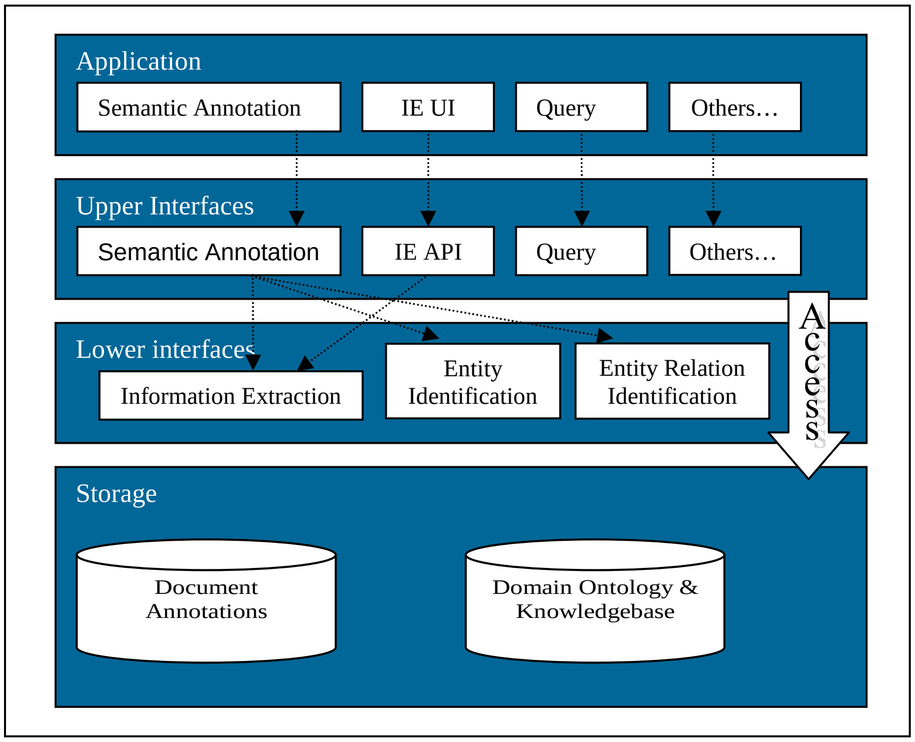


Figure 5: General architecture of a semantic annotation platform

Semantic annotation requires an ontology in order to perform concept instance mapping. Ontologies are usually architected using levels, such as upper and lower. The upper ontology consists of general concepts, while the lower ontology has a deeper specialization of the upper ontology concepts (Missikoff, Navigli, & Velardi, 2002). Some semantic annotation platforms place the responsibility on the user for constructing an initial ontology. Examples include MUSE (Maynard, 2003) and Ont-O-Mat (S. Handschuh et al., 2002). Other platforms provide an initial ontology as part of their development. The KIM platform provides an ontology called KIMO that is designed to provide a minimal open-domain ontology, and is based on OpenCyc, WordNet, DOLCHE and other upper-level resources (Popov et al., 2003). KIMO is composed of approximately 250 classes and 100 attributes and relations, and the specialization of classes is derived from an analysis of a corpus of general news (Popov et al., 2003). The Seeker platform uses TAP, which is a shallow knowledgebase that contains information about a broad range of popular culture subjects, such as movies, sports, and so forth (Dill et al., 2003). The TAP knowledgebase has about 72,000 labels that are used to tag instances found in documents. The MnM platform uses a hand-crafted ontology called KMi (Knowledge Management Institute) (Vargas-Vera et al., 2002). The AeroDAML platform uses the commercial product Aerotext, and utilizes an upper-level ontology based on WordNet, while the lower-level ontology uses the common knowledgebase of AeroText (Kogut & Holmes, 2001). Armadillo provides an example of a platform where the initial ontology is very light weight, consisting of an address-book type of ontology where members of a computer science department are discovered and populate address information, such as name, phone number, address, and so forth (Dingli et al., 2003).

2.1.1.1 Classification of Platforms

Current semantic annotation platforms use several methods for information extraction (IE) from Web documents. Figure 6 shows a hierarchical classification of annotation platforms, and this classification can be used to organize the platforms performing semantic annotation. While semantic annotation platforms have many aspects, the information extraction approach currently used to find entities within text has the most impact on the effectiveness of the platform. For this reason, the IE approach of each platform is used to organize the platforms. As semantic platforms develop, it is anticipated that the classification structure will adapt to newer approaches as well.

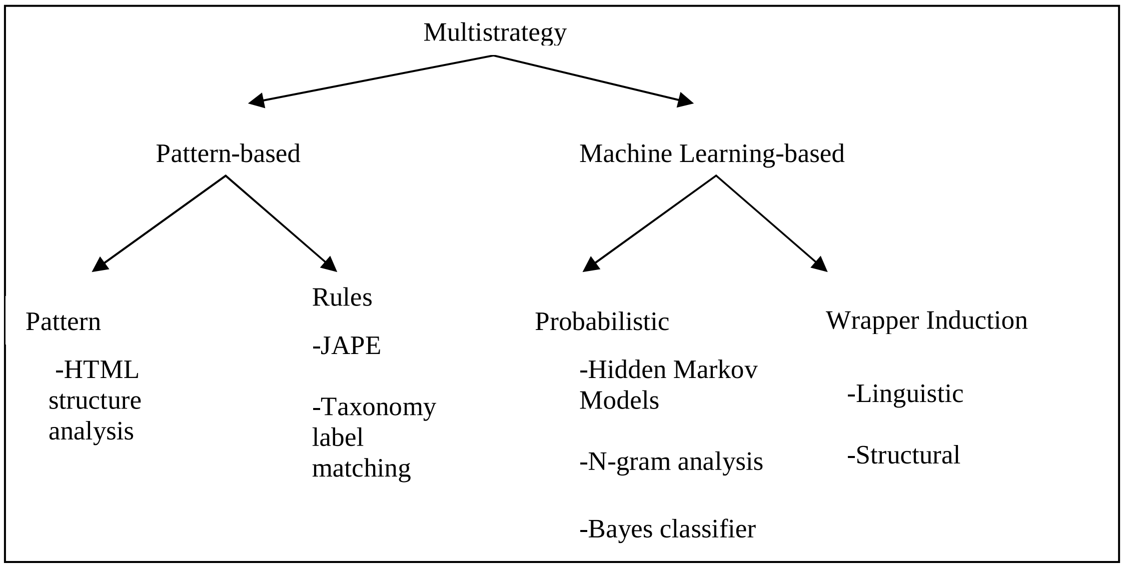


Figure 6: Classification of semantic annotation platforms

The top-level approach is multi-strategy, which uses a combination of the lower level approaches. A platform using a multi-strategy approach is able to adapt its IE methods based on the text it is processing in order to obtain the best results. The multi-strategy approach uses a high-level identification of text genre, and then executes the appropriate IE methods. This is in contrast to lower-level approaches using newer IE algorithms such as LP^2 (Ciravegna, 2001) in platforms such as KIM (Popov et al., 2003), which are able to use machine-learning to perform rule induction using both structural and linguistic information. No semantic annotation platform to date is using a complete multi-strategy approach incorporating both pattern and machine-learning approaches. The MUSE system comes the closest by using text features and then conditionally executing rules based on the text features (Maynard, 2003).

The two primary lower levels are pattern-based methods and machine-learning methods. Pattern-based methods are systems composed of manual rules. The rules are typically hand-crafted rules that define how entities can be found in text (Maynard, 2003). Examples of such systems are AeroDAML (Kogut & Holmes, 2001), MUSE (Maynard, 2003) and SemTag/Seeker (Dill et al., 2003). A limiting factor on the scalability of such systems is that the manual rule generation process can be maintenance intensive. Each time a data source changes, the pre-defined rules may also need to be changed. Machine-learning approaches use pre-annotated examples to learn how to identify entities. Rules are learned automatically, and this type of rule learning is currently used in platforms using the Amilcare toolkit (University of Sheffield, 2002), which implements the LP^2 algorithm (Ciravegna, 2001). Examples of systems using rule learning are Ont-O-Mat (S. Handschuh et al., 2002) and MnM (Vargas-Vera et al., 2002).

Hidden Markov Model (HMM) is an example of another machine-learning approach that can also be used. HMM is not currently being used by any of the semantic annotation platforms as an information extraction method.

2.1.1.1.1 Pattern Discovery

Patterns are also a widely-used technique in semantic annotation platforms. Pattern discovery works by taking a few seed samples, finding entities based on the patterns, expanding the seed samples with patterns from the new entities found, and repeating the process until no more instances are found, or the user stops the iterative process (Brin, 1998). Patterns can exploit known linguistic patterns, such as Hearst patterns (Hearst, 1992), to find entities, as is done in the Ont-O-Mat using PANKOW platform (Cimiano et al., 2004). The Ont-O-Mat platform has been updated to replace the Amilcare component with a pattern-based component, called PANKOW. The Ont-O-Mat platform (S. Handschuh et al., 2002) demonstrates the usefulness of an extensible semantic annotation platform, where components can be replaced without losing or duplicating the features already available (Cimiano et al., 2004). The Armadillo platform is also example of a platform that uses a small set of initial seeds to begin a pattern discovery process.

2.1.1.1.2 Rules

Rules can be manually-generated, as MUSE (Maynard, 2003) does with the JAPE grammar (Cunningham, Maynard, & Tablan, 2000), or they can be generated with machine-learning techniques. In this case, rules are considered the rules that are a subset of the pattern classification in Figure 6. The difference is the rules are initially manually specified by the user. Rules can take many forms. In the Seeker platform, the rules are labels (Dill et al., 2003). SemTag, the semantic tagging component of Seeker, uses the labels stored in the knowledgebase to find instances of ontology concepts, and then uses statistical likelihood to determine where in the ontology tree an instance is most likely to be contained (Dill et al., 2003). The process is not completely automatic, however, as an initial set of training data must be provided. The authors report 700 entries as the initial size of the training set (Dill et al., 2003). In the MUSE platform, rules are written using the JAPE grammar (Maynard, 2003). The MUSE platform is an interesting rule-based system because it can adapt the rules used in entity identification depending on text attributes, such as language, document type, document source, and so forth (Maynard, 2003). The result is a rule-based platform that performs competitively with machine-learning based platforms (Maynard, 2003).

2.1.1.1.3 Machine Learning

Platforms based on machine learning are divided using the machine-learning method, which are currently two approaches: probabilistic and wrapper induction. The more common of the two approaches is wrapper induction. It is anticipated that future research in semantic annotation will take advantage of machine-learning based approaches, because they help to relieve the manual effort required in building rules, which is the primary drawback of pattern-based approaches. For example, the Rainbow system is a domain-specific annotation system developed for aggregating product information from multiple web sites, and then providing search over the aggregated information (Svab Ondrej, Labsky Martin, Svatek Vojtech, 2004).

2.1.1.1.3.1 Probabilistic

Probabilistic methods use algorithms such as Hidden Markov Models (HMMs) to perform information extraction. For example, the DATAMOLD tool (Borkar, Deshmukhy, & Sarawagiz, 2001) uses HMMs for information extraction, but it has not yet been integrated into a semantic annotation platform. The Seeker platform uses a probabilistic approach to help eliminate mis-classification caused by its simple tagging approach within the SemTag component (Dill et al., 2003). SemTag has a pool of approximately 72,000 labels that it uses to find entity instances within text. It is likely that some of the labels will be duplicated but contained in different parts of the ontology. SemTag incorporates an algorithm as part of its tagging process to determine the probability of a particular label belonging to a particular class in the ontology. In the Rainbow project, where product catalog information from a specific domain is extracted from multiple sites, the information extraction portion project uses Hidden Markov Models as its primary method. While the performance is competitive with other approaches, the resulting system developed from Rainbow has not yet been incorporated into a general purpose semantic annotation platform. The most likely reason is that the generated model is domain-specific (Svab Ondrej, Labsky Martin, Svatek Vojtech, 2004).

2.1.1.1.3.2 Wrapper Induction

A wrapper is a function from a page to the set of tuples it contains (Kushmerick, Weld, & Doorenbos, 1997). Wrappers are used when a repeatable structure exists to extract information from. Many web sites have pages generated using from a back-end database, and the pages generated follow a common template. The purpose of wrappers is to reverse the page generation process in order to retrieve the original database tuples. Wrappers can be hand-crafted, or they can be learned. Manual wrappers require the user to mark areas of interest within a document. The machine can then extract entities from documents with a similar structural format as the manually marked-up document (Vargas-Vera et al., 2002). Kushmerick (1997) defined a method for performing wrapper induction, where the wrappers are automatically learned from example query responses from a data source. Wrappers are most effective when the data is presented in a structured format, such as product catalogs (Dingli et al., 2003).

Wrappers can also be linguistic-based, where the wrapper induction process discovers linguistic rules for identifying entities (Vargas-Vera et al., 2002). Amilcare implements the LP^2 algorithm (Ciravegna, 2001), which performs rule induction using both linguistic and structural information (Ciravegna, 2001). It is intended to provide a combination of IE approaches such as wrapper induction (Kushmerick et al., 1997) and linguistic approaches from the natural language processing community. The Amilcare toolkit is used by several platforms, such as MnM (Vargas-Vera et al., 2002), and Ont-O-Mat (S. Handschuh et al., 2002).

2.1.1.1.4 Multi-strategy

A multi-strategy approach uses a combination of both machine-learning and pattern-based approaches. No surveyed SAP uses the multi-strategy approach. However, this approach could be used by a platform which adapts its processing based on the document genre or specific text features. Such adaptive processing would apply the most effective processing for each type of document. The closest SAP that performs adaptive processing is the MUSE system (Maynard, 2003). MUSE uses a pipeline approach to identify text features that can be used to perform conditional processing. MUSE uses the JAPE grammar (Cunningham et al., 2000) to define rules to perform semantic annotation, and the JAPE rules are fire conditionally, based on text features. The conditional processing approach used by MUSE makes it competitive with machine-learning based approaches (Maynard, 2003).

Table 2 shows the author-reported performance of various platforms, with the exception of AeroDAML, Ont-O-Mat using Amilcare and SemTag, whose authors did not provide complete performance information. The standard measures of Precision, Recall, and F-measure, taken from the information retrieval field, were used by the remaining SAP authors in determining annotation effectiveness. In the general definition of recall and precision shown below, *accurate* and *inaccurate* refer to annotations generated semi-automatically by a SAP, while *all* refers to all annotations generated by a human annotator.

$$\textit{Annotation Recall} = \frac{\textit{accurate}}{\textit{all}}$$

$$\textit{Annotation Precision} = \frac{\textit{accurate}}{\textit{accurate} + \textit{inaccurate}}$$

F-measure is the harmonic mean of precision and recall. The highest performing machine learning-based platform is MnM. For pattern-based platforms, MUSE is best.

The worst performing is Ont-O-Mat using PANKOW. PANKOW is a recent effort to use unsupervised learning in a pattern-based system, and performance improvements are expected as the system develops further (Cimiano et al., 2004).

Table 2: Semantic annotation platform information extraction methods and performance measurements

Platform	IE Method	Precision	Recall	F-Measure
Armadillo	Pattern Discovery	91	74	87
KIM	Manual Rules	86	82	84
MnM	Wrapper Induction	95	90	n/a
MUSE	Manual Rules	93	92	93
Ont-O-Mat using PANKOW	Pattern Discovery	65	28	25
SemTag	Semi-automatic Rules	82	n/a	n/a

Semantic Annotation Platforms were developed to provide a level of automation to the semantic identification of text within documents, and to also overcome the limitations of manual annotation, such as annotator motivation and domain knowledge (Bayerl et al., 2003), changing and multiple ontologies, and providing multiple perspectives (Dill et al., 2003).

Several semantic annotation platforms currently exist, distinguished primarily by their annotation method, as that component has the largest impact on the effectiveness of semantic annotation. The two primary approaches are pattern-based and machine learning-based. Machine learning algorithms often perform more effectively than pattern-based methods, but the MUSE system shows that a rule-based system using conditional processing can perform as well as a machine learning system (Maynard, 2003).

Applications making use of annotations generated by semantic annotation platforms are beginning to appear. These applications are information retrieval by semantic entity rather than by keyword, custom web-page generation based on different user needs and perspectives, question-answering systems, and visualization of a domain.

2.1.2 Semantic Annotation for Biomedical Text-based Documents

Most work in semantic annotation for biomedical text is performed to support semantic indexing/retrieval and data mining of biomedical texts (Aronson, 2001a). Concepts are identified in the text and then indexed. Users can then retrieve the biomedical texts using the biomedical concept rather than searching by terms. Indexing biomedical text can be done in an offline mode. There is no requirement to index a text while the user is waiting for a response. Indexing text is also primarily interested in finding all possible concepts within a text. For example, the phrase *lung cancer* can map to three concepts: {*lung, cancer, lung cancer*}. Some biomedical annotation systems, such as MetaMap (Aronson, 2001a), find a single best match for phrases within a source text in order to understand the author's intentions. For example, *lung cancer* maps to the single concept {*lung cancer*} rather than the three distinct concepts listed above. The example phrase is describing the single concept lung cancer. If the two terms were split and mapped to two different concepts, {*lung*} and {*cancer*} the meaning would change. Instead of the single concept {*lung cancer*} the meaning would change to lung and any type of cancer. While most of the work identified here has been designed for semantic indexing, the methods used to find concepts in source text, as well as the methods used to evaluate the resulting system, are important to review.

2.1.2.1 General Approach

A common approach to most systems is to use the following method:

1. Construct a unit of analysis by generating subsets of words in the source text (e.g., phrase, sentence).
2. (Optional) Normalize the source text unit using UMLS (National Library of Medicine, United States, 2006b) by (a) removing possessives, (b) replacing punctuation with spaces, (c) removing stop words, (d) converting words to lower-case, (e) breaking a string into constituent words, and (f) sorting words into alphabetical order.
3. For each word in the input phrase, build a set of all concepts containing the word.
4. Find the intersection of the concept sets.
5. (Optional) Find the best matching concept based on the common word membership between the source text and concept text.

The unit of analysis is usually a phrase. Noun phrase are typically used because they have more content information (Elkin et al., 1988). Other unit of analysis include sentences (W. R. Hersh & Greenes, 1990), and unordered terms (Zou et al., 2003). Each unit of analysis has pros and cons. Phrases lose contextual value when a single concept is spread across multiple phrases (P. Nadkarni, Chen, & Brandt, 2001), (Zou et al., 2003). Sentences overcome the loss of context when using phrases, but suffer from several new problems (P. Nadkarni et al., 2001): (a) finding invalid concepts when using all permutations of words in the sentence, and (b) since more words are in a sentence than in a phrase, concept identification becomes more computationally complex. It is also possible to eliminate natural language processing and treat the entire text as a set of independent terms. The IndexFinder system (Zou et al., 2003) is an example of this approach, which overcomes the computational limitations of the sentence approach. A problem with this approach is that concept cannot be linked back to their source (e.g., phrase or sentence), since the words are permuted throughout the text. Another problem is the over-generation of concepts, which the authors control with various filters, such as semantic types and term ranges. The unit of analysis for each of the reviewed systems, as well as the method used to identify the unit of analysis, is shown in Table 3. Table 4 identifies the methods used to score candidate phrases.

Four types of matches between the terms of a source text and UMLS concept phrase have been defined in the literature (Aronson, 1996):

- *None*: there is no match of terms.

- *Simple*: there is an exact match between the terms in the text of the source and the UMLS concept text.

- *Partial*: one or more terms do not match exactly.

- *Complex*: the original source phrase is divided into two or more sets of terms which are then mapped to distinct concepts.

The most common types of matches are Simple and Partial. The last column in Table 3 lists for each system whether the system supports Simple, Partial or both. Typically the match types form a simple hierarchy. All systems support None and

Simple. Systems which support Partial also support None and Simple, and systems supporting Complex support None, Simple, and Partial as well.

The identification of a phrase is typically done using natural language processing (NLP), where a part-of-speech tagger is first executed over a sentence, and then using parts of speech to build a noun phrase. Other approaches include using NLP Heuristics which do not use strict NLP processing (P. M. Nadkarni, 1997), moving window where all combinations of a sentence words are generated to build a phrase (Wollersheim, Rahayu, & Reeve, 2002), and using pre-defined block text sizes (W. R. Hersh & Greenes, 1990).

Table 3: Attributes of several biomedical annotation systems

System	Unit of Analysis	Phrase Identification Method	Mapping
SAPHIRE (W. R. Hersh, 1990)	Sentence	Text block	Simple, Partial
MetaMap Transfer (Aronson, 1996, 2001)	Phrase	NLP	Simple, Partial, Complex
SENSE (Zieman & Bleich, 1997)	Phrase	User-specified Query	Simple
Concept Locator (P. Nadkarni et al., 2001)	Phrase	NLP Heuristics	Simple, Partial
Dynamic Taxonomy (Wollersheim et al., 2002)	Phrase	Moving Window	Simple
PhraseX (Srinivasan et al., 2002)	Phrase	NLP	Simple
KnowledgeMap (J. C. Denny et al., 2003)	Phrase	NLP	Simple, Partial
IndexFinder (Zou et al., 2003)	Unordered Terms	All words, excluding stop words	Simple, Partial

Table 4: Phrase scoring methods of several biomedical annotation systems

System	Phrase Scoring Method
SAPHIRE (W. R. Hersh, 1990)	Combines measures of term overlap, term proximity, and length of term matches
MetaMap Transfer (Aronson, 1996, 2001)	Combines several measures: *Centrality* – is source phrase head term used in concept phrase *Variation* – how far the source phrase's term variant is from the concept phrase's term *Coverage* – overlap between source phrase and concept phrase terms, ignoring gaps *Coherence* – find term sequence overlaps between source phrase and concept phrase
SENSE (Zieman & Bleich, 1997)	Translates source and concept phrase to low-level semantic factors, then performs exact matching of the semantic factors
Concept Locator (P. Nadkarni et al., 2001)	Sub-divide phrase & look for exact match
Dynamic Taxonomy (Wollersheim et al., 2002)	Normalize source phrase using UMLS tools; find exact match
PhraseX (Srinivasan et al., 2002)	Exact match
KnowledgeMap (J. C. Denny et al., 2003)	Exact match, followed by variant-generation and re-match
IndexFinder (Zou et al., 2003)	Find all matching words, regardless of location

2.1.2.2 Biomedical Semantic Annotation Platforms

Multiple systems have been previously built to implement various ways to perform concept annotation of biomedical texts. In the following subsections, several existing systems which perform mapping from a source text to UMLS concepts are reviewed. The focus is on the algorithmic approach, evaluation method, and failure analysis, when provided, for each system.

2.1.2.2.1 Concept Locator

The Concept Locator (P. Nadkarni et al., 2001) system uses a phrase-based approach to identify concepts for indexing and retrieval.

Algorithm: Input phrases from the source text are identified using the IBM Intelligent Text Miner's Feature Extraction Tool. Concept Locator uses a subset of UMLS. The subset removes redundant concepts, concepts unrelated to clinical medicine, concepts having eight or more terms, and suppressible synonyms (synonyms which result in ambiguity, such as *complications* being a synonym of *Complications Specific to Antepartum or Postpartum* (Aronson, 2001a)). To match concepts to the input phrase, the algorithm steps are as follows: 1. The input phrase is first stripped of stop words and words which do not appear in the UMLS. Any resulting phrase over five words is rejected. 2. An exact match of concept words to input phrase words is then attempted by using all words in the input phrase to match all concepts having the same set of words, regardless of word order. If the match results in mapping the input phrase to a distinct concept, the concept is output and the mapping algorithm stops. 3. If no exact match is found, two cases are handled. In the first case, a phrase consists of a single word. If the word is defined as ambiguous by UMLS, no further matching is attempted. In the second case, a phrase consists of two or more words. Subsets of words in the phrase matching is performed by finding complete concept word matches for all combinations of words {N-1, N-2,…2} where N is the total number of words in the source phrase and the word subset concept matching results in a single distinct concept. Words which are not matched are sent back to algorithm step #2. If the subset matching does not result in any concept matches, individual words in the input phrase are matched to single-word concepts. 4. If an input phrase from step 2 or 3 matches several concepts, concept disambiguation is performed by stemming both the words in the concept and the input phrase, and then comparing each for an exact match of the stemmed words. The disambiguation step is on a best-effort basis; that is, it is possible for disambiguation to fail and still result in mapping an input phrase to multiple concepts.

Evaluation: A manual evaluation of the concept mapping output was performed using a corpus of 24 biomedical documents (12 each of discharge summaries and surgery notes). A domain expert manually identified UMLS concepts in the corpus texts, and then compared the Concept Locator concept mappings to a manually-annotated corpus. It was found Concept Locator matched concepts correctly for 76.3% of the phrases.

Failure Analysis: The authors identify three categories of concept-mapping failures: (a) use of noun phrases, (b) UMLS content, and (c) matching algorithm. The use of noun phrases cannot locate concepts spread across two or more noun phrases, resulting in matching two or more concepts rather than a specific single concept. Also, spelling, grammatical errors, and proper names can confuse natural language parsers. UMLS content influences performance because it is incomplete. UMLS does not list all possible term variations, may have missing biomedical concepts, and has redundant concepts. The matching algorithm has built-in limitations, such as five-word maximum phrase length, which causes some phrases not to be mapped.

2.1.2.2.2 Dynamic Taxonomy

The Dynamic Taxonomy system (Wollersheim et al., 2002) was designed to automate the construction of indexes to be combined with ontologies for biomedical text retrieval.

Algorithm: The Dynamic Taxonomy (DT) uses two different methods for extracting phrase: NLP and moving window. The NLP approach uses part-of-speech taggers to construct phrases. DT studied three different part-of-speech taggers. The moving window approach is an algorithm the authors propose which produces input phrases by taking a combination of *N* consecutive words. A word is considered to be a text string with more than three characters. The authors studied window sizes of *N*=1 to 6. UMLS concept matching was performed by normalizing the input phrase text using the UMLS lexical tool LVG (National Library of Medicine, United States, 2006b). The UMLS concept phrase list is then scanned to find an exact match with the normalized input phrase.

Evaluation: The DT system was evaluated by first having a domain expert identify all possible concepts found in a six-page biomedical text having 2,982 words. The domain expert's annotations were then compared against the output of the DT matching process. Nine variations of the system were evaluated based on the phrase selection method used: three variations used part-of-speech taggers, and six variations used sliding window sizes of one to six words. Precision and recall were measured. The best-performing variation using part-of-speech tagging had a precision of 28% and recall of 19.5%. The recall for the six variations using sliding windows was approximately 11% while precision ranged from approximately 47% to 51%.

Failure Analysis: The authors believe that UMLS contains words which do not contain much medical content, and therefore if eliminated would result in better precision/recall scores by eliminating false positive mappings. In addition, they found that the domain expert made maximum use of the index, and in doing so suggested multiple correct concepts for a single phrase. The multiple correct concepts identified by the domain expert went from highly-general concepts to highly-specific, and often did not include words from the original source text. Finally, the authors suggested using meaning from previous paragraphs to identify meaning in subsequent paragraphs.

2.1.2.2.3 KnowledgeMap

The KnowledgeMap system (J. C. Denny, Irani, Wehbe, Smithers, & Spickard, 2003) is designed to identify concepts in biomedical educational texts for indexing.

Algorithm: KnowledgeMap's concept identification component is known as KnowledgeMap Concept Identifier (KMCI) and consists of three phases: sentence identification, concept identification, and concept disambiguation. Sentence and noun phrase identification is done by using a natural language parser. Concept identification is done by taking a noun phrase and finding a set of UMLS concepts which match the noun phrase. If no concept match is found, then variants of the terms in the noun phrase are generated and the matching process retried. Nearby noun phrases linked by grammar (such as conjunctions and prepositions) are attempted to match concepts with the current noun phrase. This overcomes the phrase-based problem of identifying concepts occurring over two or more noun phrases. In addition, KMCI will distribute modifying adjectives. For example, “large and small intestine” is converted to “large intestine and small intestine” (J. C. Denny et al., 2003). If multiple concepts for a phrase are located, disambiguation is attempted. Two resources are used to perform disambiguation. The first resource is an externally-maintained list of concept co-occurrences which occur in MEDLINE abstracts. The second resource is a dynamic list of concepts with an exact match to a source text noun phrase. Disambiguation is performed by discarding concepts which are not similar to either the text’s dynamic list of exact-match concepts or which do not co-occur with concepts in MEDLINE abstracts.

Evaluation: Evaluation of KnowledgeMap was done by first having two domain experts manually annotate five biomedical educational texts with important terms and phrases. Each text was then split into its component sentences, and each sentence was then submitted to KMCI and MetaMap, a state-of-the-art concept matching system produced by the National Library of Medicine (see Sections 2.1.2.2.5 and 4.1.2.2.3). The domain experts then determined if the concepts for the identified terms and phrases were accurate or not. Precision and recall are the evaluation measures. Recall is defined as the number of important terms and phrases identified. Precision is defined as the number of correctly identified concepts. The recall is measured at 86% and precision at 92%, which outperforms MetaMap, which has a recall of 81% and a precision of 89% (J. C. Denny, Smithers, Miller, & Spickard, 2003).

Failure Analysis: The authors identified nine primary reasons for failure. The biggest failure of concept matching (62% of failures) in KMCI is the lack of a corresponding concept in UMLS. Other sources of failure include acronym/abbreviation/hyphen handling, and overmatching. Overmatching occurs when no exact match exists between the input phrase and UMLS phrases, and additional words in UMLS concept phrases are used to find a match with the input phrase. The disambiguation stage was responsible for only 2% of failures, while accounting for 18% of successful matches.

2.1.2.2.4 IndexFinder

IndexFinder (Zou et al., 2003) is a concept matching system designed for real-time indexing applications.

Algorithm: IndexFinder uses a series of in-memory table structures to find all possible concepts by using all combinations of words in the text. IndexFinder treats the terms within a text independently regardless of order. The terms in a text are first normalized by (a) lowercasing them, (b) removing unknown acronyms/abbreviations, stop terms, and terms unknown to UMLS, and (c) mapping remaining terms to their base form. Next, for each unique term, all UMLS concept phrases containing the term are retrieved. Each retrieved phrase maintains its length and the count of terms matched so far. After all terms have been evaluated, the retrieved phrases are then evaluated based on their counts. Concepts are extracted for indexing where concept phrases have all matching terms with the source phrase.

Evaluation: The authors used a corpus of 5,783 patient reports totaling 10.8MB in size and report a text processing speed of 42.7KB/second. No evaluation was reported on the accuracy of the concepts extracted, although one was planned to evaluate the number of false positives and false negatives.

Failure Analysis: No failure analysis was reported. However, six filters are available to restrict the output. The filters focus on two aspects: (a) restricting the location of text, and (b) restricting the number of concepts generated. In order to generate concepts based on smaller units (i.e., not the full text), the term length can be restricted to less than six terms or less than 11 terms, as well as all terms. In addition, the range filter restricts terms to a certain distance, such as ten terms. Terms outside of the range are not counted in concept matching. The effect is to index only concepts occurring with terms a certain distance from each other. To restrict the generation of concepts, concept subsets can be removed if they are contained within a larger concept. In addition, concepts can be included only if they appear within certain UMLS semantic types.

2.1.2.2.5 MetaMap

MetaMap (Aronson, 2001a) is a concept matching system produced by the National Library of Medicine. MetaMap is considered a state-of-the-art system for concept matching (J. C. Denny et al., 2003). MetaMap was originally developed to support indexing applications, but has also been used in data mining, decision support, and patient record applications.

Algorithm: The MetaMap algorithm consists of five steps. Parsing of the source text by a natural language parser is first done to form noun phrases. Variant generation on each noun phrase is done to find a term’s acronyms, abbreviations, synonyms, inflectional and spelling variants. UMLS concept phrases containing the term or its variants are then identified. Each concept phrase is then evaluated to find the concept phrase which best matches the noun phrase from the source text. Concept phrase evaluation uses four metrics to measure similarity with a source phrase: centrality, variation, coverage, and cohesiveness (Aronson, 1996). Centrality measures if the phrase head term is used in the concept phrase. Variation is a distance value that determines how far the term variant is from the term, based on the variant type. Coverage measures how much the terms between the concept phrase and source noun phrase overlap, ignoring term gaps. Cohesiveness measures similarly to coverage, but factors in sequences of terms which co-occur in the concept phrase and the source noun phrase. The four scores are computed into a weighted average. The coverage and cohesiveness scores are weighted twice as heavily as centrality and variation. The final stage forms a final mapping of the source text, which may result in mapping a source text into one or more concepts. The final mappings are shown on a scale of 0 to 1000, with 1000 being an exact mapping. No explicit disambiguation step is performed.

Evaluation: Several evaluations of MetaMap have been performed to date. The National Library of Medicine (NLM) performed a failure analysis of MetaMap by using a short evaluation (Divita, Tse, & Roth, 2004). Five annotators without domain expertise annotated two documents from genetic information Web site. Two documents were chosen as the evaluation corpus size due to (a) the labor intensive activity of annotating documents, and (b) mediating conflicting concept mappings between annotators. The two documents were processed by MetaMap and the MetaMap concept output was then compared to the manually-generated annotations. The recall score for the two documents was measured at 53%; precision was not calculated.

The University of Washington (UW) also performed an evaluation of MetaMap (Pratt & Yetisgen-Yildiz, 2003). They used six domain experts to identify concepts within a corpus of 60 texts. The study used precision and recall as measures, reporting a recall of 53% for exact matches and 93% for partial matches. Precision is reported as 28% for exact matches and 55% for partial matches.

Failure Analysis: The NLM and UW evaluations also concluded with a failure analysis. The NLM study identified thirteen sources of failure. The most common failures resulted from (a) needing implicit knowledge to map a term, (b) the use of broader concepts by annotators because the UMLS Metathesaurus is incomplete, and (c) co-reference resolution. The UW study identified four types of failures: incorrect splitting of a noun phrase, concept ranking and identification failed, and noun phrase breaking which changed the meaning of the phrase.

2.1.2.2.6 PhraseX

The PhraseX program performs noun extraction (National Library of Medicine, United States, 2004). A study of UMLS phrases in MEDLINE abstracts used the output of PhraseX to perform simple concept matching (Srinivasan, Rindflesch, Hole, Aronson, & Mork, 2002). A newer application uses PhraseX as a component of a larger biomedical text indexing application (National Library of Medicine, United States, 2006a).

Algorithm: Noun phrase identification is done by first tagging a sentence's words with their part-of-speech, and then using the barrier word method (Tersmette, Scott, Moore, Matheson, & Miller, 1988) to delimit a phrase. In this case, tagger output delimits a phrase based on its part-of-speech. For example, a verb ends one phrase and begins another. PhraseX defines three types of successively complex phrases: (a) simp – phrases with a head noun, (b) macro – phrases with a preposition to the right, and (c) mega – using a finite word to divide the sentence into two phrases: one phrase before the verb and one phrase after the verb. Once a phrase is identified, the mapping is done in one of three ways: (a) exact match, (b) exact match with lower casing done to all terms in the phrase, and (c) exact matching done after UMLS normalization (lower casing, possessive removal, inflectional variation, and term sorting, among others).

Evaluation: The PhraseX evaluation was performed by downloading all MEDLINE abstracts from PubMed as of the Fall of 2001. The noun phrases were extracted resulting in approximately 175 million unique phrases. The authors report that 63% of the phrases were simp phrases, 16% were macro phrases, and 21% were mega phrases. Each unique phrase was then attempted to be mapped with one of the three matching methods. The result is 6.5% for exact match, 22.5% for lower case match, and 30% for normalized match.

Failure Analysis: Five types of failures were identified.

1. The UMLS Metathesaurus contains strings which are not useful for mapping. Examples are long descriptive strings and strings containing codes for what are known as Logical Observations Identifiers, Names and Codes (LOINC).
2. Syntactic analysis of text cannot address ambiguity due to grammar and writing style.
3. Trade names are not always included in the UMLS Metathesaurus.
4. Conservative constraints on the definition of a macro phrase.
5. There were exact matches only; partial matches were excluded.

2.1.2.2.7 SAPHIRE

The SAPHIRE system (W. R. Hersh & Greenes, 1990) was originally developed to support biomedical text indexing, but was later used to support other applications, such as extraction of concepts from patient medical records (W. R. Hersh & Donohoe, 1998).

Algorithm: The original SAPHIRE algorithm is substantially different from the current version. The original algorithm is based on strict pattern matching. All terms in the source text phrase must be present and in the same order as the UMLS phrase for a successful match (W. R. Hersh & Greenes, 1990). The latest algorithm (W. Hersh & Leone, 1995) relaxes the strict requirements and allows for partial matches and out of sequence terms. The algorithm takes a source text in the form of a phrase or sentence as input and extracts individual terms using the barrier word method (Tersmette et al., 1988). Barrier words are high-frequency words considered to be common. For each term found, a list of UMLS concepts containing the word is retrieved. UMLS concepts having high-frequency terms must also have low frequency terms as well or the concepts are excluded. This is to eliminate concepts containing low content terms. The individual term concept lists are merged, and any concept having less than one-half the number of terms from the input text is excluded. The resulting set of UMLS concepts is then scored. The highest score occurs if all terms in the concept appear in the source text. Term order is ignored. If there is no exact term match, a weighting formula scores the concept based on proportion of words between source text and concept, term proximity, and length of term matches between source text and concept (W. R. Hersh, Mailhot, Arnott-Smith, & Lowe, 2001).

Evaluation: SAPHIRE has been evaluated in the context of effectiveness as an information retrieval system (W. Hersh, Hickam, Haynes, & McKibbon, 1991), but a more recent evaluation focusing on finding best concepts rather than all concepts was completed using radiology reports (W. R. Hersh et al., 2001). Fifty radiology reports were processed using SAPHIRE to extract UMLS concepts. Precision and Recall were the evaluation measures. Precision was defined as number of correctly mapped concepts divided by number of total concepts. Recall was defined as the number of correctly mapped concepts divided by total number of correct concepts. Precision was measured at 30% and Recall at 63%.

Failure Analysis: A failure analysis of the radiology report concept matching was performed. The failures affecting recall were due to scoring errors, where the correct concept was scored lower than other concepts, or with issues regarding the barrier method of phrase identification. Failures affecting precision include negation in the phrase which was not identified, and disambiguating competing concepts. A key problem with SAPHIRE's algorithm is that it may return multiple matching concepts for a text segment which, due to partial matching, are incorrect concept mappings (P. Nadkarni et al., 2001). The disambiguation problem was handled by adding a semantic type filter, which filters out concepts belonging to a particular semantic type.

2.1.2.2.8 SENSE

The SENSE (SEarch with New Semantics) system (Zieman & Bleich, 1997) is designed to map user queries to the National Library of Medicine's Medical Subject Heading (MeSH) terms. The system addresses the mismatch between user's natural language queries and MeSH index terms. The idea is to map user queries to MeSH terms, which are indexed by biomedical information retrieval systems.

Algorithm: In order to translate source text into concepts, SENSE translates a user query (a short phrase) into what the authors call *semantic factors*. Semantic factors are base concepts which cannot be decomposed further. SENSE defines 3,400 semantic factors. Semantic factors are constructed from an input phrase by having the Semantic Analyzer component look up phrase terms in a knowledge base, which handles variants, such as spelling and plural forms, and produce identical semantic factors for all input phrases having the same meaning. The output is a suggested list of MeSH terms which can be used to perform a search.

Evaluation: No evaluation was performed.

Failure Analysis: No failure analysis was performed because no evaluation was done. However, it should be noted that the purpose of SENSE is to build a list of suggested MeSH terms. No effort is made to identify the best matching concept. Therefore, SENSE needs a disambiguation stage to filter out concepts which are not the best match for a phrase.

2.2 Text Summarization

Text summarization is a data reduction method and has roots dating back to the 1950s with the work initially done using statistical analysis of terms (Luhn, 1958). Text summarization has been defined as “the process of distilling the most important information from a source (or sources) to produce an abridged version for a particular user (or users) and task (or tasks)” (Mani & Maybury, 1999), and “a reductive transformation of source text to summary text through content reduction by selection and/or generalization on what is important in the source" (Sparck Jones, 1999). In this paper, text summarization and document summarization are used interchangeably. Text summarization is generally a three-phase model: Interpretation, Transformation and Generation (Sparck Jones, 1999). Interpretation provides some analysis of the source text (such as removing insignificant words) and puts it into an intermediate form. Transformation takes the interpreted form and builds a summary form through methods such as content selection and concept generalization. Generation then takes the summary form and generates output appropriate for the user. Generation includes, for example, how to order the highest-ranking sentences when using an extractive approach.

Summaries are designed to serve one of two purposes: (a) indicative summaries give some idea of what the source text is about, so that the user can determine whether the source text should be read completely, and (b) informative summaries are intended as surrogates for the source text, where the main ideas are captured and presented (Mani & Maybury, 1999). Summaries are generated from the original source text using one of two approaches: extractive or abstractive. Extractive approaches reuse sentences from the source text in the generated summary. Abstractive approaches rely on natural language generation to summarize a text.

There are several methods for producing text summaries: Surface-level, Entity-level, and Discourse-level as well as hybrid combinations of these approaches (Mani & Maybury, 1999). Surface-level approaches use statistical information about term occurrences (Luhn, 1958), (A. Nenkova & Vanderwende, 2005), (Vanderwende & Suzuki, 2005), term locations (Edmundson, 1999), (S. Teufel & Moens, 1999), and important, known phrases (Kupiec, Pedersen, & Chen, 1999) Entity-level approaches typically rely on graph-based structures to identify important information. Such structures are built from and typically rely on external information sources, such as thesaural relationships (Barzilay & Elhadad, 1997). Discourse-level approaches derive important information based on the structure of the text (Strzalkowski, Stein, & Wang,). In terms of performance, Hovy reports that frequency-based approaches typically have 15%-35% precision and recall, while cohesive approaches, such as Entity-level and Discourse-level approaches, typically range from 30%-60% precision and recall(E. H. Hovy, 2005). The precision and recall scores in Hovy's work are generated by comparing sentences extracted by a machine to the set of sentences extracted by a person for the same document.

In text summarization, a key goal is to identify important text which should be presented to the user. This subset of the original source text is considered to contain the main ideas of a text. In either single document or multiple document summarization, the top-n-sentences approach is the most common. In this approach, the summarization system identifies the sentences that most likely capture the main idea of a document or set of documents. The top sentences are then output, up to some limit either on the number of sentences or the number of characters, in order to produce a summary.

The first research work done in summarization used term frequency to identify important words in a text (Luhn, 1958). The sentences containing one or more high-frequency words are considered more important than other sentences which do not contain high-frequency words, or fewer of them. The idea is that frequently repeated terms are due to the author reiterating the main idea. Further research in text summarization confirms frequency (reiteration) is a strong feature (Rath, Resnick, & Savage, 1961), (Pollock & Zamora, 1975), (Edmundson, 1999), (A. Nenkova & Vanderwende, 2005). Another method for finding important text is recent work done lexical chaining (Morris & Hirst, 1991). Lexical chaining finds common links between words. For example, the words *lung* and *pulmonary* are related by the common concept *lung*. A text is analyzed to find all chains of words, and the strongest chains of words are then assumed to represent the main idea of a text.

Our research applies both approaches to biomedical text summarization, using concept chaining rather than term chaining, and using concept frequency rather than term frequency. The research covers all three summarization approaches: surface (concept frequency summarizer), entity (concept chaining summarizer), and discourse (to be integrated with the concept frequency summarizer). The subsections below review the literature for the frequency and lexical chaining methods.

2.2.1 Text Summarization Using Lexical Chaining

Lexical chaining has been used for many years for text summarization. Lexical chaining is a method for determining lexical cohesion among terms in a text (Barzilay & Elhadad, 1997). Lexical cohesion is a property of text that causes a discourse segment to “hang together” as a unit (Morris & Hirst, 1991). Lexical cohesion is important in computational text understanding for two major reasons: 1) providing term ambiguity resolution, and 2) providing information for determining the meaning of text (Morris & Hirst, 1991). Lexical chaining is useful for determining the *aboutness* of a discourse segment, without fully understanding the discourse. As a basic assumption, the text must explicitly contain semantically related terms identifying the main concept. For example, if a text is about a political candidate and does not contain terms signifying the person is a candidate, lexical chaining cannot identify the fact the person is a political candidate.

Lexical chains for text summarization were first introduced by Morris and Hirst (Morris & Hirst, 1991). Their initial work described the approach, but did not implement it because electronic versions of a thesaurus were not available at the time. A thesaurus is used to relate words semantically; for example, through synonymy and hypernym/hyponym semantic relationships. A hypernym is the word with the more general meaning among a set of related words. A hypernym relationship moves from a specific concept to a general concept. For example, given two related words {flower,plant}, *plant* is a hypernym of *flower* because *plant* is more general than *flower*. A hyponym relationship is the opposite of hypernymy, and moves from a general concept to a specific concept. For example, *flower* is a hyponym of *plant* because *flower* is more specific than *plant.* A machine implementation of lexical chaining by Barzilay and Elhadad (Barzilay & Elhadad, 1997) showed the theoretical work by Morris/Hirst could be practically realized for document summarization. While Barzilay/Elhadad proved the feasibility of computing lexical chains, their implementation ran in exponential time, making its mainstream use unlikely. A linear time algorithm was later defined and implemented by (Silber & McCoy, 2002). A more recent implementation by Galley and McKeown (Galley & McKeown, 2003) focused on improving word sense disambiguation based on the idea of one sense per discourse. All of these implementations use WordNet (Fellbaum, 1998) as the knowledge source for identifying semantic relationships between terms. WordNet is a freely-available lexical database for the English language which organizes nouns, verbs, adjectives and adverbs into synonym sets (known as a *synset*). These synsets are then linked to one another via different relations to form semantic relations between lexical concepts. WordNet is an ongoing research project developed at the Cognitive Science Laboratory of Princeton University.

The SUMMARIST system (E. Hovy & Lin, 1999) uses WordNet (Fellbaum, 1998) concept counting not for identifying salient sentences, but for topic interpretation. In topic interpretation, concept frequency counting is used to find a node in the concept hierarchy which sufficiently generalizes more specific concepts (e.g., {pear, apple} $\rightarrow$ fruit). The SUMMARIST authors cite the lack of domain-specific resources as a serious drawback to this approach. BioChain uses domain-specific resources exclusively for important sentence identification (see Section 3.2.1).

Lexical chains are an intermediate representation of source text, and are not used directly by an end-user. Instead, lexical chains are applied within a specific application. For example, lexical chaining has been used for generating hypertext links in newspaper articles (Green, 1998), indexing videoconference transcriptions (Kazman, Al-Halimi, Hunt, & Mantei, 1996), generating news story headlines (Wang et al., 2005), detecting and correcting of malapropisms (Hirst, Graeme and St-Onge, David, 1998), and summarizing documents (Barzilay & Elhadad, 1997), (W. Doran, Stokes, Carthy, & Dunnion, 2004).

2.2.2 Text Summarization Using Frequency

The use of frequency as a feature in locating important areas of a text has been proven useful in the literature (Luhn, 1958) (Rath et al., 1961) (Pollock & Zamora, 1975)

(Edmundson, 1999). This is most likely due to reiteration, where authors state important information in several different ways, in order to reinforce main points (Sparck Jones, 1999).

Term frequency was first used in extractive text summarization in the late 1950's (Luhn, 1958). Luhn justified the use of word frequency as a measure of word significance on the observation that authors typically reiterate the main points of a text. Luhn then extended the idea of word significance to identifying significant sentences. The sentences containing one or more high-frequency words are considered more important than other sentences which do not contain high-frequency words, or fewer of them. The idea is that frequently repeated terms within close proximity to one another are due to the author reiterating one of the main ideas of a text. The strong performance of subsequent work in text summarization which utilizes frequency as feature confirms that word frequency (reiteration) is a strong feature (Rath et al., 1961), (Pollock & Zamora, 1975), (Edmundson, 1999), (A. Nenkova & Vanderwende, 2005). A follow-up study to Luhn's work used five different methods for scoring sentences to compare against the scoring method proposed by Luhn. The five methods considered high-frequency words, word gaps within a sentence, and sentence length. The authors found the five methods showed high agreement in sentence selection, indicating the actual scoring method was less effective for identifying important sentences than the base frequency measure (Rath et al., 1961). Rath et al. also suggested that features other than frequency be consider. For example, cue phrases, statistically-generated keywords, title words, and word location were used as features to help identify important sentences. (Pollock & Zamora, 1975) (Edmundson, 1999). Cue phrases are well-known phrases that positively or negatively suggest a sentences relevance, such as ‘significantly’ (positive) or ‘incredibly’ (negative – too subjective). Title words are words found in the title of a text. Keywords are non-cue words found in a source text which have a high frequency. Keywords are considered to be topic indicators. The successful use of multiple features to identify sentences suggests that semantic and syntactic features need to supplement the statistical approach of word frequency (Edmundson, 1999).

Summarization using units of text larger than a single word has also been researched. The LAKE system uses keyphrases for summarization (D’Avanzo, Magnini, & Vallin, 2004). A keyphrase is a phrase which indicates one of the topics within a text. The LAKE system used over 200 part-of-speech patterns to identify phrases in a source text. Keyphrases were selected from among the list of all possible phrases in the source document by a classifier which used two features: (a) a term frequency weighting, and (b) the position of the phrase from the start of the document. Other attempts to use more than one word include the SUMMARIST system (E. Hovy & Lin, 1999). The SUMMARIST system uses WordNet (Fellbaum, 1998) to find concepts. The concepts are then counted to determine frequency and therefore importance. SUMMARIST is unique from other systems in that the concept counting is used for generalizing concepts for topic interpretation rather than for identifying salient sentences. For example, the words {pear, apple, strawberry} might be rewritten in the output summary to use the more general term {fruit}.

Most recently, the SumBasic algorithm used word frequency as part of a context-sensitive approach to identifying important sentences while reducing information redundancy (A. Nenkova & Vanderwende, 2005). SumBasic uses a probability distribution of terms in a source text, and reduces term probability as sentences containing the terms are selected. The idea of reducing term probability is to reduce information redundancy by finding sentences which include words not already in the summary. There are four steps in the algorithm. The first is to determine the probability distribution of all words found within a source text by computing the number of times a word appears in the text and dividing it by the total number of words found in the text. The second step is to score each sentence by summing the probabilities of all words within a sentence. The third step determines the sentence to be extracted by finding the highest-scoring sentence. The fourth step then multiplies the probability value of each word in the last extracted sentence by itself. This has the effect of reducing the chance that the same words will be selected again from the remaining source sentences.

2.2.3 Document Understanding Conferences

There has been much work done in the Document Understanding Conferences (DUC) (http://duc.nist.gov/). DUC provides an annual forum (or competition) for researchers to extend text summarization technology. In recent DUC conferences, several approaches for identifying sentences for extraction were used, and several of the approaches are surveyed here. The News Story system uses a pattern extraction algorithm (C5.0) to generate a decision tree to predict the words in the source text that should be part of summary (W. Doran et al., 2004). The News Story system uses eight text features: (a) term frequency (TF) of words in the document, (b) inverse document frequency (IDF) of words in external news corpus, (c) position of words from start of document, (d) lexical cohesion score between words and the document, (e) a Boolean noun flag for each word, (f) a Boolean verb flag for each word, (g) a Boolean adjective flag for each word, and (h) a noun or proper noun phrase flag. Their findings are that TF, word position and IDF have the greatest impact on summary quality. In addition, they concluded that lexical cohesion adds little as a feature in decision tree classification.

The LAKE system uses a keyphrase extraction approach that is used to identify candidate sentences (D'Avanzo et al., 2004). LAKE relies on word N-grams, which is a way to group sequences of words together. For example, the phrase *lung cancer* has unigrams {lung, cancer}, a single bigram {lung cancer} and no further N-grams where N has value greater than two (that is, three or more terms). LAKE begins by extracting all unigrams, bigrams, trigrams, and four-grams and filters them with part-of-speech patterns. A Naïve Bayes classifier trained using manual keyphrases is then used to identify relevant keyphrases. The resulting keyphrases are scored using two features: (a) keyphrase TF*IDF, and (b) distance of keyphrase from the start of document. Their results scored in the middle of all 2004 DUC submissions. The authors feel their system can be improved by finding additional features that capture the semantic properties of keyphrases. One possibility mentioned is to compute lexical chains and then use membership of a keyphrase within a chain as a feature.

The KMS system describes a system where a text is decomposed into a parse tree format (Litkowski, 2004). The parse tree is then used to identify noun phrases and score them based on a frequency analysis of terms in the noun phrases in addition to the occurrence of words in a DUC topic specification. Their performance fell in an acceptable range, and the authors observe that in general their frequency-based approach performs better than systems based on other approaches.

Finally, the GISTexter system uses a frequency-based method to identify sentences to extract (Lacatusu, Hickl, Harabagiu, & Nezda, 2004). GISTexter computes a weight for each term in a collection based on term frequency in a relevant set of documents. This weight is then used to score each sentence. The top scoring sentences are then extracted. GISTexter used the following method to generate a summary:

1. The highest ranking sentence is used as the initial summary.
2. If the summary length is less than the 665-character limit imposed by DUC, then the next highest-scoring sentence containing more information than the summary currently has is selected.
3. Step 2 is to iterate until the maximum summary length of 665 characters is exceeded or no more sentences are available.
4. Summary compression is performed, using several different approaches: (a) if (summary length - last sentence length) > 600 characters, the last sentence is removed; and (b) if the summary length > 665-character DUC limit, the summary is truncated at the last word prior to the 665-character limit, resulting in an incomplete sentence.

GISTexter performed among the top systems. The authors found the best approach for summary compression is truncation (4b listed above). This approach rated the second highest score of all systems that competed in DUC 2004.

2.2.4 Text Summarization in the Biomedical Domain

In this section, previous work related to summarization of biomedical texts is discussed. The first subsection describes efforts by others to characterize biomedical texts and construct a corpus for use in generating and evaluating summaries. The second subsection describes some recent biomedical text summarization and question-answering systems.

2.2.4.1 Characterization and Corpora

A recent survey of medical document summarization considered summarization approaches not only from text-based documents, but also from multimedia formats (Afantenos et al., 2005). The authors identify the following methods currently used to perform medical text summarization:

Extractive: Extractive approaches, as previously mentioned, take sentences from the source text and re-use them in the generated summary. There are two approaches used in this technique: statistical and graph. The statistical approach ranks each sentence and extracts the highest ranking sentence. The scoring is done in many ways, such as term frequency, keyphrase identification, and noun phrase frequency. The graph approach generates a tree representation of a text, and the most salient nodes in the tree are identified. The tree representation can be based on paragraph similarity, cohesion relationships between terms, and rhetorical structure relationships.

Abstractive: The abstractive approaches rely on natural language generation to summarize a text. The first abstractive approach uses a predefined template and the fields in the template are filled-in from information contained in the source text. The second approach uses a syntactical analysis of the source text to identify key components of each candidate sentence to form new sentences from existing sentences.

Multimedia: Many forms of medical communication include video, audio, and graphics presentations. These forms of communication have been analyzed to perform summarization on each type of multimedia, but the approaches are not directly usable for text processing, and are not discussed further.

Cognitive model based: Cognitive-based approaches try to simulate the methods of human summarizers to produce a summary of a source text. The authors mention a system which uses 79 agents based on over one hundred human strategies to produce a summary. The agents work in combination with a knowledge base, a domain-specific ontology, and rhetorical structure information to produce a summary.

The use of full-text sources as compared to their corresponding abstracts was found to be more beneficial for resolving gene-symbol ambiguity (Schuemie et al., 2004). The corpus was a set of 3,900 biomedical articles with full-text and abstract sources available. Keywords in the text sources were identified using five different methods, including MeSH terms and high-ranking TF*IDF words. Information content was measured using twelve different measures. Sections of the paper were identified using the author's section headings. The recognized headings are *Abstract*, *Background*, *Introduction*, *Methods*, *Results*, and *Discussion*.

Key related findings reported by the authors are as follows:

1. Full-text is not as easy to process as abstracts, since full-text is not always publicly-available and requires more computing and storage capacity.

2. The information density is highest in the abstract, while the information coverage is highest in the full-text.
3. The highest information coverage is in the Results section.
4. The amount of information in each section is unique to that section is 30% to 40%.
5. The use of abstracts alone for information extraction is likely to result in information loss.

Building a corpus for use in summary generation as well as summary evaluation is a large effort. Typically a multi-step process is required to first identify domain-specific sources of full-text articles, acquire them, and then process them linguistically to identify text units such as sections, paragraphs, sentences, and phrases. In addition, semantic processing may also be performed to identify domain-specific concepts. An example of such a corpus-building effort was done as part of the PERSIVAL project (K. R. McKeown et al., 2001). The PERSIVAL corpus for the cardiology domain was constructed by first sampling journals in the domain, downloading full-text article source, and then identifying medical terms in the text (S. Teufel & Moens, 2002). There were 22 journals sample from an estimated 700 journals in the domain. The journals were selected based on (a) ISI citation analysis, (b) domain expert review, and (c) electronic availability. Crawlers were then implemented for the 22 journals to download the full-text sources, resulting in a corpus of approximately 30,000 full-text articles. Each of the downloaded texts is then linguistically processed to find sentence boundaries and noun phrases. The noun phrases are then used as input to a medical terminology finder. The medical terminology finder uses two methods to find medical terms in the full text: (a) the UMLS vocabularies, and (b) a statistical method which compares general English text and medical text. The output of the medical terminology finder is not evaluated for accuracy. Each full-text article crawled and processed is stored in an XML-encoded format which contains elements to identify article sections, sentences, and medical terms, among other items. PERSIVAL uses the XML-encoded format to identify sections to extract sentences from and to fill information extraction templates with medical terms and findings.

2.2.4.2 Biomedical Text Summarization Systems

In this section, several recent systems designed for biomedical text summarization are discussed.

2.2.4.2.1 PERSIVAL

PERSIVAL (PErsonalized Retrieval and Summarization of Images, Video and Language) is a medical digital library which provides personalized information to physicians as well as laypeople (N. Elhadad & McKeown, 2001) (K. R. McKeown et al., 2001) (K. McKeown, Elhadad, & Hatzivassiloglou, 2003). PERSIVAL incorporates query generation, search, presentation and summarization to provide a complete system. The original summarization component incorporates two different methods, depending on the target audience (medical professional or layperson). For medical professionals, the summarizer uses clinical trial study articles as the input source. The format of the articles follows a common section format. Sentences are extracted from the Results section and then categorized as to whether or not they provide result information based on cue words and pattern matching. For patient summaries, relevant consumer health texts are identified, similar sentences from the consumer health texts are clustered, and then a representative sentence from each cluster is output to form a summary. A tree structure of the topics discussed by all selected consumer health text is also formed and used to identify common themes, differences, and topic detail. The layperson summarizer also identifies medical terms and provides a reference to explain them. The system evaluation focuses on the medical professional summarizer, where physicians were asked to select sentences from the clinical trial articles they found relevant. Although exact agreement was not specified, the authors report differences based on individual interests and level of experience. The Technical Article Summarizer, described below, has a more complete evaluation of summarization within PERSIVAL.

2.2.4.2.2 Technical Article Summarizer

The Technical Article Summarizer (TAS) is a newer component of the PERSIVAL digital medical library (N. Elhadad, 2006), (N. Elhadad, McKeown, Kaufman, & Jordan, 2005), (N. Elhadad, Kan, Klavans, & McKeown, 2005). The TAS component is designed to generate a personalized biomedical summary based on a patient record. Clinical trial articles are first located using query results submitted to the PubMed resource based on several patient-specific criteria, such as the patient record. The patient record is used to identify characteristics of the patient, such as symptoms and diagnoses. The clinical trial articles are then processed using a pipeline architecture which performs content selection, content organization, and content generation. The content selection stage identifies text from the clinical trial articles based on patient characteristics generated from the patient record. The content organization stage clusters semantically-related text from the clinical trial articles. Each cluster is prioritized based on its content, and is output as a single paragraph during the content generation stage. The content generation stage aggregates sentences within each cluster and uses phrasal generation to build the paragraph to be output. The TAS component was evaluated by asking eleven study participants to use the summarization output along with knowledge of a patient in three different medical scenarios. The participants were given a general summary, a list of articles returned by a search engine, and the personalized summary produced by TAS. The results show that the personalized summaries were the preferred format to provide patient-related information. The major problems with the general summary which are addressed by the personalized summary are irrelevant information and lack of continuity in the sentences.

2.2.4.2.3 Medical Text Indexer

Medical Text Indexer (MTI) (National Library of Medicine, United States, 2006a) is part of an indexing initiative at the United States National Library of Medicine that uses concept-based algorithms to index text. A full-text biomedical source can contain many concepts. Some concepts are more important in describing the text than other concepts. The use of summarization allows MTI to index the most important areas of a text. The most important concepts useful for indexing are contained in the summarized text (National Institute of Health, 2005). MTI uses the MetaMap (Aronson, 1996) concept finder to identify concepts in the text. The identified concepts are linked to each sentence in the text. A concept-by-sentence matrix is formed. This set of vectors is then dimension-reduced using latent semantic analysis. A text relationship map is built which links the various sentences in the text together. Each link in the map is assigned a weight based on the similarity between two sentences. The text sentences are scored based on the sum of outgoing link weights. The highest scoring sentences are considered the most important. The evaluation compares the generated summaries against a section model, which extracts sentences for indexing based on the section of the text they belong to. The authors find that the optimal summary size for the best indexing performance is approximately 90 sentences. Summary sizes ranged from 17 and doubled at each interval until all sentences were included. In addition, some smaller summary ranges from 68-102 sentences were evaluated. The authors also conclude that using the section model (that is, using sections of a text, such as Methods or Results) outperforms the use of summaries for MTI indexing purposes.

2.2.4.2.4 Medical Information Summarizer

The Medical Information Summarizer (MIS) provides single-document summaries as part of the results returned from a query against online medical document repositories (Chen & Verma, 2006). Starting with keywords provided from the paper's author, the system expands the set of keywords by using UMLS resources to perform variant generation on the original keywords, finding abbreviations of the keywords, and finding semantically-related keywords. Sentences are then scored based on keyword membership. Three different scoring mechanisms were evaluated: (a) using original keywords only, (b) using UMLS expanded keywords, and (c) using UMLS expanded keywords normalized for sentence length. Each keyword is weighted so that the original keywords have a value of 1.0, while UMLS expanded keywords have a value of 0.5. The evaluation measured standard precision and recall, comparing the extracted sentences to sentences from the abstract plus conclusion section. The authors found that the expanded keywords outperformed both the paper keywords and the normalized expanded keyword scoring approaches.

2.2.4.2.5 BIOSQUASH

BioSquash is an effort to provide a summary from multiple documents in response to a question (Shi et al., 2007). BioSquash is an adaption of the Squash summarizer (Melli et al., 2005). It uses UMLS as the domain-specific ontology and WordNet as the general ontology. BioSquash uses a four-stage pipeline approach. The Annotator stage tags the multiple input documents as well as the input question with named-entities and semantic roles of syntactic constituents. The Concept Similarity stage identifies concepts using UMLS and WordNet. The Extractor stage takes the output of the Annotator and Concept Similarity stages and constructs a semantic graph of the sentences in the texts. Sentences are scored based on the number of edges in the graph. The Editor stage takes the highest-scoring summary and constructs a fluent summary. Sentences are given an importance score based on sentence overlap with the question and first and last sentences in the document. The sentences are then ordered by their importance score. BioSquash was evaluated using data from the Ad-hoc Retrieval task of Genomics Track at the 2005 Text Retrieval Conference (TREC). TREC provided abstracts and summaries for 18 questions. There were 5 human summaries and 50 ROUGE reference summaries. The set of abstracts to be summarized came from (a) human selected relevant summaries, and (b) system selected summaries. Thirty abstracts for each question were summarized. ROUGE-2 and ROUGE-SU4 scores were generated using the output of BioSquash and compared to the reference summaries. The findings are that the human selected summaries outperformed the system-selected summaries of the abstracts. The BioSquash work is still preliminary: further evaluation is planned using full-texts rather than abstracts, and using human summaries of the full-text as the source for ROUGE references.

2.2.4.2.6 MITRE Text and Audio Processing

The MITRE Text and Audio Processing (MiTAP) is a system for monitoring biological threats by assimilating information from multiple textual sources (Damianos et al., 2002). MiTAP incorporates single- and multiple-document summarization components. Single document summarization is generated using statistical and rule-based classifiers. The classifiers are trained on the abstracts of text. The features generated include TF*IDF measures and synonym links between sentences. The multiple document summarizers use externally-provided summarizers: (a) NewsBlaster clusters related articles and produces a summary of them (K. McKeown et al., 2002), and (b) LingPipe extracts sentences referencing known entities (Carpenter & Baldwin, 2007). No specific evaluation was done on the summarization components, but ongoing user-studies have been done on the entire system to find usability and utility of the overall system. The findings are that the system provides more utility than general Web-based searching and result presentation.

2.2.4.2.7 Clinical Question-Answering

Biomedical text summarization has also been used as part of larger clinical question-answering system (CQA) (D. Demner-Fushman & Lin, 2006) (D. Demner-Fushman & Lin, 2007). The CQA system combines elements of question answering, information retrieval and summarization. It uses a drill-down approach to provide answers to practicing physicians of the form 'What is the best drug for treatment for *X*?' The first level provides categories of drugs. Each drug category is associated with a cluster of MEDLINE abstracts related to the drug category, which is the second level. The third level is an extractive summary of each of the abstracts. The fourth level is to view the original abstract. Each generated summary contains three main elements: (a) the main intervention described in the paper, (b) the paper title, and (c) the top-scoring outcome sentence. The outcome sentence makes a statement about the quality of the drug for treating a particular condition. The outcome sentence is determined using a supervised machine learning classifier (D. Demner-Fushman & Lin, 2005). The classifier is actually composed of several other classifiers: rule-based, unigram, n-gram classifier, position, document length, and semantic. The rule-based classifier uses cue phrases; the unigram and n-gram classifiers use Naïve Bayes on abstract terms; position determines how far away from the end of the abstract a sentence is; document length classifier determines probability of an abstract having an outcome sentence based on its length; and the semantic classifier uses UMLS concepts highly correlated with outcome statements. The classifier scores are then weighted and combined, resulting in score for each sentence. An evaluation shows that the classifier correctly predicts outcome sentences 75% of the time when a two-sentence cutoff is used, and 95% of the time when a three-sentence cutoff is used. Abstracts may have more than one outcome statement, and the cutoffs are used to match at least one of the outcome sentences.

3. APPROACH AND METHODS

This chapter describes an integrated semantic annotation and text summarization system which forms the basis of our research. The high-level structure of the annotation and summarization system is shown in Figure 7. The system takes a biomedical source text and performs analysis in several stages resulting in a final output of a text summary of the original source text. The two primary stages which are part of this research are Semantic Annotation and Text Summarization. Two other stages, Lexical Processing and User Presentation, are constructed as part of the system, but are not the focus of the research. The Lexical Processing stage is the first stage and is responsible for finding sentence boundaries within a source text, and then phrases within each discovered sentence. Lexical Processing is performed using software components from third-party sources. The final stage is the User Presentation stage, which presents to the user the generated summary. The user presentation consists of generating a simple text file containing the extracted sentences.

The two stages which are the focus of the research, Semantic Annotation and Text Summarization, are between the Lexical Processing and User Presentation stages. Once a source text has been decomposed into sentences and phrases in the Lexical Processing stage, the sentence phrases are fed into the Semantic Annotation stage, and the output is an annotated version of the source text containing sentences, sentence phrases, and sentence phrases annotated with one or more domain-specific concepts (a best mapping is attempted, but in cases of ambiguity more than one mapping may be returned.) The annotated source text from the Semantic Annotation stage is then directed into the Text Summarization stage. The Text Summarization stage uses the concept mapping information to identify the most important content in the source text, and then extract sentences from the source text which are most representative of the content. The output of the Text Summarization stage is an ordered subset of sentences from the source text. In this research, only single document summarization is considered. In the following sections, the details of each of the two primary stages, Semantic Annotation and Text Summarization, are presented.

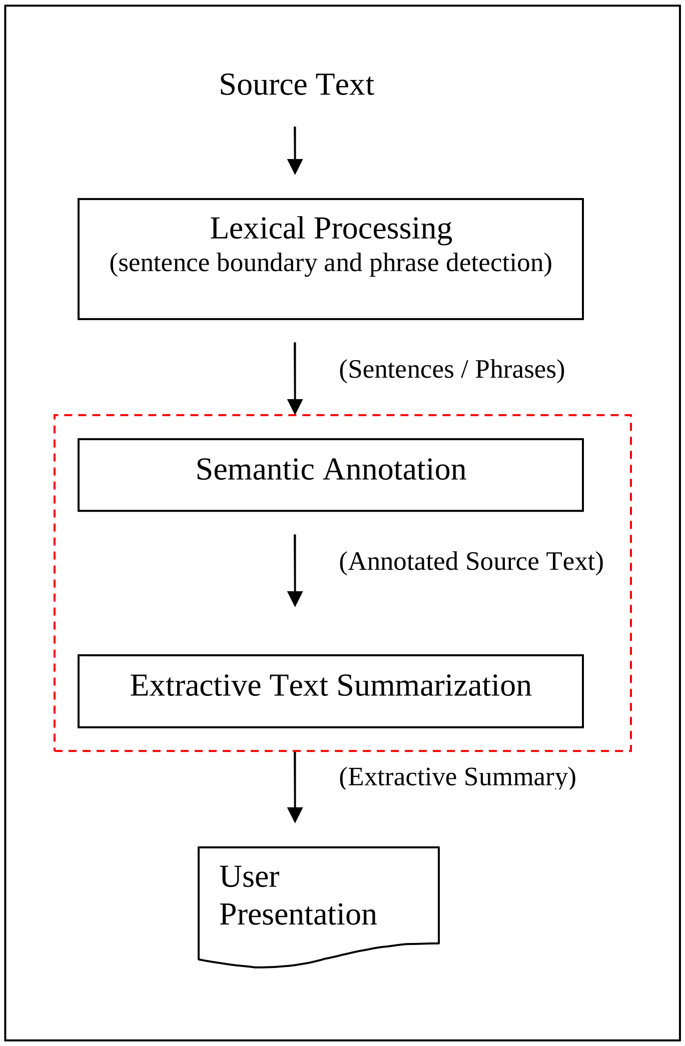


Figure 7: High-level overview of the annotation and text summarization system. The dashed box indicates the parts of the system the research focuses on.

The annotation system assumes the source phrase has already been determined from prior analysis. Source phrases can be detected using a variety of methods, such as natural language parsing and sliding windows (Wollersheim et al., 2002) and barrier words (Tersmette et al., 1988). The annotation system makes no assumptions on the selection of source phrases, other than the input unit must be a phrase. The goal of the Semantic Annotation stage is to find the best UMLS concept match for each phrase in the source text using surface-level features.

There are several types of phrases. The *source phrase* is a phrase from the source text which the system will attempt to annotate with a biomedical concept. UMLS concepts are composed of one or more synonymous phrases, which are known as *concept phrases*. A single UMLS concept may have more than one concept phrase associated with it. *Candidate phrases* are concept phrases having words in common with the source phrase. A *candidate concept* identifies the UMLS concept a candidate phrase belongs to. A *concept name* is the name given to a particular UMLS concept.

The multi-level filtering approach of the system takes a source phrase from the source text, retrieves a list of concept phrases based on the overlap of words from the source phrase and concept phrases, forming a set of candidate phrases. If only a single candidate phrase exists, its associated concept is returned. If there is more than one candidate phrase generated, an iterative process of filtering out candidate phrases begins. The candidate phrase filters are based on n-gram co-occurrences between the source phrase and the candidate phrases. The multi-level filtering is done to improve computational efficiency by applying successively more computationally complex filters, rather than scoring a candidate phrase with different measures at once. This approach is different than existing approaches, which typically score a candidate phrase completely in one pass and then rank the set of resulting concepts (W. Hersh & Leone, 1995), (Aronson, 2001a). The idea is to successively filter out concepts using basic techniques, and compute more complex candidate phrase scores for a small subset of possible candidate phrase matches. Two different types of filters are used, although other filters can be added: Coverage and Coherence. Coverage is the number of words in common between the source phrase and a candidate phrase, and coherence measures the common word ordering between the source phrase and a candidate phrase (Aronson, 1996). The Coverage and Coherence filters were chosen because they offer a complete approach to measuring word-membership and word-order between a source phrase and a candidate phrase.

Our research uses two methods for measuring both coverage and coherence: term weighting and skip-bigrams. The two methods are based on existing approaches and are novel in that they have not been applied to the semantic annotation task in previous work described in the literature. The coverage filter uses term weights. Previous approaches for measuring word coverage use binary weighting. In binary weighting, the weight of each word in a candidate phrase is zero to indicate absence of a word or one to indicate presence of a word in common with the source phrase (Aronson, 2001a), (W. Hersh & Leone, 1995). The coherence filter utilizes skip-bigrams, which have proven effective for measuring term order in machine translation and text summarization evaluation (C. Y. Lin & Och, 2004a), (Lavie, Sagae, & Jayaraman, 2004). Both the coverage and coherence methods used in this research are described in more detail below.

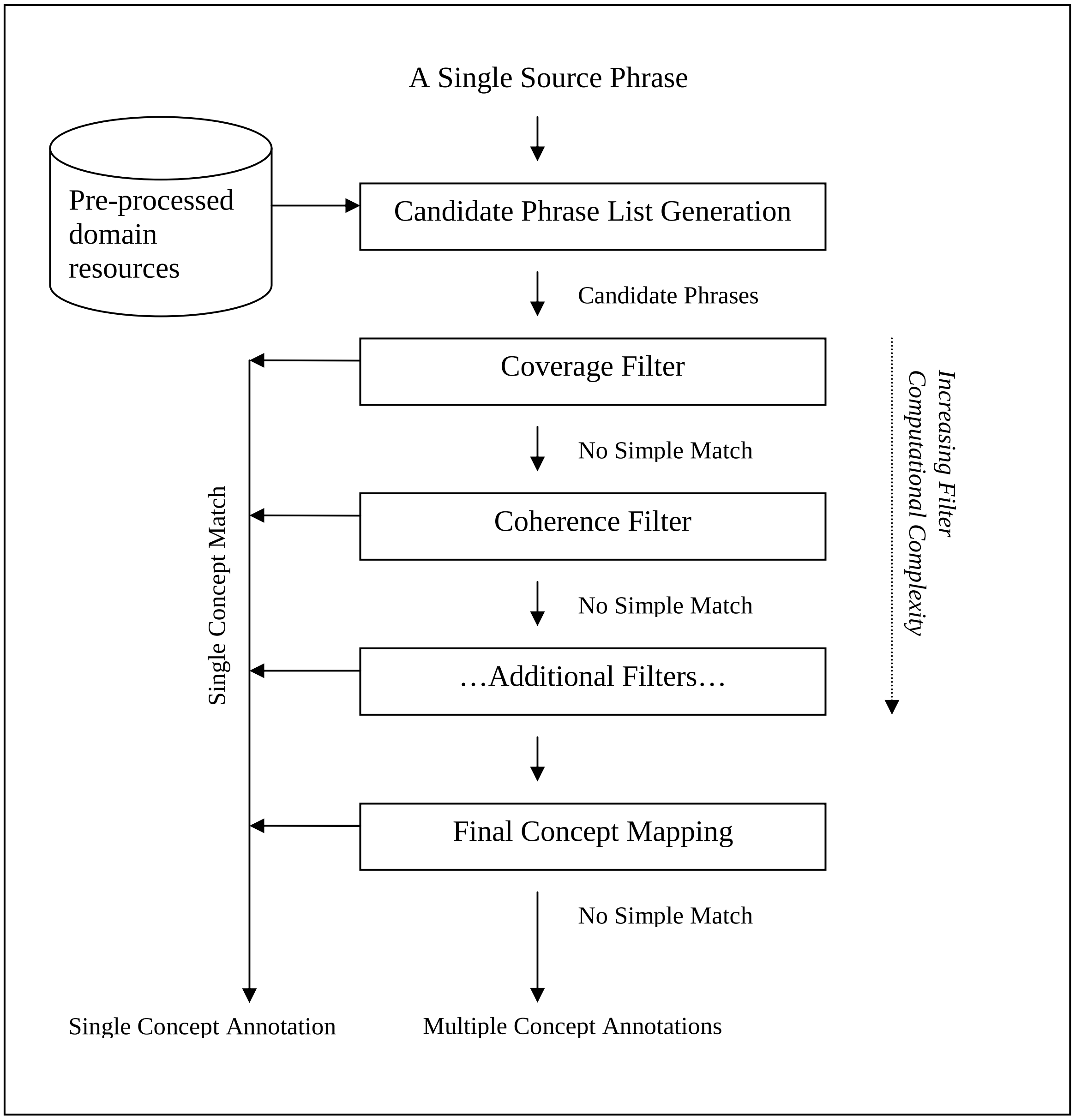


Figure 8: Stages in the multi-level annotator

The annotation system has advantages over existing concept mapping approaches. Systems such as MetaMap (Aronson, 1996) and SAPHIRE (W. Hersh & Leone, 1995) score candidate phrases completely in one pass. This has the disadvantage of computing more complex scores, such as coherence, when the simpler coverage score may have rejected the candidate phrase alone. In addition, by modeling the mapping as a series of filtering stages, the number of candidate phrases is reduced at each stage, reducing the number of complex calculations which must be performed. The MetaMap system also generates word variants and inflections at run-time. The annotation system maps the concept phrase words to their base form in a pre-processing stage (done once for each UMLS version of the data, not at each run-time). For example, the variants {*eyes*, *eyed*, *eying*} are all mapped to the base word *eye*. This allows concept phrases to map to base words so that variant generation is not required. By reducing the number of candidate phrases which have to be scored, using in-memory table structures, and eliminating time-consuming variant generation, CONANN is expected to outperform the state-of-the-art MetaMap system in terms of time to annotate a source phrase.

The overall CONANN annotation strategy for a single source phrase is as follows:

1. Construct a list of candidate phrases based on the words in common between all concept phrases and the source phrase. If only one candidate phrase remains, return its associated candidate concept.
2. Filter the list of candidate phrases based on a weighted coverage score given to each candidate phrase. If only one candidate phrase remains, return its associated candidate concept.
3. Filter the list of candidate phrases based on a weighted coherence score given to each candidate phrase. If only one candidate phrase remains, return its associated candidate concept.

4. If more than one candidate phrase remains, return the list of candidate concepts associated with each candidate phrase.

The following subsections explain in more detail each of the annotation stages:

3.1.1 Domain Resource Preparation (pre-processing)

Preprocessing of UMLS data is done before CONANN is used to perform annotation. Preprocessing organizes the words, word variants, phrases, and concepts stored in UMLS text files into a format which is faster for CONANN to process. For example, words are mapped to unique integer identifiers to reduce storage space and increase word comparison speed. Also, concept phrase words are mapped to their base form so that variant generation does not have to occur at run-time, as is the case for systems such as MetaMap (Aronson, 1996). For example, word variants within UMLS concept phrases are all mapped to a single base, such as mapping {*eye, oculus, ophthalmic*} to the base form {*ocular*}.

Each word in the UMLS is also weighted based on its usage in all concept phrases. In contrast to existing systems such as MetaMap (Aronson, 2001a), SAPHIRE (W. Hersh & Leone, 1995), and IndexFinder (Zou et al., 2003) which consider the count of words in common between a source phrase and a candidate phrase, the scoring of coverage and coherence in CONANN considers the contribution of each word in the source phrase by using a weighting mechanism. Information retrieval research uses a family of algorithms called TF*IDF, which uses the frequency of a term (TF) within a document and the frequency of a term across all documents (IDF) to find a similarity value between a user query and a document. A frequently occurring term within a document better indicates the content of the document, while a frequently occurring term across all documents is thought to give little discriminatory power, since a high proportion of documents contain the term (Baeza-Yates & Ribeiro-Neto, 1999). The inverse document frequency value (IDF) uses the frequency of a word across all documents as a way to identify words which are semantically focused (Manning & Schutze, 1999). Semantically-focused words are those words which do not frequently occur across all documents within a collection, and thus are more likely to have more discrimination power than words which frequently occur. To apply the ideas of TF*IDF from information retrieval to CONANN, each concept phrase is substituted for document. The weight of each unique word in all concept phrases is calculated using the inverse document frequency idea from information retrieval (Sparck Jones, 1972), substituting concept phrase for document, as shown in Figure 9:

$$\boxed{InversePhraseFrequency = \log \frac{N}{n_i}}$$

Figure 9: Inverse Phrase Frequency (IPF) value. N is the total number of phrases in UMLS and n_i is the total number of phrases a particular word occurs in.

Each unique UMLS word i is assigned a weight w_i based on its inverse phrase frequency (IPF) value. The importance (or weight) w_i of a word i is represented by its IPF value. Words which are more semantically focused will be given a higher weight than words which are not semantically focused. For example, assume there are 1,000 concept phrases, the word *plasma* occurs in 200 of the phrases, and the word *myeloma* occurs in 50 of the phrases. The IPF value of *plasma* will be 0.70 (log 1000/200) and the IPF value of *myeloma* will be 1.30 (log 1000/50). Therefore, if the words *plasma* and *myeloma* occur in the same phrase, the word myeloma is considered a more discriminative word than *plasma*. The idea is to give some indication of the importance of a word based on its usage within all concept phrases. Term frequency, which is typically combined with inverse document frequency for document information retrieval, is not considered since it is highly likely the frequency for each word will be one because the input unit of CONANN is a phrase, which usually does not include the same word multiple times in it. Eliminating term frequency from scoring reduces computational complexity.

Table 5 shows the list of tables generated in the pre-processing stage. The tables are all simple key-value lookup tables which map a unique key to a scalar or list value. The unique keys are generally known as identifiers. For example, a concept is uniquely identified by its concept identifier. The advantage of creating in-memory lookup tables is to speed access to key information, such as the words belonging to concept phrases, and to pre-calculate key information used in the stages, such as the Inverse Phrase Frequency weight for each word, in order to increase runtime performance. Each mapping table is useful for getting additional information about a word, phrase or concept. For example, given a word in the source text, its unique word identifier can be retrieved using the *WordToWordId* table, and the list of concept phrases the word appears in can then be obtained using *WordIdToPhraseIdList* table. To get the concept name a phrase belongs to, the *PhraseIdToConceptId* table can be used to find the concept identifier of the phrase, and then the *ConceptIdToConceptName* table is used to find the concept name. To account for word variation, the *WordToUninflectedWord* is used to normalize a word to its base form. For example, the word ‘cancers’ has the base word ‘cancer.’ The *WordToWordVariants* is used to find all UMLS-defined variants of a word. For example, the word *pulmonary* has the following set of word variants: { lung, lungs, pneumal, pneumonic, pulmonic, pulmonal, pneumonias, pneumoniae }.

Table 5: Mapping tables generated during pre-processing

Table Name	Purpose
WordToWordId	Get the unique identifier of a word
WordIdToWordIPF	Get the inverse phrase frequency weight for a word
WordIdToPhraseIdList	Retrieve all concept instances containing a given word
WordToUninflectedWord	Get the uninflected form of a word
WordToWordVariants	Retrieve all known variants of a word
ConceptIdToConceptName	Get the concept name for a given concept identifier
ConceptIdToPhraseIds	Retrieve UMLS phrases associated with a concept
ConceptLanguageModel	Get the language model of a concept
PhraseIdToConceptIds	Retrieve the concepts associated with a UMLS phrase
PhraseIdToWordIdList	Retrieve all words belonging to a particular concept phrase

To generate each of the mapping tables, UMLS source files from the MetaMap Transfer were used. Table 6 shows the source file name used for each mapping table. The MetaMap versions of the files rather than the core UMLS files are used since MetaMap Transfer has already pre-processed the core UMLS files and removed concepts which are known to be ambiguous or otherwise not useful in the concept identification task. The files used from MetaMap Transfer are as follows:

1. *sui_nmstr_str.txt* provides UMLS string identifiers and the corresponding text strings. (b) *infl.txt* provides UMLS words and their uninflected form.
2. *fullvars.txt* provides UMLS words and all known variations of the words.
3. *cui_concept.txt* provides a UMLS concept identifier and its corresponding name.
4. *sui_cui.txt* provides mappings between UMLS string identifiers and UMLS concept identifiers.

Table 6: MetaMap Transfer source files for mapping tables

Generated Table Name	MetaMap Transfer Source File(s)
WordToWordId	sui_nmstr_str.txt
WordIdToWordIPF	sui_nmstr_str.txt
WordIdToPhraseIdList	sui_nmstr_str.txt
WordToUninflectedWord	infl.txt
WordToWordVariants	fullvars.txt
ConceptIdToConceptName	cui_concept.txt
ConceptIdToPhraseIds	sui_cui.txt
ConceptLanguageModel	sui_cui.txt, sui_nmstr_str.txt
PhraseIdToConceptIds	sui_cui.txt
PhraseIdToWordIdList	sui_nmstr_str.txt

3.1.2 Candidate list generation

When a source phrase is presented to be annotated, it is first processed to remove all words which do not appear in UMLS, as well as removal of stop words. The words in the source phrase are mapped to their UMLS base form. This is done to eliminate word variation, and to allow exact matching of concept phrase words, which had the same base-form mapping done in the pre-processing step. A list of candidate phrases is then generated by finding all concept phrases which contain one or more of the base-form words in the source phrase. For example, the phrase *lung cancer* will find all candidate phrases having the words *lung* or *cancer*, which will return candidate phrases such as {*lung, chronic obstructive lung disease, lung cancer*, *liver cancer*} and so forth. Table 7 shows a partial list of candidate concepts generated based on the concept phrases having words in common with the source phrase *lung cancer*. The table shows the concept phrase which matched at least one word from the source phrase, as well as the corresponding concept's name and identifier. It is not required that a candidate phrase have all words in common, since exact mappings between a source phrase and concept phrases are expected to be rare.

Table 7: Example candidate phrase list generation

Source Phrase: *lung cancer*		
Concept Id	Concept Name	Concept Phrase
0024109	Lung	*Lung*
0024117	Chronic Obstructive Airway Disease	*Chronic Obstructive Lung Disease*
0242379	Malignant Neoplasm of the Lung	*Lung Cancer*
0684249	Carcinoma of the Lung	*Cancer of the Lung*
0279000	Liver and Intrahepatic Biliary Tract Carcinoma	*Liver Cancer*

If the source phrase is less than five words, then all words in the source phrase are used to find candidate phrases, as described above. If the source phrase is long (defined as consisting of five or more words), the number of candidate phrases generated by the words in the source phrase may be very large. One way to overcome this is to select only the most important words in the source phrase, and then use these words to select candidate phrases. The method of finding the most important words is to find the IPF weight of each word in the source phrase, calculate the standard deviation of the retrieved IPF weights, and then use all source phrase words whose IPF weight is greater than a threshold to find candidate phrases. The chosen threshold is one standard deviation of the mean IPF weight. Selecting greater than or equal to one standard deviation allows approximately 32% (Kiess, 2002)of the words in the source phrase to be used to retrieve candidate phrases. Figure 10 shows an example of the long source phrase *chronic obstructive pulmonary disease finding* having word IPF weights of {1.5, 2, 2, 1, 1}. The mean is 1.5, the standard deviation is 0.5, and the minimum IPF weight is therefore 2.0. To generate candidate phrases in this example, only the words {*obstructive*, *pulmonary*} are used since they are the most discriminative words based on their IPF weight. In the example, all concept phrases having either the words *obstructive* or *pulmonary* will be passed to the next filter (i.e., Coverage filter). The idea is to use the only the most important words in the source phrase to limit the number of candidate phrases retrieved. Each stage of the filter should seek to find a single best matching candidate phrase, and if not possible, reduce the number of candidate phrases passed to the next filter stage, which is assumed to be more computationally complex.

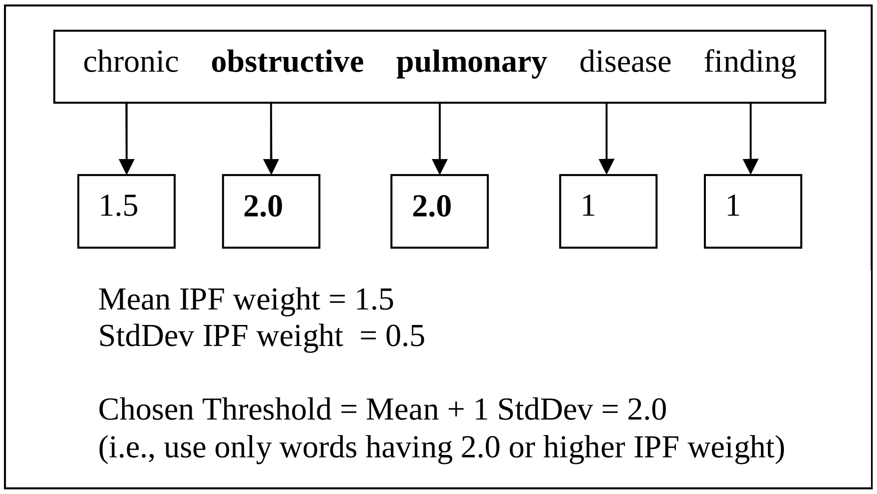

chronic **obstructive** **pulmonary** disease finding
1.5
2.0
2.0
1
1
Mean IPF weight = 1.5
StdDev IPF weight = 0.5
Chosen Threshold = Mean + 1 StdDev = 2.0
(i.e., use only words having 2.0 or higher IPF weight)

Figure 10: Example of long source phrase using IPF weights to find significant words

3.1.3 Coverage Filter

Once a list of candidate phrases is retrieved, coverage (overlap of words in common) is measured to filter out less important candidate phrases. The idea is to find the list of candidate phrases having the best coverage of the source phrase words, based on the IPF weight of each word in common between the source phrase and each candidate phrase. The Coverage score for each candidate phrase can be computed quickly using table lookup operations. In existing work, a coverage score for a candidate phrase is measured using a count of the number of words in common (Aronson, 1996), (Zou et al., 2003). In our research, weighted unigram filtering is used to measure coverage. The combined IPF weights of all words in common between a source phrase and a candidate phrase is used as the coverage score for each candidate phrase and is called the PhraseCoverageIPF weight, defined in Figure 11 as:

$$\boxed{\text{PhraseCoverageIPF} = \sum_{i=1}^{N} \text{IPF}_i}$$

Figure 11: Phrase Coverage Weight (PhraseCoverageIPF). *N* is the total number of words in common between the source phrase and the candidate phrase. *IPF* is the Inverse Phrase Frequency weight of a word *i* in common between the source phrase and the candidate phrase.

Once the PhraseCoverageIPF weights are computed for all candidate phrases, the standard deviation of the PhraseCoverageIPF weights for the set of candidate phrases is calculated. A threshold value is chosen as the mean IPF weight plus two standard deviations, which captures the top 5% (Kiess, 2002) of the highest-weighted candidate phrases. All candidate phrases whose PhraseCoverageIPF weight is greater than or equal to the threshold value are passed to the next filter (i.e., Coherence filter). There are two exceptions to consider: (a) if there is an exact match between a source phrase and one of the candidate phrases, the candidate concept associated with the candidate phrase is returned; and (b) if no candidate phrase has a PhraseCoverageIPF weight greater than or equal to the threshold, the candidate phrases with the highest PhraseCoverageIPF weight are passed to the next stage (if there is only one such candidate phrase with the highest PhraseCoverageIPF weight, its corresponding candidate concept is returned).

Examples of each case are shown in Tables 8 through 10. In each case, the threshold value is shown using one standard deviation in order to demonstrate the algorithm with a small dataset. In the actual coverage filter implementation, it was found that the threshold of two standard deviations was better at reducing the large number of candidate phrases generated during the initial phrase list generation stage. Table 8 shows an example of the coverage filter processing for the source phrase *lung cancer* where there is an exact match. The IPF weights for each word in the source phrase are first retrieved. Each candidate phrase from the candidate generation step is scored by summing the IPF weights for *lung* and *cancer* when they exist in the candidate phrase. After all candidate phrases are scored with this PhraseCoverageIPF weight, exact candidate phrase matches with source phrase PhraseCoverageIPF weight and source phrase word ordering are returned. Although the phrase coverage filter is not concerned with word order, exact matches are checked since they are considered the best matches possible. Exact matches are immediately accepted and are not required to meet the minimum coverage weighting threshold. In the Table 8 example, *Lung Cancer* is returned because it is an exact match both with the source phrase PhraseCoverageIPF weight and with the source phrase *lung cancer*. *Cancer of the Lung* is not returned because it is not an exact match with the source phrase, even though it has the same PhraseCoverageIPF weight as the source phrase. In Table 8, the mean PhraseCoverageIPF weight is 0.90, the standard deviation is 0.34, and the threshold value, the mean plus one standard deviation, is 1.24. Since an exact match was found, the threshold value is not used in this case.

Table 8: Example of Coverage filtering for exact match with source phrase

Source Phrase: *lung cancer*
(IPF weights: lung=0.75, cancer=0.50, total=**1.25**)

Candidate Phrase	PhraseCoverageIPF weight
Lung	0.75
Chronic Obstructive Lung Disease	0.75
Lung Cancer	**1.25** **← Exact Weight & String**
Cancer (of the) **Lung**	**1.25** **← Exact Weight**
Liver Cancer	0.50

Scoring Details:
Mean PhraseCoverageIPF weight = 0.90
StdDev of PhraseCoverageIPF weights = 0.34
Chosen threshold = Mean PhraseCoverageIPF weight + 1 StdDev = 1.24
There is an exact match between the source phrase and candidate phrase in PhraseCoverageIPF weight and also in the source phrase string for *Lung Cancer*.

Table 9 shows the case where there is no exact match between the source phrase and the candidate phrases. In this case, the function of the Coverage filter is to reduce the size of the candidate list. This is accomplished by retaining all candidate phrases whose PhraseCoverageIPF weight is greater than or equal to a threshold weight. The mean PhraseCoverageIPF weight is 0.90, the standard deviation is 0.16, and the threshold weight, the mean plus one standard deviation, is 0.58. Therefore, any candidate phrase which has a PhraseCoverageIPF weight greater than or equal to 0.58 is passed to the next filter. In Table 9 there are two candidate phrases having a PhraseCoverageIPF weight of 0.60, so both of these candidate phrases (shown in bold) are passed to the next filter (e.g., coherence filter). If only one candidate phrase had a PhraseCoverageIPF weight equal to or greater than 0.60, its candidate concept is returned and no candidate phrases are passed to the next filter.

Table 9: Example of Coverage filter with no exact match for source phrase

Source Phrase: *lung cancer disease*
(IPF weights: lung=0.30, cancer=0.30, disease=0.30, total=**0.90**)

Candidate Phrase	PhraseCoverageIPF weight
Lung	0.30
Chronic Obstructive Lung Disease	0.60 **>= 0.58**
Liver Cancer	0.30
Lung Cancer	0.60 **>= 0.58**
Cancer	0.30

Scoring Details:
Mean PhraseCoverageIPF weight = 0.42
StdDev of PhraseCoverageIPF weights = 0.16
Chosen threshold = Mean PhraseCoverageIPF weight + 1 StdDev = 0.58
Two PhraseCoverageIPF weights >= 0.58 are passed to the next filter:
Chronic Obstructive Lung Disease
Lung Cancer

Table 10 shows a variation of the no-exact-match case shown in Table 9, where there is no exact match between the source phrase and the candidate phrases, and none of the candidate phrases have a PhraseCoverageIPF weight greater than or equal to the threshold value. If the strict logic shown in Table 10 is followed, then no candidate phrases will be passed to the next filter. To resolve this, the approach used is to find the candidate phrases with the highest PhraseCoverageIPF weight. If only one such candidate phrase exists, its corresponding candidate concept is returned. In Table 10, the mean

PhraseCoverageIPF weight is 0.68, the standard deviation is 0.26, and the threshold weight, the mean plus one standard deviation, is 0.94. Therefore, any candidate phrase which has a PhraseCoverageIPF weight greater than or equal to 0.94 is passed to the next filter. As Table 10 shows, no candidate phrase has a PhraseCoverageIPF weight greater than or equal to 0.94. In this case, the highest PhraseCoverageIPF weight is found, which is 0.90. There are two candidate phrases having a PhraseCoverageIPF weight equal to 0.90, so the two corresponding candidate phrases are passed to the next filter (e.g., coherence filter). If only one candidate phrase had the highest PhraseCoverageIPF weight, its corresponding candidate concept is returned and no candidate phrases are passed to the next filter.

Table 10: Example of Coverage filter with highest scores used

Source Phrase: *lung cancer disease*
(IPF weights: lung=0.50, cancer=0.40, disease=0.40, total=1.30)

Candidate Phrase	PhraseCoverageIPF weight
Lung	0.50
Chronic Obstructive Lung Disease	0.90 **← Highest Weight**
Liver Cancer	0.40
Lung Cancer	0.90 **← Highest Weight**

Scoring Details:
Mean PhraseCoverageIPF weight = 0.68
StdDev of PhraseCoverageIPF weights = 0.26
Chosen threshold = Mean PhraseCoverageIPF weights + 1 StdDev = 0.94
(No PhraseCoverageIPF weight is >= 0.94, so candidate phrases with highest weights are passed to the next filter:
Chronic Obstructive Lung Disease
Lung Cancer

3.1.3.1 Extensions of the Coverage Filter

The base coverage filter can be extended to provide improvements in annotation precision. Figure 12 shows an extension of the coverage filter which incorporates two heuristics. The heuristics are designed to add candidate phrases to the list of top candidate phrases returned by the original Coverage filter without requiring the phrases to meet the minimum threshold value. The first heuristic is to compare the PhraseCoverageIPF weight of the candidate phrase to the summed IPF score of the source phrase. If the two weights are equal, a check is performed to see whether the two phrases are an exact match. If the two phrases match exactly, the candidate phrase is automatically added to the list of returned phrases. This first heuristic is identical to the base coverage filter, except that it does not enforce the minimum threshold value for exact candidate phrase matches with the source phrase.

The second heuristic is to add a candidate phrase to the list of phrases returned by the original Coverage filter if (a) it consists of a single word, and (b) the single candidate phrase word also appears in the list of source phrase words. During development of CONANN, it was discovered that single-word concepts were being filtered out, resulting in lower precision scores. The idea of maximally weighting a single-word candidate phrase is that the candidate phrase has no possibility of extra noise words, since the candidate phrase has only one word to describe its corresponding concept. Therefore, the single-word candidate phrase should always be considered. The inclusion of a single-word candidate phrases is particularly useful in the phrase-counting final mapper, which relies on counting the number of candidate phrases which map to the same concept.

```
Input:
 Source phrase words
 Candidate phrases

Output:
 Candidate phrases (filtered subset)

Procedure:
 Score Candidate Phrases:
 for each candidatePhrase
   for each candidatePhrase word contained in source phrase
     if candidatePhrase has only one word then
       set candidatePhrase score to MAX_VALUE
     else
       add word's IPF value to candidatePhraseScore

   Check for Exact Matches:
     for each candidate phrases whose candidatePhraseScore =
      PhraseCoverageIPF(sourcePhrase )
         if exactMatch(sourcePhrase, candidatePhrase)
           add candidatePhrase to returnedCandidatePhrases

   Filter:
     if no exact matches then
       add all candidate phrases to returnedCandidatePhrases whose
       minimum score is >= (averageScore + (2 * stdDevAverageScore))

     if no returnedCandidatePhrases
       add all candidate phrases to returnedCandidatePhrases whose minimum score is =
           highest candidatePhraseScore

 Return:
     output returnedCandidatePhrases
```

Figure 12: CoverageFilterExtended: An extended Coverage Filter algorithm utilizing the exact match and single-word candidate phrase maximal weighting heuristics.

3.1.4 Coherence Filter

A filter to measure coherence is introduced, where coherence is a measure of the order of terms in the phrase. Coherence is measured by looking at the order of the words in common between the source phrase and each candidate phrase. The idea is that the common syntactic ordering of the source and candidate phrases will remove candidate phrases which have some words in common but are in a different order, indicating the candidate phrase may be expressing a different concept than the source phrase. The Coherence filter uses, pairs of ordered words which allow for intervening words, known as skip-bigrams (C. Y. Lin & Och, 2004a). The skip-bigrams are generated by walking the candidate phrase words from beginning to end and pairing each word with the word that follows it. For example, the phrase *peripheral plasma cell myeloma* has the set of complete skip-bigrams as shown in Figure 13:

peripheral plasma
peripheral cell
peripheral myeloma
plasma cell
plasma myeloma
cell myeloma

Figure 13: Complete skip-bigram (i.e., no gap defined) for phrase *peripheral plasma cell myeloma*

The number of intervening words, called a gap, can be limited. The skip-bigram gap can be set from 0 to any specified number, and indicates the number of allowed intervening words. The lower the gap size, the more restrictive the order of words is enforced. For a given gap size *n,* the skip-bigrams generated include all skip-bigrams for lower levels of *n*. For example, a gap size of two will include skip-bigrams with gap sizes zero, one and two. Figure 14 shows the skip-bigrams generated for a gap size of zero:

peripheral plasma
plasma cell
cell myeloma

Figure 14: Skip-bigrams with gap zero for phrase *peripheral plasma cell myeloma*

A gap size of one produces the list shown in Figure 15 (which includes skip-bigrams of gap size zero as well as skip-bigrams of gap size one):

peripheral plasma
peripheral cell
plasma cell
plasma myeloma
cell myeloma

Figure 15: Skip-bigrams with gap one for phrase *peripheral plasma cell myeloma*

The skip-bigram statistic can be computed as precision and recall measures (C. Y. Lin & Och, 2004a), as shown in Figure 16. In both measures, the number of common skip-bigrams between the source phrase and a candidate phrase within a specified gap is computed. For precision, this common skip-bigram count is divided by the total number of skip-bigrams of the source phrase within a specified gap. For recall, the common skip-bigram count is divided by the total number of skip-bigrams of the candidate phrase within a specified gap. The skip-bigram precision measures the degree of skip-bigram matching with the correct phrase (i.e., source phrase), while the skip-bigram recall measures the degree of skip-bigram overlap with the retrieved phrase (i.e., the candidate phrase).

$$\text{Precision} = \frac{\text{CommonSkipBigramsWithinGap(SourcePhrase, CandidatePhrase}}{\text{CountSkipBigramsWithinGap(SourcePhrase)}}$$

$$\text{Recall} = \frac{\text{CommonSkipBigramsWithinGap(SourcePhrase, CandidatePhrase)}}{\text{CountSkipBigramsWithinGap(CandidatePhrase)}}$$

Figure 16: Skip-bigram Precision and Recall metrics. *CommonSkipBigramsWithinGap(SourcePhrase, CandidatePhrase)* is the number of the bigrams in common between the source phrase and the candidate phrase within the specified gap, and the *CountSkipBigramsWithinGap(somePhrase)* is the number of skip-bigrams within the specified gap distance of *somePhrase*. (C. Y. Lin & Och, 2004a)

For example, the source phrase *cancer of the lung* demonstrates how the skip-bigram filter works. According to a Metathesaurus search of the UMLS Knowledge Source Server (http://umlsks.nlm.nih.gov), there are two potential UMLS concepts for this phrase with concept identifiers and concept names *C0242379: Malignant Neoplasm of the Lung* and *C0684249: Carcinoma of the Lung*. In this example, each concept has several concept instances but we assume only one concept instance is chosen as the candidate phrase for each concept. The candidate phrase for concept C0242379 is *Lung Cancer*. The candidate phrase for concept C0684249 is *Cancer of the Lung,* which becomes *Cancer Lung* after stop word removal. The source phrase after stop word removal is *cancer lung*. The skip-bigrams with a gap of zero for C0242379 is *lung cancer*, for C0684249 is *cancer lung* and for the source phrase is *cancer lung*. The skip-bigram recall score for C0242379 is 0 (0/1) while the skip-bigram recall score for C0684249 is 1 (1/1). Therefore, the returned UMLS concept is C0684249: *Carcinoma of* *the Lung*. Figure 17 shows an example of how the skip-bigram filter works using the source phrase *cancer of the lung*.

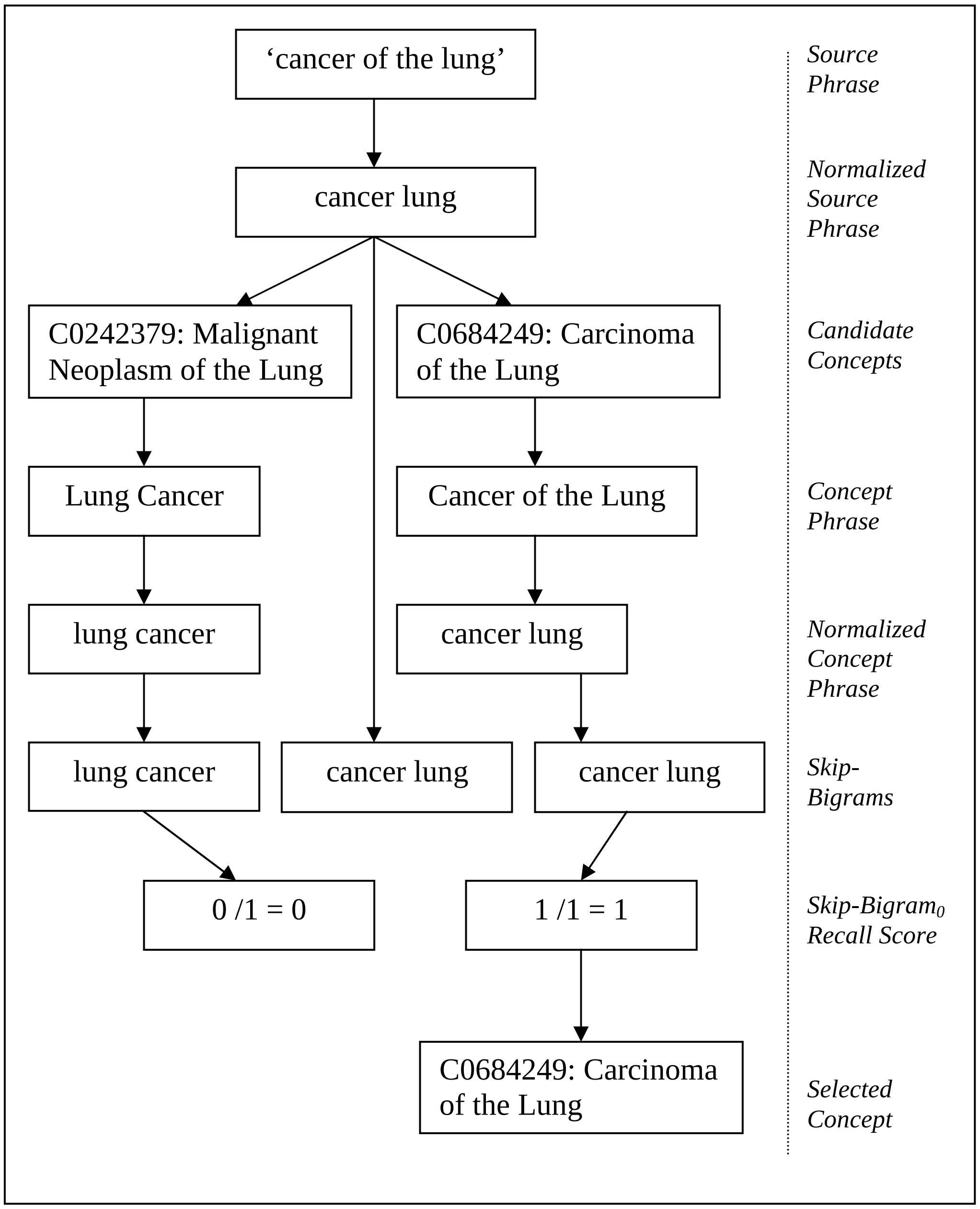


Figure 17: Example skip-bigram filtering

The performance of skip-bigrams has been evaluated in machine translation evaluation and summary evaluation, and has been shown to perform at or above state-of-the-art measures with less complexity (C. Y. Lin & Och, 2004b). CONANN uses the Recall measure, since it has been shown in machine translation evaluation research that n-gram recall is the biggest factor in evaluations using n-gram measures (Lavie et al., 2004). In addition, the original SAPHIRE system used a high-precision approach to match all words in the source phrase in their original order, and found that this resulted in missed concept mappings (W. Hersh & Leone, 1995). The original SAPHIRE system used exact word order to try and eliminate false-positive matches where all words appeared in a phrase but in a different order resulting in finding a different meaning than the source phrase intended. In the current implementation, CONANN calculates the skip-bigram recall scores for all candidate phrases using a complete skip-bigram that is less restrictive than the high-precision approach of SAPHIRE. Whereas the original SAPHIRE system required exact word order, CONANN allows for gaps between words. Allowing for gaps between words imposes word order as a requirement, but does not completely require all words to match between the candidate phrase and the source phrase. This enforcement of word order while allowing intervening words is the primary advantage of using the skip-bigram approach. The concept associated with the highest-scoring candidate phrase is returned. If there are ties in candidate phrase scores, the concepts associated with the tied candidate phrases are returned.

3.1.5 Additional Filters

The annotation system is not limited to the two filters presented above. The coverage and coherence filters were designed to measure word membership and word order. Additional filters can be added, and may even modify the coverage and/or coherence filters with additional heuristics. An example of an additional filter is concept disambiguation. The concept disambiguation filter could be implemented by using UMLS-provided concept co-occurrence information. The general idea would be to make two passes over the source text, annotating first all unambiguous concepts, and then using the unambiguous discovered concepts along with UMLS co-occurrence metrics to disambiguate remaining concepts.

3.1.6 Final Concept Mapping

The selection of the concept(s) which best match the source text phrase is the final step in CONNANN. The core problem is to take the list of candidate phrases remaining after all filters have been applied, and pick the best matching concept(s) based on the remaining candidate phrases. Two different approaches are currently implemented and have been evaluated in CONANN:

Candidate Phrase Counting: If more than one candidate phrase remains in the list of candidate phrases after all filters have been applied, the candidate phrases are passed to a final stage to perform text-to-concept mapping (or simply concept mapping). Final concept mapping finds the best matching candidate phrase among the remaining candidate phrases. The Candidate Phrase Counting approach is to sum the number of candidate phrases belonging to each UMLS concept, and then choose the concept(s) with the largest number of candidate phrases. The final mapping of a source phrase to a UMLS concept is performed after the coverage and coherence filters have been applied to a list of candidate phrases. The remaining candidate phrases are then grouped by the concepts they belong to. Each candidate concept is then scored based on the number of candidate phrases it contains. The highest scoring candidate concept is then output as the concept for the source phrase. In the event of tie scores, multiple candidate concepts can be output. The idea is that the number of candidate phrases per concept after filtering gives an indication of the matching likelihood of a source phrase to a concept.

Language Model: CONANN uses a multinomial unigram language model to find the concept most likely to have generated the source phrase. A list of candidate phrases is first retrieved from the output of the coverage filter (see Section 3.4). Each candidate phrase belongs to one or more concepts. A list of concepts is generated from the candidate phrases to form a set of candidate concepts.

Each candidate concept is assigned a score based on its probability of generating the source phrase. The probability score is calculated as shown in Figure 18, which is a standard unigram language mixture model (Manning, Raghavan, & Schütze, 2007) which combines a source phrase word (*w*) probability within the concept language model ($M_{concept}$) with the source phrase word probability of the entire UMLS phrase collection ($M_{conceptCollection}$). We initially set λ=0.5 to balance the concept language model with the collection model. The extreme values of λ=0.1 and λ=0.9 were also evaluated, but did not notice any change in the final concept annotation output. To allow for more word variation, the source phrase words were expanded to include all source phrase word variants (provided by the UMLS resources) of each source phrase word. For example, the source phrase word *lung* would be expanded to include the word variant *pulmonary*. Any concept instance including the word *pulmonary* would then be added to the candidate list.

$M_{conceptCollection}$ is calculated as part of pre-processing and contains the probability of UMLS words occurring across all UMLS phrases. $M_{concept}$ is the language model for a particular concept based on the concept instances for the particular concept. For example, the concept *Lung Cancer* might contain the concept instances {*lung cancer*, *pulmonary carcinoma*}. $M_{concept}$ for the concept *Lung Cancer* is the language model constructed from these two concept instances.

Each candidate concept is assigned a score by retrieving the $M_{concept}$ and $M_{conceptCollection}$ probabilities of each source phrase word and applying the retrieved probability values as shown in Figure 18. The idea is to get the probability that the concept generated the source phrase, using each concept's language model which is composed of one or more concept instances defined by domain experts. The highest-probability candidate concept is then output as the best-matching concept for the source phrase. In the case of ties, all of the highest-scoring concepts are output.

$$\boxed{P(SrcPhrase \mid M_{concept}) = \prod_{w \in SrcPhrase} ((1-\lambda)P(w \mid M_{concept}) + \lambda P(w \mid M_{conceptCollection}))}$$

Figure 18: Multinomial Unigram Language Mixture Model (*w* denotes a word of the source phrase, *SrcPhrase*, including the word variants of each source phrase word; $M_{concept}$ is the language model for a specified concept, and $M_{conceptCollection}$ is the language model for all UMLS concepts)

3.2 Biomedical Text Summarization

For biomedical text summarization, two different approaches are proposed, designed, and implemented to use concepts rather than terms to identify salient portions of text within a source text. The first is concept chaining, which uses ideas from existing research work done in lexical chaining and applied to biomedical text summarization applications. The basic idea is to link together the concepts found in a text based on semantic types. The semantic types with the strongest chains are then representative of the text's main topics. The second approach uses the idea of reiteration, where an author repeats important points. Reiteration is reflected in the frequency of content terms used. Instead of using terms, the use of frequently occurring concepts is chosen. The novel contributions of the text summarization component of the system are as follows: (a) the use of concepts and an associated semantic network to chain concepts together to find text themes (BioChain); (b) the use of concept, rather than term, frequency to identify text themes; (c) the development of a text summarization algorithm which matches the concept or term distribution of the source text to the generated summary (FreqDist algorithm); (d) the determination of the optimal length of a summary; and (e) the identification of the location within texts human summarizers draw text.

3.2.1 Biomedical Text Summarization Using Concept Chaining (BioChain)

This section describes a novel summarization system called *BioChainSumm*, which utilizes the concept chaining approach (called *BioChain*). BioChain applies the concepts and methods of lexical chaining to biomedical text using concepts rather than terms.

BioChain uses identified biomedical concepts in the source text and chains them based on their biomedical semantic type(s) (see Section 1.6). Figure 19 shows the flow of the BioChainSumm text summarizer utilizing the BioChain concept chaining method. The basic idea is to first identify the strongest chains (as indicated by the number and type of concepts found in the source text), and then score each sentence in the source text by counting the number of strong chain concepts each sentence contains. The highest-scoring sentences are then extracted to form a summary.

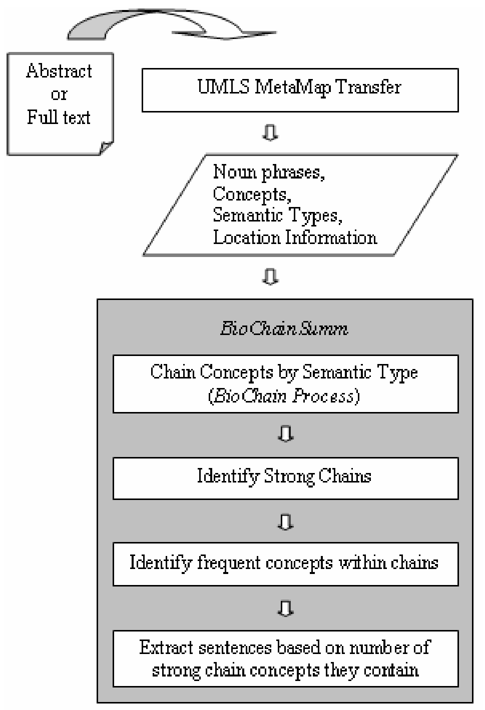


Figure 19: Text summarization process for BioChainSumm.

A concept chain is created for each semantic type defined in the UMLS Semantic Network (135 total). A concept chain is also called a semantic type chain or a semantic chain. Each entry in a semantic type chain contains a list of concepts belonging to the semantic type. Each concept entry in a semantic chain contains a concept name, a concept identifier, a sentence number, a section number (roughly paragraph), and a source noun phrase. A concept entry is constructed for each instance of a concept found in the source text. If a concept belongs to multiple semantic types (i.e., multiple concept chains), the concept appears in multiple chains.

Once all concepts within a source text have been identified and linked into semantic type chains, the chains are then scored to identify the strongest chains. Each chain is scored by multiplying the frequency of the most frequent concept in the chain by the number of distinct concepts in the chain. This formula incorporates a combination of features as proposed by (W. P. Doran, Stokes, Dunnion, & Carthy, 2004) and Barzilay/Elhadad (Barzilay & Elhadad, 1997).

Once all chains are scored, strong chains, which identify the semantic types occurring most often, are determined. Lexical chaining research generally uses two standard deviations above the mean of all chain scores (Barzilay & Elhadad, 1997), and that method is also used in BioChainSumm. The strong chains are sorted into descending order based on their score. Strong concepts within the strongest chains are then identified using two different methods: (a) most frequent concept within each chain (multiple concepts having the same frequency count are considered equal) (abbreviated as MostFrequentStrongChainConcept), and (b) all concepts within a chain (abbreviated AllStrongChainConcepts). Sentences from the source text are then scored based on the number of strong concepts they contain. After sentences have been scored, sentences are sorted into descending order based on their score. The top-*n* sentences in the sorted list are extracted, re-sorted into their order of appearance in the original text, and presented to the user. Figure 20 presents the pseudo-code for the summarization algorithm, which consists of several stages: concept mapping, concept chaining, strong chain identification, and sentence scoring and extraction. Each major stage in the process is detailed in the following sections.

```
BioChain(source-sentences, summary-size)

Concept Chaining:
    FOR EACH concept and semantic type found
        APPEND the concept to the semantic type chain

Strong Chain Identification:
    // Score each chain
    FOR each semantic type chain
        FIND the concept with the highest frequency
        FIND the number of distinct concepts within the chain
        SET the chain score to concept frequency count * number of distinct concepts

    // Find minimum score required to be a strong chain
    COMPUTE ChainScoreAvg = Avg(all chain scores)
    COMPUTE ChainScoreStdDev = StdDev(all chain scores)
    COMPUTE StrongChainMinScore = ChainScoreAvg + (2 * ChainScoreStdDev)

    // Find all strong chains
    FOR EACH semantic type chain
        IF (chain score >= StrongChainMinScore) THEN Chain is a strong chain

Sentence Scoring and Extraction:
    // Score sentences
    //      Two variations
    //        Variation #1: use most frequent concept in the strong chain
    //        Variation #2: use all concepts in the strong chain
    FOR EACH sentence
        IF (sentence CONTAINS strong chain concept) THEN
            INCREMENT sentence score by number of
                strong chain concept instances in sentence

    // Sentence Extraction
    //      Note: N is the number of sentences to output
    SORT sentences into descending order by sentence score
    EXTRACT top N sentences
    SORT top N sentences into original appearance order
    PRESENT top N sentences as summary
```

Figure 20: BioChainSumm summarization algorithm

3.2.1.1 Concept Chaining (BioChain)

Concepts are identified using the UMLS MetaMap Transfer application (United States National Library of Medicine, 2005b), and then chained based on their semantic type(s). A concept chain is created for each semantic type defined in the UMLS Semantic Network (135 total). Each concept chain contains a list of concepts belonging to the semantic type. Each concept entry in a concept chain (or semantic chain) contains a concept name, concept identifier, sentence number, section number (roughly paragraph number), and source text noun phrase. If a concept belongs to multiple semantic types (i.e., multiple concept chains), the concept appears in multiple chains.

3.2.1.2 Identification of Strong Chains

There has been no definitive measure for scoring chains, and the literature suggests changes in scoring methodology do not adversely impact chaining results (W. P. Doran et al., 2004). There are three types of strong chain features: (a) reiteration, (b) density, and (c) length (Morris & Hirst, 1991). Reiteration is repetition of concepts throughout a text. Density is physical proximity of concepts; that is, concepts closer together are more likely to be related. Length is the number of concept instances within a chain. The chosen scoring method, shown in Figure 21, includes a combination of features as proposed in (W. P. Doran et al., 2004) and (Barzilay & Elhadad, 1997). Once all chains are scored, strong chains, which identify the semantic types occurring most often in the source text, are computed. Lexical chaining research generally uses two standard deviations above the mean of all chain scores (Barzilay & Elhadad, 1997), as shown in Figure 22.

*Score(Chain) = Frequency of most frequent concept * number of distinct concepts*

Figure 21: Chain scoring

*Strong(Chain) = Score(Chain) > (Average(Scores) + 2 * StandardDeviation(Scores))*

Figure 22: Strong chain identification

3.2.1.3 Identification of Frequent Concepts and Summarization

Summarization identifies sentences most likely capture the main ideas of a text. BioChainSumm uses concept chaining to first identify the main themes of a biomedical text and then the sentence extraction method to generate a summary. Sentence extraction begins by first sorting the strong chains into descending order based on chain score explained in Section 3.2.1.3. In each strong chain, either the most frequent concept in the chain (MostFrequentStrongChainConcept) or all of the concepts (AllStrongChainConcepts) are used to score sentences. Each sentence is assigned a score based on how many concepts it contains from each strong chain.

For MostFrequentStrongChainConcept, a sentence is scored higher if it contains the most frequent concept from a strong chain. Multiple concepts having the same frequency count are considered equal. When using AllStrongChainConcepts, a sentence receives a higher score if it contains any of the concepts from a strong chain. Each strong chain concept found increases the sentence score value by one. Once all sentences have been scored, the sentence list is sorted into descending order based on the computed sentence score. The top-*n* sentences are then extracted, where *n* is a user-defined upper bound on the number of sentences to select for a summary of the original text. After the specified number of sentences has been extracted to form a summary, the sentences in the summary are re-sorted into their order of appearance in the original text, and presented to the user.

3.2.2 Biomedical Text Summarization Using Concept Frequency Distribution (FreqDist)

A new summarizer based on the FreqDist concept frequency distribution algorithm, FreqDistSumm, is proposed, designed, and implemented. FreqDist is a frequency-based and redundancy-sensitive algorithm. The FreqDistSumm summarizer creates a summary of a source text which has approximately the same frequency distribution of concepts as the source text. Figure 23 shows an outline of the *FreqDist* algorithm to generate a summary given the full-text of some source (source text) using a frequency distribution approach. The basic idea of the frequency distribution approach is that the frequency distribution of terms or concepts in the source text and the generated summary should be as similar as possible. There are two stages in the summary generation process: Initialization and Summary Generation. In the initialization stage, the unit items (terms, concepts, etc.) of the source text are counted to form a frequency distribution model of the source text. In our research, the focus is on concepts as unit items. Concepts are identified using the UMLS MetaMap Transfer application (United States National Library of Medicine, 2005b). A pool of sentences from the source text is also created. A summary frequency distribution model is duplicated from the source text's frequency distribution model. The frequency of the unit items in the summary frequency distribution model are initially set to zero because the summary is initially empty. In the Summary Generation stage, new sentences are selected to be added to the summary. Identifying the next sentence to be added to the summary is accomplished by finding the sentence which most closely aligns the frequency distribution of the summary to the frequency distribution of the original source text. A candidate summary is first initialized to the summary generated so far. For each sentence in the sentence pool, the sentence is added to the candidate summary to see how much it contributes to the candidate summary. To determine the sentence’s contribution, the candidate summary frequency distribution is compared for similarity to the source text’s frequency distribution. The comparison generates a similarity score. This similarity score is assigned to the sentence as the sentence score. After all sentences from the sentence pool have been scored (evaluated for their contribution to the candidate summary), the highest scoring sentence is added to the summary and removed from the sentence pool. This process is iterative, and repeats until the desired length of the summary is reached.

```
FreqDist(source-text, important-sentences, summary size)
Initialization:
// Note: '-model' means  'frequency distribution model'
INITIALIZE source-model to unit-items in source-text
INITIALIZE summary-model,
        candidate-model from source-model
  SET all frequency values to 0

INITIALIZE sentence-pool to source-text sentences

Summary Generation:
REPEAT
  INITIALIZE sentence-pool scores to 0
  INITIALIZE best-score to 0
  INITIALIZE best-sentence to first sentence in pool

  FOR each sentence-entry in sentence-pool
    INITIALIZE candidate-model from summary-model

    ADD sentence unit-item frequencies to candidate-model

    SET sentence-entry.score =
        similarity(source-model, candidate-model)

    IF sentence-entry.scorescore > best-score
      SET best-score to sentence-entry.score
      SET best-sentence to sentence-entry
    ENDIF
  ENDFOR

  ADD unit-items from best-scoring sentence
        to summary-model

  REMOVE best-sentence from sentence-pool

UNTIL desired summary size reached or
        sentence-pool exhausted
```

Figure 23: FreqDist - an algorithm for generating summaries using a frequency distribution approach.

Five similarity functions were compared to find which type of function worked best to evaluate a candidate summary's frequency distribution to the original source text frequency distribution. Each frequency distribution (candidate summary and original source text) is modeled as a vector of unit items. Similarity functions are then applied to the two vectors. Figure 24 shows the five similarity functions used. The notations are as follows: *ui* is unit item; *srcUIs* and *sumUIs* is all unit items in source text and candidate summary, respectively; *src(ui)* and *sum(ui)* is indexed unit item in the source text and candidate summary, respectively. Cosine similarity (Baeza-Yates & Ribeiro-Neto, 1999), Dice's coefficient (Dice, 1945), Euclidean distance and vector subtraction (Subhash, 1996) are all well-known vector comparison methods. In addition, an approach to vector model comparison considering only unit item frequency was tried (D. L. Lee, Chuang, & Seamons, 1997). Cosine similarity uses the cosine angle value between the vectors for similarity. Dice's coefficient looks at the number of common terms between the two vectors. Euclidean distance measures the distance between the vectors in Euclidean space. For vector subtraction, the absolute value of the difference of each unit item in each vector is summed to form a distance score. The unit item frequency approach attempts to simulate cosine similarity without the computational complexity by only considering unit item frequency (D. L. Lee et al., 1997).

$$\boxed{score = \frac{\sum_{ui=1}^{srcUIs} sum(ui) \times src(ui)}{\sqrt{\sum_{ui=1}^{srcUIs} sum(ui)^2 \times \sum_{ui=1}^{srcUIs} src(ui)^2}}}$$

(a) Cosine similarity

$$\boxed{score = \frac{2 * count(srcUIs \cap sumUIs)}{count(srcUIs) + count(sumUIs)} |}$$

(b) Dice's coefficient

$$\boxed{score = (\sum_{ui=1}^{srcUIs} (sum(ui) - src(ui))^2)^{1/2}}$$

(c) Euclidean distance

$$\boxed{score = \sum_{ui=1}^{srcUIs} |(src(ui) \times sum(ui))|}$$

(d) Unit item frequency

$$\boxed{score = \sum_{ui=1}^{srcUIs} |(src(ui) - sum(ui))|}$$

(e) Vector subtraction

Figure 24: Similarity functions to evaluate a candidate summary's frequency distribution to the original source text frequency distribution. (a) cosine similarity, (b) Dice's coefficient, (c) Euclidean distance, (d) unit item frequency, and e) vector subtraction. Notations used: *ui* is unit item; *srcUIs* and *sumUIs* is all unit items in source text and candidate summary, respectively; *src(ui)* and *sum(ui)* is indexed unit item in the source text and candidate summary, respectively.

3.2.3 Biomedical Text Summarization Combining BioChain and FreqDist (ChainFreq)

The BioChain and FreqDist algorithms use different approaches for identifying relevant sentences for building an extractive summary. A problem not addressed in the current BioChainSumm summarizer is reducing information redundancy. Sentences containing the strongest concepts in the text are extracted without a complimentary method for reducing redundancy from sentences already selected. To overcome this limitation, the BioChain and FreqDist algorithms are combined to form a hybrid algorithm, called ChainFreq. The hybrid ChainFreq algorithm first uses the BioChain algorithm to identify candidate sentences containing strong concepts. The candidate sentences (*Sc*) and their corresponding concepts (*Cc*) are then passed to the FreqDist algorithm, which produces a set of summary sentences from the candidate sentences. A summary frequency distribution model is then created from the *Cc*, and the frequency counts are initialized to zero. The FreqDist algorithm then selects sentences containing concepts in the same distribution as the original source text with respect to *Cc* which reduces redundancy to the same proportion it exists in the source text.

Figure 25 shows how the two summarization methods, BioChain and FreqDist, are combined to form the new hybrid summarizer, ChainFreqSumm. First, all source sentences with their corresponding concept annotations are collected and passed to the BioChain algorithm. Concepts are identified using the UMLS MetaMap Transfer application (United States National Library of Medicine, 2005b), The BioChain algorithm takes advantage of domain-specific knowledge, specifically UMLS semantic types, to find sentences which are important in the domain. There is no limit on the number of sentences generated by the BioChain algorithm. The subset of source-text sentences identified by the BioChain algorithm are then passed to the FreqDist method. The FreqDist method then finds a further subset of sentences whose concept distribution best aligns with the concept distribution of the source text. A user-defined summary size limits the number of sentences output at this stage. Both the BioChain algorithm and the FreqDist algorithm work together to (a) find the important sentences according to the domain (using the BioChain algorithm), and (b) reduce redundancy by further reducing the number of important sentences based on how well their concept distribution aligns with the source text's concept distribution (using the FreqDist algorithm), which has the effect of reducing redundancy.

```
Initialization:
  INITIALIZE source-sentences to source-text sentences;
  INITIALIZE important-sentences to NULL;

Summary Generation:
  important-sentences = BioChain(source-sentences, 100);

  important-sentences = FreqDist(source-text,
                                 important-sentences,
                                 summary-size);

RETURN  important-sentences as final summary;
```

Figure 25: Hybrid summarization method *ChainFreqSumm* using the *BioChain* method to identify sentences, and the *FreqDist* method to remove redundancy.

3.2.4 Update Task in DUC 2007

In this subsection the FreqDistUpdate system used in the DUC 2007 update task is described. FreqDistUpdate is a first entry in the DUC evaluations. FreqDistUpdate uses ideas from the FreqDist text summarizer (FreqDistSumm). FreqDistSumm has been shown to perform well in biomedical text summarization, and this is a first adaptation to use it within a general domain. While FreqDistUpdate did not perform well in the automated evaluation scores, it did perform better in the manual evaluation. The frequency distribution method is a promising approach for the update task. Improvements in implementation and approach will likely lead to better performance in future DUC evaluations of the update task.

In the DUC 2007 update task, systems are asked to produce short summaries of newswire articles, assuming a user has read a set of previous, related article texts. The idea is to present new information that the user has not already read from the set of preceding article texts. This task is an appropriate place to test the existing FreqDist algorithm in a new way and in a new domain. The basic idea behind FreqDistSumm is to create a summary which has approximately the same frequency distribution of unit items (i.e., terms or concepts) as the source text. In this way, the summary captures the expressions of a text in the same degree they are expressed in the source text. This approach has worked well in biomedical text summarization work (Reeve et al., 2006).

For the update summarization task, three summaries were generated for each topic. The summaries are based on three document sets labeled A, B, and C. Summary generation is done by (a) reading the sentences from all documents in a document set, (b) determining the frequency distribution of all terms within the document set, and then (c) building a summary so that the summary term frequency distribution is as close as possible to the current document set’s term frequency distribution. To account for information accumulated from a previous summary, the summary term frequency distribution is initialized to the previous summary’s term frequency distribution. Sentences from the current document set are then scored based on how well they presented new information (terms) as compared to the previous summary.

The update summarization task required the generation of three 100-word multi-document summaries for each of ten topics. Within each topic, there are three document clusters labeled A, B, and C. Each document cluster is chronologically ordered and contains approximately ten documents related to a topic. The task is to generate three summaries from the contents of each document set given a topic statement (information need). Summary A summarizes the texts in document cluster A. Summary B summarizes the texts in document cluster B assuming the reader already has the information from the documents in document cluster A. Summary C proceeds the same way, assuming the reader has already read the documents in document clusters B and C. There will be approximately 10 topics in the test data, with 25 documents per topic.

FreqDistSumm began by first constructing a list of important words from the topic statement. The topic statement words were generated with a simple method which first replaced a known set of delimiters, defined as {(,), ;, :}, with spaces. The topic sentence was then split into words based on a space character as the delimiter. Semantically unimportant words, such as ‘a’ and ‘the‘, were removed from the list. The words remaining in the important word list served to boost the scores of these words if they were found in the texts within a document cluster.

For Document Clusters A, B, and C, all documents within each cluster were read and parsed into sentences using the LingPipe sentence chunker (Carpenter & Baldwin, 2007). The sentence chunker was initialized to use the Indo-European sentence model, which was provided as part of the LingPipe toolkit. The sentences from all documents in the cluster were combined to form a single list representing all sentences within the cluster. The result of the reading and parsing was three lists of sentences, one for each cluster.

For Document Cluster A, the summarizer was then passed the list of sentences and the list of important words. The first step in the summarizer was to initialize all of the sentences with a score of zero. A hash table containing all words in the sentence list and their frequency counts was generated. The base FreqDist algorithm shown in Figure 23 was then applied. Several modifications to the algorithm were done to account for important words from the topic statement and also the 100-word maximum summary length requirement. Important words within each sentence were counted. If a sentence did not contain one or more important words, it was penalized so that it chance of being selected was very low. The idea was to select sentences which had words in common with the topic statement. For summary length, a sentence was not selected unless its length plus the length of the summary generated so far was less than 100 words. The result is that a lower-scoring sentence would be selected if a higher-scoring sentence caused the summary length to exceed 100 words. Once all sentences were selected, they were sorted into their original order of appearance and a summary was generated.

In Document Cluster B, the same basic approach was applied, but with Document Cluster A being passed as a parameter to the summarizer, in addition to the set of cluster B sentences and important words. The words from the Document Cluster A sentences were used to prime the frequency distribution of the summary to be generated. The idea is to account for frequencies of words have already been seen and selected, so that the likelihood of words from the Document Cluster A summary being selected again in the Document Cluster B summary will decrease.

Finally the summary for Document Cluster C was done identically to the summary for Document Cluster B, except for using sentences from cluster C as the source text input.

4. EVALUATION

This chapter provides information about the semantic annotation and text summarization evaluation methodologies. The evaluations are done in an automated fashion. Text summarization is evaluated by measuring n-gram overlap of a system-generated summary to several model summaries generated by domain experts. The CONANN semantic annotator is evaluated against the concept output of a state-of-the-art biomedical concept annotator as well as its performance in identifying salient sentences for text summarization.

4.1 Background of Evaluation Corpus

To provide a set of data for evaluating semantic annotations as well as summary performance, a corpus of 24 biomedical texts was generated from a citation database of oncology clinical trial papers. The database contains approximately 1,200 papers physicians feel are important to the field (Brooks & Sulimanoff, 2002). Of the 1,200 papers cited, 24 were randomly selected. The number of papers chosen (24) was based on the minimum requirements of the ROUGE summary evaluation tool (C. Lin, 2004) as well as the resources available to complete the manual processing of each paper. The PDF versions of these 24 papers were then obtained and converted to plain-text format. The papers were then manually processed to remove graphics, tables, figures, captions, citation references, and the bibliography section. The resulting texts were further split into an abstract text and a full-text source text (without the abstract).

A corpus of model summaries was provided by the Drexel University College of Medicine. Each of the 24 texts was summarized by three different domain experts, resulting in three model summaries for each of the 24 texts. The domain experts are medical students in their final year of study. A model summary is a summary written by a person representing that person’s version of an ideal summary of the full-text source. The task presented to each human summarizer was to select 20% of the sentences within each full-text source to form a general summary of the full-text source. In effect, the human summarizers are performing the same extractive task as the system summarizer. An automated evaluation can then be done to compare a system-generated summary of a full-text source to the model summaries of the same full-text source.

Three human summaries of each text were also generated at a 20% compression rate using sentences from the original source text. The sentences are identified by their section location in the source text (e.g., Introduction, Methods, Results, Discussion and Appendix). In addition, each sentence is ranked for its importance in contributing to the manual summary.

To develop a corpus for semantic annotation, the 24 papers are processed by MetaMap to find all noun phrases in the 24 papers, resulting in a corpus of 4,410 unique phrases. The corpus was pruned to retain only those phrases which MetaMap annotated with a single concept, allowing for meaningful mapping comparisons between the two systems, MetaMap and CONANN. There were 1,628 phrases with a single MetaMap concept annotation. This set of phrases was used to perform the evaluation. As a baseline, the precision of MetaMap concept annotation is assumed to be 100%.

4.2 Biomedical Semantic Annotation Evaluation

Evaluation of the annotation system is done using intrinsic and extrinsic evaluations. The intrinsic evaluation is done by comparing CONANN’s concept output to the concept output of the MetaMap system (Aronson, 2001a), and determining precision and recall values. The extrinsic measure evaluates the performance of CONANN’s concept output on a text summarization task.

4.2.1 Intrinsic Annotation Evaluation

The MetaMap system output is used as baseline to measure against. MetaMap takes a phrase as input and generates the best matching UMLS concept(s). CONANN’s annotator output for the same phrase is then generated and compared to the concept(s) generated by MetaMap. To measure the amount of time it takes for MetaMap to annotate the test corpus of phrases, MetaMap was executed using the 1,628 phrases as input. The MetaMap API (Devita, 2006) is used to annotate each phrase. MetaMap provides various APIs to annotate different text chunk sizes, including document, document section, sentence, or term. The term method is used so that MetaMap does not need to expend effort finding phrase boundaries, as it would do if passed a document, document section, or sentence to annotate. CONANN is then executed against the same set of 1,628 phrases and its annotation time measured. CONANN also produces concept annotations for the list of phrases. These mappings are then compared to MetaMap, producing the annotation precision metric. Three runs of each system were performed, and the system restarted after each run to remove variations caused by the operating environment, such as file system caching.

Accuracy is measured by comparing CONANN’s annotation of each phrase to the MetaMap’s annotation output for each phrase. There are two measures for the intrinsic evaluation: (a) precision, and (b) phrase annotation time. The first measure looks at the accuracy of the concept annotation, and the second measure looks at the speed of the concept annotation. The Annotation Precision measure uses the same idea as in the precision measure in information retrieval, but adapted to fit concept mapping (W. R. Hersh et al., 2001). Annotation Precision is defined as the fraction of mapped concepts which are correct, as shown in Figure 26. In this evaluation, two types of matching are used. *Single Concept* matching counts a correct match only if CONANN directly generates a single concept which exactly matches the MetaMap single concept. In *Relaxed Matching,* CONANN generates five top concepts. A correct match is counted if any of the five concepts generated by CONANN match the MetaMap single concept. The idea is to see if the correct MetaMap concept is among the highest-scoring CONANN concepts. Recall is not considered because the source phrase corpus that is correctly annotated by MetaMap is only provided to CONANN to annotate, and so recall is not meaningful for this evaluation. For measuring speed, the average time to annotate a phrase is used. This measure is calculated by taking the total annotation time divided by the total number of phrases annotated. Annotation time is defined as the time it takes to annotate a single phrase, and does not include the annotator initialization. Total annotation time is the time it takes to annotate all phrases in the corpus, excluding annotator initialization.

$$\boxed{Precision = \frac{\#of\ correct\ concepts}{total\,\#\ of\ concepts\, mapped}}$$

Figure 26: Annotation Precision metric

For the coverage filter, two different candidate phrase scoring approaches are used: Naïve and Involvement. The Naïve approach (shown in Figure 27) simply sums the word weights and assigns the resulting sum as the candidate phrase score. The word weights are either zero or one for binary weighting, or the Inverse Phrase Frequency (IPF) values if using the IPF approach. The use of two different word weights allows for contrasting the performance of using Inverse Phrase Frequency weighting with binary weighting in the scoring of candidate phrases.

$$\boxed{NaiveCandidateScore = \sum_{i=1}^{N} WordWeight_i}$$

Figure 27**:** Naive candidate scoring method. *N* is the number of words in common between a source phrase and a candidate phrase. *WordWeight* is either 0 or 1 for binary weighting or the inverse phrase frequency value for IPF weighting.

A second method for scoring candidate phrases comes from the MetaMap Transfer system and is called Involvement (Aronson, 2001b). Involvement first computes the normalized word weights of words in common between a source phrase and a candidate phrase in both directions (phrase involvement), and then averages the two phrase involvement values to determine the candidate phrase score. For example, consider the source phrase words {A, B, C} and candidate phrase words {A, B} and assume binary word weights. The candidate phrase involvement is 2/2, since candidate phrase words {A,B} are contained in the source phrase. The source phrase involvement is 2/3 since the source phrase words {A,B} are also in the candidate phrase words, but source phrase word {C} is not. The candidate phrase score is then the average of (2/2 + 2/3)/2, or 0.83. As in the Naïve scoring method, the Involvement score is calculated using binary values (as in the example), and also using IPF values. The formal calculation for the involvement score is shown in Figure 28.

$$\boxed{Involvement = \frac{\dfrac{\sum_{i=1}^{NC} WordWeight_i}{|CandidatePhraseWords|} + \dfrac{\sum_{i=1}^{NS} WordWeight_i}{|SourcePhraseWords|}}{2}}$$

Figure 28: Involvement Score. *NC* is the number of words in the candidate phrase that are in the source phrase, *NS* is the number of words in source phrase that are in candidate phrase, and *WordWeight* is either: 0 or 1 for binary weighting, or inverse phrase frequency value for IPF weighting. (Aronson, 2001b)

4.2.2 Extrinsic Annotation Evaluation

The output of a concept annotator is a list of phrases and their associated domain-specific concepts. This output is an intermediate format, not directly useable by an end-user. The extrinsic evaluation is a complimentary evaluation to the intrinsic, designed to show the usefulness of the concept output in some task. Text summarization was selected as the end-user task. The use of text summarization as the end-user task leverages our work done in text summarization using MetaMap, so there is a good baseline and a working system in place to compare the performance of CONANN with. Two probabilistic summarizers are used: (a) FreqDist (Reeve et al., 2006), and (b) a version of SumBasic (A. Nenkova & Vanderwende, 2005) modified to use concepts rather than terms. Both summarizers only use concept frequency as the sole feature to select salient sentences. The output of a concept annotator is used for the input to the summarizers. Both summarizers' performance is entirely reliant on the frequency of concepts identified in the texts. It is expected if the concept annotation is accurate, summarization performance will improve because the concepts will have identified important areas within a text. Conversely, if the concept annotation is not accurate, text summarization performance will degrade.

The biomedical text corpus is annotated using both CONANN and MetaMap. The FreqDist and modified version of the SumBasic summarizers are then used to generate a summary of each of the 24 texts using the concept output from both annotators. The summary output from both summarizers is evaluated against manual summaries generated from domain experts using the ROUGE tool (Recall-Oriented Understudy for Gisting Evaluation) (C. Lin & Hovy, 2003). ROUGE is an automated evaluation tool which compares a system-generated summary from an automated system with one or more ideal summaries produced by people, called model summaries. ROUGE uses n-gram co-occurrence to determine the overlap between a summary and its corresponding model summaries. An n-gram can be considered as one or more consecutive words. The ROUGE parameters from the DUC 2005 conference (National Institute of Standards and Technology (NIST), 2005) are used to evaluate system summarizer performance. Two recall scores are extracted from the output of ROUGE to measure each summarizer: ROUGE-2 and ROUGE-SU4, which are also the measures used by DUC 2005. ROUGE-2 evaluates bigram co-occurrence while ROUGE-SU4 evaluates skip-bigrams with a maximum distance of four words. The ROUGE scores indicate the n-gram overlap between the source text and the model summaries. The range of ROUGE scores is 0.0 to 1.0. If a system-generated summary and a single model summary overlap exactly, the ROUGE score is 1.0. If there is no overlap, the ROUGE score is 0.0.

4.3 Biomedical Text Summarization Evaluation

Summarization evaluation is typically done by comparing a system-generated summary to summaries generated by people (C. Lin & Hovy, 2003), (Harnly, Nenkova, Passonneau, & Rambow, 2005). The 2005 Document Understanding Conference (National Institute of Standards and Technology (NIST), 2005) was reviewed for its approaches for evaluating system-generated summaries. DUC2005 used two different approaches: NIST and ISI/Columbia. Both approaches require the use of model summaries, which are manually created summaries of a source text. The system-generated summary and the model summaries are then compared to form an evaluation of the system-generated summary.

The NIST approach uses two manual methods for evaluation and one automated method. The manual evaluation methods are subjective measures of *quality* and *responsiveness*, and no comparison is done with the model summaries. The quality evaluation method is implemented by asking an evaluator to read the system-generated summary, and then answer a series of five questions about it which inquire about grammaticality, information redundancy, anaphora resolution, focus, and structure and coherence (National Institute of Standards and Technology, 2005b). The responsiveness evaluation (National Institute of Standards and Technology, 2005a) looks at the quality of information in the system-generated summary, and the granularity of the information supplied. The idea is to understand how well the system-generated summary responded to a specified information need. The automated method uses a tool called ROUGE (Recall Oriented Understudy for Gisting Evaluation) (C. Lin, 2005) for comparing a system-generated summary to a set of model summaries.

The ISI/Columbia approach uses a manual method called the Pyramid method (A. Nenkova & Passonneau, 2004). The Pyramid method evaluates the common information content between a system-generated summary and a set of model summaries produced by people. Information content is defined as an expression of an idea without regard to the terms used to express it. This analysis is difficult to do automatically, and so human annotators are used to annotate all summaries with their information content, and the overlap of the information content between a system-generated summary and its model summary is measured using frequency of occurrence. Both systems generate a score indicating how well the system-generated summary correlates with the summaries produced by people for the same text.

In order to evaluate a system-generated summary, the output of the system summarizer is compared to several model summaries using an automated tool. The ROUGE evaluation method is used because (a) it is completely automated as compared to Pyramid, which requires at least some manual annotation, and (b) requires fewer model summaries as compared to Pyramid, which requires about five model summaries

(Harnly et al., 2005). The ROUGE (Recall-Oriented Understudy for Gisting Evaluation) tool (version 1.5.5) (C. Lin & Hovy, 2003) developed by the Information Science Institute at the University of Southern California is an automated tool which compares a system-generated summary from an automated system with one or more ideal summaries produced by people. ROUGE uses n-gram co-occurrence to determine the overlap between a summary and the models. An n-gram can be considered as 1 or more consecutive words. ROUGE was used in the 2004 and 2005 Document Understanding Conferences (DUC) (National Institute of Standards and Technology (NIST), 2005) as the evaluation tool. The ROUGE parameters from the DUC 2005 conference (National Institute of Standards and Technology (NIST), 2005) are used to evaluate system summarizer performance. Two recall scores are extracted from the output of ROUGE to measure each summarizer: ROUGE-2 and ROUGE-SU4. ROUGE-2 evaluates bigram co-occurrence while ROUGE-SU4 evaluates *skip-bigrams* with a maximum distance of four words. ROUGE-2 and ROUGE-SU4 are also the measures used by DUC 2005. The recall scores indicate the n-gram overlap between the source text and the model summaries. It is difficult to compare ROUGE results outside of the corpus and model summaries used in the evaluation. For this reason, several summarizers from publicly-available sources are also used in order to provide some meaningful comparison among them using the same corpus and set of model summaries.

The Pyramid method (A. Nenkova & Passonneau, 2004) is mostly a manual evaluation effort, although there has been some recent work done to automate part of the evaluation (Harnly et al., 2005). Pyramid evaluation begins by identifying all units of model summary text no bigger than a clause which expresses an idea. These clauses are called contributors. The contributors all expressing the same idea are gathered together into a Semantic Content Unit (SCU). An SCU is composed of a unique index (i.e., SCU1, SCU2, etc.), a weight which is the number of model summaries it appears in, and a label which expresses the idea of the contributors (A. Nenkova & Passonneau, 2004). After all SCUs have been identified, a pyramid is formed based on the SCU weights, where each level of the pyramid has SCUs of the same weight, with the highest weighted SCU at the top of the pyramid and descending weighted SCUs forming lower levels of the pyramid. A system-generated summary is then also annotated in the same way as the model summaries. The pyramid expresses the content which should be in the summary, with the top level expressing the most important content. The system-generated summary is given a score computed by first counting the SCU overlap between the model summary SCUs and the system-generated SCUs at each level of the pyramid, multiplying the overlap count by the tier in the pyramid where the SCU occurs, and summing the overlap scores for each level of the pyramid (A. Nenkova & Passonneau, 2004).

4.3.1 Summarizers Used for Evaluation

Eight extractive summarizers are used in the evaluation to compare the performance of our summarization approaches. The BaseLine and SumBasic summarizers were implemented for this evaluation, and each has multiple variations. The Lemur MMR, MEAD, AutoSummarize in Microsoft Word, OTS, and SWESUM summarizers are publicly available. The eight extractive summarizers were selected based on the type of summarization method and availability. There are roughly four categories of summarizers selected: baseline, frequency-based, multiple feature, and redundancy-sensitive. Two summarizers in each category were selected. The two baseline summarizers are Baseline-Lead, which sequentially selects the first 20% of sentences in the source text, and Baseline-Random, which randomly selects 20% of the sentences in the source text. The frequency-based summarizers are AutoSummarize in Microsoft Word (Microsoft Coporation, 2002)and Open Text Summarizer (OTS) (Rotem, 2003). AutoSummarize is a feature of the Microsoft Word (Microsoft Coporation, 2002) word processing software, and although exact details of the algorithm are not documented, online help for the product indicates sentences using frequently-used words are given a higher score than sentences containing low frequency words. OTS is an open source project where stemming can also be performed to eliminate word variations. The two summarizers using multiple features to identify sentences are SweSum (Dalianis, 2000) and MEAD (Radev et al., 2004). SweSum is a multi-lingual summarizer for Swedish and English text using features such as sentence position and numerical data identification. MEAD is a single and multi-document summarizer using features such as position of sentence within the text, overlap of each sentence with the first sentence, sentence length, and a centroid method based on a cluster of related documents. Finally, the two summarizers which reduce information redundancy are Lemur Maximal Marginal Relevance (MMR) (The Lemur Project, 2006)and SumBasic (A. Nenkova & Vanderwende, 2005). Lemur MMR iteratively selects sentences having a high query similarity to an automatically-generated query, and which are also maximally dissimilar to sentences already included in the summary. SumBasic uses a probability distribution of terms in the text, and reduces term probability as sentences containing the terms are selected. SumBasic was also adapted to use concepts as the input source, rather than terms.

Each summarizer generated a summary that was equal to 20% of the length of the source text. For example, if a source text consists of 100 sentences, then 20 sentences are selected and extracted by each summarizer and presented as the summary. Selecting a summary size was problematic. The news summarization domain typically selects a size of less than five sentences. This represents about 20% of the size of a typical news story (Goldstein, Kantrowitz, Mittal, & Carbonell, 1999). It has been generally thought that a summary should be no shorter than 15% and no longer than 35% of the source text (E. H. Hovy, 2005). The following is a brief description of the approaches used by each summarizer.

In addition to the FreqDistSumm summarizer, we also implemented several variations of BioChainSumm. BioChainSumm is divided into two primary approaches based on the concept selection criteria in strong chains for the sentence scoring: (a) using the most frequent concept in the strongest chains (MostFrequentStrongChainConcept), and (b) using all of the concepts within the strongest chains (AllStrongChainConcepts). Each of these primary approaches also implements variations by providing each sentence (or each chain) with a certain weight. The first variation is adding a sentence position heuristic where each sentence is initially scored as 1/N, where N is the number of sentences in the source text. The concept chain score is then added to this base score. The idea is that sentences in the beginning of the text are more important than sentences at the end of a text (Dalianis, 2000). A second variation is to filter out semantic types. The idea is that not all semantic types are important to focus on for a particular user's information needs. Our domain expert identified the semantic types important within the oncology clinical trial domain. A chain is scored as zero if not in the list shown in Table 11. The final variation is to combine the sentence position heuristic with semantic type filtering.

Table 11: Important UMLS semantic types for oncology clinical trials

UMLS Semantic Type	UMLS Semantic Type Name
T37	Injury or Poisoning
T51	Event
T52	Activity
T61	Therapeutic or Preventative Procedure
T62	Research Activity
T67	Phenomena or Process
T81	Quantitative Concept
T169	Functional Concept
T170	Intellectual Product
T191	Neoplastic Process

4.3.1.1 BaseLine

The purpose of the baseline summarizers is to give some indication of the level of performance of a naïve summarization implementation. Two baseline summarizers were implemented. The first baseline summarizer is called Baseline-Lead, and it sequentially selects the first 20% of sentences in the source text. The second baseline summarizer is called Baseline-Random, and it randomly selects 20% of the sentences in the source text.

4.3.1.2 Lemur MMR

The Lemur MMR application (The Lemur Project, 2006) is a summarizer built using the idea of Maximal Marginal Relevance (MMR) (Carbonell & Goldstein, 1998).

Carbonell describes marginal relevance as finding relevant sentences which contain minimal similarity to previously selected sentences. Maximal marginal relevance in text summarization attempts to maximize the dissimilarity of the information content between sentences within a summary. Lemur MMR iteratively selects sentences based on similarity to a query, and then selects sentences having a high query similarity which are also maximally dissimilar to sentences already included in the summary. In the evaluation no query was specified, so Lemur MMR automatically generated a query based on the source text. No domain specific knowledge sources were provided to the summarizer.

4.3.1.3 MEAD

MEAD (Radev et al., 2004) is a single- and multiple-document summarizer using multiple features to score sentences. Some of the features include position of sentence within the text, overlap of sentence with the first sentence, sentence length, and a centroid method based on a cluster of related documents. For the evaluation, the MEAD Demo located at http://tangra.si.umich.edu/clair/md/demo.cgi was used. No domain specific knowledge sources were provided to the summarizer.

4.3.1.4 AutoSummarize

The AutoSummarize is a feature of the Microsoft Word (Microsoft Coporation, 2002) word processing software. AutoSummarize is based on a word frequency algorithm. Each sentence in a document is given a score based on the words the sentence contains. Although the exact details of the algorithm are not documented, the online help for the product states that sentences using frequently-used words are given a higher score than sentences containing low frequency words. No domain specific knowledge sources were provided to the summarizer.

4.3.1.5 Open Text Summarizer (OTS)

The Open Text Summarizer (OTS) is an open source project which provides a library for summarizing arbitrary texts (Rotem, 2003). It is based on a frequency-based approach, where the most frequently occurring words are assumed to be indicators of the text theme. Stemming can also be performed to eliminate word variations, so that for example, *year* is not distinguished from the plural *years*. Users can provide their own set of stemming rules. For the evaluation, the pre-built summarization library was used with no change to the stemming rules. No domain-specific resource is used.

4.3.1.6 SumBasic

The SumBasic algorithm (A. Nenkova & Vanderwende, 2005) is a recent frequency-based algorithm. The original algorithm works using terms as the unit item to count. For this evaluation, is has been modified so that the unit items can be terms or concepts. SumBasic incorporates a component for ensuring coverage of weaker concepts within a text. There are four steps in the algorithm. The first is to determine the probability distribution of all concepts found within a source text by computing the number of times a unit item appears in the text divided it by the total number of unit items found in the text. The second step is to score each sentence by summing the probabilities of all unit items within a sentence. The third step determines the sentence to be extracted by finding the highest-scoring sentence. The fourth step then reduces the probability of each unit item appearing in future extracted sentences by multiplying each probability of each unit item in the last extracted sentence by itself. The implementation using terms as unit items first had a stop word list applied. For the implementation using concepts, the UMLS Metathesaurus was used as the domain-specific resource (the same as the BioChainSumm summarizer).

4.3.1.7 SWESUM

SweSum (Dalianis, 2000) is a multi-lingual summarizer for Swedish and English text. SweSum uses multiple features for scoring sentences, such as sentence position and numerical data identification. Sentences located earlier in a text are scored higher than sentences at the end of the text. Sentences containing numerical data are given additional weight. User-specified keywords can also be provided to boost sentence scores for those sentences containing the keywords. For the evaluation, the online version located at http://swesum.nada.kth.se/index-eng-adv.html was used. The text type was set to *Academic* and the summarization size was to 20%. No other parameters were set, and no domain specific knowledge sources were provided to the summarizer.

4.3.2 Biomedical Text Characterization

To explore the text structure of biomedical texts, three studies are completed. The first study looks at the distribution of biomedical concepts within a set of biomedical full-text sources, as well as the biomedical concepts within the author abstracts of the same set of texts. The idea is to see if the distribution of concepts mirrored the well-known Zipf distribution of terms in text (Zipf, 1949).

The second study tries to define a minimum compression ratio which can be used for biomedical text. This is important, because a smaller summary size results in less data which must be processed by a person to acquire the same amount of information as a larger summary. System summaries are generated at five different compression ratios: 1%, 5%, 10%, 15% and 20%. The system summaries are evaluated against model summaries produced by domain experts. The model summaries match the compression size of the system-generated summaries. Compression rate is the percentage of the sentences from the source text.

The third study looks at the sections within papers where human summarizers draw their sentences. The idea is to see if some sections are considered more important than others. This information can be used to weight different sections of a text more heavily than others if people draw sentences from one section more than another section. Using this information about the section location of extracted sentences, the study will see if there are any commonalities between human summarizers in the sections they pull summary sentences from.

The second and third studies uses the corpus of biomedical texts annotated by domain experts with the sentences they would extract from the source text to form a summary. The sentences are selected from one sentence up to 20% of the number of sentences in the text, as well as the major section of the paper each selected sentence appears in.

4.3.2.1 Concept Distribution

The concept distribution study is done to gain insight into (a) the most frequent concepts within oncology texts, and (b) determine if the there is any relationship between the frequency of concepts in the abstract and the frequency of concepts within the corresponding full text. The study data is generated by first annotating all text within the evaluation corpus with biomedical concepts. The annotation is done separately for the full-text and the abstract so that an analysis of each is done. The concepts are then aggregated across (a) the set of full-text sources, and (b) the set of corresponding abstracts.

4.3.2.2 Biomedical Summary Size

A primary goal of text summarization is to reduce the amount of data which must be processed by the human reader. In news summarization, the ideal size of a summary is approximately 20 percent (Goldstein et al., 1999). The ideal size of a biomedical text summary has not been studied in the literature. To determine the ideal size of a biomedical text (specifically, clinical trials in oncology), the approach is to (a) generate summaries using two summarizers, SumBasic and FreqDist at compression rates of 1%, 5%, 10%, 15% and 20%, (b) generate model summaries using the same compression rate, and (c) evaluate the generated summaries using the ROUGE tool.

5. EVALUATION RESULTS AND DISCUSSION

This chapter presents the evaluation results based on the evaluation methodologies described in Section 4. Section 5.1 focuses on the characteristics of biomedical text. It describes (a) the distribution of UMLS concepts within biomedical text, (b) the size of an ideal biomedical summary, and (c) the sections of a biomedical text where human summarizers select sentences from. Sections 5.2 and 5.3 present an intrinsic and an extrinsic evaluation of two semantic annotation methods. Sections 5.4 and 5.5 discuss text summarization evaluations use the ROUGE tool on two novel text summarization algorithms (concept chaining and frequency distribution). Section 5.6 presents significance testing results of various summarizers. Finally, Section 5.7 presents the results of the FreqDistUpdate summarizer in the 2007 Document Understanding Conference.

5.1 Biomedical Text Characteristics

The following subsections detail the results of the study on the distribution of concepts within the corpus of biomedical texts and the ideal size of a summary size. In addition, the sections of text where human summarizers select important sentences from are evaluated.

5.1.1 Concept Distribution

A TreeMap visualization (Johnson & Shneiderman, 1991) was developed to display frequency of concepts within the corpus of biomedical texts. Figures 29 and 30 show two-level hierarchical TreeMap views of the resulting concept frequency data. The first level in the hierarchy is the UMLS *Semantic Type*. The semantic type is used within UMLS to broadly organize related concepts. Each semantic type forms a cell. The size of the semantic type cell gives an indication of the total number of concepts within the semantic type cell. A larger semantic type cell indicates that the semantic type has more concepts than a smaller semantic type cell. Within each semantic type cell, concepts cells for each related concept are constructed with a range of color intensities. A lighter color intensity within a concept cell indicates a more frequently occurring concept than a darker intensity cell.

Figure 29 shows that the top five semantic types across the set of abstracts are *Quantitative Concept*, *Intellectual Product*, *Therapeutic or Preventative Procedure*, *Functional Concept*, and *Temporal Concept*. The most frequent concepts are *Survival Aspects*, *Methods*, *Methodology*, *Result*, *Month*, *Patients*, and *Continuance of Life*. This seems reasonable, as published oncology clinical trials papers typically evaluate methods for extending the life of patients.

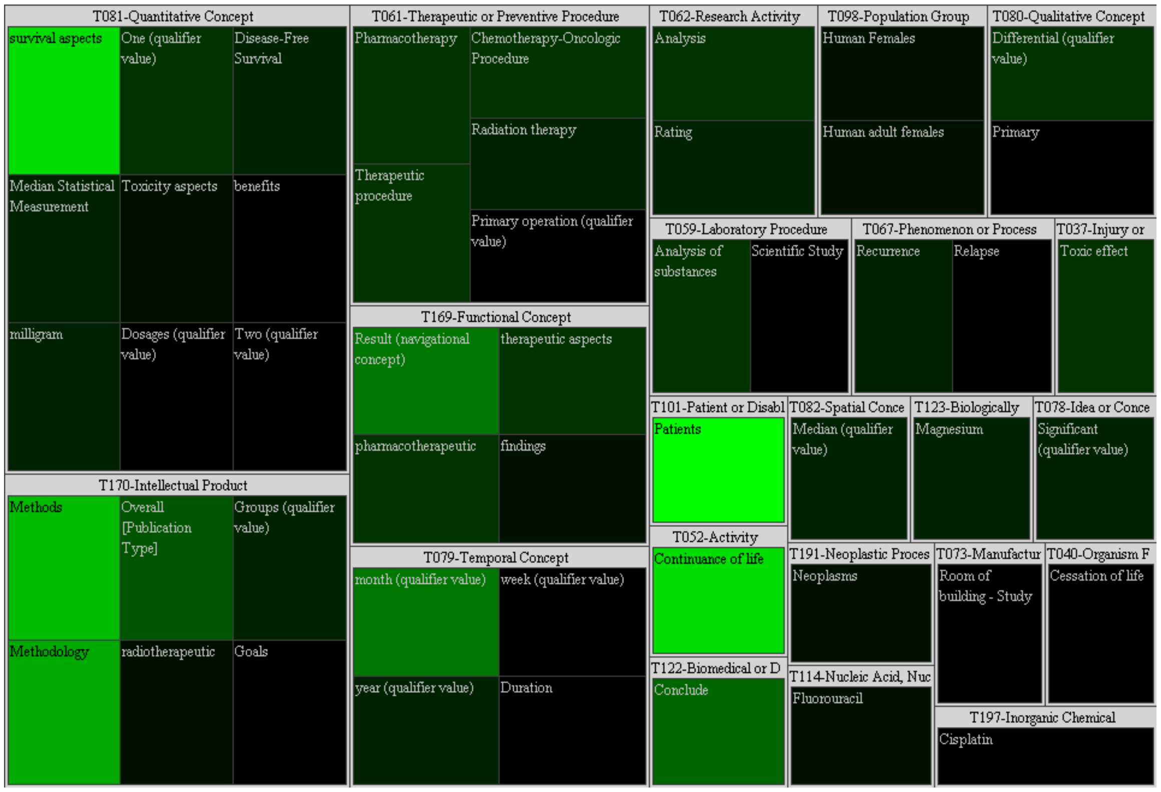


Figure 29: TreeMap visualization of concept distribution across 24 abstracts

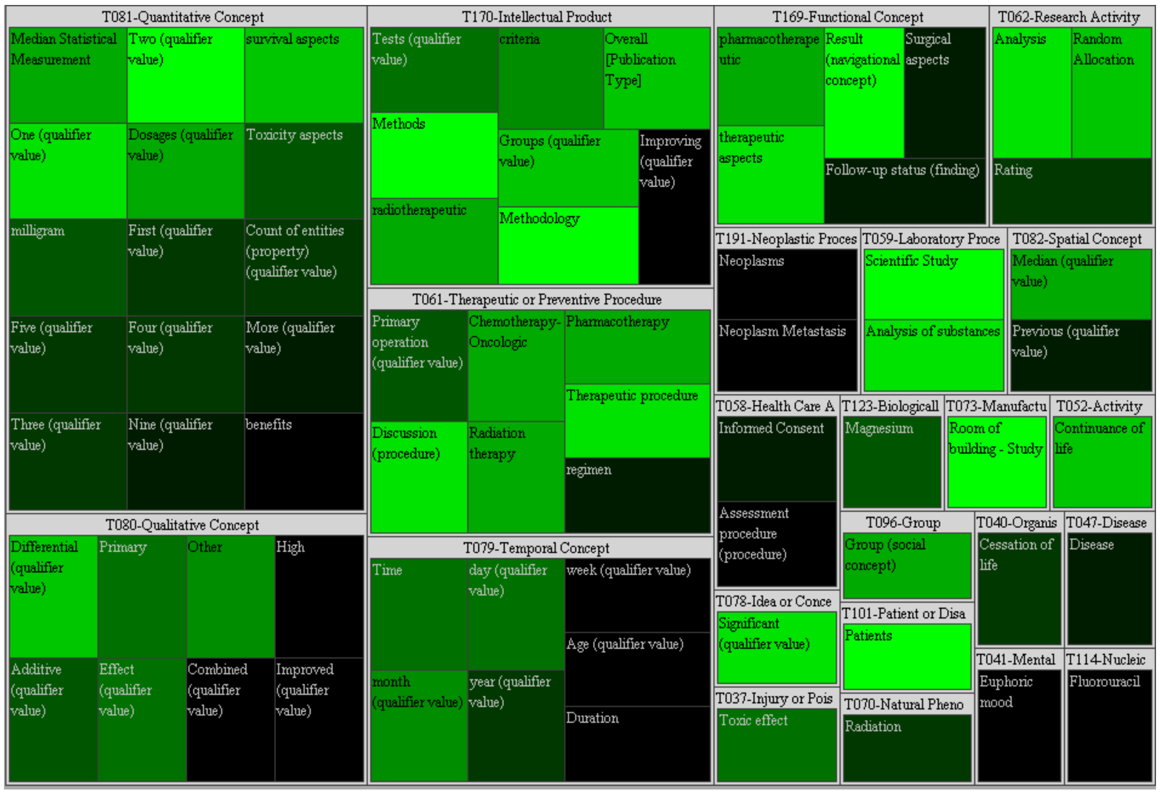


Figure 30: TreeMap visualization of concept distribution across 24 full-text sources

Within the set of full-text sources, the top five semantic types are *Quantitative Concept*, *Qualitative Concept*, *Intellectual Product*, *Therapeutic or Preventative Procedure*, and *Functional Concept*, as shown in Figure 30. The most frequent concepts are numeric *qualifier values*, *Methods*, *Methodology*, *Therapeutic Procedure*, *Discussion*, *Result*, *Scientific Study*, and *Patients*.

In terms of semantic types, the primary difference between the full-text source and the corresponding abstracts is that the former has more qualitative concepts than the latter. This is likely attributed to the inclusion of background information (e.g., introductory material and previous work) which is not usually provided in an abstract. At the concept level, the full-text sources make extensive use of so-called qualifier values, such as ‘one’, ‘two’, ‘effect’ and so forth, which modify other concepts. These qualifier values exist both in the Quantitative and Qualitative semantic types, and occur in greater frequency than concepts which appear frequently in the abstract, such as *Continuance of Life*. The main observations made during this study are (a) the semantic types and concept frequency distribution between a source text and abstract are largely the same; and (b) not all concepts can be treated equally, and it may make sense to weight some concepts as more important than others. For example, it might be argued that the qualifier concept *One* is not as important as the concept *Continuance of Life*. In addition, although it may be useful to filter unimportant parts of text by ignoring concepts within a particular semantic type, it might also be useful to filter specific types of concepts. For example, a physician wanting primarily the results in a generated summary may elect to exclude all qualitative concepts. A more specific method than removing concepts by semantic type is to remove, for example, all qualifier concepts across all semantic types. This may be useful for reducing the number of concepts which do not contribute to identifying text themes.

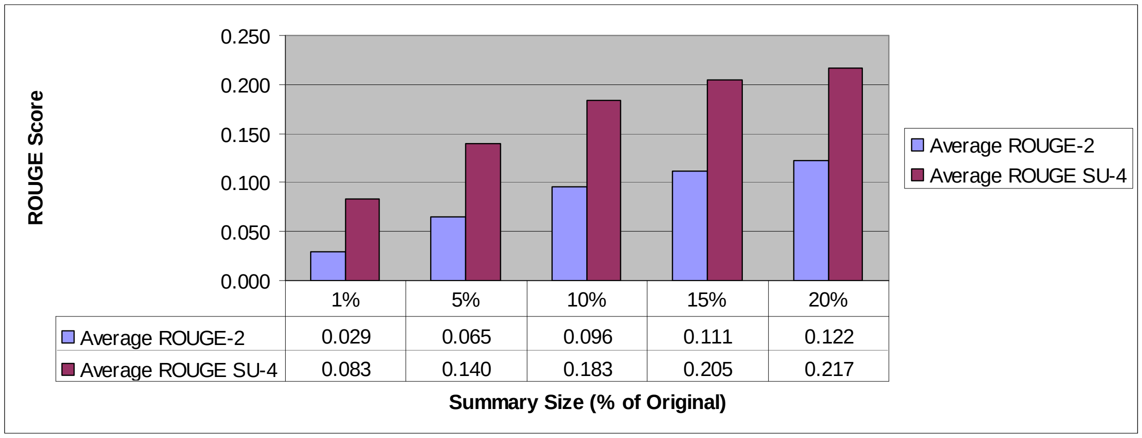
5.1.2 Summary Size

This section presents the results of evaluating the ideal size of a biomedical summary. Five different compression ratios are examined using two different frequency-based summarizers. The use of concepts and terms as unit items to identify salient sentences is also used within each of the two summarizers. Table 12 shows the ROUGE-2 and ROUGE-SU4 scores for the FreqDist and SumBasic summarizers at varying summary sizes. Term and concept versions of each summarizer were used.

Table 12: ROUGE Scores by summarizer at varying summary sizes

Summarizer	Score Type	1%	5%	10%	15%	20%
FreqDist-Concept	ROUGE-2	0.01561	0.05817	0.09063	0.10540	0.12069
FreqDist-Term	ROUGE-2	0.01561	0.05526	0.10289	0.11944	0.13241
SumBasic-Concept	ROUGE-2	0.02432	0.06128	0.08102	0.10303	0.11003
SumBasic-Term	ROUGE-2	0.06098	0.08532	0.10909	0.11800	0.12611
FreqDist-Concept	ROUGE-SU4	0.06766	0.13398	0.18256	0.20260	0.21851
FreqDist-Term-Dice	ROUGE-SU4	0.06766	0.12737	0.19341	0.21382	0.22941
SumBasic-Concept	ROUGE-SU4	0.07277	0.13387	0.15964	0.19106	0.19941
SumBasic-Term	ROUGE-SU4	0.12280	0.16383	0.19802	0.21173	0.22089

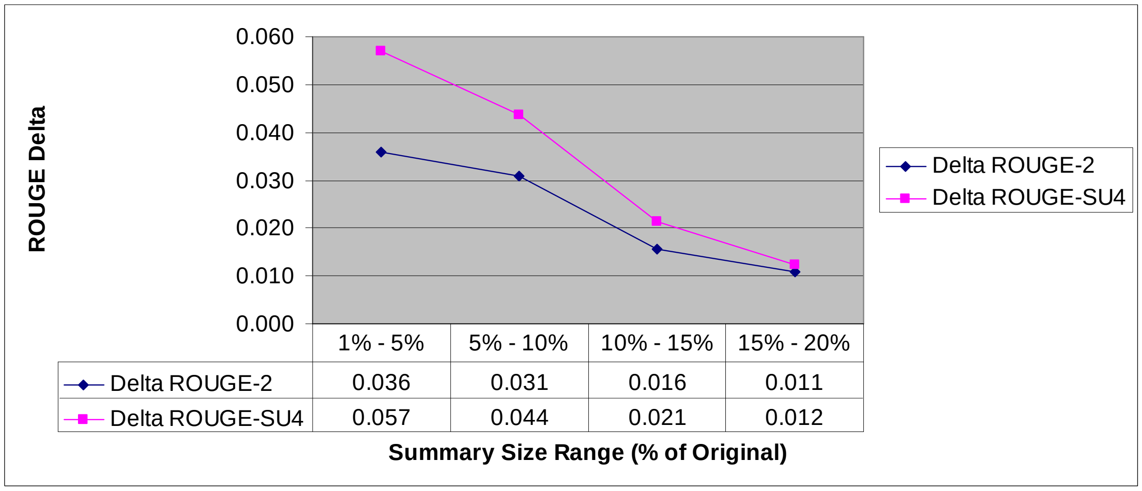
The general trend across all summarizers is a continual improvement in ROUGE scores as the size of the summary increases. For ROUGE-2, the FreqDist-Concept summarizer improves from the 1% size to the 20% size by 87%, while the term version improves by 88%. The SumBasic-Concept summarizer similarly improves from the 1% size to the 20% size by 78%, while the term version improves by 52%. Using ROUGE-SU4 scores, FreqDist-Concept improves by 69%, while the FreqDist-Term summarizer improves by 71%. The SumBasic-Concept summarizer improves by 64% from the 1% size to the 20% size, while SumBasic-Term improves by 44%. The chart in Figure 31 shows this trend visually. The chart uses the average ROUGE scores of all summarizers at each summary size level and plots them. At the 1% compression level, the SumBasic summarizer using terms performs very well, while SumBasic using concepts performs above average but not at the level of terms. The FreqDist summarizer does not perform well until the 10% level. The FreqDist algorithm models the summary after the source text, and based on this evaluation FreqDist needs about 10% of the original source text to provide a well-performing summary.



	1%	5%	10%	15%	20%
▨ Average ROUGE-2	0.029	0.065	0.096	0.111	0.122
■ Average ROUGE SU-4	0.083	0.140	0.183	0.205	0.217

Figure 31: Graph of ROUGE scores at varying summary sizes

Figure 32 plots the decreasing rate of change of ROUGE scores between various summary sizes. The ROUGE scores for all summarizers are first averaged at each summary size. For each successive summary size range, the difference between the score averages is obtained. The summary size ranges are from 1% to 5%, 5% to 10%, 10% to 15%, and 15% to 20%. For example, the difference of average scores at the 1% to 5% summary size range is obtained by subtracting the 1% average score from the 5% average score, resulting in a *Delta* score. Delta ROUGE-2 measures the change in ROUGE-2 scores, while Delta ROUGE-SU4 measures the change in ROUGE-4 scores. The idea is to understand at what point the least amount of summary improvement is obtained by increasing summary size. Figure 32 makes it easy to see the rate of change (i.e., summary improvement) is good at 1%-5% and also 5%-10%, but then starts to more rapidly decline at the 10%-15% and 15% -20% summary size ranges.

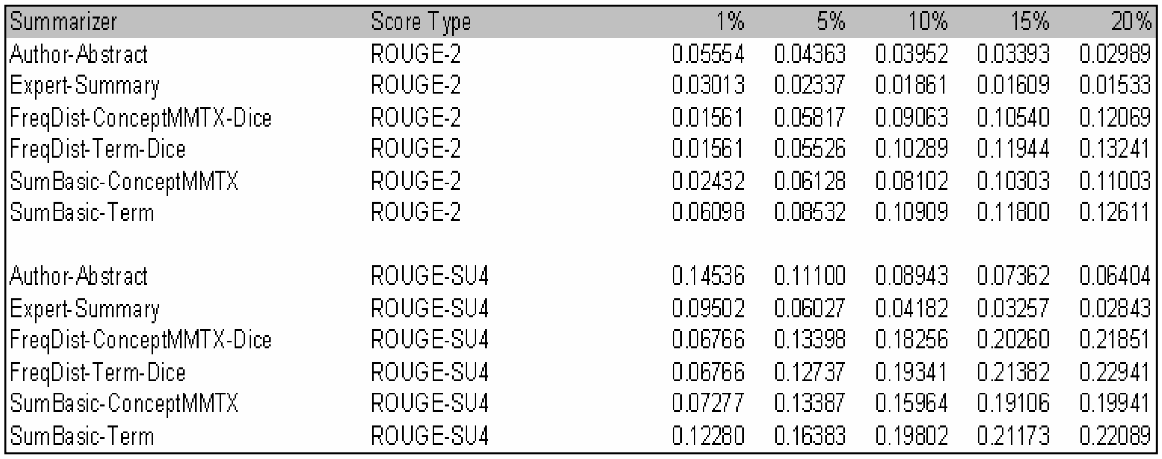


	1% - 5%	5% - 10%	10% - 15%	15% - 20%
Delta ROUGE-2	0.036	0.031	0.016	0.011
Delta ROUGE-SU4	0.057	0.044	0.021	0.012

Figure 32: Graph of ROUGE scores rate-of-change between varying summary sizes

The goal of this study is to find the level at which the smallest summary can be generated. Using Figure 32, it can be seen that summarizer improvement tends to decline somewhere in the 10%-15% range, and therefore the ideal summary size for biomedical text is approximately 10% to 15% of the original source text. This is different than general news summarization, where the ideal size is approximately 20% (Goldstein et al., 1999). The smaller size for biomedical text is likely due to the longer length of biomedical text, which is likely to take advantage of reiteration. The suggested summary size of 10% to 15% is only a general guideline. As Table 12 shows, at the 1% size, the SumBasic-Term summarizer performs the best, and its performance tends to increase until about 10% summary size, where additional summary sizes do not add significantly to the score. This observation holds true for both the ROUGE-2 and ROUGE-SU4 scores. It is also notable that the FreqDist-Concept summarizer outperforms the FreqDist-Term summarizer until about the 10% summary size, where the term version then begins to outperform the concept version. It can be observed that while the optimal summary size can be generally determined, the performance at various summary sizes is influenced by the summarizing method as well as the unit type (concept or term.)

A similar study was also performed using two human-generated summaries as system summaries. The first summary is the abstract and the second is a domain expert's synthesized summary. The two summary type's performance is measured against the model summaries produced by domain experts at various compression ratios. The idea is to get a sense of the performance of using the abstract as a summary. The abstract size was left unmodified (i.e., it was not reduced or enlarged based on the compression ratio). Similarly, the domain expert's synthesized summary is a short (typically 1-2 sentences) that summarizes the biomedical text for other domain experts using their own terminology and without using sentences from the original source text. Figure 33 shows the performance of the text abstract (Author-Abstract) and domain expert summary (Expert-Summary). In addition, the performance of term and concept variations of the FreqDist and SumBasic summarizers is shown for comparison. For ROUGE-2 scores, Author-Abstract outperforms all automated summarizers except for SumBasic-Term at the 1% compression ratio. As the compression ratio increases, the performance of the Author-Abstract decreases. For ROUGE-4 scores, Author-Abstract outperforms all summarizers at the 1% level, and then its performance decreases as the compression ratio increases. The Author-Abstract is outperformed in ROUGE-2 and ROUGE-4 beginning at the 5% compression ratio. The reason for the high-performance of Author-Abstract at the 1% level is that the size of Author-Abstract is larger than the summaries generated the other summarizers. The larger size allows for Author-Abstract to have a better chance to match terms of the model summaries. Similarly, its performance decreases with larger compression ratios because the other summarizers are generating larger summaries than the Author-Abstract. In the same way, the Expert-Summary performed well at the small compression ratios and then decreased as the compression ratio was increased and the other summarizers generated text greater than or equal in size to the Expert-Summary. If we assume that the size of an abstract is 10% of the full-text (5000 source text words = 500 word summary), then the performance of the Author-Abstract and Expert-Summary is the lowest using both ROUGE-2 and ROUGE-SU4. This indicates that the Author-Abstract and Expert-Summary do not align very well the model summaries produced by domain experts.



Summarizer	Score Type	1%	5%	10%	15%	20%
Author-Abstract	ROUGE-2	0.05554	0.04363	0.03952	0.03393	0.02989
Expert-Summary	ROUGE-2	0.03013	0.02337	0.01861	0.01609	0.01533
FreqDist-ConceptMMTX-Dice	ROUGE-2	0.01561	0.05817	0.09063	0.10540	0.12069
FreqDist-Term-Dice	ROUGE-2	0.01561	0.05526	0.10289	0.11944	0.13241
SumBasic-ConceptMMTX	ROUGE-2	0.02432	0.06128	0.08102	0.10303	0.11003
SumBasic-Term	ROUGE-2	0.06098	0.08532	0.10909	0.11800	0.12611
Author-Abstract	ROUGE-SU4	0.14536	0.11100	0.08943	0.07362	0.06404
Expert-Summary	ROUGE-SU4	0.09502	0.06027	0.04182	0.03257	0.02843
FreqDist-ConceptMMTX-Dice	ROUGE-SU4	0.06766	0.13398	0.18256	0.20260	0.21851
FreqDist-Term-Dice	ROUGE-SU4	0.06766	0.12737	0.19341	0.21382	0.22941
SumBasic-ConceptMMTX	ROUGE-SU4	0.07277	0.13387	0.15964	0.19106	0.19941
SumBasic-Term	ROUGE-SU4	0.12280	0.16383	0.19802	0.21173	0.22089

Figure 33: Human summarization performance

5.1.3 Human Summarizer Sentence Selection

Sentence selection was broken down into five categories: Introduction, Methods, Results, Discussion and Appendix. The number of sentences as well as the percentage of sentences (shown in parenthesis) selected in each of these sections by three human summarizers is shown in Table 13. Summarizer #1 data included only about 8 texts, while Summarizers #2 and #3 provided data for all 24 texts. A visual depiction of the same data is shown in Figure 34. There is a clear trend between Summarizers #1 and #2 to use the same sections in the same proportion. They appear to agree that the Results section is the most important section to draw sentences from (44% and 41%, respectively), followed by the Methods (26% and 24%) and Discussion (26% and 23%) sections. Interestingly, neither draws information from the Appendix. Summarizer #2 contrasts Summarizers #1 and #3 by selecting sentences primarily from the Methods (33%) and Discussion (31%) sections. At 14% of sentences selected overall, the Results section does not seem to hold much content for Summarizer #2, although it is possible information in the Results are reiterated in the Discussion section and Summarizer #2 has decided to draw sentences from that section.

Table 13: Number of sentences selected in each section by summarizer using all sentences

Summarizer	Introduction	Methods	Results	Discussion	Appendix
Summarizer #1	8 (4%)	47 (26%)	80 (44%)	48 (26%)	0 (0%)
Summarizer #2	124 (19%)	221 (33%)	94 (14%)	205 (31%)	17 (3%)
Summarizer #3	80 (12%)	155 (24%)	271 (41%)	149 (23%)	0 (0%)

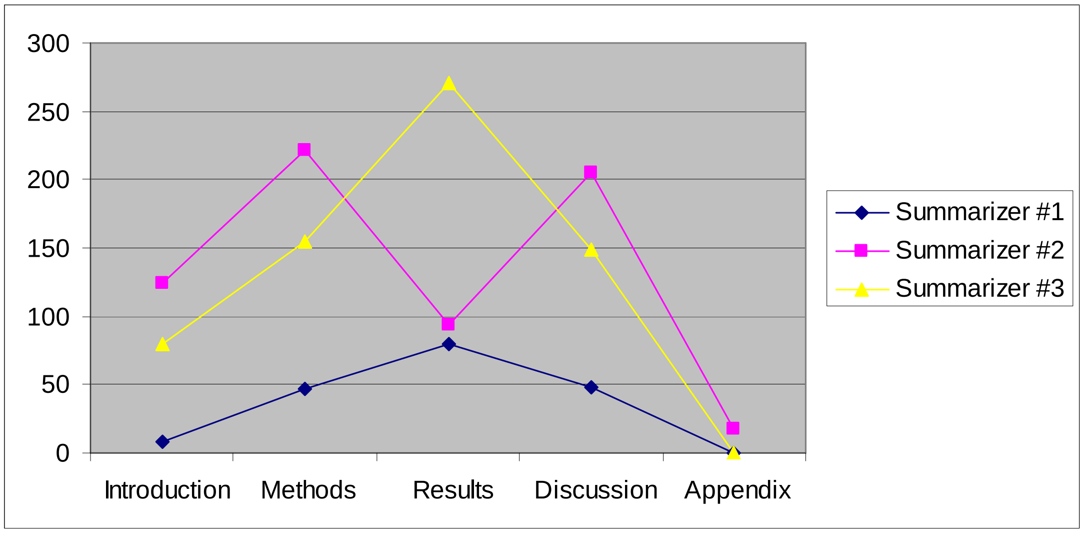


Figure 34: Graph of all sentence selection by section and summarizer

The sentences selected by each summarizer were also ranked in decreasing order of importance, regardless of which section of they were drawn from. To see where the most important sentences were drawn from, the procedure above was repeated using the top one-third of sentences selected by each summarizer. The number of sentences as well as the percentage of sentences (shown in parenthesis) selected in each of these sections by three human summarizers is shown in Table 14. A graph depicting the same data is shown in Figure 35. Summarizers #1 and #3 are again similar using the Methods, Results and Discussion section in about the same percentages. The only difference between the two is that Summarizer #3 selects 14% of sentences from the Introduction, while Summarizer #1 selects only 5%. Summarizer #2's selection of top sentences looks very similar to their selection using all sentences, as shown in Figure 34.

The purpose of the study is to determine if particular sections can be weighted more heavily than other sections when extracting sentences to form a summary. It seems clear that the Methods and Discussion sections are important to all summarizers in the study, although in different degrees. The Results section is also considered most important by two of the summarizers, but less important than Methods and Discussion by one of the summarizers. It is unclear, then, how the Results section should be weighted. The Introduction section, considered less important than Methods, Results, or Discussion sections for all summarizers, is again used more by two summarizers than the third summarizer. The Appendix section is used only by one summarizer, and even then only used to select about 3% of sentences. A possible approach, then, is to extract sentences by preferring sections up to a certain size. For example, select sentences first from the Results section up to say 40%, then select Methods and Sections up to 25% each, then select 5% from Introduction, and the remaining 5% could be used as wildcard to select sentences from any section. Other approaches could also be used based on this study. However, given the summarizer inconsistency in this small study, it is not possible to come to firm conclusions about the importance of each section beyond the fact that Methods and Discussion and important to all summarizers.

Table 14: Number of sentences across entire corpus selected in each section using top one-third of all sentences

Summarizer	Introduction	Methods	Results	Discussion	Appendix
Summarizer #1	6 (5%)	40 (30%)	65 (49%)	22 (16%)	0 (0%)
Summarizer #2	63 (14%)	168 (37%)	75 (16%)	135 (30%)	14 (3%)
Summarizer #3	63 (14%)	134 (30%)	192 (43%)	62 (13%)	0 (0%)

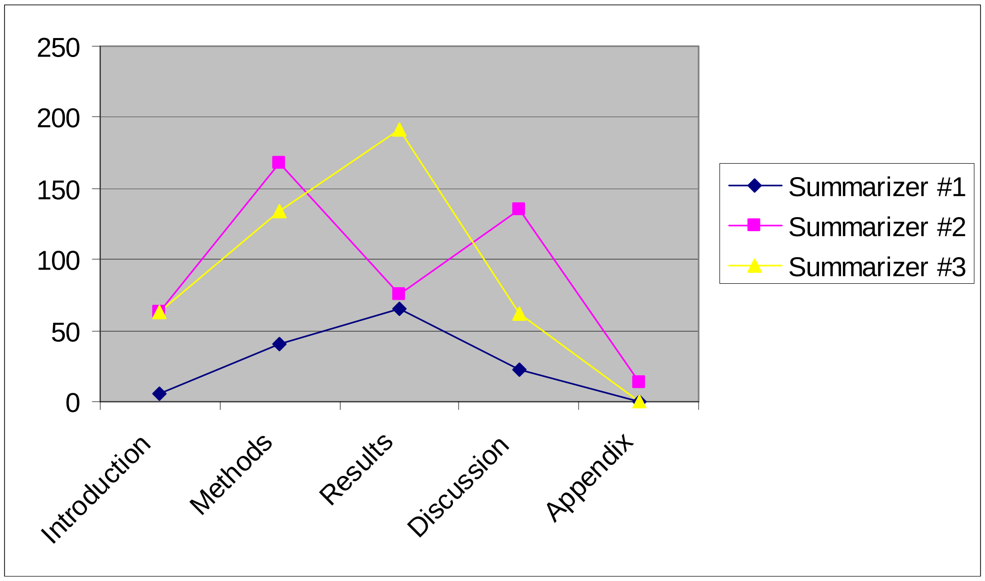


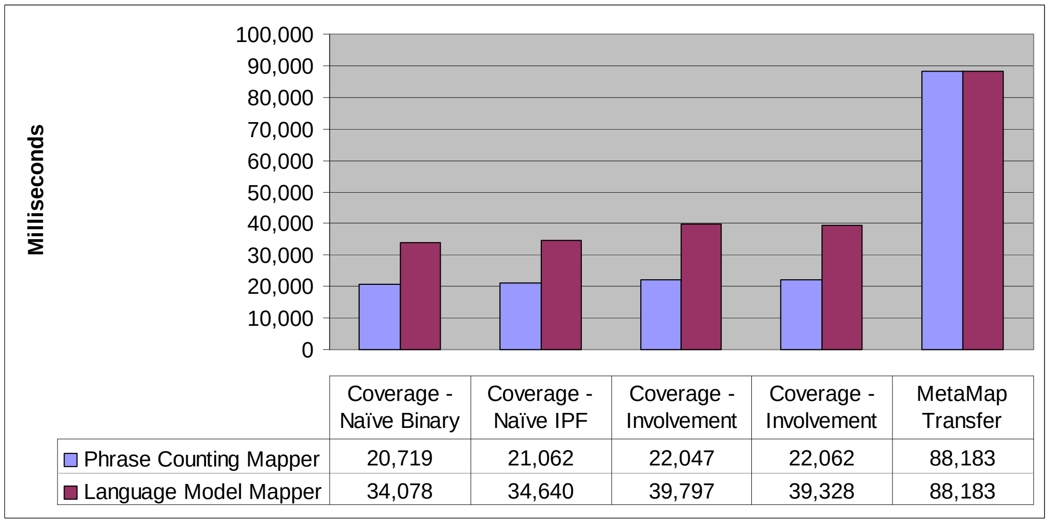
Figure 35: Graph of top one-third of sentences selected, by section and summarizer

5.2 Semantic Annotation

This section presents an intrinsic and an extrinsic evaluation of the CONANN semantic annotator. The contribution of the Coverage and Coherence filters as well as two final concept mappers using phrase counting and language models is presented. An additional evaluation is performed on the Extended Coverage Filter described in Chapter 4.

5.2.1 Intrinsic Evaluation

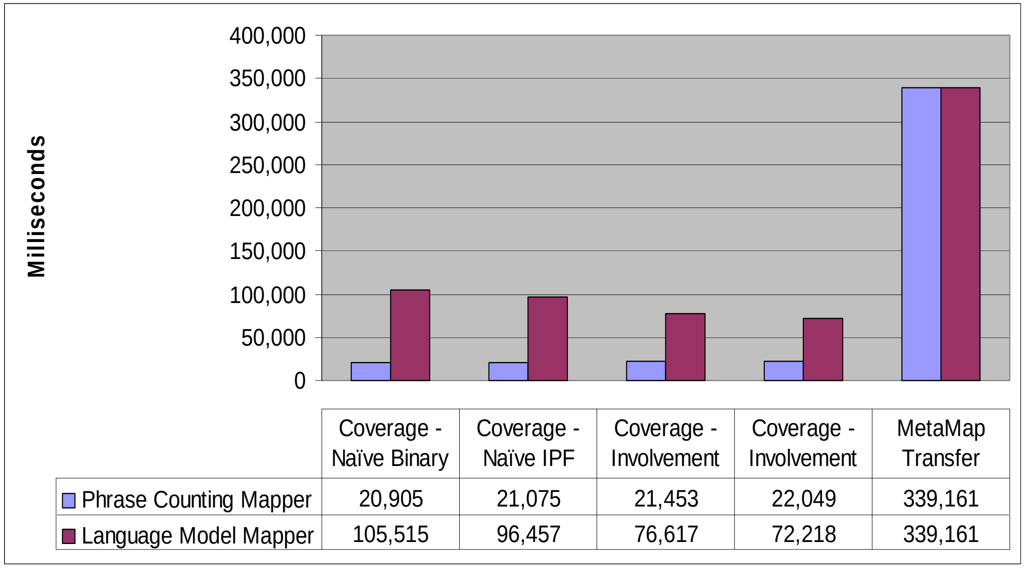
As shown in Figure 36, MetaMap initialization time was 88 seconds, while for our CONANN, initialization time ranged from a low of 20 seconds using the Phrase Counting Concept Mapper (PCCM) to 40 seconds for the Language Model Concept Mapper (LMCM). The LMCM took longer to initialize than the PCCM because it required loading the approximately 700,000 language models (one for each UMLS concept).



	Coverage - Naïve Binary	Coverage - Naïve IPF	Coverage - Involvement	Coverage - Involvement	MetaMap Transfer
🟦 Phrase Counting Mapper	20,719	21,062	22,047	22,062	88,183
🟪 Language Model Mapper	34,078	34,640	39,797	39,328	88,183

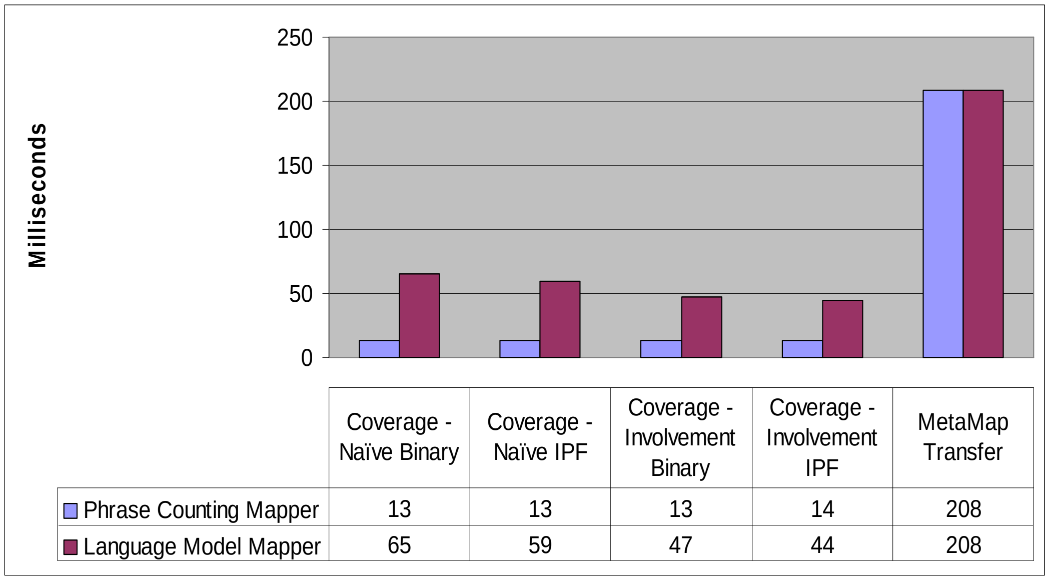
Figure 36: Annotator initialization time

Figure 37 presents the total time to annotate all 1,628 phrases in the evaluation corpus. MetaMap total annotation time was 5.7 minutes, while CONANN PCCM ran with a high of 22 seconds using the Coverage-Involvement filter while CONANN LMCM ran with a high of 1.76 minutes using the Coverage-NaiveBinary filter. Figure 38 shows the average time to annotate each phrase. Average Phrase Annotation Time is calculated by taking the total annotation time and dividing it by 1,628, which is the total number of phrases annotated. MetaMap average time to annotate a phrase was 208 milliseconds, while CONANN ranged from a high of 14 milliseconds for the PCCM to 64 milliseconds for the LMCM.



	Coverage - Naïve Binary	Coverage - Naïve IPF	Coverage - Involvement	Coverage - Involvement	MetaMap Transfer
▨ Phrase Counting Mapper	20,905	21,075	21,453	22,049	339,161
■ Language Model Mapper	105,515	96,457	76,617	72,218	339,161

Figure 37: Total annotation time



	Coverage - Naïve Binary	Coverage - Naïve IPF	Coverage - Involvement Binary	Coverage - Involvement IPF	MetaMap Transfer
▨ Phrase Counting Mapper	13	13	13	14	208
■ Language Model Mapper	65	59	47	44	208

Figure 38: Average phrase annotation time

Table 15: CONANN precision using MetaMap as baseline system

Map	*Filter*	*Exact Match*	*Top 5*
Phrase Counting Concept Mapper (PCCM)	Coverage - Naïve Binary	0.66	0.78
	Coverage - Naïve IPF	0.63	0.75
	Coverage - Involvement Binary	0.75	0.90
	Coverage - Involvement IPF	0.67	0.87
Language Model Concept Mapper (LMCM)	Coverage - Naïve Binary	0.76	0.86
	Coverage - Naïve IPF	0.74	0.83
	Coverage - Involvement Binary	0.85	0.93
	Coverage - Involvement IPF	0.81	0.91

Table 15 shows the CONANN annotator precision for each coverage filter and concept mapper combination. Two precision scores are presented. The first is the precision when matching CONANN's output exactly with MetaMap's output. The second is the precision when matching any of the top five concepts produced by CONANN for a single phrase with the single MetaMap concept. The best performing PCCM (0.75) uses the binary involvement filter. Precision increases to 0.90 when using the top five CONANN concepts, indicating that an improved final concept mapper method could increase precision. The best performing LMCM (0.85) also uses the binary involvement filter, and its precision increases to 0.93 when using the top five concepts. LMCM outperforms PCCM in exact matching from 13%-20% depending on the coverage filter. LMCM performance can be attributed to the fact LMCM considers all concept instances of a concept, while PCCM considers only those concept instances which have passed through the coverage filter.

The intrinsic evaluation shows the best-performing word coverage filter for both phrase counting and language model mapping is binary involvement. The highest precision final concept mapper is language model, but has the tradeoff of longer average annotation phrase annotation time at over three times longer than PCCM. The precision of LMCM is 13-20% higher than the PCCM for the exact match. For average annotation time, LMCM is over four times faster than MetaMap while PCCM is nearly 15 times faster. The tradeoff CONANN makes for higher annotation performance is less precision when compared to MetaMap. The impact of this precision tradeoff is evaluated in the extrinsic evaluation, discussed in the next section on extrinsic evaluation.

5.2.2 Extrinsic Evaluation

Tables 16 and 17 show the ROUGE-2 and ROUGE-SU4 scores using the FreqDist summarizer with both CONANN and MetaMap annotation output. For the ROUGE-2 and ROUGE-SU4 metrics, both the FreqDist and SumBasic summarizers using any of the CONANN variations outperform FreqDist and SumBasic using MetaMap. Tables 16 and 17 show CONNAN with IPF word weights in word coverage filtering outperforms MetaMap annotations from 1.5% to 6.7% in the extrinsic text summarization task (marked with * in Tables 16 and 17).

For the Phrase Counting Concept Mapper (PCCM), ROUGE-2 and ROUGE-SU4 metrics indicate the best performing FreqDist and SumBasic summarizers use the binary involvement filter. For the Language Model Concept Mapper (LMCM), the ROUGE-2 and ROUGE-SU4 metrics show for FreqDist that the coverage filter used made no difference. However, for SumBasic, the best performing filter is Naïve Binary.

We conclude from the extrinsic evaluation results shown in Tables 16 and 17 that even with the lower intrinsic evaluation precision of the CONANN variations shown in Table 15, the extrinsic summarization task is able to use the CONANN concept output to identify important areas of text and improve system-generated text summarization performance as compared to human model summaries.

Table 16: ROUGE scores using Phrase Counting Concept Mapper (PCCM)

Summarizer	***ROUGE-2 Score***	***ROUGE-SU4 Score***
FreqDist - MetaMap	0.12080	0.21864
FreqDist - Coverage - Naïve Binary	0.12872	0.22199
FreqDist - Coverage - Naïve IPF*	0.12897	0.22252
FreqDist - Coverage – Involvement Binary	0.13018	0.22361
FreqDist - Coverage – Involvement IPF*	0.12872	0.22199
SumBasic - MetaMap	0.11412	0.19868
SumBasic - Coverage - Naïve Binary	0.11210	0.20191
SumBasic - Coverage - Naïve IPF*	0.11702	0.20670
SumBasic - Coverage – Involvement Binary	0.11834	0.21039
SumBasic - Coverage – Involvement IPF*	0.11827	0.20770

Table 17: ROUGE scores using Language Model Concept Mapper (LMCM)

Summarizer	***ROUGE-2 Score***	***ROUGE-SU4 Score***
FreqDist - MetaMap	0.12080	0.21864
FreqDist - Coverage - Naïve Binary	0.12897	0.22252
FreqDist - Coverage - Naïve IPF*	0.12897	0.22252
FreqDist - Coverage – Involvement Binary	0.12897	0.22292
FreqDist - Coverage – Involvement IPF*	0.12897	0.22292
SumBasic - MetaMap	0.10920	0.19868
SumBasic - Coverage - Naïve Binary	0.12028	0.21212
SumBasic - Coverage - Naïve IPF*	0.11614	0.20794
SumBasic - Coverage – Involvement Binary	0.11839	0.21053
SumBasic - Coverage - Involvement IPF*	0.11839	0.21053

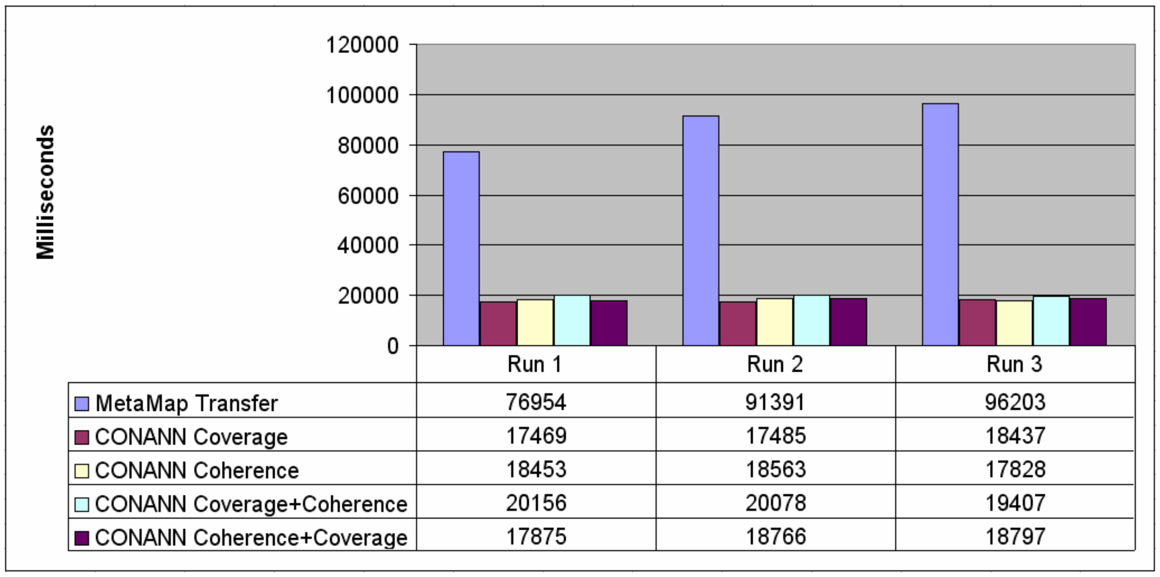
5.3 Semantic Annotation Using an Extended Coverage Filter

This section presents an intrinsic and an extrinsic evaluation of the CONANN semantic annotator using phrase counting in combination with an extended coverage filter. The extended coverage filter includes two heuristics in addition to the basic coverage filtering: (a) if an exact source and candidate phrase match occurs, the candidate phrase is automatically returned and does not need to meet minimum score requirements, and (b) if a candidate phrase consists of a single word and that word is in the list of source phrase words, the candidate phrase is given a maximum score. Complete details of the method are described in Section 4.

5.3.1 Intrinsic Evaluation

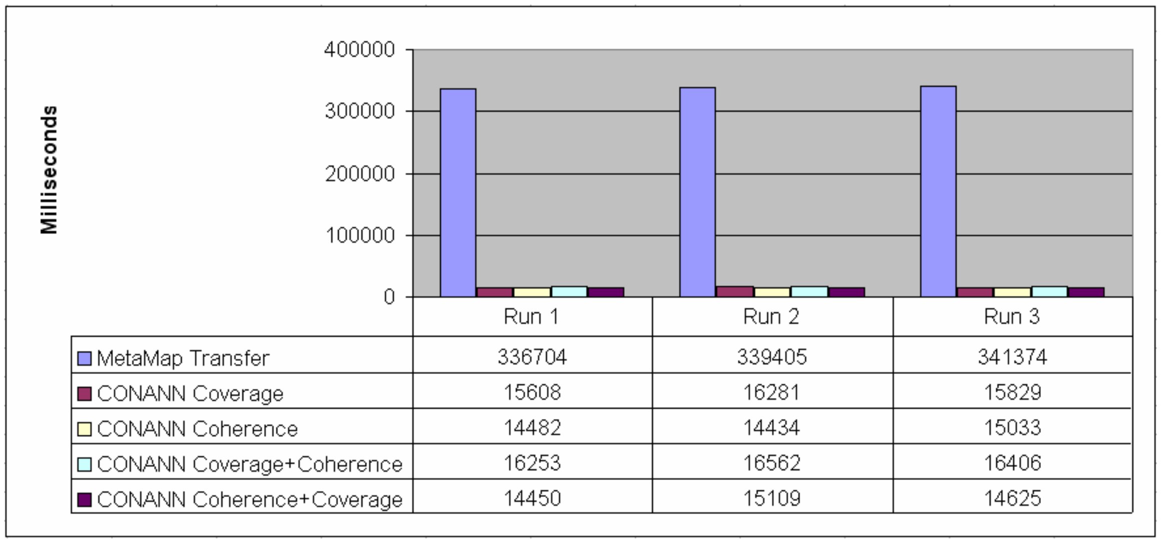
The first measurement is annotator initialization time, which is the time to load domain-specific resources into memory and prepare for annotation. Figure 39 shows the initialization time for each run of both annotators. For MetaMap, initialization time ranged from 1.3 to 1.6 minutes, while for our CONANN, initialization time ranged from 17 to 20 seconds. CONNAN is over four times faster in initialization time than MetaMap. Both systems exhibit stable initialization behavior.

Figure 40 presents the total time to annotate all 1,628 phrases in the evaluation corpus. MetaMap total annotation time was consistent across all three runs at 5.7 minutes, while CONANN ranged from 14.5 to 16.5 seconds on all three runs. CONNAN is over 20 times faster in total annotation time than MetaMap. Figure 41 shows the average time to annotate each phrase for each run of the annotator. Average Phrase Annotation Time is calculated by taking the total annotation time and dividing it by 1,628, which is the total number of phrases annotated. MetaMap average time to annotate a phrase was 208 milliseconds, while CONANN ranged from 9 to 10 milliseconds per phrase across all three runs. CONNAN is over 20 times faster in average annotation time per phrase than MetaMap.



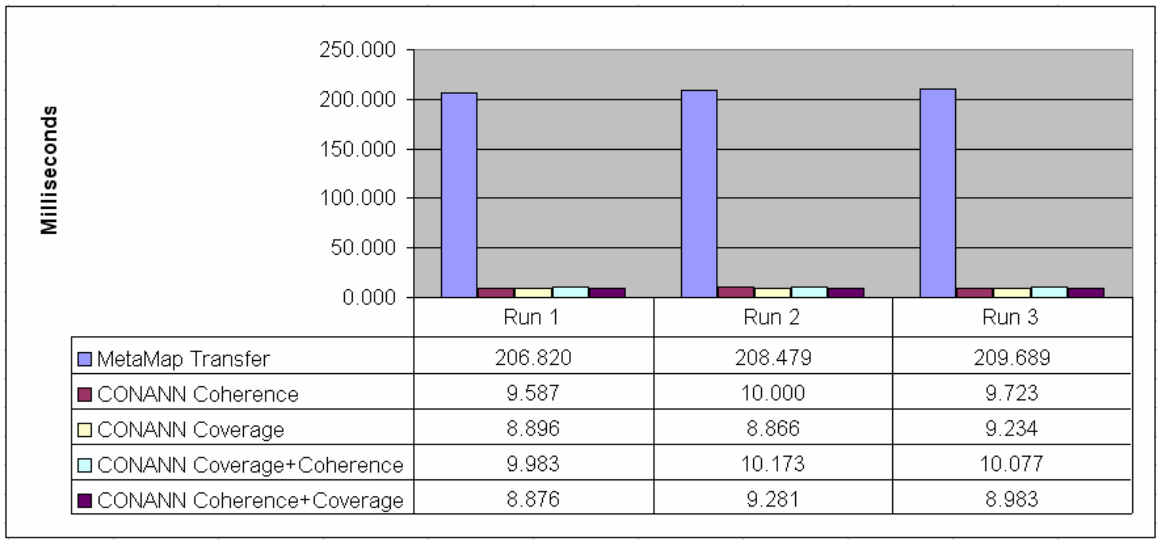
	Run 1	Run 2	Run 3
MetaMap Transfer	76954	91391	96203
CONANN Coverage	17469	17485	18437
CONANN Coherence	18453	18563	17828
CONANN Coverage+Coherence	20156	20078	19407
CONANN Coherence+Coverage	17875	18766	18797

Figure 39: Annotator initialization time



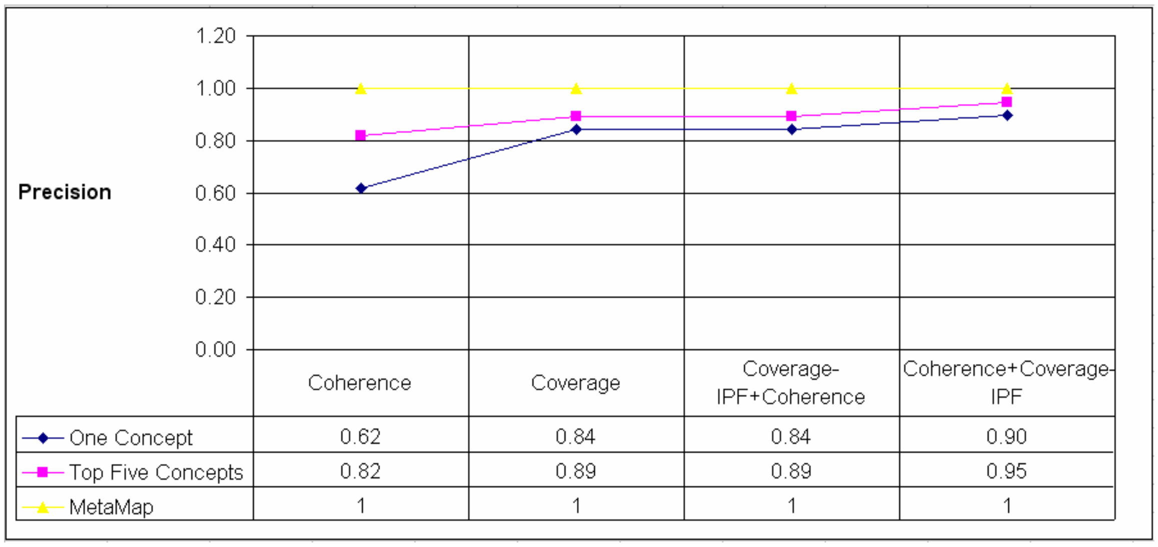
	Run 1	Run 2	Run 3
MetaMap Transfer	336704	339405	341374
CONANN Coverage	15608	16281	15829
CONANN Coherence	14482	14434	15033
CONANN Coverage+Coherence	16253	16562	16406
CONANN Coherence+Coverage	14450	15109	14625

Figure 40: Total annotation time



	Run 1	Run 2	Run 3
MetaMap Transfer	206.820	208.479	209.689
CONANN Coherence	9.587	10.000	9.723
CONANN Coverage	8.896	8.866	9.234
CONANN Coverage+Coherence	9.983	10.173	10.077
CONANN Coherence+Coverage	8.876	9.281	8.983

Figure 41: Average phrase annotation time



	Coherence	Coverage	Coverage-IPF+Coherence	Coherence+Coverage-IPF
One Concept	0.62	0.84	0.84	0.90
Top Five Concepts	0.82	0.89	0.89	0.95
MetaMap	1	1	1	1

Figure 42: Intrinsic precision of CONANN using Coherence+ExtendedCoverage method

While CONANN is over four times faster in initialization and over twenty times faster in average annotation time, the trade-off for the faster performance is less precision as compared to MetaMap (i.e., it is assumed that the precision of MetaMap concept annotation is 100%). CONANN was measured at 90% precision for exact concept matching, and 95% precision for relaxed concept matching using the best performing Coherence+ExtendedCoverage filter. The ExtendedCoverage+Coherence filter had seven percent worse precision than Coherence+ExtendedCoverage, indicating that the filtering order is important. The worst performing filter is Coherence alone when selecting a single concept, but jumps significantly higher when relaxed matching is used, indicating the correct concepts are available but candidate phrase order alone is not enough to achieve final best mapping. The ExtendedCoverage filter alone had a precision equal to ExtendedCoverage+Coherence, indicating the ExtendedCoverage filter is a strong filter which removes candidate phrases which would have been selected by the Coherence filter. To counter this effect, placing the Coherence filter before the ExtendedCoverage filter results in candidate phrases with strong ordering being selected, which are then further refined by using the semantic focusing of the ExtendedCoverage filter. Figure 42 summarizes the CONANN precision scores for each filter.

5.3.2 Extrinsic Evaluation

Table 18 shows the ROUGE-2 and ROUGE-SU4 scores for evaluating text summarization performance using the CONANN (Coherence+ ExtendedCoverage) and MetaMap annotator output. For the ROUGE-2 metric, MetaMap slightly outperforms CONANN using FreqDist (1% difference), while CONANN outperforms MetaMap using SumBasic by 7%. The results are similar for the ROUGE-SU4 scores. FreqDist using MetaMap has an approximately 2% advantage over FreqDist using CONANN. SumBasic using MetaMap has an approximately 5% advantage over SumBasic with CONANN. CONANN performs very closely to MetaMap in the extrinsic text summarization task. In addition, CONANN has a time advantage of performing annotation over twenty times faster than a state-of-the-art system, facilitating its use in online environments.

Table 18: Extrinsic text summarization task performance of CONANN using Coherence+ExtendedCoverage method

Summarization Method	ROUGE-2 Score	ROUGE-SU4 Score
FreqDist using MetaMap	0.1207	0.2200
FreqDist using CONANN	0.1192	0.2161
SumBasic using CONANN	0.1178	0.2098
SumBasic using MetaMap	0.1094	0.2003

5.4 Text Summarization Using Concept Chains (BioChain)

Tables 19 and 20 show the ROUGE-2 and ROUGE-SU4 scores, respectively, for each of the summarizers used to evaluate BioChainSumm, the automated concept-based summarizer. All of the BioChainSumm summarizer variations performed above the baseline summarizers Lead and Random, as well as the general-purpose summarizers MEAD, AutoSummarize, SweSum, and OTS. The best performing BioChainSumm summarizer is MostFrequentStrongChainConcept, which uses the most frequent concept within each strong chain to score sentences, as shown in the 4th row of Tables 19 and 20. The lower performance of AllStrongChainConcepts indicates that using more concepts did not retrieve additional sentences that would improve the summary as shown in the 6th, 7th, and 9th rows of Tables 19 and 20. It is also possible it may have retrieved too many irrelevant sentences, leaving out critical sentences. Adding the sentence position heuristic, where sentences at the beginning of the source text are scored higher than sentences at the end of the source text, degraded performance (MostFrequentStrongChainConcept-Position) as shown in the 5th row of Tables 19 and 20. Whereas in the news genre sentences at the beginning (lead sentences) often provide a good summary by themselves, it is not true for biomedical texts. Looking at the structure of texts in the evaluation corpus, the lead sentences are extracted from the Introduction section, which usually does not provide enough information for a summary. Based on the poor performance of lead sentences, research to investigate the location of where most of the model summary sentences are extracted from, such as the Discussion and Results section, was done. Semantic type filtering also reduced performance Semantic type filtering excludes concepts from certain UMLS semantic types. The semantic types to exclude are defined by a domain expert. The result of semantic type filtering (MostFrequentStrongChainConcept-Position-Filtered) is shown in the 11th row of Tables 19 and 20. While this may be helpful for personalized summaries, where the user wants to focus on summarizing specific aspects of a text, it proved harmful when summarizing the entire text. Combining sentence position and semantic type filtering resulted in the worst performance for all BioChainSumm summarizers (MostFrequentStrongChainConcept-Position-Filtered). This is to be expected, as neither feature individually improved performance. The BioChainSumm summarizers using all concepts within each strong chain (AllStrongChainConcepts) did not perform as well as using the most frequent concept in each strong chain as shown in the comparison of the 4th row with the 9th row of Table 19 and the comparison of the 4th row with the 8th row of

Table 20. Adding the sentence position heuristic also reduced performance (AllStrongChainConcepts-Position) as shown in the comparison of the 4th (9th) row with the 5th (10th) row of Table 19, respectively and the comparison of the 4th (8th) row with the 5th (9th) row of Table 20, respectively. This is the same effect as in the MostFrequentStrongChainConcept-Position summarizer. Interestingly, adding the semantic type filtering (AllStrongChainConcepts-Filtered) increased performance. This is the opposite effect of the MostFrequentStrongChainConcept-Position-Filtered summarizer. The reason semantic type filtering improved performance is it restricts the concepts used to a smaller subset than using all concepts. This may have resulted in more salient sentences being extracted with restricted number of concepts like the same way as the MostFrequentStrongChainConcept case, which uses the most frequent concept within each strong chain to score sentences and gives the highest performance among the BioChainSumm variations. While combining sentence position with semantic type filtering also improved performance (AllStrongChainConcepts-Position-Filtered) over the AllStrongChainsConcepts alone, the result is less than AllStrongChainConcepts-Filtered, again indicating adding sentence position hurts performance in biomedical text summarization.

The Baseline-Lead summarizer, which takes the first 20% of the sentences within the text, is the worst-performing summarizer. In news text summarization, this approach is difficult to compete against (Brandow, Mitze, & Rau, 1995). Features effective in summarizing news articles, such as sentence position, are not as effective in biomedical text summarization. This is further evidenced by the fact that the summarizers originally designed for general-purpose summarization (OTS, SweSum, AutoSummarize) used a sentence position as one of features and performed below the Baseline-Random summarizer. Baseline-Random randomly selects 20% of the sentences within the source text. The fact that random sentence selection performs unexpectedly well may be due to the length and structure of clinical trial texts which lend themselves to repetition. Therefore, random sentence selection is more likely to select a relevant sentence in longer texts such as biomedical texts. This is unlike the news genre where information content is much more compact, and so random sentence selection makes it difficult to select a relevant sentence.

The best performing summarizers are SumBasic (using both biomedical concepts and terms) and Lemur MMR. These summarizers performed above BioChainSumm. Both of these summarizers, SumBasic and Lemur MMR, use an information redundancy approach when building summaries. Their goal is to reduce the addition of information already included in the summary. SumBasic reduces the probability of re-selecting a term or concept when it has already been selected. Lemur MMR selects sentences which contribute more information to the summary. The high performance of (a) the Baseline-Random summarizer, which can perform competitively only with repetitive information, and (b) the high performance of the SumBasic and Lemur MMR summarizers shows that information redundancy is very much a concern in biomedical texts. BioChainSumm performs above general-purpose summarizers and random sentence selection, indicating it is useful for identifying important themes within a source text. BioChainSumm may be further improved by adding information redundancy to the sentence selection process, similar to the SumBasic and Lemur MMR summarization approaches. The evaluation shows BioChainSumm outperforms most general-purpose summarizers, as well as two baseline summarizers. BioChainSumm did not outperform two summarizers which focused on reducing information redundancy. The ROUGE performance of concept chaining (i.e., BioChain) shows it is an effective technique for identifying themes within a source text, but additional work on information redundancy is required to get optimal performance from a summarizer using the concept chaining approach. The following section describes the performance of the FreqDist Summarizer that takes into account information redundancy.

Table 19: ROUGE-2 scores for BioChainSumm summarizer evaluation

Row #	Summarizer	ROUGE-2
1	SumBasic-Term	0.11673
2	SumBasic-Concept	0.10940
3	Lemur-MMR	0.10708
4	BioChain-MostFrequentStrongChainConcept	0.10419
5	BioChain-MostFrequentStrongChainConcept-Position	0.10175
6	BioChain-AllStrongChainConcepts-Filtered	0.10043
7	BioChain-AllStrongChainConcepts-Position-Filtered	0.10043
8	BioChain-MostFrequentStrongChainConcept-Filtered	0.09868
9	BioChain-AllStrongChainConcepts	0.09708
10	BioChain-AllStrongChainConcepts-Position	0.09708
11	BioChain-MostFrequentStrongChainConcept-Position-Filtered	0.09660
12	MEAD	0.09254
13	Baseline-Random	0.08001
14	AutoSummarize	0.07977
15	SweSum	0.07513
16	OTS	0.07474
17	Baseline-Lead	0.07076

Table 20: ROUGE-SU4 scores for BioChainSumm summarizer evaluation

Row #	Summarizer	ROUGE-SU4
1	SumBasic-Term	0.21112
2	SumBasic-Concept	0.20034
3	Lemur-MMR	0.19874
4	BioChain-MostFrequentStrongChainConcept	0.19173
5	BioChain-MostFrequentStrongChainConcept-Position	0.18832
6	BioChain-AllStrongChainConcepts-Filtered	0.18659
7	BioChain-AllStrongChainConcepts-Position-Filtered	0.18659
8	BioChain-AllStrongChainConcepts	0.18557
9	BioChain-AllStrongChainConcepts-Position	0.18557
10	BioChain-MostFrequentStrongChainConcept-Filtered	0.18179
11	BioChain-MostFrequentStrongChainConcept-Position-Filtered	0.17946
12	MEAD	0.17629
13	Baseline-Random	0.16396
14	AutoSummarize	0.15171
15	SweSum	0.15115
16	OTS	0.14919
17	Baseline-Lead	0.13953

5.5 Text Summarization Using Concept Frequency Distribution (FreqDist)

Table 21 shows the ROUGE-2 scores for each summarizer. The best performing summarizes are the context-based SumBasic and our FreqDist. The FreqDist summarizer, when using Dice’s coefficient for its similarity measure, outperforms all of the other summarizers using both terms and concepts as unit items. The performance of FreqDist using concepts and terms is close. The SumBasic summarizer performs better using terms rather than concepts, where the use of terms scored one percentage point better than the use of concepts. Our FreqDist summarizer performs best when using Dice’s coefficient as the similarity measure between the summary and the source text. Dice is a measure of the common membership of unit items in the summary and source text. Other similarity measures, such as cosine, take into consideration not only membership, but also the weight (frequency) of each unit item. The frequency distribution model approach requires no additional weighting of unit items to obtain good results. However, the use of frequency weights in comparing source text and candidate summaries also performs above the baseline and general-purpose summarizers using Cosine and Unit Item Frequency. The use of frequency weights does not outperform the use of simple unit item membership. The worst performing summarizers are the ones based on the FreqDist algorithm using the Vector Subtraction and the Euclidean distance similarity measures. These two similarity measures do not work well regardless of the unit items (i.e., terms or concepts). However, in both methods, the use of concepts outperforms the use of terms. The poor performance of Vector Subtraction and Euclidean distance similarity measures is likely due to their susceptibility to outlier values in the vectors, and is in line with other studies in distributional similarity (L. Lee, 1999).

The MEAD summarizer, which employs a combination of features to identify significant sentences, outperformed the Random sentence and Lead sentence baseline summarizers, and in fact fell just below the SumBasic and FreqDist summarizers in the performance table. The general purpose summarizers AutoSummarize and SweSum performed comparably, performing below the Random sentence baseline but above the Lead sentence baseline. The simple use of frequency without considering either additional features (MEAD) or context sensitivity (SumBasic/FreqDist) is not effective with the summarization of biomedical text.

Table 21: ROUGE-2 scores for FreqDist summarizer evaluation

FreqDist-Term_Dice	0.22176
FreqDist-Concept_Dice	0.21997
SumBasic-Term	0.21112
FreqDist-Term_UnitFrequency	0.20707
SumBasic-Concept	0.20034
FreqDist-Concept_Cosine	0.19932
FreqDist-Concept_UnitFrequency	0.19932
MEAD	0.17629
FreqDist-Term_Cosine	0.17358
Baseline-Random	0.16396
AutoSummarize	0.15171
SweSum	0.15115
Baseline-Lead	0.13953
FreqDist-Concept_VectorSubtraction	0.11435
FreqDist-Concept_Euclidean	0.09236
FreqDist-Term_Euclidean	0.07516
FreqDist-Term_VectorSubtraction	0.05716

Table 22 shows the ROUGE-SU4 scores for each summarizer. In general, the ordering of the summarizer performance is about the same as in ROUGE-2. The best performing summarizers are the same as in ROUGE-2: our FreqDist and SumBasic. In both cases, the use of terms outperforms the use of concepts, but only by a margin of about 0.75 percentage points in both cases. Our FreqDist summarizer again performs best when using Dice's coefficient as the similarity measure between the summary and the source text. The Cosine and Unit Frequency also performed above the baseline and general-purpose summarizers. The use of the Vector Subtraction and Euclidean distance similarity methods with FreqDist was at the bottom of the performance list, as in ROUGE-2. The MEAD and FreqDist with Cosine similarity performed about the same using terms. The AutoSummarize and SweSum summarizers also performed closely, and were not much better than the Lead sentence summarizer. The Lead sentence baseline summarizer gave the worst performance when excluding the Vector Subtraction and Euclidean versions of FreqDist. The Random sentence baseline summarizer was in the middle of the performance table.

Table 22: ROUGE-SU4 scores for FreqDist summarizer evaluation

FreqDist-Term_Dice	0.12653
FreqDist-Concept_Dice	0.12070
SumBasic-Term	0.11673
FreqDist-Term_UnitFrequency	0.11664
SumBasic-Concept	0.10940
FreqDist-Concept_Cosine	0.10781
FreqDist-Concept_UnitFrequency	0.10781
FreqDist-Term_Cosine	0.09310
MEAD	0.09254
Baseline-Random	0.08001
AutoSummarize	0.07977
SweSum	0.07513
Baseline-Lead	0.07076
FreqDist-Concept_VectorSubtraction	0.05607
FreqDist-Concept_Euclidean	0.04356
FreqDist-Term_Euclidean	0.03429
FreqDist-Term_VectorSubtraction	0.02862

It is interesting to note the baseline summarizer using random sentence selection performed nearly in the middle of the performance rankings for both ROUGE-2 and ROUGE-SU4. The high performance of random sentence selection is due to the high information redundancy within the lengthy text. Context-sensitive methods, such as SumBasic and our FreqDist methods, significantly outperform the random baseline. Context-sensitive methods consider the sentences already selected for a summary before choosing the next sentence to add to a summary. Context-sensitive methods are used to reduce information redundancy within a summary. Excluding the FreqDist summarizers using the Vector Subtraction and Euclidean distance methods, the use of the lead sentences (i.e., Baseline-Lead in Tables 21 and 22) of a biomedical text generates the worst performance. This is important to note, because in text summarization work using the news genre, the lead sentence method often generates a very good summary (Goldstein et al., 1999). This is because news stories are usually written so that the most important information appears at the beginning of the text, and the least important information at the end. However, in biomedical texts this assumption is invalid, as shown in Tables 21 and 22.

Using context-sensitive frequency methods, the use of concepts does not outperform the use of terms. However, terms and concepts perform closely. The use of concepts rather than terms is valuable for building personalized summarizers, where it is easier for the user to select important concepts to summarize than important terms. This is because the concepts are defined for a domain, whereas terms are selected by author(s) of a paper and used in the text of the paper. To personalize a summary without domain-specific concepts, the user needs to know the important terms appearing in a text. In general, it is not easy for users to know terms in papers in advance before they read these papers.

5.6 Text Summarization Using BioChain and FreqDist (ChainFreq)

Tables 23 and 24 show the ROUGE-2 and ROUGE-SU4 scores comparing the performance of the ChainFreqSumm summarizer. The ChainFreqSumm summarizer combines the BioChain and FreqDist algorithms into a single algorithm in order to leverage the BioChain strength identifying text themes and the information redundancy control of FreqDist. As can be seen from the Baseline-Random summarizer, randomly picking sentences performs well. This is an indication that biomedical texts contain a large amount of redundancy. As can also be seen from the BioChainSumm evaluation, redundancy can decrease summarizer performance. The hybrid ChainFreqSumm summarizer (BioChain method plus the FreqDist method) is an attempt to find a subset of the most important sentences using domain-specific criteria, and then remove redundancy from the subset. As show in Table 24, the ChainFreqSumm summarizer performs best when all concepts in the strong chains are used, which is the opposite of what occurs when the BioChain method is used alone. This is most likely because using all strong concepts results in a larger pool of sentences for the FreqDist method to select from. Using the ROUGE-SU4 metric, the hybrid ChainFreqSumm summarizer is the best performer, but is slightly outperformed by the FreqDistSumm term method when the ROUGE-2 metric is used. The result in combining the two approaches is that the use of concept approaches for finding salient sentences is improved over the individual methods of FreqDist and BioChain. We conclude that a summarizer which (a) first identifies a subset of important sentences based on domain-specific criteria, and (b) then prunes the subset of sentences by removing redundancy leads to an effective domain-specific summarizer.

Table 23: ROUGE-2 scores for ChainFreqSumm evaluation

Summarizer	ROUGE-2 Score
FreqDistSumm-Term-Dice	0.12653
ChainFreqSumm-AllStrongChainConcepts-Dice	0.12216
FreqDistSumm-Concept-Dice	0.12070
SumBasic-Term	0.11673
SumBasic-Concept	0.10940
Lemur-MMR	0.10708
ChainFreqSumm-MostFrequentStrongChainConcept-Dice	0.10652
BioChainSumm-MostFrequentStrongChainConcept	0.10419
BioChainSumm-AllStrongChainConcepts	0.09708
Mead	0.09254
Baseline-Random	0.08001
MSWord	0.07977
SweSum	0.07513
OTS	0.07474
Baseline-Lead	0.07076

Table 24: ROUGE-SU4 scores for ChainFreqSumm evaluation

Summarizer	ROUGE-SU4 Score
ChainFreqSumm-AllStrongChainConcepts-Dice	0.22303
FreqDistSumm-Term-Dice	0.22176
FreqDistSumm-Concept-Dice	0.21997
SumBasic-Term	0.21112
ChainFreqSumm-MostFrequentStrongChainConcept-Dice	0.20158
SumBasic-Concept	0.20034
Lemur-MMR	0.19874
BioChainSumm-MostFrequentStrongChainConcept	0.19173
BioChainSumm-AllStrongChainConcepts	0.18557
Mead	0.17629
Baseline-Random	0.16396
MSWord	0.15171
SweSum	0.15115
OTS	0.14919
Baseline-Lead	0.13953

5.7 Significance Testing of ROUGE Summarization Scores

Analysis of variance using ANOVA methods on three sets of data of was performed. The idea is to see if the performance of the domain-specific summarizers BioChain and FreqDist statistically outperformed general-purpose summarizers, and if the performance of concepts versus terms is significant. The ROUGE scores indicate that the BioChain and FreqDist summarizers outperform nearly all baseline summarizers. The ANOVA analysis is performed twice for each summarizer: once for ROUGE-2 scores and once for ROUGE-SU4 scores. The approach taken here is to (a) group together ROUGE scores from multiple variations of the same summarizer, and then (b) compare the ROUGE score mean of this group with the ROUGE score mean of several baseline summarizers. The BioChain and FreqDist summarizers each have multiple variations (for example, one using concepts and one using terms). These variations of the same summarizer are grouped together to form a mean. The summarizer variations are then compared to the mean of several baseline summarizers. This method of analysis is different than most existing ANOVA testing of ROUGE scores, which seek to test the significance of a single summarizer against multiple baseline summarizers. ROUGE scores are designed to be a ranking mechanism and often fail significance level tests between scores (Evans, 2005). If a summarizer fails significance level testing, its use may still be justified by above-average ROUGE scores and by including more features not available in other summarizers.

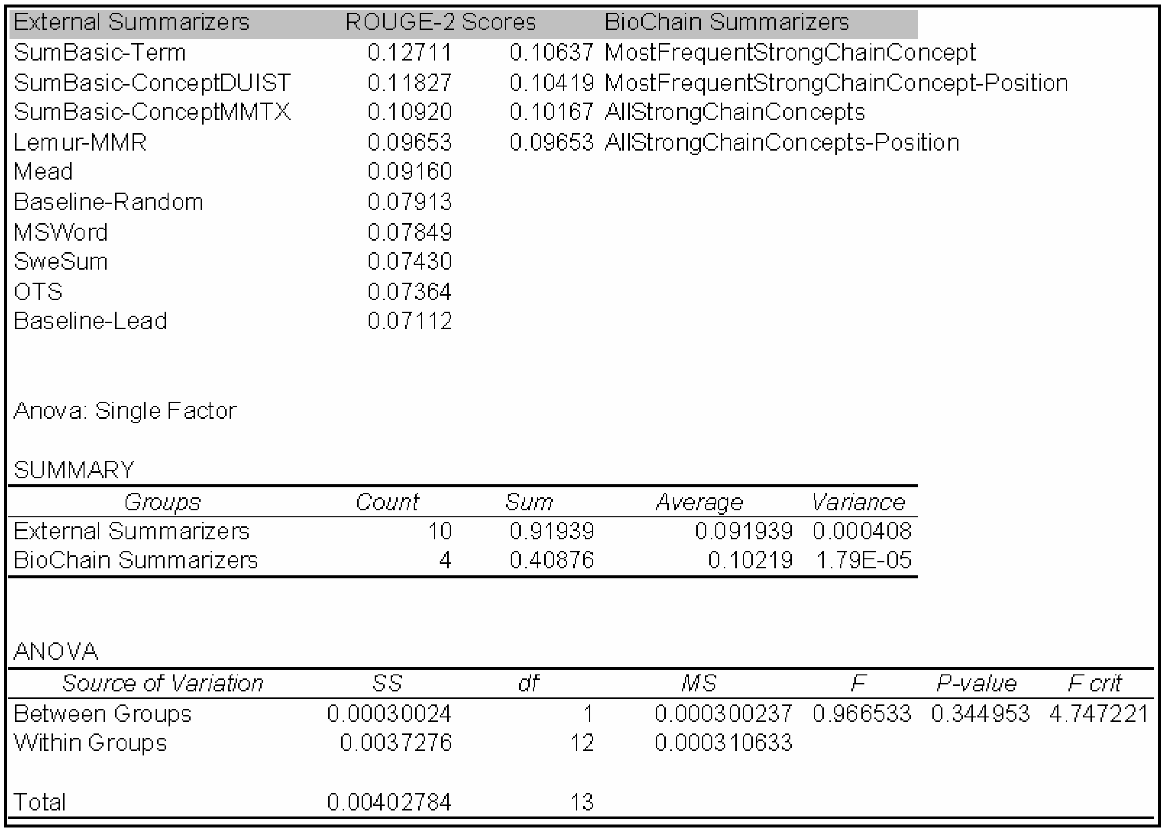
The analysis is done using the ANOVA:Single Factor method. Two sections of output are generated. The Summary section shows the descriptive statistics for the two groups (Count, Sum, Average, and Variance). The ANOVA section shows the *F* and *F*-

critical values. Each ANOVA test uses an alpha value equal to 0.05. If the *F* value is greater than the *F-critical* value then the null hypothesis that the two means are equal can be rejected. Also, the *P-value* is the probability of obtaining an *F-value* bigger than the *F-critical* value by chance. If the *P-value* is less than the alpha value (chosen to be .05 for this study), the result is statistically significant and the null hypothesis can be rejected. Both the *F* and *F-critical* values as well as the *P-value* are reported.

5.7.1 BioChainSumm

The BioChainSumm summarizer ROUGE scores are compared with the scores of a group of external summarizers using ANOVA Single Factor. The ROUGE-2 and ROUGE-SU4 scores are divided into two groups: external summarizers and BioChainSumm summarizers. There are ten external summarizers and four BioChainSumm summarizers. Two ANOVA Single Factor tests are performed: one for ROUGE-2 scores, as shown in Figure 43 and one for ROUGE-SU4 scores, as shown in Figure 44. The idea is to see if there is a statistically significant difference between the BioChainSumm score mean and the external summarizer score mean for both ROUGE-2 and ROUGE-SU4 scores. That is, how much of the variability of the scores is due to random errors, and how much is due to summarizer performance. The null hypothesis in both cases is that the two means (BioChainSumm and external summarizers) are the same. In the ROUGE-2 case, the result is F(10, 4) = 0.97, p < .05. The *F-value* is 0.97, which is less than the *F-critical* value of 4.75, which means the null hypothesis cannot be rejected. Also, the *P-value* of 0.35 is greater than the alpha value of 0.05 does not show a statistically significant result. In the ROUGE-SU4 case, the result is F(10, 4) = 0.56, p <

.05. The *F-value* is 0.56, which is less than the *F-critical* value of 4.75, which means the null hypothesis cannot be rejected. Also, the *P-value* of 0.47 is greater than the alpha value of 0.05 does not show a statistically significant result. The conclusion that can be reached from running both sets of ROUGE scores through ANOVA is that the BioChainSumm summarizers are not statistically significantly different when compared to the scores of the external summarizers, and that this variation is not due to chance, but due to the performance of the summarizers. In this case, best-performing variation of BioChainSumm, MostFrequentStrongChainConcept, outperforms all other external summarizers except for the SumBasic summarizer variations. Even though it is not statistically significantly different, BioChainSumm still performs well according to ROUGE ranking, and is more useful than term-based summarization because it allows for easier summary customization through the use of domain-specific concepts.



External Summarizers	ROUGE-2 Scores		BioChain Summarizers
SumBasic-Term	0.12711	0.10637	MostFrequentStrongChainConcept
SumBasic-ConceptDUIST	0.11827	0.10419	MostFrequentStrongChainConcept-Position
SumBasic-ConceptMMTX	0.10920	0.10167	AllStrongChainConcepts
Lemur-MMR	0.09653	0.09653	AllStrongChainConcepts-Position
Mead	0.09160		
Baseline-Random	0.07913		
MSWord	0.07849		
SweSum	0.07430		
OTS	0.07364		
Baseline-Lead	0.07112		

Anova: Single Factor

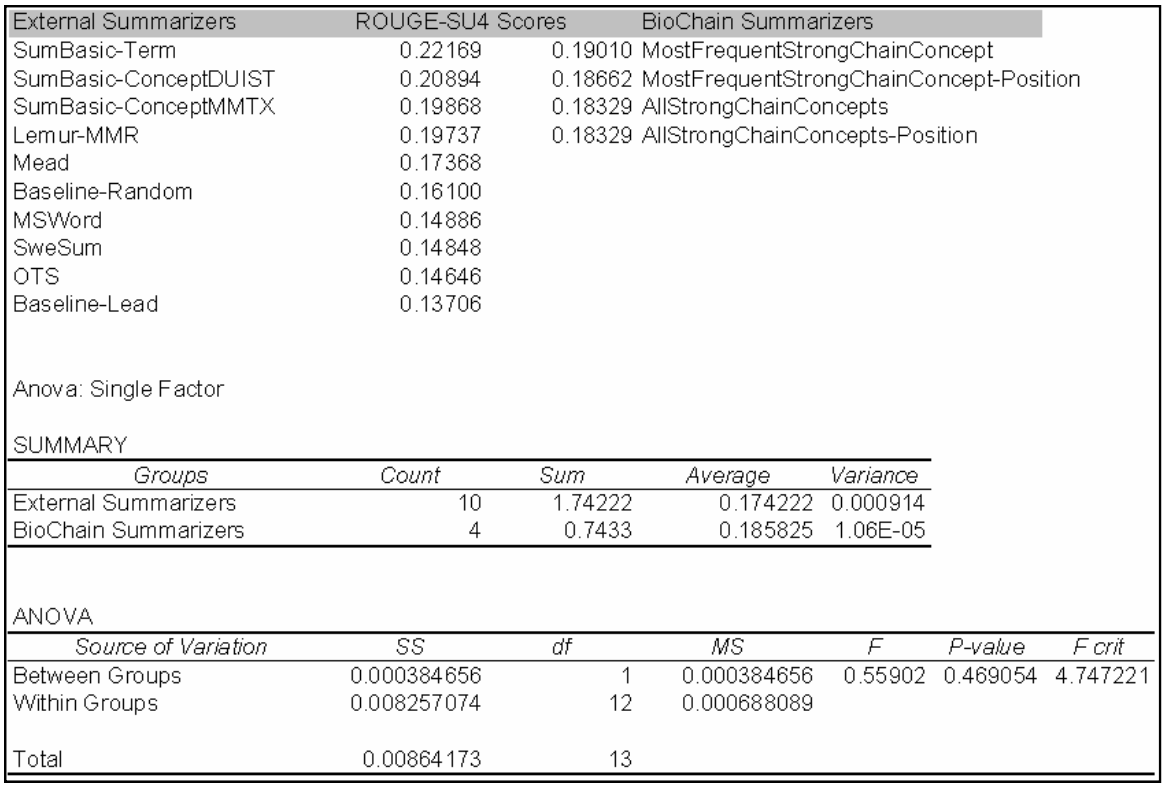
SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
External Summarizers	10	0.91939	0.091939	0.000408
BioChain Summarizers	4	0.40876	0.10219	1.79E-05

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	0.00030024	1	0.000300237	0.966533	0.344953	4.747221
Within Groups	0.0037276	12	0.000310633			
Total	0.00402784	13				

Figure 43: BioChainSumm significance level testing for ROUGE-2 scores



External Summarizers	ROUGE-SU4 Scores		BioChain Summarizers
SumBasic-Term	0.22169	0.19010	MostFrequentStrongChainConcept
SumBasic-ConceptDUIST	0.20894	0.18662	MostFrequentStrongChainConcept-Position
SumBasic-ConceptMMTX	0.19868	0.18329	AllStrongChainConcepts
Lemur-MMR	0.19737	0.18329	AllStrongChainConcepts-Position
Mead	0.17368		
Baseline-Random	0.16100		
MSWord	0.14886		
SweSum	0.14848		
OTS	0.14646		
Baseline-Lead	0.13706		

Anova: Single Factor

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
External Summarizers	10	1.74222	0.174222	0.000914
BioChain Summarizers	4	0.7433	0.185825	1.06E-05

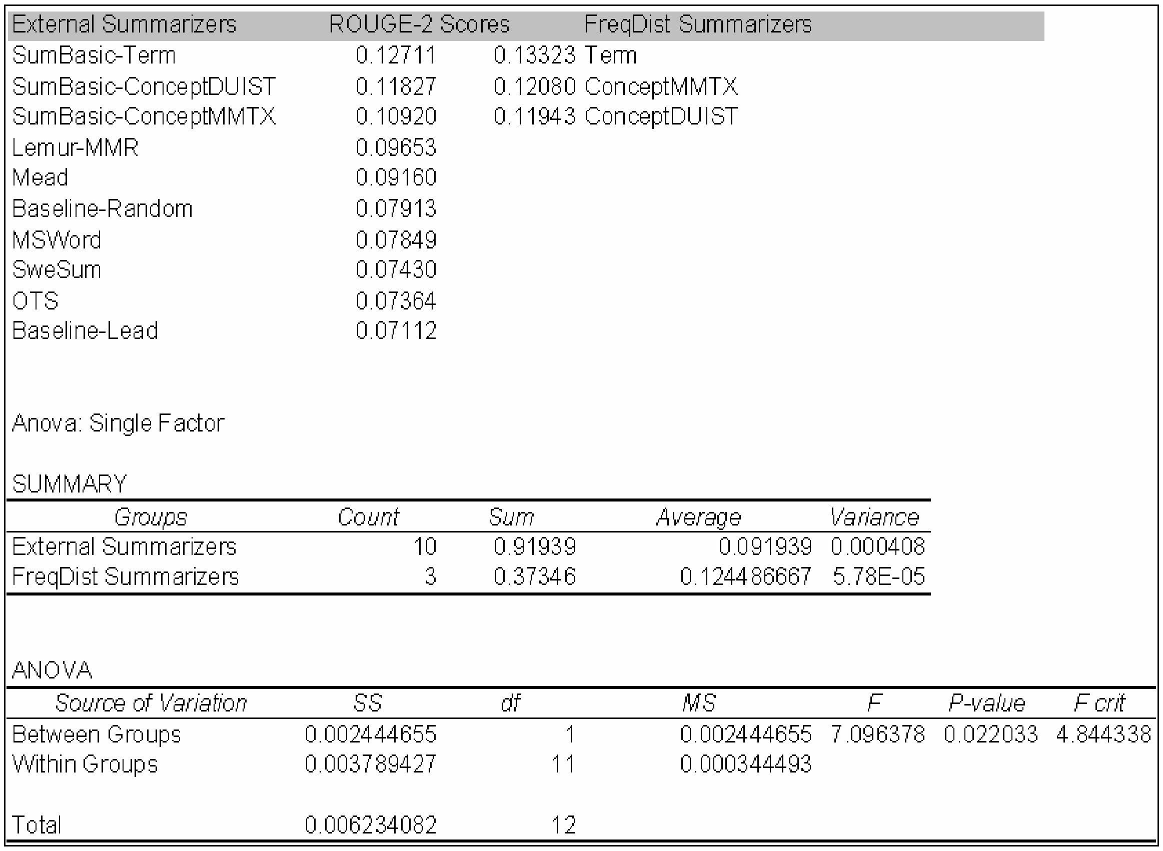
ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	0.000384656	1	0.000384656	0.55902	0.469054	4.747221
Within Groups	0.008257074	12	0.000688089			
Total	0.00864173	13				

Figure 44: BioChainSumm significance level testing for ROUGE-SU4 scores

5.7.2 FreqDistSumm

Two ANOVA Single Factor tests are performed for FreqDist analysis: one for ROUGE-2 scores, as shown in Figure 45 and one for ROUGE-SU4 scores, as shown in Figure 46. There are ten external summarizers and three FreqDist summarizer variations. The null hypothesis in both ROUGE cases is that the two means (FreqDist and external summarizers) are the same. In the ROUGE-2 case, the result is F(10, 3) = 7.10, $p < .05$. The *F-value* is 7.10, which is greater than the *F-critical* value of 4.84, which means the null hypothesis can be rejected. Also, the *P-value* of 0.35 is less than the alpha value of 0.05 which shows a statistically significant result. In the ROUGE-SU4 case, the result is F(10, 3) = 6.71, $p < .05$. The *F-value* is 6.71, which is greater than the *F-critical* value of 4.84, which means the null hypothesis can be rejected. Also, the *P-value* of 0.03 is less than the alpha value of 0.05 which shows a statistically significant result. The conclusion that can be reached from running both sets of ROUGE scores through ANOVA is that the FreqDist summarizers are statistically significantly different when compared to the scores of the external summarizers.



External Summarizers	ROUGE-2 Scores		FreqDist Summarizers
SumBasic-Term	0.12711	0.13323	Term
SumBasic-ConceptDUIST	0.11827	0.12080	ConceptMMTX
SumBasic-ConceptMMTX	0.10920	0.11943	ConceptDUIST
Lemur-MMR	0.09653		
Mead	0.09160		
Baseline-Random	0.07913		
MSWord	0.07849		
SweSum	0.07430		
OTS	0.07364		
Baseline-Lead	0.07112		

Anova: Single Factor

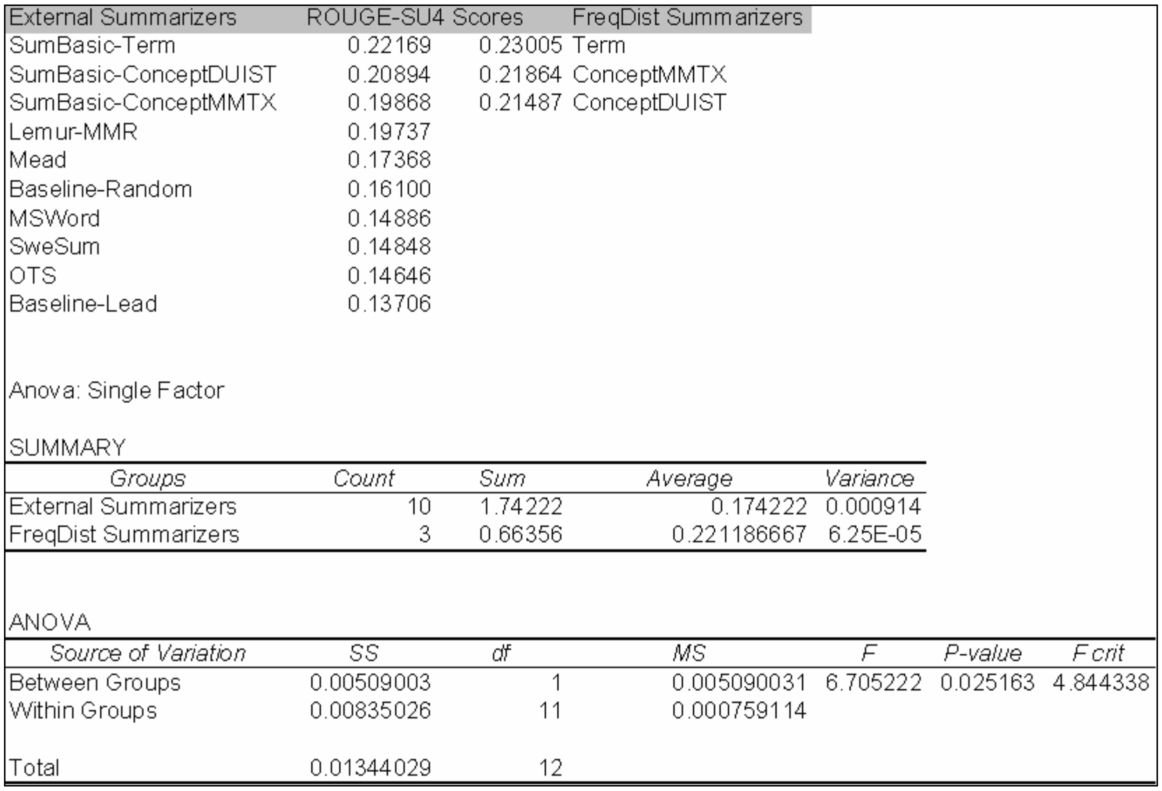
SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
External Summarizers	10	0.91939	0.091939	0.000408
FreqDist Summarizers	3	0.37346	0.124486667	5.78E-05

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	0.002444655	1	0.002444655	7.096378	0.022033	4.844338
Within Groups	0.003789427	11	0.000344493			
Total	0.006234082	12				

Figure 45: FreqDist significance level testing for ROUGE-2 scores



External Summarizers	ROUGE-SU4 Scores	FreqDist Summarizers	
SumBasic-Term	0.22169	0.23005	Term
SumBasic-ConceptDUIST	0.20894	0.21864	ConceptMMTX
SumBasic-ConceptMMTX	0.19868	0.21487	ConceptDUIST
Lemur-MMR	0.19737		
Mead	0.17368		
Baseline-Random	0.16100		
MSWord	0.14886		
SweSum	0.14848		
OTS	0.14646		
Baseline-Lead	0.13706		

Anova: Single Factor

SUMMARY

Groups	Count	Sum	Average	Variance
External Summarizers	10	1.74222	0.174222	0.000914
FreqDist Summarizers	3	0.66356	0.221186667	6.25E-05

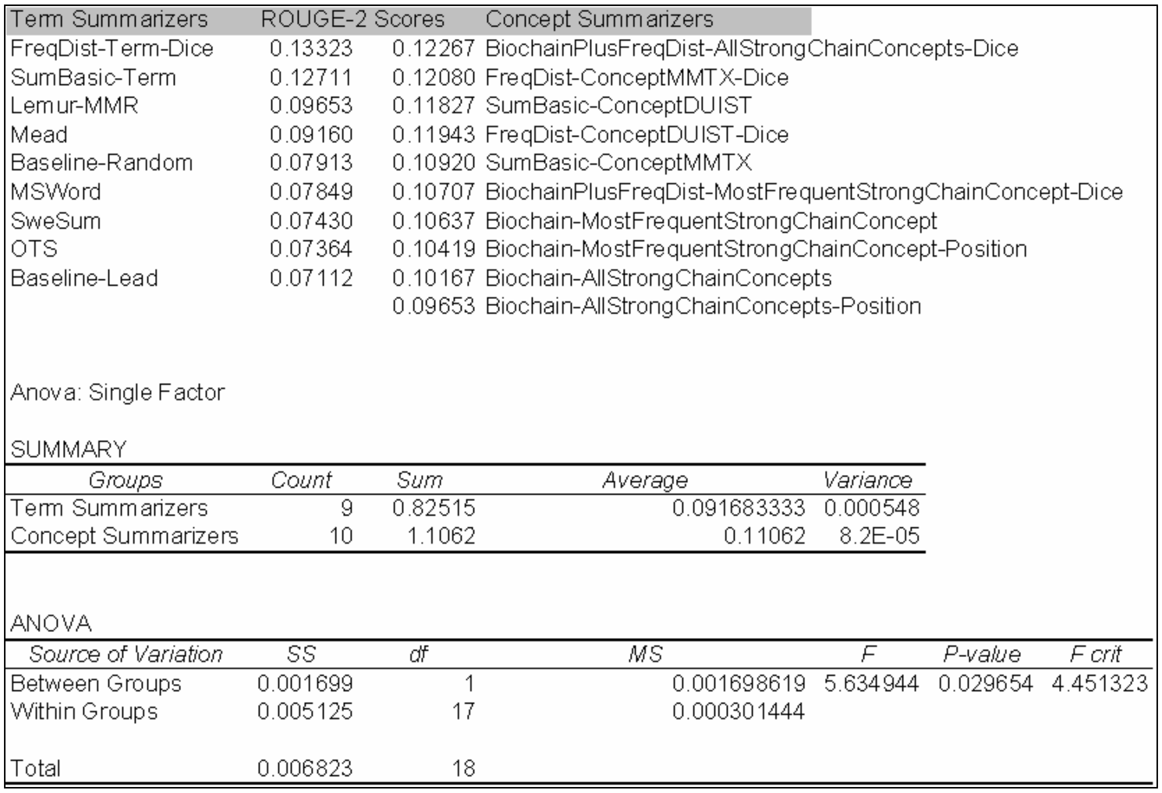
ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	0.00509003	1	0.005090031	6.705222	0.025163	4.844338
Within Groups	0.00835026	11	0.000759114			
Total	0.01344029	12				

Figure 46: FreqDist significance level testing for ROUGE-SU4 scores

5.7.3 Concepts versus Terms

The use of concepts in summarizers to find important areas of text as opposed to the use of terms is studied by first creating two groups of summarizers: term and concept. Two ANOVA Single Factor tests are performed: one for ROUGE-2 scores, as shown in Figure 47 and one for ROUGE-SU4 scores, as shown in Figure 48. There are nine term-based summarizers and ten concept-based summarizer variations. The null hypothesis in both ROUGE cases is that the two means (term- and concept-based summarizers) are the same. In the ROUGE-2 case, the result is F(9, 10) = 5.63, *p* < .05. The *F-value* is 5.63, which is greater than the *F-critical* value of 4.45, which means the null hypothesis can be rejected. Also, the *P-value* of 0.30 is less than the alpha value of 0.05 shows a statistically significant result. In the ROUGE-SU4 case, the result is F(9, 10) = 5.01, *p* < .05. The *F-value* is 5.01, which is greater than the *F-critical* value of 4.45, which means the null hypothesis can be rejected. Also, the *P-value* of 0.04 is less than the alpha value of 0.05 which shows a statistically significant result. The conclusion that can be reached from running both sets of ROUGE scores through ANOVA is that the concept-based summarizers are statistically significantly different when compared to the scores of the term-based summarizers.



Term Summarizers	ROUGE-2 Scores		Concept Summarizers
FreqDist-Term-Dice	0.13323	0.12267	BiochainPlusFreqDist-AllStrongChainConcepts-Dice
SumBasic-Term	0.12711	0.12080	FreqDist-ConceptMMTX-Dice
Lemur-MMR	0.09653	0.11827	SumBasic-ConceptDUIST
Mead	0.09160	0.11943	FreqDist-ConceptDUIST-Dice
Baseline-Random	0.07913	0.10920	SumBasic-ConceptMMTX
MSWord	0.07849	0.10707	BiochainPlusFreqDist-MostFrequentStrongChainConcept-Dice
SweSum	0.07430	0.10637	Biochain-MostFrequentStrongChainConcept
OTS	0.07364	0.10419	Biochain-MostFrequentStrongChainConcept-Position
Baseline-Lead	0.07112	0.10167	Biochain-AllStrongChainConcepts
		0.09653	Biochain-AllStrongChainConcepts-Position

Anova: Single Factor

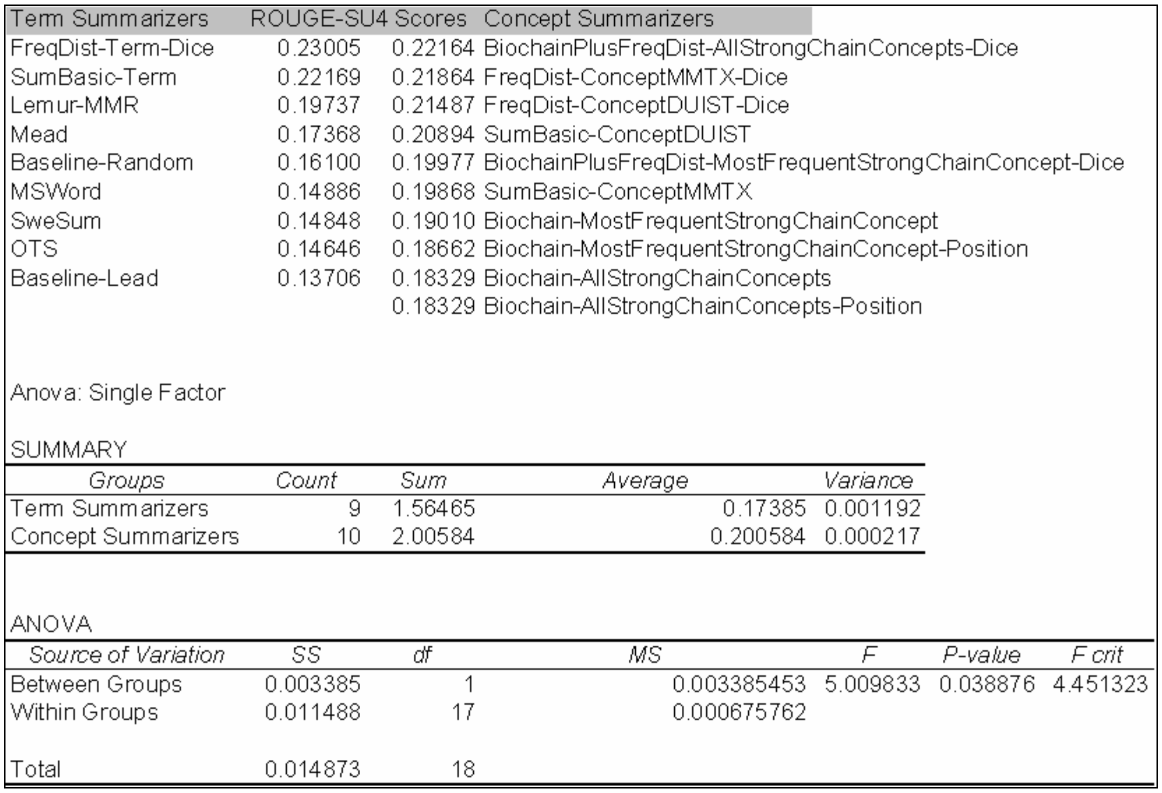
SUMMARY

Groups	Count	Sum	Average	Variance
Term Summarizers	9	0.82515	0.091683333	0.000548
Concept Summarizers	10	1.1062	0.11062	8.2E-05

ANOVA

Source of Variation	SS	df	MS	F	P-value	F crit
Between Groups	0.001699	1	0.001698619	5.634944	0.029654	4.451323
Within Groups	0.005125	17	0.000301444			
Total	0.006823	18				

Figure 47: Term versus Concept summarizer significance level testing for ROUGE-2 scores



Term Summarizers	ROUGE-SU4 Scores		Concept Summarizers
FreqDist-Term-Dice	0.23005	0.22164	BiochainPlusFreqDist-AllStrongChainConcepts-Dice
SumBasic-Term	0.22169	0.21864	FreqDist-ConceptMMTX-Dice
Lemur-MMR	0.19737	0.21487	FreqDist-ConceptDUIST-Dice
Mead	0.17368	0.20894	SumBasic-ConceptDUIST
Baseline-Random	0.16100	0.19977	BiochainPlusFreqDist-MostFrequentStrongChainConcept-Dice
MSWord	0.14886	0.19868	SumBasic-ConceptMMTX
SweSum	0.14848	0.19010	Biochain-MostFrequentStrongChainConcept
OTS	0.14646	0.18662	Biochain-MostFrequentStrongChainConcept-Position
Baseline-Lead	0.13706	0.18329	Biochain-AllStrongChainConcepts
		0.18329	Biochain-AllStrongChainConcepts-Position

Anova: Single Factor

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
Term Summarizers	9	1.56465	0.17385	0.001192
Concept Summarizers	10	2.00584	0.200584	0.000217

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	0.003385	1	0.003385453	5.009833	0.038876	4.451323
Within Groups	0.011488	17	0.000675762			
Total	0.014873	18				

Figure 48: Term versus Concept summarizer significance level testing for ROUGE-2 scores

5.8 Update Task in DUC 2007

In the DUC 2007 update task, systems are asked to produce short summaries of newswire articles, assuming a user has read a set of previous, related article texts. The idea is to present new information that the user has not already read from the set of preceding article texts. The 2007 DUC was our first chance to participate in a DUC event. The FreqDist Summarizer, which had good results in the biomedical domain, was adapted to the DUC Update Task and was named FreqDistUpdate. The FreqDistUpdate summarizer creates a summary of a source text which has approximately the same frequency distribution of terms as the source text. Several modifications to the base frequency distribution summarizer were implemented for the DUC Update Task to account for a statement expressing an information need, as well as summary length limitations.

NIST provided four different evaluations of each of the 22 systems submitted: ROUGE, Basic Elements (BE), Pyramid, and Responsiveness. The ROUGE, Basic Element, and Pyramid evaluations use increasingly larger units of text to measure overlap between a system-generated summary and a set of model summaries. ROUGE (C. Lin, 2005) measures the n-gram overlap between a system summary and the model summaries. Basic Elements moves beyond simple n-gram matching to find minimal semantic units, which are defined to be heads of syntactic units, such as noun phrases, and also relationship triples (E. Hovy, Lin, & Zhou, 2005), (E. Hovy, Lin, Zhou, & Fukumoto, 2006). Pyramid uses a set of manually annotated semantic units derived from the system summary and the model summaries (A. Nenkova & Passonneau, 2004). The Responsiveness score is a human assessment on a scale of 1 (low) to 5 (high) of how much information the summary provides in order to address the information need defined in the topic statement.

Model summaries were written using ten different human summarizers. There were four update summaries written for each document cluster. Two baseline summarizers were also provided by NIST. The Baseline 1 summarizer returns the first 100 words from the most recent document in a cluster. The Baseline 2 summarizer is an HMM-based summarizer which performed well in DUC-2004.

The official NIST results placed the FreqDistUpdate system 21 out of 22 systems based on the ROUGE-2 and ROUGE-SU4 scores. FreqDistUpdate did better in the BE evaluation, where FreqDistUpdate placed 14 out of 22, ahead of Baseline 1 Summarizer but below Baseline 2 Summarizer. The Pyramid evaluation assigned FreqDistUpdate an average score of 2.23 out of a possible five, with scores ranging from 1 to 4 for each of the document set summaries. The Baseline 1 Summarizer produced an average Pyramid score of 1.69, while the BaseLine 2 Summarizer had an average Pyramid score of 2.70. In the Responsiveness evaluation FreqDistUpdate placed 15 out of 22. It is interesting that the ROUGE scores placed FreqDistUpdate much lower than both BE and Responsiveness. It appears the ROUGE scores reflect that FreqDistUpdate did not select the same terms as the model summaries, while the BE evaluation, which focus more on semantic than syntactic units, and the Responsiveness evaluation, which is performed by humans, reflected that FreqDistUpdate did select information which was considered important to the human assessors. The use of different evaluation systems in DUC is useful for providing insight into different aspects of a summarization system.

There are several areas where the FreqDistUpdate system can be improved:

1. A list of important words derived from the topic statement is a heuristic which needs to be empirically evaluated whether or not its inclusion is helpful. Also, the penalization of sentences which do not include any important words may be too restrictive.

2. An evaluation of whether it is better to include prior summaries or the entire prior source text when selecting content for Document Clusters A and B needs to be completed.

3. The documents in each cluster were treated as one large source text, but it may be more valuable to generate update summaries of each document in the

cluster, and then generate a final cluster summary from the cluster's update summary.

4. A bug in FreqDistUpdate which for Document Cluster C which considered only Document Cluster A's information content, when it really should have considered information content from Document Clusters A and B, needs to be fixed. The result of this error is that the summary for Document Cluster C considered only the summary it had seen from the summary of Document Cluster A.

The 2007 DUC was our first chance to participate in a DUC event. We adapted the existing FreqDist text summarizer, which had good results in the biomedical domain, to the DUC Update Task. The FreqDistUpdate summarizer creates a summary of a source text which has approximately the same frequency distribution of terms as the source text. Several modifications were made to the base FreqDist summarizer, shown in Figure 23, for the DUC Update Task to account for a statement expressing an information need, as well as summary length limitations. While FreqDistUpdate did not do as well as hoped, some ideas to improve FreqDistUpdate performance in future DUC evaluations has been gathered.

6. CONCLUSION AND FUTURE WORK

In this chapter we conclude this dissertation work by highlighting our research contributions and by discussing future directions. We provide some suggestions about how our current work can improve other research such as personalization and multi-document summarization.

6.1 Summary of Contributions

The research presented in this dissertation is divided into two main areas: semantic annotation and text summarization. Biomedicine is used as the domain. In semantic annotation, the significance of the research is applying statistical language modeling from speech recognition field to biomedical semantic annotation in the natural language processing field, and in doing so knowledge was integrated from one disciplinary area to another. The practical result of the semantic annotation research is that the annotation time can be significantly decreased (by 3x-16x) with precision competitive with a state-of-the-art annotator. An improvement in the speed performance allows an online system to support texts unknown to the system ahead-of-time in order to support applications such as data mining, semantic indexing, semi-automatic annotation, and text summarization. The significance of the text summarization research is the use of domain-specific concepts in place of terms for biomedical text summarization.
The use of biomedical concepts overcomes biomedical language variation. Applications of text summarization using domain-specific summarization include personalized summaries, multiple document summarization, and question-answering systems.

The primary contributions of this research are as follows:

1. Design, implementation and evaluation of an ontology-based biomedical annotator. A phrase-unit concept annotator, called CONANN, that is more readily usable in online environments than existing systems was designed, implemented, and evaluated. CONANN is novel in several respects:

- The use of incremental filters to iteratively remove unlikely candidate phrases.
- The use of Inverse of Phrase Frequency to weight words based on their semantic importance.
- The use of language modeling for text-to-concept mapping.

The incremental filter approach removes unlikely candidate phrases in the earliest stage possible. The coverage filter uses semantically-focused words in a given ontology, called *Inverse Phrase Frequency*, to measure word membership. The coherence filter uses skip-bigrams, which allow gaps between words in common, to measure word order. To find the best-matching concept among candidate phrases which have passed the coverage and coherence filters, the phrase counting and language modeling final concept mappers are introduced, designed, and implemented. The phrase counting final mapper counts the number of candidate phrases which belong to a concept. The concept with the highest phrase count is then selected as the best match. The language model approach builds a language model for each UMLS concept using each concept's instances. Candidate phrases are then evaluated as to their probability of having been generated by the language of the concept. The language model approach is unique in that considers all of a concept’s instances together, whereas existing approaches in the literature consider each concept instance separately.

CONANN is evaluated using two different evaluation approaches. An intrinsic evaluation compares CONANN’s concept output to the concept output generated by a state-of-the-art concept annotator, MetaMap Transfer. The extrinsic evaluation measures the use of the generated concept annotations in a text-summarization task. The output of MetaMap and CONANN were used to generate biomedical text summaries, and the summaries were then evaluated using the ROUGE tool.

When compared to MetaMap, CONNAN was evaluated to be 2 to 4 times faster in initialization time, and from 3 to 14 times faster in average time to annotate a phrase. The highest precision as compared to MetaMap ranges from 90% to 95%. In the text summarization task, CONANN outperforms MetaMap from 1.5% to 6.7%.

The coverage filter was found to work very well alone, while the opposite is true of the coherence filter, leading to the conclusion that the ordering of words within UMLS concept instances is less important than the specific words included. It was also discovered that when applying both filters, the ordering of the filters is significant. The coverage filter tends to remove concept phrases where the coherence filter is effective. Therefore, the coherence filter needs to be applied before the coverage filter. Although the coherence filter is not effective alone, applying it along with the coverage filter improves the overall annotation precision score. In the coverage filter, the use of the involvement score outperformed the use of the naïve score using both binary and inverse phrase frequency values as weights. The use of inverse phrase frequency over the use of binary weights could not be conclusively shown, as each weight outperformed the other in different scoring approaches. The final mapping leads to a trade-off of precision versus speed. The phrase counting method is faster than the language model mapping, but results in lower precision than the language model mapping method. In addition, both phrase counting and language model mapping appear to be more effective at generating matching concepts in the top five candidate concepts. Future work will focus on finding the best exact match out of the top five or top ten concepts using approaches such as word sense disambiguation.

2. Design, implementation and evaluation of single-document text summarizers for biomedical text which uses domain concepts. Three biomedical text summarizers using two novel methods for identifying salient sentences, BioChain and FreqDist, as well as a third summarizer which combines the two methods, were designed, implemented and evaluated. The BioChain approach uses biomedical concepts identified within a source text and chains the concepts together based on each concepts semantic type. The FreqDist approach finds the frequency distribution of terms or concepts in the source text and then iteratively selects sentences from the original source text to form a summary which has a frequency distribution of terms or concepts as similar as possible to the source text's frequency distribution of terms or concepts. An advantage of FreqDist over BioChain is that FreqDist restricts redundancy of information to the same level as it is found in the source text. The two approaches were also merged into a third summarizer, ChainFreq, to take advantage of BioChain's strength in identifying important areas of a text, and FreqDist's strength in controlling information redundancy.

The hybrid ChainFreqSumm summarizer, based on a combination of the BioChain and FreqDist algorithms, was the best performing of all summarizers evaluated using ROUGE-SU4 scores. The hybrid summarizer showed that the use of concepts can outperform the use of terms for identifying salient sentences.

It was found that the use of domain-specific summarizers outperformed the use of nearly all general-purpose summarizers. The use of concepts was found to be competitive with and in some cases better than the use of terms for identifying text themes for single document summarization. The concept chaining approach is good at identifying important areas of biomedical text, but does not control redundancy in the generated summaries. The frequency distribution method restricts redundancy to the same level as it appears in the source text. New methods for novelty detection are likely to further improve redundancy detection.

3. Study of characteristics of biomedical texts, specifically concept distribution and summary size. The distribution of domain-specific concepts within a biomedical text corpus was studied, as was the ideal summary size of a biomedical text and the sections where human summarizers selected sentences.

The main observations of the concept distribution study are (a) the semantic types and concept frequency distribution between a source text and abstract are largely the same; and (b) not all concepts can be treated equally, and it may make sense to weight some concepts as more important than others.

The size of an ideal summary was evaluated using term and concept versions of the FreqDist and SumBasic summarizers. The summarizers generated summaries at compression ratios of 1%, 5%, 10%, 15%, and 20%. The model summaries for each text in the corpus were also set to the same compression ratios. The general trend across all summarizers is a continual improvement in ROUGE scores as the size of the summary increases. The ideal summary size for biomedical text is approximately 10% to 15% of the original source text, which is different than general news summarization, where the ideal size is approximately 20% of the original source text.

6.1.1 Answers to Research Questions

It was shown that the use of language modeling and phrase-counting techniques for performing biomedical concept mapping outperforms a state-of-the-art biomedical concept annotator based on word metrics by 2 to 4 times in initialization time, and by 3 to 14 times in average time to annotate a phrase.

It was also shown that the use of information retrieval techniques such as inverse phrase frequency to measure words in common between a source phrase and a UMLS candidate phrase, and language modeling of concept instances, can achieve 90% to 95% precision and better text summarization performance compared to a state-of-the-art biomedical concept annotator.

The use of concepts was shown to perform competitively with the use of terms for identifying salient source-text sentences. The use of concept chains in text summarization outperformed baseline summarizers and most general-purpose summarizers. The general-purpose summarizers that the concept-based summarizers scored below incorporated information redundancy removal techniques. The use of frequency distribution in text summarization was shown to outperform general-purpose summarizers as well as the baseline summarizers. Terms slightly outperformed the use of concepts. However, by combining the BioChain and FreqDist methods, the use of concepts was shown to outperform the use of terms in single-document biomedical text summarization.

6.2 Recommendations for Future Research

Our research can be extended in many directions. Several ideas for performing future research are:

Personalized Summaries: The use of concepts can be more useful than terms for generating personalized summaries. An envisioned system allows a user to select domain-specific concepts important to the user, and then have the summarizer generate a summary where those concepts are more highly weighted than the other concepts appearing in the source text. In our research, some work was performed towards this goal. One of the text summarizers for BioChainSumm used semantic type filtering. A list of important UMLS semantic types for oncology research was provided by a domain expert. In BioChain, only those semantic types listed by the domain expert were allowed to be strong chains. It was found that this approach reduced text summarization performance and the resulting summary was more appropriate to a highly-experienced domain expert. This was likely due to the fact that the model summaries were summaries of the entire text, and that the generated summaries were no longer generic, but personalized due to the filtering out of some semantic types. A key problem will be finding a method to evaluate personalized summaries. Current text summarization evaluation methods require the use of multiple model summaries to perform an automated evaluation.

Multiple-document Summarization: The text summarization work done in this dissertation research focuses on single-document text summarization. The use of concepts may be more beneficial for multiple-document summarization, where the language is expected to vary more across multiple texts than within a single text. The domain-specific concepts in UMLS are comprised of multiple concept instances. The concept instances are able to capture the variation in language. For example, a reference in one paper to *lung cancer* and a reference in a second paper to *pulmonary carcinoma* are likely discussing the same concept. A term-based approach will not capture the semantic similarity of the two phrases, but a concept-based approach will. A key problem is finding a corpus of biomedical texts and corresponding model summaries which are designed for evaluating multiple document biomedical text summarization.

FreqDist in DUC Update Task: The use of the frequency distribution algorithm presented in this research is promising for the new DUC Update Task. In the DUC Update Task, systems are asked to produce short summaries of newswire articles, assuming a user has read a set of previous, related article texts. The idea is to present new information that the user has not already read from the set of preceding article texts. A key part of the FreqDist algorithm is to incorporate sentences into a summary which have new information. Through participation in the DUC 2007 competition, several areas were identified where the submitted FreqDistUpdate system can be improved:

1. A list of important words derived from the topic statement is a heuristic which needs to be empirically evaluated as to whether or not its inclusion is helpful.

2. An evaluation needs to be done to determine if FreqDistUpdate identifies new information better by including prior summaries or prior source text.

3. The DUC Update Task requires generating a summary for a cluster of related documents. FreqDistUpdate may perform better by generating a summary based on the content of each of the related documents. In the submitted system, the documents in each cluster were treated as one large source text. However, it may be more valuable to first generate summaries of each document in the cluster, and then construct an update summary for the entire cluster from the individual document summaries.

LIST OF REFERENCES

Afantenos, S. D., Karkaletsis, V., & Stamatopoulos, P. (2005). Summarization from Medical Documents: A Survey. *Journal of Artificial Intelligence in Medicine, 33*(2), 157-177.

Aronson, A. R. (1996). *MetaMap: Mapping Text to the UMLS Metathesaurus.* Unpublished manuscript.

Aronson, A. R. (2001a). Effective Mapping of Biomedical Text to the UMLS Metathesaurus: The MetaMap Program. *Proceedings of the AMIA Symposium 2001,* 17-21.

Aronson, A. R. (2001b). *MetaMap Evaluation.* Unpublished manuscript.

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern Information Retrieval* (1st ed.). Harlow, England: Addison-Wesley.

Barzilay, R., & Elhadad, M. (1997). Using Lexical Chains for Text Summarization. *In Proceedings of the Intelligent Scalable Text Summarization Workshop (ISTS'97), ACL,* Madrid, Spain. 10-18.

Bayerl, P. S., Lüngen, H., Gut, U., & Paul, K. I. (2003). Methodology for Reliable Schema Development and Evaluation of Manual Annotations. *Proceedings of the Workshop on Knowledge Markup and Semantic Annotation at the Second International Conference on Knowledge Capture (K-CAP 2003),* Florida, USA.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American, 284*(5), 34-43.

Borkar, V., Deshmukhy, K., & Sarawagiz, S. (2001). Automatic Segmentation of Text into Structured Records. *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data,* Santa Barbara, California, USA. *, 30*(2) 175-186.

Brandow, R., Mitze, K., & Rau, L. F. (1995). Automatic condensation of electronic publications by sentence selection. *Information Processing and Management: an International Journal, 31*(5), 675-685.

Brin, S. (1998). Extracting Patterns and Relations from the World Wide Web. *Proceedings of the WebDB Workshop at 6th International Conference on Extending Database Technology,* Valencia, Spain. *, 1590* 172-183.

Brooks, A. D., & Sulimanoff, I. (2002). Evidence-Based Oncology Project. *Surgical Oncology Clinics of North America* (pp. 3-10)

Carbonell, J., & Goldstein, J. (1998). The use of MMR, diversity-based reranking for reordering documents and producing summaries. *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval,* Melbourne, Australia. 335-336.

Carpenter, R. L., & Baldwin, F. B. (2007). *LingPipe - A suite of Java libraries for the linguistic analysis of human language.* Retrieved January 26, 2007, from http://www.alias-i.com.

Chandrasekaran, B., Josephson, J. R., & Benjamins, V. R. (1999). What are ontologies and why do we need them? *IEEE Intelligent Systems, 14*(1), 20-26.

Chen, P., & Verma, R. (2006). A Query-Based Medical Information Summarization System Using Ontology Knowledge. *Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems,* Salt Lake City, Utah, USA. 37-42.

Cimiano, P., Handschuh, S., & Staab, S. (2004). Towards the Self-Annotating Web. *Thirteenth International Conference on World Wide Web,* New York, NY, USA. 462-471.

Ciravegna, F. (2001). Adaptive Information Extraction from Text by Rule Induction and Generalisation. *Proceedings of 17th International Joint Conference on Artificial Intelligence (IJCAI 2001),* Seattle, Washington, USA. 1251-1256.

Cohen, A. M., & Hersh, W. R. (2005). A survey of current work in biomedical text mining. *Briefings in Bioinformatics, 6*(1), 57-71.

Cunningham, H., Maynard, D. & Tablan, V. (2000). *JAPE: A Java Annotation Patterns Engine (Second Edition). Technical report CS--00--10, University of Sheffield, Department of Computer Science.*

D'Avanzo, E., Magnini, B., & Vallin, A. (2004). Keyphrase Extraction for Summarization Purposes: The LAKE System at DUC-2004. *Proceedings of the 2004 Document Understanding Conference,* Boston, USA.

Dalianis, H. (2000). *SweSum - A Text Summarizer for Swedish* No. TRITA-NA-P0015). Stockholm, Sweden: NADA, KTH.

Damianos, L., Day, D., Hirschman, L., Kozierok, R., Mardis, S., McEntee, T., et al. (2002). Real Users, Real Data,Real Problems: The MiTAP System for Monitoring Bio Events. *Proceedings of the Conference on Unified Science and Technology for Reducing Biological Threats and Countering Terrorism (BTR 2002),* Albuquerque, NM, USA. 166-177.

Demner-Fushman, D., & Lin, J. (2006). Answer extraction, semantic clustering, and extractive summarization for clinical question answering. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, 841-848.

Demner-Fushman, D., & Lin, J. (2007). Answering Clinical Questions with Knowledge-Based and Statistical Techniques. *Computational Linguistics, 33*(1), 63-103.

Demner-Fushman, D., & Lin, J. (2005). Knowledge Extraction for Clinical Question Answering: Preliminary Results. *Proceedings of the AAAI-05 Workshop on Question Answering in Restricted Domains,* Pittsburgh, Pennsylvania, USA.

Denny, J. C., Smithers, J. D., Miller, R. A., & Spickard, A. (2003). "Understanding" Medical School Curriculum Content Using KnowledgeMap. *Journal of the American Medical Informatics Association, 10*(4), 351-362.

Denny, J. C., Irani, P. R., Wehbe, F. H., Smithers, J. D., & Spickard, A.,3rd. (2003). The KnowledgeMap project: development of a concept-based medical school curriculum database. *Proceedings of the Annual AMIA Symposium*, 195-199.

Devita, G. (2006). *MMTx API Documentation for Release 2.3.* Unpublished manuscript.

Dice, L. R. (1945). Measures of the amount of ecologic association between species. *Ecology, 26*, 297-302.

Dill, S., Eiron, N., Gibson, D., Gruhl, D., Guha, R., Jhingran, A., et al. (2003). SemTag and Seeker: Bootstrapping the Semantic Web via Automated Semantic Annotation. *Twelfth International World Wide Web Conference,* Budapest, Hungary. 178-186.

Dingli, A., Ciravegna, F., & Wilks, Y. (2003). Automatic Semantic Annotation using Unsupervised Information Extraction and Integration. *Proceedings of the Workshop on Knowledge Markup and Semantic Annotation at the Second International Conference on Knowledge Capture (K-CAP 2003),* Florida, USA.

Divita, G., Tse, T., & Roth, L. (2004). Failure analysis of MetaMap Transfer (MMTx). *Medinfo, 11*(Pt 2), 763-767.

Doran, W., Stokes, N., Carthy, J., & Dunnion, J. (2004). Comparing Lexical Chain-based Summarisation Approaches Using an Extrinsic Evaluation. *Proceedings of the Global WordNet Conference (GWC 2004),* 112-117.

Doran, W. P., Stokes, N. S., Dunnion, J., & Carthy, J. (2004). Assessing the Impact of Lexical Chain Scoring Methods and Sentence Extraction Schemes on Summarization. *Proceedings of the 5th International conference on Intelligent Text Processing and Computational Linguistics CICLing-2004,* Seoul, South Korea. *, 2945* 627-635.

Doran, W., Stokes, N., Newman, E., Dunnion, J., Carthy, J., & Toolan, F. (2004). News Story Gisting at University College Dublin. *Proceedings of the Document Understanding Conference (DUC-2004).*

Edmundson, H. P. (1999). New Methods in Automatic Extracting. In I. Mani, & M. T. Maybury (Eds.), (pp. 23-42). Cambridge, MA: MIT Press.

Elhadad, N., Kan, M. Y., Klavans, J., & McKeown, K. (2005). Customization in a unified framework for summarizing medical literature. *Journal of Artificial Intelligence in Medicine, 33*(2), 179-198.

Elhadad, N., & McKeown, K. R. (2001). Towards generating patient specific summaries of medical articles. *Proceedings of the NAACL Workshop on Automatic Summarization,* Pittsburgh, PA. 31-39.

Elhadad, N. (2006). User-Sensitive Text Summarization: Application to the Medical Domain. (Ph.D., Columbia University).

Elhadad, N., McKeown, K., Kaufman, D., & Jordan, D. (2005). Facilitating physicians' access to information via tailored text summarization. *Proceedings of the Annual AMIA Symposium,* Washington, DC. 226-230.

Elkin, P., Cimino, J., Lowe, H., Aronow, D., Payne, T., Pincetl, P., et al. (1988). Mapping to MeSH(the art of trapping MeSH equivalence from within narrative text). 185-190.

Evans, D. K. (2005). Identifying similarity in text: Multi-lingual analysis for summarization. (Doctor of Philosophy, Columbia University, Graduate School of Arts and Sciences). *Columbia University, 1* (1), 1-168.

Fellbaum, C. (1998). *WORDNET: An Electronic Lexical Database*. Cambridge, MA: The MIT Press.

Frustaci, S., Gherlinzoni, F., De Paoli, A., Bonetti, M., Azzarelli, A., Comandone, A., et al. (2001). Adjuvant Chemotherapy for Adult Soft Tissue Sarcomas of the Extremities and Girdles: Results of the Italian Randomized Cooperative Trial. *Journal of Clinical Oncology, 19*(5), 1238-1247.

Galley, M., & McKeown, K. (2003). Improving Word Sense Disambiguation in Lexical Chaining. *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence,* Acapulco,Mexico. 1486-1488.

Goldstein, J., Kantrowitz, M., Mittal, V., & Carbonell, J. (1999). Summarizing text documents: sentence selection and evaluation metrics. *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval,* Berkeley, California, United States. 121-128.

Green, S. J. (1998). Automatically Generating Hypertext in Newspaper Articles by Computing Semantic Relatedness. *Proceedings of the Joint Conference on New Methods in Language Processing and Computational Natural Language Learning,* 101-110.

Handschuh, S., Staab, S., & Volz, R. (2003). On Deep Annotation. *International WWW Conference,* 431-438.

Handschuh, S., Staab, S., & Ciravegna, F. (2002). S-CREAM -- Semi-automatic CREAtion of Metadata. *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02),*

Harnly, A., Nenkova, A., Passonneau, R., & Rambow, O. (2005). Automation of Summary Evaluation by the Pyramid Method. *Proceedings of the Recent Advances in Natural Language Processing,* Borovets, Bulgaria.

Hearst, M. (1992). Automatic Acquisition of Hyponyms from Large Text Corpora. *Proceedings of the Fourteenth International Conference on Computational Linguistics,* Nantes, France. *, 2* 539-545.

Hersh, W. R., Mailhot, M., Arnott-Smith, C., & Lowe, H. J. (2001). Selective Automated Indexing of Findings and Diagnoses in Radiology Reports. *Journal of Biomedical Informatics, 34*(4), 262-273.

Hersh, W., Hickam, D. H., Haynes, R. B., & McKibbon, K. A. (1991). Evaluation of SAPHIRE: an automated approach to indexing and retrieving medical literature. *Proceedings / the ... Annual Symposium on Computer Application [sic] in Medical Care. Symposium on Computer Applications in Medical Care*, 808-812.

Hersh, W., & Leone, T. J. (1995). The SAPHIRE server: a new algorithm and implementation. *Proceedings of the Annual Symposium on Computer Applications in Medical Care. Symposium on Computer Applications in Medical Care*, 858-862.

Hersh, W. R., & Donohoe, L. C. (1998). SAPHIRE International: a tool for cross-language information retrieval. *Proceedings of the AMIA Annual Symposium.*, 673-677.

Hersh, W. R., & Greenes, R. A. (1990). SAPHIRE--an information retrieval system featuring concept matching, automatic indexing, probabilistic retrieval, and hierarchical relationships. *Computers and biomedical research, an international journal, 23*(5), 410-425.

Hirst, Graeme and St-Onge, David. (1998). Lexical chains as representations of context for the detection and correction of malapropisms. In C. Fellbaum (Ed.), *WordNet: An electronic lexical database* (). Cambridge, MA: MIT Press.

Hovy, E. H. (2005). Automated Text Summarization. In R. Mitkov (Ed.), *The Oxford Handbook of Computational Linguistics* (pp. 583-598). Oxford: Oxford University Press.

Hovy, E., & Lin, C. (1999). Automated Text Summarization in SUMMARIST. In I. Mani, & M. T. Maybury (Eds.), *Advances in Automatic Text Summarization* (pp. 81-94). Cambridge, MA: MIT Press.

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

Hovy, E., Lin, C., & Zhou, L. (2005). Evaluating DUC 2005 using Basic Elements. *Proceedings of Document Understanding Conference (DUC-2005),* Vancouver, B.C. Canada.

Jaques, D. P. (Ed.). (2002). *Surgical Oncology Clinics of North America: Prospective Randomized Clinical Trials in Oncology* (1st ed.). Philadelphia, PA USA: W.B. Saunders Company.

Johnson, B., & Shneiderman, B. (1991). Tree-Maps: a space-filling approach to the visualization of hierarchical information structures. *Proceedings of the 2nd conference on Visualization '91,* San Diego, CA, USA. 284-291.

Kazman, R., Al-Halimi, R., Hunt, W., & Mantei, M. (1996). Four Paradigms for Indexing Video Conferences. *IEEE Multimedia, 3*(1), 63-73.

Kiess, H. O. (2002). *Statistical Concepts for the Behavioral Sciences* (Third ed.). Boston, MA: Allyn and Bacon.

Kogut, P., & Holmes, W. (2001). AeroDAML: Applying Information Extraction to Generate DAML Annotations from Web Pages. *Proceedings of the Workshop on Knowledge Markup and Semantic Annotation at the First International Conference on Knowledge Capture (K-CAP 2001),* Victoria, BC.

Kosala, R., & Blockeel, H. (2000). Web Mining Research: A Survey. *SIGKDD Explorations, 2*(1), 1-15.

Kupiec, J., Pedersen, J., & Chen, F. (1999). A Trainable Document Summarizer. In I. Mani, & M. T. Maybury (Eds.), (pp. 55-60). Cambridge, MA: MIT Press.

Kushmerick, N., Weld, D. S., & Doorenbos, R. (1997). Wrapper Induction for Information Extraction. *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI '97),* Nagoya, Japan. 729-737.

Lacatusu, F., Hickl, A., Harabagiu, S., & Nezda, L. (2004). Lite-GISTexter at DUC 2004. *Proceedings of the 2004 Document Understanding Conference,*

Lavie, A., Sagae, K., & Jayaraman, S. (2004). The Significance of Recall in Automatic Metrics for MT Evaluation. *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas,* Washington, DC USA.

Lee, D. L., Chuang, H., & Seamons, K. (1997). Document ranking and the vector-space model. *Software, IEEE, 14*(2), 67-75.

Lee, L. (1999). Measures of Distributional Similarity. *Proceedings of the 37th annual meeting of the Association for Computational Linguistics,* 25-32.

Lin, C. Y., & Och, F. J. (2004a). Automatic Evaluation of Machine Translation Quality Using Longest Common Subsequence and Skip-Bigram Statistics. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics,* Barcelona, Spain. 605-612.

Lin, C. Y., & Och, F. J. (2004b). Orange: a Method for Evaluating Automatic Evaluation Metrics for Machine Translation. *Proceedings of the 20th International Conference on Computational Linguistics,* Geneva, Switzerland. 501-507.

Lin, C. (2004). Looking for a Few Good Metrics: Automatic Summarization Evaluation - How Many Samples Are Enough? *Proceedings of the NTCIR Workshop 4,* Tokyo, Japan.

Lin, C., & Hovy, E. H. (2003). Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. *Proceedings of 2003 Language Technology Conference (HLT-NAACL 2003),* Edmonton, Canada. , *1*(1) 71-78.

Lin, C. (2005). *Recall-Oriented Understudy for Gisting Evaluation (ROUGE).* Retrieved August 20, 2005, from http://www.isi.edu/~cyl/ROUGE/

Litkowski, K. C. (2004). Summarization Experiments in DUC 2004. *Proceedings of the 2004 Document Understanding Conference,* Boston, USA.

Luhn, H. P. (1958). The Automatic Creation of Literature Abstracts. *IBM Journal of Research and Development, 2*(2), 159-165.

Maedche, A., & Staab, S. (2001). Ontology Learning for the Semantic Web. *IEEE Intelligent Systems, 16*(2), 72-79.

Mani, I., & Bloedorn, E. (1998). Machine Learning of Generic and User-Focused Summarization. *Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence,* Madison, Wisconsin, United States. 820-826.

Mani, I., & Maybury, M. T. (1999). In Mani I., Maybury M. T. (Eds.), *Advances in Automatic Text Summarization*. Cambridge, MA: MIT Press.

Manning, C. D., Raghavan, P., & Schütze, H. (2007). *Introduction to Information Retrieval* (1st ed.). Cambridge, England: Cambridge University Press.

Manning, C. D., & Schutze, H. (1999). *Foundations of Statistical Natural Language Processing* (1st ed.). Cambridge, Massachusetts: The MIT Press.

Maynard, D. (2003). Multi-Source and Multilingual Information Extraction. *Expert Update.*

McKeown, K., Barzilay, R., Evan, D., Hatzivassiloglou, V., Klavans, J., Sable, C., et al. (2002). Tracking and Summarizing News on a Daily Basis with Columbia's Newsblaster. *Proceedings of HLT 2002: Human Language Technology Conference,* San Diego, CA, USA. 269-273.

McKeown, K. R., Chang, S. F., Cimino, J., Feiner, S., Friedman, C., Gravano, L., et al. (2001). PERSIVAL, a system for personalized search and summarization over multimedia healthcare information. 331-340.

McKeown, K., Elhadad, N., & Hatzivassiloglou, V. (2003). Leveraging a common representation for personalized search and summarization in a medical digital library. *Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries,* Houston, Texas, USA. 159-170.

Melli, G., Wang, Y., Liu, Y., Kashani, M., Shi, Z., Gu, B., et al. (2005). Description of Squash, the SFU question answering summary handler for the DUC-2005 summarization task. *Proceedings of the Document Understanding Conference (DUC-2005),* Vancouver, Canada. 103-110.

Microsoft Coporation. (2002). *Microsoft Word 2002*. Redmond, Washington, USA:

Missikoff, M., Navigli, R., & Velardi, P. (2002). The Usable Ontology: An Environment for Building and Assessing a Domain Ontology. *1st International Semantic Web Conference (ISWC2002),* 39-53.

Morris, J., & Hirst, G. (1991). Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text. *Computational Linguistics, 17*(1), 21-43.

Nadkarni, P., Chen, R., & Brandt, C. (2001). UMLS Concept Indexing for Production Databases. *Journal of the American Medical Informatics Association, 8,* 80-91.

Nadkarni, P. M. (1997). Concept locator: a client-server application for retrieval of UMLS metathesaurus concepts through complex boolean query. *Computers and biomedical research, an international journal, 30*(4), 323-336.

National Institute of Health. (2005). *Identification of Important Text in Full Text Articles Using Summarization.* Retrieved April 21, 2007, from http://ii.nlm.nih.gov/resources/Summarization_and_FullText.pdf

National Institute of Standards and Technology. (2005a). *DUC 2005 - Responsiveness Task.* Retrieved September 28, 2006, from http://duc.nist.gov/duc2005/responsiveness.assessment.instructions

National Institute of Standards and Technology. (2005b). *DUC 2005 Quality Questions.* Retrieved September 28, 2006, from http://duc.nist.gov/duc2005/quality-questions.txt

National Institute of Standards and Technology (NIST). (2005). *Document Understanding Conferences.* Retrieved August 20, 2005, from http://www-nlpir.nist.gov/projects/duc/

National Library of Medicine, United States. (2004). *PhraseX and the SPECIALIST Minimal Commitment Parser.* Retrieved September 3, 2006, from http://ind.nlm.nih.gov/MTI/phrasex.shtml

National Library of Medicine, United States. (2006a). *Medical Text Indexer.* Retrieved September 3, 2006, from http://ind.nlm.nih.gov/mti.shtml

National Library of Medicine, United States. (2006b). *Specialist Lexicon and Lexical Tools.* Retrieved August 31, 2006, from http://www.nlm.nih.gov/research/umls/meta4.html

Nenadic, G., Mima, H., Spasic, I., Ananiadou, S., & Tsujii, J. (2002). Terminology-driven literature mining and knowledge acquisition in biomedicine. *International Journal of Medical Informatics, 67*(1), 33-48.

Nenkova, A., & Passonneau, R. (2004). Evaluating content selection in summarization: The pyramid method. Paper presented at the *Proceedings of HLT/NAACL 2004,* Boston, MA.

Nenkova, A., & Vanderwende, L. (2005). *The Impact of Frequency on Summarization* No. MSR-TR-2005-101). Redmond, Washington: Microsoft Research.

Pollock, J. J., & Zamora, A. (1975). Automatic Abstracting Research at Chemical Abstracts Service. *Journal of Chemical Information and Computer Sciences, 15*(4), 226-232.

Popov, B., Kiryakov, A., Kirilov, A., Manov, D., Ognyanoff, D., & Goranov, M. (2003). KIM - Semantic Annotation Platform. *2nd International Semantic Web Conference (ISWC2003), , 2870* 834-849.

Pratt, W., & Yetisgen-Yildiz, M. (2003). A study of biomedical concept identification: MetaMap vs. people. *AMIA.Annu.Symp.Proc., ,* 529-533.

Radev, D., Allison, T., Blair-Goldensohn, S., Blitzer, J., Celebi, A., Drabek, E., et al. (2004). MEAD - a platform for multidocument multilingual text summarization. *Proceedings of Language Resources and Evaluation 2004 (LREC 2004),* Lisbon, Portugal.

Rath, G. J., Resnick, A., & Savage, R. (1961). The Formation of Abstracts by the Selection of Sentences.[Electronic version]. *American Documentation, 2*(12), 139-208.

Reeve, L., & Han, H. (2006). A Comparison of Semantic Annotation Systems for Text-based Web Documents. In D. Taniar, & J. W. Rahayu (Eds.), *Web Semantics and Ontology* (1st ed.,). Hershey, PA USA: Idea Group.

Reeve, L., Han, H., Nagori, S. V., Yang, J., Schwimmer, T., & Brooks, A. D. (2006). Concept Frequency Distribution in Biomedical Text Summarization. *Proceedings of the ACM Fifteenth Conference on Information and Knowledge Management (CIKM'06),* Arlington, VA, USA. 604-611.

Rose, P. G., Bundy, B. N., Watkins, E. B., Thigpen, J. T., Deppe, G., Maiman, M. A., et al. (1999). Concurrent Cisplatin-Based Radiotherapy and Chemotherapy for Locally Advanced Cervical Cancer. *New England Journal of Medicine, 340*(15), 1144-1153.

Rotem, N. (2003). *Open Text Summarizer (OTS).* Retrieved July 3, 2006, 2006, from http://libots.sourceforge.net

Schuemie, M. J., Weeber, M., Schijvenaars, B. J., van Mulligen, E. M., van der Eijk, C. C., Jelier, R., et al. (2004). Distribution of information in biomedical abstracts and full-text publications. *Bioinformatics (Oxford, England), 20*(16), 2597-2604.

Shi, Z., Melli, G., Wang, Y., Liu, Y., Gu, B., Kashani, M., et al. (2007). Question Answering Summarization of Multiple Biomedical Documents. *Proceedings of the 20th Canadian Conference on Aritificial Intelligence (CanAI '07),* Montreal, QC.

Silber, G. H., & McCoy, K. F. (2002). Efficiently Computed Lexical Chains as an Intermediate Representation for Automatic Text Summarization. *Computational Linguistics, 28*(4), 487-496.

Sparck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation, 28*(1), 11-21.

Sparck Jones, K. (1999). Automatic Summarizing: Factors and Directions. In I. Mani, & M. T. Maybury (Eds.), *Advances in Automatic Text Summarization* (pp. 2-12). Cambridge, MA: MIT Press.

Srinivasan, S., Rindflesch, T. C., Hole, W. T., Aronson, A. R., & Mork, J. G. (2002). Finding UMLS Metathesaurus concepts in MEDLINE. *Proceedings / AMIA ... Annual Symposium. AMIA Symposium*, 727-731.

Strzalkowski, T., Stein, G., & Wang, J. B, Wise. A Robust Practical Text Summarizer. 1999. *Advances in Automatic Text Summarization*, 137–154.

Subhash, S. (1996). *Applied Multivariate Techniques* (1st ed.). USA: John Wiley and Sons.

Svab Ondrej, Labsky Martin, Svatek Vojtech. (2004). RDF-Based Retrieval of Information Extracted from Web Product Catalogues. *Proceedings of the 27th Annual ACM SIGIR Conference on Research and Development in Information Retrieval, Semantic Web Workshop,* Sheffield, UK.

Tallis, M. (2003). Semantic Word Processing for Content Authors. *Second International Conference on Knowledge Capture,* Sanibel, Florida, USA.

Tersmette, K. W. F., Scott, A. F., Moore, G. W., Matheson, N. W., & Miller, R. E. (1988). Barrier word method for detecting molecular biology multiple word terms. 207-211.

Teufel, S., & Moens, M. (2002). Summarizing scientific articles: experiments with relevance and rhetorical status. *Computational Linguistics, 28*(4), 409-445.

Teufel, S., & Moens, M. (1999). Argumentative classificatin of extracted sentences as a first step towards flexible abstracting. In I. Mani, & M. T. Maybury (Eds.), (pp. 155-171). Cambridge, MA: MIT Press.

The Lemur Project. (2006). *Lemur Language Modeling Toolkit.* Retrieved July 3, 2006, 2006, from http://www.lemurproject.org/

United States National Library of Medicine. (2004). *UMLS Semantic Network Fact Sheet.* http://www.nlm.nih.gov/pubs/factsheets/umlssemn.html

United States National Library of Medicine. (2005a). *ClinicalTrials.gov.* http://www.clinicaltrials.gov/

United States National Library of Medicine. (2005b). *MetaMap Transfer.* http://mmtx.nlm.nih.gov/

United States National Library of Medicine. (2005c). *Unified Medical Language System (UMLS).* http://www.nlm.nih.gov/research/umls/

United States National Library of Medicine. (2006a). *PubMed.* http://www.ncbi.nlm.nih.gov/entrez/query.fcgi

United States National Library of Medicine. (2006b). *UMLS Metathesaurus Fact Sheet.* http://www.nlm.nih.gov/pubs/factsheets/umlsmeta.html

University of Sheffield. (2002). *Amilcare.* Retrieved December 28, 2004, from http://nlp.shef.ac.uk/amilcare/amilcare.html

Vanderwende, L., & Suzuki, H. (2005). Frequency-based Summarizer and a Language Modeling Extension. *Proceedings of the MultiLingual Summarization 2005 Workshop.*

Vargas-Vera, M., Motta, E., Domingue, J., Lanzoni, M., Stutt, A., & Ciravegna, F. (2002). MnM: Ontology Driven Semi-Automatic and Automatic Support for Semantic Markup. *The 13th International Conference on Knowledge Engineering and Management (EKAW 2002)*, 379-391.

Wang, R., Stokes, N., Doran, W., Newman, E., Dunnion, J., & Carthy, J. (2005). LexTrim: A Lexical Cohesion based Approach to Parse-and-Trim Style Headline Generation. *Proceedings of the 6th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2005),* Mexico City, Mexico. , *3406* 645-648.

Wollersheim, D., Rahayu, W., & Reeve, J. (2002). Evaluation of Index Term Discovery in Medical Reference Text. Paper presented at the *Proceedings of the International Conference on Information Technology and Applications,* Bathhurst, NSW, Australia.

Yesilada, Y., Harper, S., Goble, C., & Stevens, R. (2004). Ontology Based Semantic Annotation for Visually Impaired Web Travellers. *Proceedings of the 4th International Conference on Web Engineering (ICWE 2004),* Munich, Germany. *3140* 445-458.

Zieman, Y. L., & Bleich, H. L. (1997). Conceptual mapping of user's queries to medical subject headings. *Proceedings of the conference of the American Medical Informatics Association, Annual Fall Symposium*, 519-522.

Zipf, G. (1949). *Human Behavior and the Principle of Least Effort*. Cambridge, MA: Addison-Wesley.

Zou, Q., Chu, W. W., Morioka, C., Leazer, G. H., & Kangarloo, H. (2003). IndexFinder: A Method of Extracting Key Concepts from Clinical Texts for Indexing. *Proceedings of the AMIA Annual Symposium,* 763-767.

APPENDIX A – Example Summarization Output

Paper Title:
Concurrent Cisplatin-Based Radiotherapy and Chemotherapy for Locally Advanced Cervical Cancer (Rose et al., 1999)

Paper Abstract (Rose et al., 1999):
1) Purpose: Cisplatin, mitolactol (dibromodulcitol), and ifosfomide have been the most active single agents in squamous carcinoma of the cervix identified so far by the Gynecologic Oncology Group (GOG).

2) Combinations of cisplatin plus ifosfamide and cisplatin plus mitolactol are prospectively compared with cisplatin alone.

3) Patients and Methods: Patients were randomized to receive cisplatin 50 mg/m2 or the same dose of cisplatin plus mitolactol (C + M) 180 mg/m 2 orally on days 2 to 6, or cisplatin plus ifosfamide (CIFX)
5 g/m 2 given as a 24-hour infusion plus mesna 6 g/m 2 during and for 12 hours after the ifosfamide infusion, every 3 weeks for up to six courses.

4) Of 454 patients entered, 438 were eligible and analyzed for response and survival.

5) Results: CIFX had a higher response rate (31.1% v 17.8%, P = .004) and longer progression-free survival (PFS) time (P = .003) compared with cisplatin alone.

6) The median times to progression or death were 4.6 and 3.2 months, respectively.

7) C + M showed no significant improvement in these parameters compared with cisplatin alone.

8) Survival was associated with initial performance score (PS; 0 was more favorable; P < .001) and with age (younger was unfavorable, P = .025).

9) There was no significant difference in overall survival between cisplatin and either of the combinations.

10) Leukopenia, renal toxicity, peripheral neurotoxicity, and CNS toxicity were more frequent with CIFX (P < .05).

11) Conclusion: CIFX improved the response rate and PFS duration in advanced cervix cancer compared with cisplatin alone, but at the cost of greater toxicity and with no improvement in survival.

Domain Expert's Summary (Jaques, 2002):

1) There was a significantly greater frequency of response among patients treated with CIFX compared to cisplatin alone (31.1% vs 17.8%) .

2) Progression free survival was also significantly longer for the CIFX group when compared to cisplatin alone .

3) There was no significant difference in survival between cisplatin and either of the other combination regimens .

4) Toxicity was significantly worse for the two combination arms when compared to cisplatin alone .

FreqDistSumm Summary using 5% of original source text:

1) Cisplatin is said to be the most active anticancer drug in cervical cancer, although 100 mg/m did not significantly improve the complete response (CR) rate compared prospectively with 50 mg/m (12.7 v 10%, respectively) and there was no appreciable difference in response duration, progression-free interval (PFI), or survival.

2) Thus, there is no convincing reason to use a cisplatindose higher than 50 mg/m .

3) However, shortly after activation, the study was amended so that the serum creatinine concentration was to be within the institution's normal limits and the serum albumin level was to be >= 3 g/dL .

4) Ineligible patients included those with cervical neoplasms other than squamous cell carcinoma or with nonmeasurable cervical cancer, WBC count less than 4,000/uL and/or platelet count less than 100,000/uL, abnormal liver function (bilirubin, AST, or alkaline phosphatase level > two times normal not related to the cancer), bilateral hydronephrosis, GOG PS 3 or 4, past or concomitant malignancy other than skin (excluding melanoma), prior therapy with cytotoxic drugs except when used as a radiation sensitizer, radiationtherapy within 3 weeks of entry, lesions measurable only by ultrasound, or pregnancy or lactation .

5) Patients were prospectively stratified according to whether they had received prior radiation-sensitizer treatment (hydroxyurea, cisplatin, or fluorouracil) and by PS, and were then centrally randomized with equal probability to receive (1) cisplatin 50 mg/m with appropriate hydration every 3 weeks for a maximum of six courses, or (2) cisplatin 50 mg/m on day 1 plus mitolactol (180 mg/m orally for 5 days) on days 2 through 6 every 3 weeks, or (3) cisplatin 50mg/m plus ifosfamide 5.0 g/m over 24 hours plus mesna 6 g/m given concurrently with ifosfamide and for 12 hours after, every 3 weeks, again for a maximum of six courses .

6) Between June 1990 and January 1994, 454 women entered the study of whom 16 were ineligible (wrong stage, n = 2, wrong cell type, n = 9, wrong primary tumor, n = 2, prior chemotherapy, n = 2, second primary tumor, n = 1), which left 438 eligible patients (including 10 who received no drug, all were included in the intent to-treat analysis) .

7) The results of this trial in cervix cancer reflect a common set of findings in combination chemotherapy of advanced solid tumors, namely, a higher response rate (but not a high CR rate) with a combination compared with single-agent therapy at the cost of more toxicity and no survival benefit .

APPENDIX B – Example Annotation Output

Sample source phrases from the evaluation corpus and their UMLS concept mappings generated by CONANN.

Source Phrase	UMLS Concept Identifier	UMLS Concept Name
pelvic lymphadenectomy	C0193883	Pelvic lymphadenectomy (procedure)
advances	C0205179	Advanced phase
only eligible patients	C0030705	Patients
staging	C0027646	Diagnostic Neoplasm Staging
body	C0242821	Human Body
First	C0205435	First (qualifier value)
poor prognosis	C0278252	Prognosis bad (finding)
radiosensitivity	C0034537	Radiation Tolerance
indeed poor prognostic features	C0220901	prognostic
pulmonary disease	C0024115	Lung diseases
complete blood counts	C0009555	Blood Count, Complete
10th percentile	C1264641	Percentile (property) (qualifier value)
pulmonary toxicity	C0600688	Toxic effect
underlies	C0444455	Underlay (qualifier value)
namely cmf	C0950521	CMF
ib	C0022104	Irritable Bowel Syndrome
pelvic examination	C0200045	Manual pelvic examination (procedure)|
ct scan	C0441633	Scanning
stepwise logistic regression	C0206031	Logistic Regression

VITA

Education

2007 Drexel University

Ph.D., Information Science and Technology

- Thesis: Semantic Annotation and Summarization of Biomedical Text
- Advisor: Dr. Hyoil Han, Drexel University

2001 Widener University

M.E., Computer and Software Engineering

1990 Rutgers University

B.A., Computer Science

Publications

Refereed Conference Papers

- Lawrence Reeve and Hyoil Han (2007). *CONANN: An Online Biomedical Concept Annotator*. Proceedings of the 2007 Data Integration in the Life Sciences conference (DILS'07), Philadelphia, PA USA.
- Lawrence Reeve, Hyoil Han, Saya V. Nagori, Jonathan C. Yang, Tamara A. Schwimmer, and Ari D. Brooks (2006). *Concept Frequency Distribution in Biomedical Text Summarization*. Proceedings of the 15th Conference on Information and Knowledge Management. (15% acceptance)
- Lawrence Reeve, Hyoil Han, and Ari D. Brooks (2006). *BioChain: Using Lexical Chaining Methods for Biomedical Text Summarization*. Proceedings of the 21st Annual ACM Symposium on Applied Computing, Bioinformatics track. (32% acceptance)
- Lawrence Reeve and Hyoil Han (2005). *Survey of Semantic Annotation Platforms*. Proceedings of the 20th Annual ACM Symposium on Applied Computing, Web Technologies and Applications track. (37% acceptance)
- Lawrence Reeve (2004). *Adapting the TileBar Interface for Visualizing Resource Usage*. Proceedings of the 30th International Conference for the Resource Management and Performance Evaluation of Enterprise Computing Systems.

Refereed Journal Papers

- Lawrence Reeve, Hyoil Han and Ari D. Brooks (2007). *Biomedical Text Summarization Using Concept Chains*. International Journal of Data Mining and Bioinformatics.
- Lawrence Reeve, Hyoil Han and Ari D. Brooks (2007). *The Use of Domain-Specific Concepts in Biomedical Text Summarization*. Journal of Information Processing and Management, Special Issue on Summarization.

Refereed Book Chapters

- Lawrence Reeve and Hyoil Han (2006). *A Comparison of Semantic Annotation Systems for Text-based Web Documents*. Web Semantics and Ontology, David Taniar and J. Wenny Rahayu (Eds.), Idea Group Publishing.
- Lawrence Reeve, Hyoil Han, and Chaomei Chen (2005). *Information Visualization and the Semantic Web*. Visualizing the Semantic Web, Vladimir Geroimenko and Chaomei Chen (Eds.), Springer.