Online Biomedical Concept Annotation Using Incremental Filtering and Language Modeling

1st Author
1st author's affiliation
1st line of address
2nd line of address
Telephone number, incl. country code
1st author's email address

2nd Author
2nd author's affiliation
1st line of address
2nd line of address
Telephone number, incl. country code
2nd E-mail

3rd Author
3rd author's affiliation
1st line of address
2nd line of address
Telephone number, incl. country code
3rd E-mail

ABSTRACT
In this paper, we describe an online biomedical concept annotator, CONANN, which takes a source phrase and finds the best-matching domain-specific concept. Domain concepts are defined in resources such as the United States National Library of Medicine's Unified Medical Language System Metathesaurus. We show that techniques from the information retrieval field, specifically inverse document frequency and language modeling, can be applied to filtering and mapping domain-specific concepts. An advantage of such an approach is to improve annotation speed, facilitating the use of concept annotation in online environments. Our main contributions are 1) the design of a phrase-unit concept annotator which is more readily usable in online environments than existing systems, 2) an incremental filter design to remove unlikely concept matches, and 3) the use of a language model for each domain-specific concept. An intrinsic evaluation comparing CONANN's concept output to a state-of-the-art concept annotator shows our system has an annotation precision of 89% and an average phrase annotation time which is four times faster. In addition, an extrinsic evaluation using the generated concepts in a text summarization task shows no significant degradation when using CONANN.

Categories and Subject Descriptors
H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing – *indexing methods, thesauruses.*

General Terms
Algorithms, Measurement, Performance, Design, Economics, Experimentation.

Keywords
Biomedical semantic annotation, biomedical concept mapping, concept annotation

1. INTRODUCTION
The biomedicine community maintains large and continuously-updated information sources. For example, United States National Library of Medicine's PubMed service contains in excess of 16 million citations from over 5,000 worldwide biomedicine-related journals (United States National Library of Medicine, 2006). The PubMed service consists of citations and abstracts in addition to linking with full-texts. For physicians and biomedical researchers, finding and using relevant texts within these large resources can be challenging. To address this challenge, annotation systems using domain-specific concepts, rather than terms, have been developed. Examples of such systems include MetaMap Transfer (MetaMap)[1], SAPHIRE[2], and KnowledgeMap[3]. These systems annotate text with biomedical concepts defined in resources such as the Unified Medical Language System (UMLS) [4] and National Cancer Institute (NCI) Thesaurus [5]. Among the benefits of using concepts, rather than terms, is 1) synonym merging, where synonymous phrases are merged to a single concept, and 2) the use of a domain-specific language for querying. Biomedical concept annotations have been used in applications for indexing and retrieval, data mining, decision support, patient records, medical curriculum searching, and text summarization [1] [3] [6].

The task of a concept annotator is to map each text unit (typically a phrase) of a source text into one or more domain-specific concepts. In some systems, such as MetaMap [1], efforts are made to find a best-matching concept, while in other systems, such as IndexFinder [7], 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 where the meaning of small text units (e.g. phrases), is needed.

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 [8]. 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.

In this paper, we describe our biomedical concept annotator, CONANN, which supports both dynamic and static concept annotation. The current concept resource is UMLS, but can support other concept resources as well. Its design is intended to achieve faster annotation time per phrase while maintaining annotation accuracy competitive with existing biomedical annotation systems. Such an online biomedical annotation concept system has the advantages of supporting texts unknown to the system ahead of time, as well as providing for constantly changing concept resources, which is common in a field such as biomedicine. These advantages overcome the limitations of purely static biomedical concept annotators, which form the majority of such systems today.

Our main contributions are 1) development of an online as opposed to offline concept annotator, 2) the use of an incremental filtering approach to remove unlikely concepts from a candidate pool, and 3) the use of a language model to perform final concept mapping using the filter output. The use of a language model for each concept is particularly distinctive, as previous systems consider each concept's synonymous phrases as independent, while we combine all phrases for a particular concept into a single model.

This paper is organized as follows. Section 2 provides background on the biomedical resource we use for concept annotation of texts and also describes the general concept mapping process. Section 3 discusses the architecture of our concept annotator system, as well as the algorithms for two important parts of the mapping process. Section 4 describes previous work in this area. Section 5 describes our evaluation methods, and Section 6 discusses the evaluation results. Section 7 concludes.

2. BACKGROUND

2.1 Biomedical Concept Resource

Finding meaning in biomedical documents which can be processed by machines is accomplished by first creating one or more ontologies, and then linking information within each document to specifications contained in each ontology using a markup language [9]. Ontologies are conceptualizations of a domain that typically are represented using domain vocabulary [10]. Automated semantic annotation is the process of mapping data instances to an ontology [11], [12]. The resulting annotations from the semantic annotation processing are what provide the link between information stored within a document and the ontology [9]. 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) [4]. UMLS contains many sub-components, but only the Metathesaurus is used in this work.

The UMLS Metathesaurus contains concepts and real-world instances of the concepts, including a concept name and its synonyms, lexical variants, and translations [13]. 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 associated with the concept. A *concept instance* is a phrase belonging to a UMLS concept. Each UMLS concept is associated with one or more synonymous concept instances. Concept instances are derived from biomedical vocabulary sources. The key idea is that a single UMLS concept may have multiple ways of being expressed (instances). The UMLS Metathesaurus organizes concept instances into concepts. A *concept name* is the name given to a particular UMLS concept.

Table 1: A UMLS concept and its concept instances

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

2.2 Text-To-Concept Mapping Process

The task of a biomedical concept annotator is to map small text units in the source text to concept instances which in turn, determine the concept name the text unit should have. For example, assuming our ontology consists of only the single concept and the five concept instances shown in Table 1, then in the sentence "The patient has multiple myeloma," the phrase "multiple myeloma" is mapped to the concept Multiple Myeloma. In reality, however, such simple mappings are rare.

Most biomedical concept annotators use the following generalized method to map a source text unit to a domain-specific concept:

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 by (a) removing possessives, (b) replacing punctuation with spaces, (c) removing stop words, (d) convert words to lower-case, (e) breaking a string into constituent words, and (f) sorting words into alphabetical order [14];

3. For each word in the input phrase, build a set of all concepts based on their concepts instances 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.

Each system also determines the types of matches between a source text unit and the concept resource it will support. Four types of matches between the terms of a source text and the terms of concept instances have been defined in the literature [15]: 1) *None*: there is no match of terms; 2) *Simple*: there is an exact match between the terms; 3) *Partial*: one or more terms do not match exactly; and 4) *Complex*: two or more sets of terms map to distinct concept instances. Not all systems support all types of

matches. In addition, some systems focus on finding all possible matches, while other systems find the best possible match.

3. Concept Annotator - CONANN

In this section, we discuss the general design of our CONANN concept annotator and describe how the UMLS domain resource is pre-processed for use within CONANN. In addition, we detail the algorithms for the coverage filter and the final concept mapping.

3.1 Architectural Overview

CONANN finds the best matching concept for a source phrase and uses an incremental approach, as shown in Figure 1.

There are several phrase types used by CONANN. A *source phrase* is a phrase from the source text which the system will attempt to annotate with a biomedical concept. A *concept instance* is phrase belonging to a UMLS concept (each UMLS concept is associated with one or more synonymous phrases as shown in table 1). *Candidate phrases* are concept instances 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.

A list of candidate phrases is generated based on the overlap of words from the source phrase and all concept instances. If only a single candidate phrase exists, its associated concept name is returned. If there is more than one candidate phrase generated, filtering process to remove unlikely candidate phrases begins. The candidate phrase filters are based on n-gram co-occurrences between the source phrase and the candidate phrases. The incremental filtering is done to improve computational efficiency by applying successively more computationally complex filters. 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 [16], [1]. The idea is to successively filter out candidate concepts using basic techniques, and compute more complex candidate phrase scores for a small subset of possible candidate phrase matches.

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

1. *Candidate Phrase Generation*: Construct a list of candidate phrases based on the common words between all concept instances and the source phrase. If only one candidate phrase remains, return its associated concept name.

2. *Incremental Filtering*: Filter the list of candidate phrases using one or more filters (see Section 3.2.2). After each filter is applied, if only one candidate phrase remains, its associated concept name is returned. In this work, we apply only a single filter which measures word coverage between a source phrase and a concept instance. In future work, additional filters will be applied.

3. *Final Mapping*: If more than one candidate phrase remains after all filters have been applied, the candidate phrases are passed to a final stage to perform concept mapping. The final concept mapping is responsible for finding the best matching candidate phrase among the remaining candidate phrases. Many possible approaches to final concept mapping are possible. In this paper, we describe a language model approach to final concept mapping (see Section 3.4). The final concept mapping returns the best-matching concept name (or concept names in cases of scoring ties).

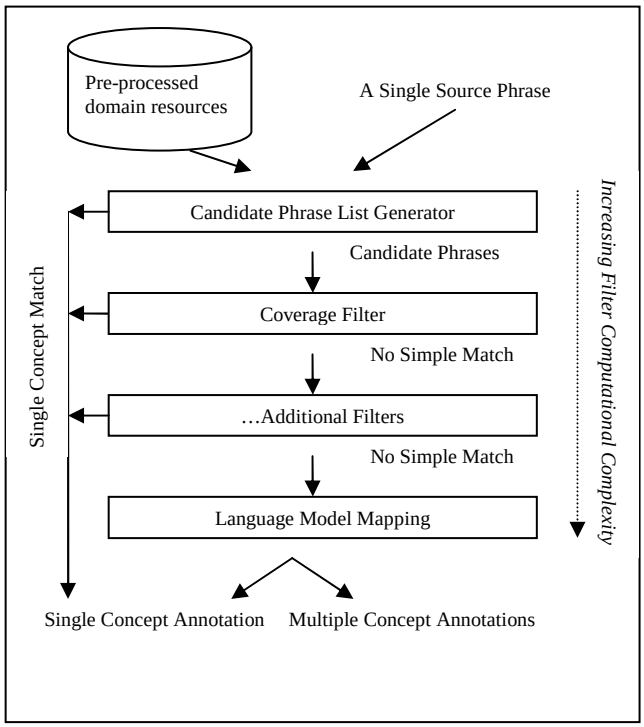

Figure 1: Architecture of CONANN

3.2 Domain Resource Pre-processing

Domain resource processing is done once with each new domain resource and then stored externally. Domain resource processing consists of converting UMLS text-based resources into a format usable for fast in-memory access (Section 3.2.1). In addition, calculations which can be performed ahead of time, such as inverse phrase frequency and concept language models (described in Sections 3.2.2 and 3.2.3, respectively) are completed.

3.2.1 Table Generation for Fast Lookup

CONANN uses a set of pre-processed hash tables to allow fast in-memory lookup of words and candidate phrases. The use of such in-memory tables has been shown to dramatically increase the response time of concept annotators [7]. Nine hash tables are generated for rapid in-memory lookup, divided into 5 categories: concepts, phrases, words, variants, and language model. The hash tables are built based on the UMLS text-based files, specifically the ones included in the latest MetaMap release, version 2.4.b [17]. These tables are loaded from external storage at CONANN startup as part of CONANN initialization, and remain in main memory until the annotator is shutdown.

3.2.2 Inverse Phrase Frequency Calculation

As part of pre-processing, each word in the UMLS is given a weight based on its usage in all concept instances within UMLS.

In information retrieval, inverse document frequency value (IDF) uses the frequency of a word across all documents as a way to identify semantically-focused words [18]. Semantically-focused words do not frequently occur across all documents within a collection, and thus are more likely to have more discrimination. To apply the ideas of TF*IDF [19] to CONANN, each UMLS concept instance is substituted for document, resulting in a weight called Inverse Phrase Frequency (IPF), shown in Figure 2. Each unique UMLS word is assigned its semantic importance based on its inverse phrase frequency value. More semantically important words will be given a higher weight than semantically unimportant words. The idea is to give some indication of the importance of a word in UMLS based on its usage within all UMLS concept instances. Term frequency, typically combined with IDF in information retrieval, is not considered here since it is highly likely the frequency for each word within a phrase will be one, due to the short length of phrases.

$$Inverse\ \ Phrase\ \ Frequency\ = \log \frac{N}{n_i}$$

Figure 2: Inverse Phrase Frequency. *N* is the number of UMLS phrases, *i* is a UMLS word, and n_i is the number of phrases word *i* appears in.

3.2.3 Concept Language Models

A unigram language model is built for each of the concepts in UMLS. Each UMLS concept contains one or more synonymous phrases, called concept instances, as previously shown in Table 1. The language model contains the word identifier, frequency of the word in the concept, and the probability of the word occurring across all concept instances within the concept, that is, $P(w) = \frac{|w|}{N}$, where *w* is a word in a particular concept's concept instances, |*w*| is a count of the number of times the word appears in all of the concept instances of a concept, and *N* is the total number of words in all of the particular concept's concept instances. Since there are 797,152 concepts defined in the UMLS resource used, 797,152 unigram language models were generated.

3.3 Candidate Phrase List Generation

The first step in annotating a source phrase is to find a list of all possible candidate phrases in UMLS. This pool of phrases represents possible matches. Finding candidate phrases is done by first removing all words from the source phrase which do not appear in UMLS. Stop words in the PubMed stop list are also removed [20]. The words in the source phrase are mapped to their UMLS base form, which removes inflection. For example, 'cancers' is mapped to 'cancer.' This is done to eliminate word variation, and to allow exact matching of concept instance 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 instances 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* and *cancer*, which will return candidate phrases such as {*lung, chronic obstructive lung disease, lung cancer, liver cancer*} and so forth. It is not required that a candidate phrase have all words in common, since exact mappings between a source phrase and concept instances are expected to be rare. In addition to finding all candidate phrases having one or more of the base words, the same process is repeated for all word variants of the base word. For example, *pulmonary* is a variant of *lung*, so phrases such as *pulmonary carcinoma* will be added to the list of candidate phrases.

3.4 Coverage Filter

Coverage measures the overlap of common words between a source phrase and a candidate phrase. In contrast to existing systems such as MetaMap [1], SAPHIRE [16], and IndexFinder [7], which consider the count of common words between a source phrase and a candidate phrase, the scoring of coverage in CONANN considers the contribution of each word in the source phrase, as measured by each word's IPF value. Each candidate phrase is given a score determine by the sum of its words' IPF values, called the PhraseCoverageIPF score, as shown in Figure 3.

$$PhraseCoverageIPF = \sum_{i=1}^{N} IPF_i$$

Figure 3: Phrase Coverage IPF. *N* is the number of words in a phrase. *IPF* is the inverse phrase frequency value of $word_i$.

Once the PhraseCoverageIPF values are computed for all candidate phrases, the standard deviation of the PhraseCoverageIPF values for the set of candidate phrases is calculated. A threshold value is chosen as the mean plus one standard deviation. All candidate phrases whose PhraseCoverageIPF values is greater than or equal to the threshold value are passed to the final concept mapper. 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 phrases have a PhraseCoverageIPF value greater than or equal to the threshold, the candidate phrases with the highest PhraseCoverageIPF value are passed to the final concept mapper (if there is only one such candidate phrase with a high PhraseCoverageIPF, its candidate concept is returned).

3.5 Language Model Concept Mapping

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 coverage filter (Section 3.4). Since each candidate phrase belongs to one or more concepts, a list of concepts is generated from the candidate instances to form a set of candidate concepts. Each candidate concept is then assigned a score based on its probability of generating the source phrase. The probability score is calculated as shown in Figure 4, which is a standard unigram language mixture model [21] which combines the source word (*w*) probability within the concept language model ($M_{concept}$- see Section 3.2.3) with the word probability of the entire UMLS phrase collection ($M_{conceptCollection}$). $M_{conceptCollection}$ is the set of IPF values calculated as part of pre-processing (see Section 3.2.2.) We initially set λ=0.5 to balance the concept language model with the collection model. We also evaluated the extreme values of λ=0.1 and λ=0.9, but did not notice any change in the final concept annotation output. To allow for more word variation, we also expand the source phrase words to include all source phrase word

variants of each source phrase word. CONANN uses the source phrase word variants that are provided by the UMLS resources. The highest-scoring candidate concepts are then output as the best-matching concept for the source phrase. In the case of ties, all of the highest-scoring concepts are output.

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

Figure 4: Multinomial Unigram Language Mixture Model

4. PREVIOUS WORK

Most work in semantic annotation for biomedical text is performed to support semantic indexing/retrieval and data mining of biomedical texts [1]. Our work is most closely related to MetaMap[1], KnowledgeMap[3], and SAPHIRE[2]. We focus on scoring candidate phrases, since that is one of the primary differences between systems, SAPHIRE uses simple and partial mapping, and for candidate phrase scoring combines measures of term overlap, term proximity, and length of term matches. KnowledgeMap uses simple and partial matching, and for candidate phrase scoring uses an exact match approach and if no matches are found, performs iterative variant-word-generation and re-matching. KnowledgeMap also offers a disambiguation stage which uses concept co-occurrence information derived from existing medical texts to find a best-matching concept. MetaMap uses simple, partial and complex mapping. MetaMap scores candidate phrases using a mixture of four different scores: a) *Centrality* where a source phrase head term used in concept instance; b) *Variation* how far a source phrase term variant is from concept instance term; c) *Coverage* which measures the overlap between source phrase and concept instance terms, ignoring gaps; and d) *Coherence* which finds term sequence overlaps between source phrase and concept instance. Compared to SAPHIRE, our CONANN uses simple and partial matching, but does not score every candidate phrase for final mapping. Like KnowledgeMap and MetaMap, we incorporate word variants of the source phrase, but we do not incorporate disambiguation or exact matching as KnowledgeMap does or extensive word variants generation as MetaMap does. Our system reduces computational complexity by deferring complex scoring until after most candidate phrases have been eliminated. In addition, we build a language model of each concept's phrases, whereas existing systems consider each candidate phrase as independent of one another, even from the same concept.

Other related systems include SENSE [22], which translates source and concept instance to low-level semantic factors, then performs exact matching of the semantic factors; Concept Locator [23] which simply sub-divides a phrase & looks for exact matches; PhraseX[24] which focuses on phrase identification and performs an exact match with candidate phrases; and IndexFinder [7] which treats the source text as a bag of words and finds all matching words, regardless of their location.

None of the related concept annotators implement a language model approach. CONANN uses a unigram language model, which is widely used in information retrieval research [21] [25]. Other language model applications, such as speech recognition, use more than one n-gram to capture the order of the text [26]. The unigram model is more advantageous in concept annotation applications because it allows for gaps in word order when matching a source phrase to a concept instance, where exact matches usually do not occur. In addition, the unigram language model has proven an effective approach, outperforming the vector space model [25].

5. EVALUATION

Evaluation of the annotation system is done using both an intrinsic and an extrinsic method. The intrinsic evaluation is intended to evaluate the speed and accuracy of CONANN against an existing biomedical concept annotator. The extrinsic evaluation is designed to measure the effect of annotation output on a task. We chose text summarization using concepts as the task.

5.1 Intrinsic Evaluation

The intrinsic evaluation is intended to evaluate the speed and accuracy of CONANN against an existing biomedical concept annotator. We use the MetaMap system [1] provided by the United States National Library of Medicine as the baseline system.

The MetaMap application [17] maps biomedical text to concepts stored in the Metathesaurus. 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.

The corpus of noun phrases was generated from a citation database of approximately 1,200 oncology clinical trial papers physicians feel are important to the field [27]. Of the 1,200 papers cited, 24 were randomly selected based on the minimum requirements of the ROUGE summary evaluation tool [28]. The PDF versions of these 24 papers were then converted to plain-text format and manually processed to remove the abstract, graphics, tables, figures, captions, citation references, and the bibliography section. The resulting text was processed by MetaMap to find all noun phrases and their corresponding concept annotations 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. There were 1,628 phrases with a single MetaMap concept annotation. This set of phrases was used to perform the evaluation.

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. CONANN was then executed against the same set of 1,628 phrases and its annotation time measured. CONANN also produced concept annotations for the list of phrases. These mappings were then compared to MetaMap, producing the annotation precision metric shown in Figure 5. Three back-to-back 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. As shown in Figure 5, the Annotation Precision measure uses the same idea as in information retrieval, but adapted to fit concept mapping [29]. Annotation Precision is defined as the fraction of mapped concepts which are correct. 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, which is calculated by taking the total annotation time devided by the total number of phrases annotated, as shown in Figure 6.

$$Annotation\ \Pr ecision = \frac{\#\ of\ Correct\ Concepts}{Total\ \#\ of\ Concepts\ Mapped}$$

Figure 5: Annotation precision metric

$$AveragePhraseTime = \frac{Total\ AnnotationTime}{Total\ \#\ of\ Concepts\ Mapped}$$

Figure 6: Average phrase time metric

5.2 Extrinsic 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. We selected text summarization as the end-user task. We used two probabilistic summarizers, FreqDist [30] and a version of SumBasic [31] modified to use concepts rather than terms. Both summarizers only use concept frequency as the sole feature to select salient sentences. Both summarizers' performance is entirely reliant on the frequency of concepts identified in the texts. It is expected if the concept output is accurate, summarization performance will improve because the concepts will have identified important areas within a text. Conversely, if the concept identification is not accurate, text summarization performance will degrade.

The corpus of 24 texts (see Section 5.1) is annotated using both CONAN 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 annotators is evaluated against manual summaries generated from domain experts using the ROUGE tool (Recall-Oriented Understudy for Gisting Evaluation) [32]. 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. ROUGE uses n-gram co-occurrence to determine the overlap between a summary and the models. An n-gram can be considered as one or more consecutive words. The ROUGE parameters from the DUC 2005 conference [33] 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 4 words. ROUGE-2 and ROUGE-SU4 are also the measures used by DUC 2005. The ROUGE scores indicate the n-gram overlap between the source text and the model summaries.

SumBasic [31] is frequency-based summarization algorithm. It operates by first finding a probability distribution of all concepts within a source text. It then iteratively finds the highest-scoring sentence by summing the probabilities of all concepts within a sentence. After the highest-scoring sentence is selected, the probability of each concept in the selected sentence is reduced so that weaker concepts have a chance to be incorporated into a summary. The FreqDist algorithm [30] builds a frequency distribution model of the text's concepts. It then selects sentences which, when added to the summary constructed so far, best match the frequency distribution of the source text. The goal is to get the frequency distribution of the summary and the source text to match as closely as possible.

6. RESULTS

The intrinsic evaluation compares the speed and accuracy of our CONANN versus the MetaMap baseline system. The extrinsic evaluation compares the use of the generated concept annotations on a text summarization task.

6.1 Intrinsic Evaluation Results

The intrinsic evaluation resulted in 1,454 concept annotation matches from CONANN with the 1,628 total single-concept annotations produced by MetaMap (see Section 5.1). This is a precision of 0.89. Several measurements of time were also performed, using three runs of each system. The first measurement is annotator initialization time, which is the time to load domain-specific resources into memory and prepare for annotation. Figure 7 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 32 to 33 seconds. Both systems exhibit stable initialization behavior.

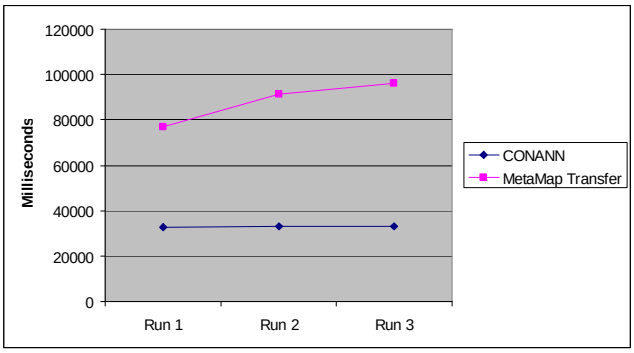

Figure 7: Annotator initialization time

Figure 8 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 ran at 1.3 minutes on all three run. CONANN and MetaMap are stable across all three runs. Figure 9 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 average 48 milliseconds per phrase across all three runs.

In summary, CONANN is faster both in initialization time and annotation time. Initialization time is 2.5 to three times faster, while average annotation time per phrase is more than four times faster. The trade-off for the faster performance is lower precision than 100%, which was measured at 89%. Section 6.2 presents an evaluation to determine the impact of lower precision on a task which uses the concept output.

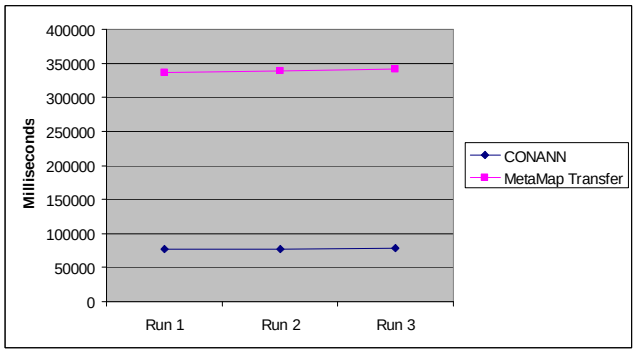

Figure 8: Total annotation time

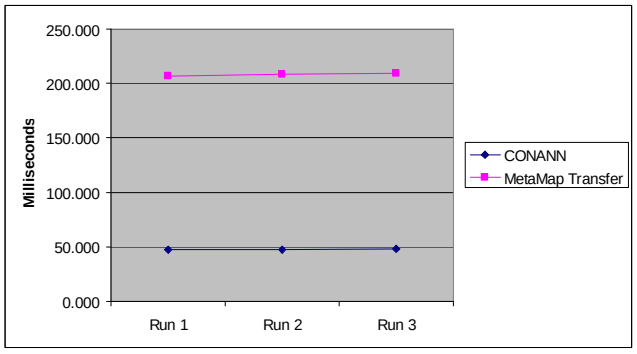

Figure 9: Average phrase annotation time

6.2 Extrinsic Evaluation Results

Table 2 shows the ROUGE-2 and ROUGE-SU4 scores for evaluating text summarization performance using the CONANN and MetaMap annotator output. For the ROUGE-2 metric, MetaMap slightly outperforms CONANN using both the FreqDist (one percent difference) and SumBasic (less than one percent difference) summarizers. FreqDist using MetaMap has a one percent advantage over FreqDist using CONANN. SumBasic using MetaMap has a less than one percent advantage over SumBasic with CONANN. Using the ROUGE-SU4 metric, MetaMap again slightly outperforms CONANN. In this case, the FreqDist summarizer using MetaMap has a 1.77 percent advantage over FreqDist using CONANN, while SumBasic using MetaMap has a 1.5 percent advantage over using SumBasic with CONANN. These differences make sense when considering the CONANN annotation precision was measured at 0.89 when compared to MetaMap. There is a small advantage to using MetaMap in a text summarization task, but this advantage may be is likely eliminated in an online environment, where CONANN has a time advantage such as average more than four times faster of annotation time for a single source phrase (see Section 6.1).

Table 2: Text summarization task performance

ROUGE-2 Scores	
FreqDist using MetaMap	0.1207
FreqDist using CONANN	0.1192
SumBasic using MetaMap	0.1094
SumBasic using CONANN	0.1085
ROUGE-SU4 Scores	
FreqDist using MetaMap	0.2200
FreqDist using CONANN	0.2161
SumBasic using MetaMap	0.2003
SumBasic using CONANN	0.1973

7. CONCLUSION

We presented an online biomedical concept annotator, CONANN, which takes a source phrase, identifies potential matching concepts and phrases in a domain-specific thesaurus, uses an incremental filter approach to remove candidate phrases using a variation of inverse document frequency, and maps the source phrase to best-matching concepts based on the probability of the language model of each candidate concept generating the source phrase.

An intrinsic evaluation was performed to compare CONANN's concept output to MetaMap's output. In addition, an extrinsic evaluation was performed to measure the usefulness of the concept output of each annotator. CONANN initialization time is 2.5 to three times faster and average annotation time per phrase is four times faster than a state-of-the-art concept. The speed advantage is at some cost in accuracy, as the precision compared to MetaMap is 89%. However, this loss of accuracy did not significantly impact the use of the CONANN's output in a text summarization task.

We have shown that average annotation time per phrase can be significantly reduced over existing annotation systems for applications which require annotation at the phrase level. We have also shown that the use of language models is effective at mapping source phrases to biomedical concepts. Future work includes integrating synonymous word variants into the concept language models, finding methods to reduce the size of the initial candidate list, and incorporating concept disambiguation. Our

eventual goal is to provide a biomedical concept annotator operating at the phrase level which has high accuracy compared to existing systems, and which can operate in an online environment. Such a system would be useful for ad-hoc physician and biomedical research tasks such as summarizing texts and semantic search.

8. REFERENCES

[1] A. R. Aronson, "Effective mapping of biomedical text to the UMLS metathesaurus: The MetaMap program," in *Proceedings of the AMIA Symposium 2001*, 2001, pp. 17-21.

[2] W. R. Hersh and R. A. Greenes, "SAPHIRE--an information retrieval system featuring concept matching, automatic indexing, probabilistic retrieval, and hierarchical relationships," *Comput. Biomed. Res.*, vol. 23, pp. 410-425, Oct. 1990.

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

[4] United States National Library of Medicine, "Unified Medical Language System (UMLS)," 2005.

[5] Center for Bioinformatics, United States National Cancer Institute. (2006, National cancer institute thesaurus.

[6] L. Reeve, H. Han and A. D. Brooks, "BioChain: Using lexical chaining methods for biomedical text summarization," in *Proceedings of the 21st Annual ACM Symposium on Applied Computing, Bioinformatics Track*, 2006, pp. 180-184.

[7] Q. Zou, W. W. Chu, C. Morioka, G. H. Leazer and H. Kangarloo, "IndexFinder: A method of extracting key concepts from clinical texts for indexing," in *Proceedings of the AMIA Annual Symposium*, 2003, pp. 763-767.

[8] A. Dingli, F. Ciravegna and Y. Wilks, "Automatic semantic annotation using unsupervised information extraction and integration," in *Proceedings of the Workshop on Knowledge Markup and Semantic Annotation at the Second International Conference on Knowledge Capture (K-CAP 2003)*, 2003,

[9] T. Berners-Lee, J. Hendler and O. Lassila, "The Semantic Web," *Sci. Am.*, vol. 284, pp. 34-43, May 2001. 2001.

[10] B. Chandrasekaran, J. R. Josephson and V. R. Benjamins, "What are ontologies and why do we need them?" *IEEE Intelligent Systems*, vol. 14, pp. 20-26, 1999.

[11] S. Handschuh, S. Staab and R. Volz, "On deep annotation," in *International WWW Conference*, 2003, pp. 431-438.

[12] L. Reeve and H. Han, "A comparison of semantic annotation systems for text-based web documents," in *Web Semantics and Ontology* ,1st ed., vol. 1, D. Taniar and J. W. Rahayu, Eds. Hershey, PA USA: Idea Group, 2006,

[13] United States National Library of Medicine. (2006, 28 March 2006). UMLS metathesaurus fact sheet.

[14] National Library of Medicine, United States. (2006, Specialist lexicon and lexical tools. *2006(August 31)*, pp. 1.

[15] A. R. Aronson, "MetaMap: Mapping Text to the UMLS Metathesaurus," 1996.

[16] W. Hersh and T. J. Leone, "The SAPHIRE server: a new algorithm and implementation," *Proc. Annu. Symp. Comput. Appl. Med. Care.*, pp. 858-862, 1995.

[17] United States National Library of Medicine, "MetaMap Transfer," 2005.

[18] C. D. Manning and H. Schutze, *Foundations of Statistical Natural Language Processing*. ,1st ed.Cambridge, Massachusetts: The MIT Press, 1999, pp. 620.

[19] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, pp. 11-21, 1972.

[20] United States National Library of Medicine. (2006, December 14, 2006). PubMed stop word list. *2007(1/18)*,

[21] C. D. Manning, P. Raghavan and H. Schütze, *Introduction to Information Retrieval*. ,1st ed.Cambridge, England: Cambridge University Press, 2007,

[22] Y. L. Zieman and H. L. Bleich, "Conceptual mapping of user's queries to medical subject headings," *Proc. AMIA. Annu. Fall. Symp.*, pp. 519-522, 1997.

[23] P. M. Nadkarni, "Concept locator: a client-server application for retrieval of UMLS metathesaurus concepts through complex boolean query," *Comput. Biomed. Res.*, vol. 30, pp. 323-336, Aug. 1997.

[24] S. Srinivasan, T. C. Rindflesch, W. T. Hole, A. R. Aronson and J. G. Mork, "Finding UMLS Metathesaurus concepts in MEDLINE," *Proc. AMIA. Symp.*, pp. 727-731, 2002.

[25] X. Liu and W. B. Croft, "Statistical language modeling for information retrieval," in *Annual Review of Information Science and Technology* , vol. 39, B. Cronin, Ed. American Society for Information Science and Technology, 2005, pp. 1-31.

[26] R. Rosenfeld, "Two decades of statistical language modeling: where do we go from here?" *Proceeding of the IEEE*, vol. 88, pp. 1270-1278, 2000.

[27] A. D. Brooks and I. Sulimanoff, "Evidence-based oncology project," in *Surgical Oncology Clinics of North America* , vol. 11, Anonymous 2002, pp. 3-10.

[28] C. Lin, "Looking for a few good metrics: Automatic summarization evaluation - how many samples are enough?" in *Proceedings of the NTCIR Workshop 4*, 2004,

[29] W. R. Hersh, M. Mailhot, C. Arnott-Smith and H. J. Lowe, "Selective Automated Indexing of Findings and Diagnoses in Radiology Reports," *J. Biomed. Inform.*, vol. 34, pp. 262-273, 2001.

[30] L. Reeve, H. Han, S. V. Nagori, J. Yang, T. Schwimmer and A. D. Brooks, "Concept frequency distribution in biomedical text summarization," in *Proceedings of the ACM Fifteenth Conference on Information and Knowledge Management (CIKM'06)*, 2006,

[31] A. Nenkova and L. Vanderwende, "The impact of frequency on summarization," Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005-101, 2005.

[32] C. Lin and E. H. Hovy, "Automatic evaluation of summaries using N-gram co-occurrence statistics," in *Proceedings of 2003 Language Technology Conference (HLT-NAACL 2003)*, 2003, pp. 71-78.

[33] National Institute of Standards and Technology (NIST), "Document Undertanding Conferences," vol. 2005, July 5, 2005. 2005.