Soft Pattern Matching Models for Definitional Question Answering

HANG CUI
and
MIN-YEN KAN
and
TAT-SENG CHUA
National University of Singapore

We explore probabilistic lexico-syntactic pattern matching, also known as soft pattern matching, in a definitional question answering system. Most current systems use regular expression based hard matching patterns to identify definition sentences. Such rigid surface matching often fares poorly when faced with language variations. We propose two soft matching models to address this problem: one based on bigrams and the other on Profile Hidden Markov Model (PHMM). Both models provide a theoretically sound method to model pattern matching as a probabilistic process that generates token sequences. We demonstrate the effectiveness of the models on definition sentence retrieval for definitional question answering. We show that both models significantly outperform the state-of-the-art manually constructed hard matching patterns on recent TREC data.

A critical difference between the two models is that the PHMM has a more complex topology. We experimentally show that the PHMM can handle language variations more effectively but requires more training data to converge.

While we evaluate soft pattern models only on definitional question answering, we believe that both models are generic and can be extended to other areas where lexico-syntactic pattern matching can be applied.

Categories and Subject Descriptors: H.3.3 [**Information Search and Retrieval**]: Retrieval models; I.2.7 [**Natural Language Processing**]: Text analysis

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: Soft patterns, definitional question answering

1. INTRODUCTION

Definition questions, such as questions like "What is TB?" or "Who is Aaron Copland?", account for a significant number of queries submitted to Web search engines [Voorhees 2001].

The most straightforward way to answer such questions is to look up the appropriate definition in a dictionary or an encyclopedia. Many current search engines,

Authors' address: Department of Computer Science, School of Computing, National University of Singapore, Singapore, 117543.

such as Google and Yahoo, adopt such an approach to defining terms, relying on existing online definitional resources. However, the public's interest is often focused on recent development of events and people. New terms, organizations and personalities, such as *Enron*, *Clay Aiken* and *SARS*, which are of great interest to the public, are often first described in news. Descriptions for these newly-minted terms of interest are not found in authoritative sources such as dictionaries or encyclopedias, but their descriptions can often be found in breaking news web sites. Moreover, definitions of people and organizations are often changing over time, thus even with a suitably large pool of human editors, it is impossible for manually-edited resources to keep all definitions up-to-date.

Therefore, automatic definitional question answering (QA) systems have been developed. They complement the use of such manually-edited definitions as they accumulate and produce definitions for these types of terms that are not covered by standard authoritative resources. In response to such questions, a typical definitional QA system extracts definition sentences that contain descriptive information about the search term from multiple documents and summarizes these sentences into definitions.

Traditional IR systems are only part of the solution: given a topic, search engines can only retrieve relevant documents, but do not distill these documents down to a single, coherent definition. The synthesis of a complete definition of such new terms requires the identification and collation of definition sentences (as opposed to whole documents or web pages) across multiple relevant articles.

The construction of complete and fluent definitions for terms incorporates many technologies. In this paper, we focus on a core component of this task: namely, the identification of definition sentences from relevant documents for a specified search target. A definition sentence contains descriptive information that can be included in an *extended definition* of the term. Such an extended definition answers not only "what/who is X?", but also "what/who is X like?" [Lannon 1991], i.e., the extended definition aggregates all relevant information about the target.

Definitional QA systems are based on either sentence retrieval or linguistic construct extraction (e.g., BBN [Xu et al. 2003] and Columbia [Blair-Goldensohn et al. 2004] systems). In this study, we focus on sentence retrieval for answering definition questions due to two reasons: First, the diversity of definition sentences makes definition sentence retrieval a difficult problem. Sentences are longer units than linguistic constructs and thus provide better coverage of relevant information and context which aids comprehension. Generally, recall is considered more important than precision for a corpus-based definitional QA system. Second, we believe that a QA system based on sentence retrieval and summarization provides an adaptable testbed for experimenting with various techniques. Other modules, such as the one to find fine grained definition units, can be easily integrated into this system.

Definition questions are about event and non-event targets. Definitions of events such as "The return of Hong Kong to Chinese sovereignty" require information to be extracted and ordered chronologically. Suitable techniques have been developed in the event-based summarization community [Mani et al. 2004], and thus we deal with definitions of non-event targets, such as people, organizations and other terms.

Most approaches applicable to definition sentence retrieval use pattern matching. Many systems create patterns manually: Harabagiu et al. [2003] employed

38 manually-constructed definition patterns for plain text news articles; Liu et al. [2003] mined topic-specific definitions using hand-crafted rules from web pages. However, we find that the manually constructed lexico-syntactic rules are too rigid to cover the notion of extended definitions, as such sentences exhibit diversity in patterns [Cui et al. 2004], and rigid rules often fail to match definition sentences due to inserted or deleted tokens. Such mismatches are common in natural language texts because authors often use diverse expressions to convey the same meaning. A similar problem occurs in other applications that make use of lexico-syntactic pattern matching, such as information extraction.

One promising technique to address this is soft pattern (SP) matching. In our previous work, we have shown that soft patterns are able to significantly outperform hard patterns in extracting definition sentences as they model language variations probabilistically [Cui et al. 2004; 2005]. Different from regular expression based hard-matching patterns employed in existing systems, we treat definition patterns as sequences of lexical and syntactic tokens. Therefore, pattern matching can be considered as probabilistic generation of test sequences based on training sequences. In this article, we develop a definitional QA system around soft pattern matching and formalize two soft pattern matching models. The first model is derived from the bigram language model with linear interpolation of unigram and bigram probabilities. The second model is based on the Profile Hidden Markov Model (PHMM). While the language model and the PHMM have been studied in other areas, their use in modeling lexico-syntactic pattern matching is a novel contribution of our work. Note that we constrain our focus on soft matching models for lexico-syntactic patterns on plain text. While the corpus we use for evaluation is constructed from online news, all HTML tags and hyperlinking information for the documents have been removed. Tackling patterns in other structures, such as parsing trees and non-linear structures of Web pages involving hyperlinks, is beyond the scope of this study.

In summary, the main contributions of this paper are:

—We propose two soft pattern models that significantly improve the performance of the baseline definitional QA system that uses hard patterns in F_3 scores on both TREC-13 and -14 test sets. Such improvement indicates that definition sentences exhibit diversified language patterns that are not captured well by hard matching of manually-constructed rules. While we only demonstrate soft pattern models on definitional QA, both soft pattern models can be extended to other applications that employ lexico-syntactic pattern matching.

—We validate the hypothesis drawn in our previous study [Cui et al. 2005] that given more training data, the PHMM outperforms the simple bigram model. In this study, we complete the verification of the hypothesis experimentally by employing more labeled definition sentences from previous years' TREC data for training the soft pattern models. We further analyze the results produced by the PHMM and find that only a small proportion of definition sentences retrieved by the PHMM but missed by the bigram model actually benefit from editing operations in the PHMM. We conjecture that noisy data sets will benefit more from the PHMM.

We first review the background of definition extraction in Section 2. In Section 3, we present the architecture of our definitional QA systems and discuss the specific implementation details. In Section 4, we discuss our core work on the two formal soft matching models. We then show empirical evidence that bear out the effectiveness of the soft pattern approach on TREC datasets. We discuss the results and limitations of our work, and examine future directions for soft pattern matching.

2. BACKGROUND

In this section, we review related work on automatic definition extraction. We categorize the definition extraction systems into two groups – domain-specific definition extraction systems and open-domain definition extraction (or definitional QA) systems. We classify our own soft pattern based system in the latter category.

2.1 Domain-Specific Definition Extraction

There has been much work aiming to extract definitions for terms from structured or unstructured text. Identifying expansion form for abbreviations or acronyms is perhaps the simplest form of definition extraction. Schwartz and Hearst [2003] presented an algorithm that searches expanded forms for acronyms in biomedical text. The algorithm searches for the form "`short form` (`long form`)" or "`long form` (`short form`)" and examines whether each letter in the short term comes from each word in the long form. Such definitions for abbreviations are relatively simple to identify, and thus it is sufficient to apply only string processing techniques. Zahariev [2003] introduced dynamic programming in matching definitions to handle more complicated acronyms, which may have multiple letters from a single word in the expansion form.

DEFINDER [Klavans and Muresan 2001] is part of a digital library project and aims to provide readable definitions of medical terms to patients. While developed for a specific domain, the two primary techniques it employs are largely domain-independent: (1) shallow text pattern analysis – patterns such as "`is called`" and "`is the term used to describe`," and (2) syntactic analysis for recognizing more complex structures like appositive and apposition. In their evaluation, Klavans and Muresan [2001] showed that online medical dictionaries have lower coverage compared to the results automatically extracted by DEFINDER. Their results show that automatic definition extraction systems complement manually-constructed dictionaries. We believe that the coverage of standard authoritative sources is lower in the open-domain context as new terms are coined frequently. Therefore, developing automatic systems for definition extraction is indispensable.

As both the accuracy of manually-constructed definitions and the coverage of automatically-extracted definitions are positive qualities, researchers often combine both types of resources. For instance, in [Muresan et al. 2003], glossaries identified from existing web sites and definitions extracted from unstructured text by DEFINDER are integrated to determine conceptual connections between different term databases.

Schiffman et al. [2001] produced biographical summaries (i.e., to answer "`who is`" questions). They combined a data-driven statistical method with machine learned rules to generate definitions. The biographical information is identified

```
Qid 1933: Who is Vlad the Impaler?
1933 1 okay 16th century warrior prince
1933 2 vital Inspiration for Bram Stoker 1897 novel "Dracula"
1933 3 okay Buried in medieval monastery on islet in Lake Snagov
1933 4 vital Impaled opponents on stakes when they crossed him
1933 5 okay Lived in Transylvania (Romania)
1933 6 okay Fought Turks
1933 7 okay Called "Prince of Darkness"
1933 8 okay Possibly related to British royalty
```

Fig. 1. A Sample Definition Question and Answer Nuggets from TREC

by appositives and special predicates led by verbs that are associated with typical actions of people.

More recently, the ubiquity of the Web has generated interest on finding definitions. Liu et al. [2003] proposed mining topic specific definitions from the Web. The basic idea is to utilize a set of hand-crafted rules to find definition sentences on web pages. They also tried to utilize the structure of web pages to identify subtopics of each main topic, which could be considered part of the extended definition of the main topic.

The above systems automatically extract definitions from plain text or Web pages, but are domain-specific. Our aim is to present a comprehensive definition extraction system that works on news articles and for a wide spectrum of terms. Many such systems have been showcased at TREC's questions answering competition, which we review next.

2.2 TREC Definitional Question Answering

The Text REtrieval Conference (TREC), a yearly research competition sponsored by the US government, has had a separate competition track on definitional question answering since 2003. Entrants' systems are evaluated over a corpus of over one million news articles from various news agencies. The definitional QA task requires the participating systems to extract and return interesting information about a particular person or term, such as "`Who is Vlad the Impaler?`" or "`What is a prion?`" The evaluation of definition questions is based on a manual check of how many answer nuggets (determined by the human assessor) are covered by system responses and allows partial credit for incomplete answers [Voorhees 2003a]. Figure 1 illustrates an example question from TREC and its corresponding answer nuggets.

TREC assesses definitional QA systems with respect to content precision and recall and does not attempt to judge definitions with respect to fluency or coherence. This is in line with the focus on our work in retrieving relevant definition sentences but differs from the general task of definition generation in which such stylistic criteria matter. As seen in Figure 1, content nuggets are categorized into *vital* pieces of information and *okay* ones that would be desirable to include in such extended definitions.

In early TREC, definition questions are mixed with factoid questions and are answered by a phrase as short definition. As such, the systems, such as the FALCON

[Harabagiu et al. 2000] and IBM system [Prager et al. 2001], employ simple manually constructed patterns to extract proper phrases or hypernyms from WordNet to define the search target.

However, extended definitions are thought to be more useful to users as they incorporate more description and context of the target term, which may better facilitate comprehension. Recent definitional QA systems have applied sophisticated analyses to retrieve such descriptive sentences. Table I summarizes the techniques employed by some representative TREC systems that perform well in the official evaluations. An exhaustive listing of techniques on a per system basis is presented in Table VII of the appendix.

From the table, it is clear that definitional QA systems mainly rely on two types of information to identify definitions: definitional linguistic constructs and statistical ranking. Let us examine these two components in more detail.

Definitional linguistic constructs. All systems try to identify specific definitional linguistic constructs that mark definition sentences. Examples of such definitional linguistic constructs include appositives and copulas. Appositives, such as "`Gunter Blobel, a cellular and molecular biologist, ...`," are mostly used in news to introduce a person or a new term. To recognize such linguistic constructs, the systems employ pre-compiled patterns, either on surface text (e.g., BBN, MIT and LCC) or on syntactic parsing trees (e.g., Amsterdam, Columbia and Korea University). According to component evaluations [Xu et al. 2004; Cui et al. 2004], definition pattern matching is the most important component in a definitional QA system. Definition patterns can also be defined based on specific question patterns and entity classes [Harabagiu et al. 2005]. Since surface patterns are more adaptable and easier to deploy without the requirement of task-specific parsing, we discuss only surface textual patterns that are represented in lexical/syntactic tokens. We list some definition patterns in Table VIII in the Appendix.

It is worth noting that the patterns employed by current definitional QA systems are equivalent to those that have been used by information extraction (IE) systems in the past. Traditional IE systems rely on a set of textual rules which are generalized from training examples. Muslea [1999] surveyed three kinds of pattern learning algorithms that perform extraction pattern generalization on free text and online documents, as well as automatic wrapper induction. These algorithms generalize the context around the target of interest, in terms of syntax and semantics, and abstract such contextual constraints as rules. A pattern is matched if each surrounding word of the candidate extraction matches its corresponding constraint in the pattern. Therefore, we deem the pattern matching using generalized rules as *hard matching*, as it requires an exact, slot-by-slot match. A pattern is not matched if any slot is not matched. Virtually all definitional QA systems that employ manual patterns (e.g., [Harabagiu et al. 2003; Hildebrandt et al. 2004]) or automatic rule induction algorithms(e.g., [Peng et al. 2005; Cui et al. 2004]) are thus hard pattern matching systems, as their patterns perform slot-by-slot matching.

We identify two drawbacks of using such generalized pattern rules for extracting definitions:

(1) **Inflexibility in Matching**: As stated, hard matching rules fail to match when there are even small variations between the training instances and the

Table I. Summary of Techniques Employed by TREC Systems

TREC Systems	**Definitional Linguistic Constructs**					**Statistical Ranking**	
	Surface Patterns[a]	Patterns on Parsing Trees[b]	Appositives and Copulas[c]	Relative Clauses[d]	Predicates and Verb Phrases[e]	Centroid Vector or Profile[f]	Mining External Definitions[g]
NUS[h] [Yang et al. 2003]	×		×	×		×	
BBN [Xu et al. 2003; Xu et al. 2004]	×	×	×	×	×	×	×
Columbia [Blair-Goldensohn et al. 2003]		×	×		×	×	
LCC [Harabagiu et al. 2003]	×		×				
MIT [Katz et al. 2004]	×		×	×	×	×	×
IBM PIQUANT [Chu-Carroll et al. 2004; Prager et al. 2003]	×		×	×		×	×
Amsterdam [Ahn et al. 2004]		×	×			×	×
Sheffield [Gaizauskas et al. 2004]	×		×	×		×	×
Korea University [Han et al. 2004]		×	×	×	×	×	×

[a] Lexico-syntactic surface patterns, such as "`<TARGET> , the $NNP`".
[b] Pattern rules for extracting specified constructs from syntactic parsing trees for sentences.
[c] Appositives – e.g., "*Gunter Blobel, a cellular and molecular biologist, ...*".
Copulas – e.g., "*Stem cell is a cell from which other types of cells can develop.*"
[d] e.g., "*... Gunter Blobel who won the Nobel Prize for ...*"
[e] Predicates and verb phrases are mainly for describing a person or special relations. They are identified by a set of specialized verbs, which are often coupled with people's behaviors, such as "born" and "vote".
[f] To construct a centroid vector or profile for each target and then use it to rank the relevance of candidate sentences or constructs. The centroid vector contains a set of highly relevant words to the target, which could be selected by frequent words in external definitions/biographies or extracted candidate sentences or constructs.
[g] To use other definitions obtained from definitional web sites, such as online biographies and encyclopedias. The weights of the relevant words to the target are augmented if they also appear in external definitions.
[h] This system refers to our system used in TREC-12, before we proposed the soft pattern models.

test text, such as extra or missing tokens. Such variations in natural language text are common in extended definition sentences and are a hallmark of fluent, well-crafted articles. Similar problems occur in information extraction, but are usually less serious as IE tends to extract domain-specific and task-specific information.

(2) **Inconsistent Weighting of Patterns**: Most systems use statistical metrics to rank the importance of retrieved constructs, but treat each definitional pattern with the same level of importance. However, different definition patterns should be weighted differently. For instance, appositives are the most popular syntactic pattern for definitions, and thus should be weighted heavily. Many systems lack a consistent method to determine the importance of the various definition patterns. The fertility of each pattern can then be utilized when ranking extracted definition candidates.

To circumvent the above problems, we proposed an alternate pattern generation and matching technique, *soft pattern matching*, for definition sentence identification [Cui et al. 2004; 2005]. Unlike current definition patterns, soft pattern matching learns a holistic definition pattern from all training instances and assigns weight to a pattern instance according to the distribution of tokens in each slot in the training data. More importantly, it does not treat pattern matching simply as a binary decision, but allows partial matching by calculating a generative degree-of-match probability between the test instance and the set of training instances.

The definition of soft patterns encompasses several existing approaches to information extraction. Several graphical models for IE can thus be viewed as soft pattern matching in this framework, but they perform more than surface pattern matching. Skounakis et al. [2003] applied hierarchical HMMs to the task of extracting binary relations in biomedical texts. They constructed two types of HMMs to represent words and phrases, which are two levels of emission units. Earlier work by McCallum et al. [2000] demonstrates the application of Maximum Entropy Markov Model (MEMM) to segmentation and extraction of FAQs from web documents. These variations of HMMs also model pattern matching as token sequence generation and are able to deal with variations in test instances. However, they cannot be applied to definition pattern matching directly because the topologies they employ are task-specific.

In this paper, we focus our discussion on lexico-syntactic patterns used in definitional QA systems. There are other patterns beyond textual patterns. For instance in TREC 2005, LCC [Harabagiu et al. 2005] employed two additional types of pre-compiled patterns - question patterns and entity classes. Question patterns comprise a list of factoid questions whose answers are considered essential nuggets according to the type of the target. Entity classes indicate named entities relevant to the target in the corpus. These two types of patterns could be considered as pre-defined templates for searching for definitions for different targets. Since such template-like patterns need intense manual labor and expertise to construct, we do not consider them in this paper.

Statistical ranking. The second component that many current definitional QA systems make use of is statistical ranking, which used to weight the importance of extracted definition candidates. A statistical ranking method constructs a centroid

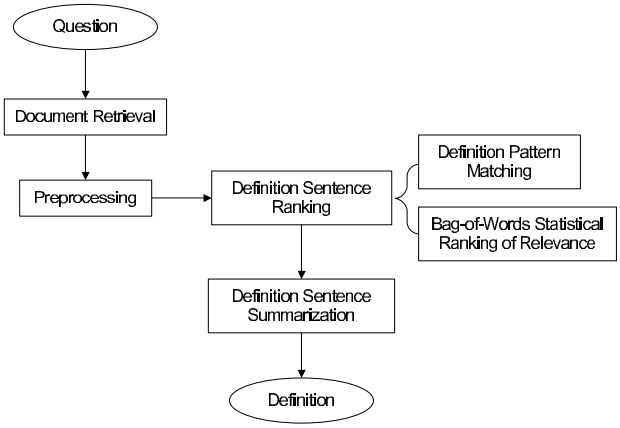

Fig. 2. Illustration of the Architecture of the Definitional QA System

vector, or *profile*, for the search target and rank the definition candidates by calculating the similarity between the candidates and the centroid vector. Centroid words are relevant, non-trivial words correlated with the search target, selected from the extracted candidates by measuring their co-occurrence with the target or their corpus frequency in an external resource. Our centroid ranking method, discussed in Section 3.1, is based on the co-occurrence technique but also generalizes the lexical tokens into syntactic tags to create more generic patterns. We will discuss how to replace centroid words with their syntactic tags in Section 4.1.

TREC systems (e.g., [Ahn et al. 2004]) also utilize definitions extracted from online encyclopedias and biographical web sites, which provide a much larger and neater resource for definitions. External definitions are usually utilized to reinforce the definition candidates from the corpus. Candidates with a higher degree of overlap with the external definitions are augmented in weights. We will not discuss the use of external definitions in detail since we focus on building a self contained definitional QA system.

3. SYSTEM OVERVIEW

Figure 2 shows the overall system architecture of our definitional QA system. Given a definition question, our system executes the following four steps to construct an appropriate answer.

(1) Document retrieval. Given a definition question, we first extract the search target as a query and feed it into a standard document retrieval system. The retrieved documents are split into sentences.

(2) Preprocessing. We process the retrieved sentences into pattern instances on which soft definition pattern generation and matching are performed. Specifically, in the training phase, labeled definition sentences are processed into pattern instances. In the testing phase, each test sentence is also transformed to a pattern instance and statistically matched against the previously trained pattern. We first replace the words that are specific to the search targets with their general syntactic (POS or chunk) tags. Remaining words are stemmed. We refer to these remain-

```
BE$ discovered by NNP <TARGET> , DT$ NN ,
DT$ JJ <TARGET> , DT$ JJ NN
NNP , known as <TARGET> , BE$ held
<TARGET> including NN in DT$
<TARGET> BE$ CD$ of DT$
```

Fig. 3. Sample Pattern Instances Generated after Pre-processing

ing lexical words and substituted general syntactic tags collectively as *tokens*. We then take the tokens surrounding the search target as pattern instances. Figure 3 illustrates several sample pattern instances.

(3) Definition sentence ranking. The ranking module ranks the input sentences based on how likely they are definition sentences for the target. We rank definition sentences using two features: definition patterns and bag-of-words relevance. To rank sentences, we combine the pattern matching and bag-of-words ranking scores using simple linear weights.

(4) Definition sentence summarization. This module produces the final answer by selecting from top ranked sentences and removing redundant sentences. Sentence editing and re-ordering may be employed to create a final coherent summary (e.g., [Jing 2000]), but this post-processing is beyond the scope of this paper.

The key steps, i.e., pattern instance generalization (Step 2) and soft pattern matching in ranking definition sentences (Step 3), deserve an in-depth discussion, and will be described in detail in the following section. In the remainder of this section, we discuss the bag-of-word relevance approach (part of Step 3) and the final summarization sentence selection (Step 4).

3.1 Bag-of-Words Statistical Ranking of Relevance

In order to accumulate as many relevant sentences for the search target as possible, we adopt *centroid ranking*, a bag-of-words statistical ranking technique for weighting the relevance of a passages with respect to a given target. Centroid ranking has been applied in summarization [Radev et al. 2004], and in definitional question answering [Xu et al. 2004; Cui et al. 2004].

In multi-document summarization, Radev et al. [2004] select centroid words by taking words that are most representative across documents by computing words' global $TF \times IDF$ weights. However, in the definitional QA context, centroid words must bear very specific information describing the search target. Therefore, we adopt a local relevance metric of words with respect to the search target based on their co-occurrences with the search target. To assess the importance of each word independent of the search target, we use inverse document frequency (IDF)[1]. A word's local co-occurrence and the global IDF scores are combined to represent the relevance of a word to the search target. The centrality of a word is then implemented in our system by the following equation:

[1]We use the statistics from Web Term Document Frequency and Rank site (http://elib.cs.berkeley.edu/docfreq/) to approximate words' IDF within the corpus.

Input: *ranked_sentences* – Rank list of sentences in descending order of definitional scores
num_selected_sentences – Number of selected sentences in the output list
Output: *selected_sentences* – List of selected definition sentences

```
Add the first sentence of ranked_sentences to selected_sentences
Remove that sentence from ranked_sentences
N = 1
for each sentence stc in the remaining of ranked_sentences
    for each sentence sel_stc in selected_sentences
        sim = CosineSimilarity(stc, sel_stc)
        record the maximum similarity max_sim
    if max_sim < similarity threshold η
      add stc into selected_sentences and remove stc from ranked_sentences
      N = N + 1
    if N > num_selected_sentences
      return
end
```

Fig. 4. Definition Sentence Summarization Algorithm

$$Centrality_T(w) = -\log \frac{SF(T,w)}{SF(T) \times SF(w)} \times IDF(w) \quad (1)$$

where T denotes the search target and w is a candidate word occurring in the context of T. $SF(w_1, w2)$ is the number of sentences that contain both w_1 and w_2 and $SF(w)$ is the number of sentences that contain w. $IDF(w)$ represents the inverse document frequency of w.

Given the input sentences, stop words are removed and the remaining words are stemmed. Centrality scores for the remaining stemmed words are calculated and those words whose scores exceed a standard deviation over the mean are selected as centroid words.

For each target, the system constructs a centroid vector from the resulting centroid words. Similar to the work by Blair-Goldensohn et al. [2004] and Xu et al. [2004], we then rank the candidate sentences by their similarity with the centroid vector, using cosine similarity. Sentences that are highly ranked are considered candidate definition sentences.

In addition to corpus statistics, we also make use of external definitions for the search targets to supplement centroid word selection. In our system, we rely on Answers.com to search for existing definitions for a given target. The main reason for utilizing external definitions is that there are only a few occurrences in the corpus for some targets, and thus it would be difficult to obtain reliable co-occurrence statistics for those contextual words. Therefore, we augment the centrality scores, i.e., the weights, of those centroid words which also appear in the external definitions.

3.2 Definition Sentence Summarization

In Step 4, the system constructs the final definition from the ranked candidate sentences. This is done by selecting the top-ranked sentences that suit the length requirement and avoid including redundant content. We adopt a variation of Maximal

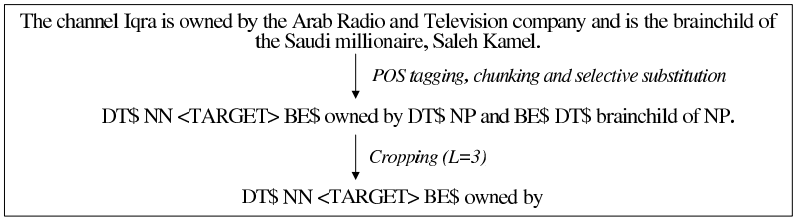

Fig. 5. Illustration of Generalization of Pattern Instances

Marginal Relevance (MMR) [Carbonell and Goldstein 1998] to select non-redundant sentences from the top of the list of sentences ranked by definition weighting scores. The sentence selection algorithm is presented in Figure 4. Different from the approach taken by Carbonell and Goldstein, who ranked all passages with MMR, our method examines sentences in descending ranked order and stops when the length of the definition is satisfied. This method takes advantage of the previous ranking step and results in a more efficient algorithm.

4. SOFT PATTERN MATCHING MODELS

In this section, we discuss the two soft pattern models: the bigram model and the profile HMM, respectively. Before introducing the soft pattern models, let us discuss how to generalize training definition sentences and test sentences into pattern instances, which are abstract token sequences representing the original sentences. Soft pattern models perform training and testing on the basis of pattern instances.

4.1 Generalization of Pattern Instances

Given a group of potential definition sentences, our goal is to learn the local contextual patterns surrounding the given search target. We do not handle long-distance dependencies, as observation shows that definition sentences are identified mainly by adjacent words and punctuation.

Definition (*Pattern Instance*). A token sequence that contains lexical and/or syntactic tokens to the left or right of the search target after performing transformation steps of tagging and chunking, selective substitution and window cropping on a candidate sentence.

The process of generalizing pattern instances is illustrated in Figure 5. It comprises three steps:

(1) **Tagging and chunking** – The sentences are first processed with part-of-speech (POS) tagging and chunking by NLProcessor, a commercial parser developed by Infogistics.

(2) **Selective substitution** – We then perform selective substitution of certain lexical items by their syntactic classes in order to generate representative patterns and to counter overfitting. The substitution attempts to replace words that are specifically related to the search term (i.e., target-specific words in Table II) with more general tags. The lexical forms of these words are specific to the search target and do not help to form general definition patterns

Table II. Heuristics Used for Selective Substitution

Token	Substitution	Examples (from the example sentence in Figure 5)
Any part of the search target	`<TARGET>`	`Iqra` → `<TARGET>`
Target-specific words (centroid words)	Corresponding syntactic classes determined by part-of-speech tags	`channel` → `NN`
Noun phrases by chunking	`NP`	`Arab Radio and Television company` → `NP`
`is, am, are, was, were`	`BE$`	`is` → `BE$`
`a, an, the`	`DT$`	`the` → `DT$`
All numeric values	`CD$`	`63` → `CD$`
Adjectival and adverbial modifiers	*Deleted*	
All other words and punctuations	*No substitution*	`owned, by, of, and, brainchild` are unchanged

and hence are replaced by their part-of-speech tags. Likewise, we perform the same substitution to noun phrases identified by chunking, as different scenarios usually do not share the same noun phrase instances. Moreover, we collapse adjacent syntactic tags of the same type into one. The substitution rules that we use and some examples are listed in Table II.

(3) **Window cropping** – We need to consider the local context around the `<TARGET>`. The context is modeled as a window centered on `<TARGET>` according to the pre-defined size L, i.e., the number of tokens on both sides of `<TARGET>`. Thus we get pattern instances with size $2L + 1$ including the search target.

Determining Substituted Words. In Step (2) of generalizing pattern instances, target-specific words are replaced with their part-of-speech tags. In this study, target-specific words are determined by centroid words, which are defined as those non-trivial words that are highly relevant to the search target. Centroid words are locally determined in the sense that they vary with each search target. A possible alternate way to determine target-specific words is to examine the complementary set of those frequent words in the training pattern instances. As definition patterns are supposed to be generic across targets, there are frequently used words, such as "known" and "born", across pattern instances. We could then construct the pattern instances by keeping these common words and substitute other words for their generic syntactic tags. However, we do not adopt this alternative as it would bias the learned definition patterns towards the frequent patterns from certain types of targets, because keywords from low-frequency targets' definitions would be ignored.

4.2 Bigram Model

The first soft matching pattern model we introduce is based on n-gram language models. Language modeling has been extensively studied in speech recognition, part-of-speech tagging and syntactic parsing [Rosenfeld 2000]. N-gram language modeling is one important approach which models local sequential dependencies be-

tween adjacent tokens. Trigrams ($n = 3$) are a common choice when large training corpora are available. We use a bigram ($n = 2$) model for soft pattern matching, as we have limited training data.

While the original bigram model is simply a product of probabilities of all bigrams in a sequence, we apply linear interpolation [Manning and Schütze 1999] of unigrams and bigrams to represent probability of bigrams. The reason is two-fold: (1) to smooth probability distribution in order to generate more accurate statistics for unseen data, and (2) to incorporate conditional probability of individual tokens appearing in specific slots. Note that we refer to slots as the positions of tokens relative to the target. In particular, we model a sequence of pattern tokens as:

$$\begin{aligned} P(t_1 \dots t_L) &= P(t_1|\mu) \prod_{i=2}^{L} (\lambda P(t_i|t_{i-1}, \mu) + (1-\lambda) P(t_i, \mu)) \\ &= P(t_1|S_1) \prod_{i=2}^{L} (\lambda P(t_i|t_{i-1}) + (1-\lambda) P(t_i|S_i)) \end{aligned} \tag{2}$$

where μ stands for the bigram model and $P(t_i|S_i)$ stands for the conditional probability of token t_i appearing in slot S_i. λ is the mixture weight combining the unigram and bigram probabilities. L denotes the number of slots in the model. Note that we use the conditional probability of a unigram being in a slot to represent unigram probability. This is because the position of a token is important in modeling: for instance, a comma always appears in the first slot right of the target in an appositive expression. Incorporating individual slot probabilities enables the bigram model to allow partial matching, which is a characteristic of soft pattern matching. Even when some slots cannot be matched, the bigram model can still yield a high match score by combining matched slots' unigram probabilities.

As test instances are often different in length, we normalize the log-likelihood of Equation 2 by the length l of the test instance:

$$P_{norm}(t_1 \dots t_L) = \frac{1}{l} (\log P(t_1|S_1) + \sum_{i=2}^{L} \log(\lambda P(t_i|t_{i-1}) + (1-\lambda) P(t_i|S_i)) \tag{3}$$

where l denotes the number of tokens in the test instance.

Next, we estimate unigram and bigram probabilities by their maximum likelihood (ML) estimates:

$$P_{ML}(t_i|S_i) = \frac{|t_i(S_i)|}{\sum_k |t_k(S_i)|} \tag{4}$$

$$P_{ML}(t_i|t_{i-1}) = \frac{|t_i(S_i) t_{i-1}(S_{i-1})|}{|t_i(S_i)|} \tag{5}$$

where $t_i(S_i)$ denotes that token t_i appears in slot S_i and $|t|$ denotes the frequency of the token t. $t_i(S_i)t_{i-1}(S_{i-1})$ represents two adjacent tokens t_i and t_{i-1}, which appear in slots S_i and S_{i-1}, respectively. In language modeling, ML estimates often suffer from sparse data. As we count tokens with respect to slot positions, the training data is even more sparse. We employ smoothing to counter the problem. For simplicity, we use Laplace smoothing on unigram probabilities (recall that bigram

probabilities have already been smoothed by interpolation):

$$P(t_i|S_i) = \frac{|t_i(S_i)| + \delta}{\sum_k |t_k(S_i)| + \delta|N(t)|} \tag{6}$$

where $|N(t)|$ gives the total number of unique tokens in our training data and δ is a constant, which is set to 2 in our experiments.

Note that we count frequencies of words and general syntactic tags separately. General tags typically have a much higher frequency compared to individual words, and would thus skew the distribution if combined with words. Thus we need to separate the two types and estimate each token's unigram probability against its own set.

Estimating the Mixture Weight λ. We use the Expectation Maximization (EM) algorithm [Dempster et al. 1977] to find optimal settings of λ. Specifically, we estimate λ by maximizing the likelihood of all training instances given the bigram model. The estimation process is as follows:

$$\begin{aligned} \lambda &= argmax_\lambda \sum_{j=1}^{|INS|} P(t_1^{(j)} \dots t_{l(j)}^{(j)}|\mu) \\ &= argmax_\lambda \sum_{j=1}^{|INS|} \frac{1}{l_{(j)} - 1} \sum_{i=2}^{l_{(j)}} \log\left(\lambda P(t_i^{(j)}|t_{i-1}^{(j)}) + (1-\lambda)P(t_i^{(j)}|S_i^{(j)})\right) \end{aligned} \tag{7}$$

$P(t_1|S_1)$ is ignored because it does not affect the estimation of λ. λ can be estimated using the EM iterative procedure:

(1) Initialize λ to a random estimate between 0 and 1, say 0.5.
(2) Update λ using:

$$\lambda' = \frac{1}{|INS|} \times \sum_{j=1}^{|INS|} \frac{1}{l_{(j)} - 1} \sum_{i=2}^{l_{(j)}} \frac{\lambda P(t_i^{(j)}|t_{i-1}^{(j)})}{\lambda P(t_i^{(j)}|t_{i-1}^{(j)}) + (1-\lambda)P(t_i^{(j)}|S_i^{(j)})} \tag{8}$$

where INS denotes all training instances and $|INS|$ is the number of training instances which is used as a normalization factor.

(3) Repeat Step 2 until λ converges.

We set λ to 0.3 according to the experimental results.

4.3 Profile Hidden Markov Model

Although the bigram model allows partial matching, it lacks the ability to deal with gaps in test instances. For instance, given training instances such as "`<TARGET> which is known for ...`", the trained bigram model cannot give reasonable match scores to test instances such as "`<TARGET> which is best known for ...`" and "`<TARGET> , whose xxx is known for ...`" even though they are simple variants of the training instances in which insertions or deletions occur. Such gaps can be captured by Profile HMMs, which allow insertion and deletion editing operations in the matching process. Figure 6 shows the topology of a PHMM.

The PHMM contains a sequence of match states, which are denoted by M_i ($i = 1 \dots L$). These match states correspond to slots in pattern instances and

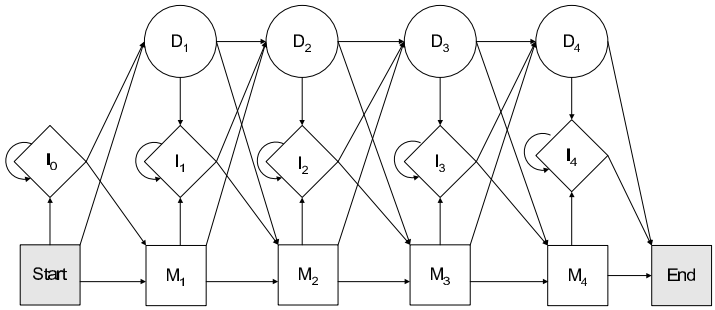

Fig. 6. Illustration of Topology of the PHMM Model

determine the model length L. Each match state can emit a token t from all tokens in the training instances with the emission probability $P(t|M_i)$. For each match state, there is a deletion state, denoted by D_i, which does not emit a token and bypasses the corresponding match state. Insertion states can emit any token t with the emission probability $P(t|I_i)$. Insertion states insert tokens after match or deletion states, as with the word *best* in the earlier example. While transitions from match states and deletion states always move forward in the model, insertion states allow self-loops, corresponding to multiple insertions. A token sequence representing a pattern instance can be generated by moving through this model with state transition probabilities $P(S_i|S_j)$. The deletion and insertion states allow the PHMM to model missing or unobserved words in training. Specifically, the probability of a sequence of tokens $t_1 \dots t_N$ generated by moving through states $S_0 \dots S_{L+1}$ (the Start and End states are S_0 and S_{L+1}) is:

$$P(t_1 \dots t_N|S_0 \dots S_{L+1}, \mu) = P(S_{L+1}|S_L) \prod_{i=1}^{L} P(t_{n(i)}|S_i) P(S_i|S_{i-1}) \qquad (9)$$

where μ stands for the model, and $P(t_{n(i)}|S_i)$ is set to 1 when S_i is a deletion state. To recognize a definition, we choose the most probable state path in the above equation to approximate the probability of the sequence being given all possible state paths, as the most probable state path often gives a much higher probability than any other path. Equation 9 can be efficiently calculated by the forward-backward algorithm [Manning and Schütze 1999]. We employ the Viterbi algorithm [Manning and Schütze 1999] to find the most probable state path. In Figure 7, we show an example to illustrate how the PHMM finds the optimal path to account for the gaps between training instances and the test instance. Although the training data does not contain any instance that has "`known`" in Slot 1 and "`NNP`" in Slot 4, the PHMM automatically selects the path that goes through a deletion state to skip Slot 1 and uses an insertion state to emit "`NNP`". Thus, the tokens are re-aligned with their most probable occurring slots such that the unseen test instance can still obtain a reasonable generative probability.

Estimation of the Model. During training, we need to estimate transition and emission probabilities for the PHMM. The training process can be accomplished by employing the standard Baum-Welch algorithm [Manning and Schütze 1999]. Corresponding to our adaptation to the calculation of sequence probability, we use

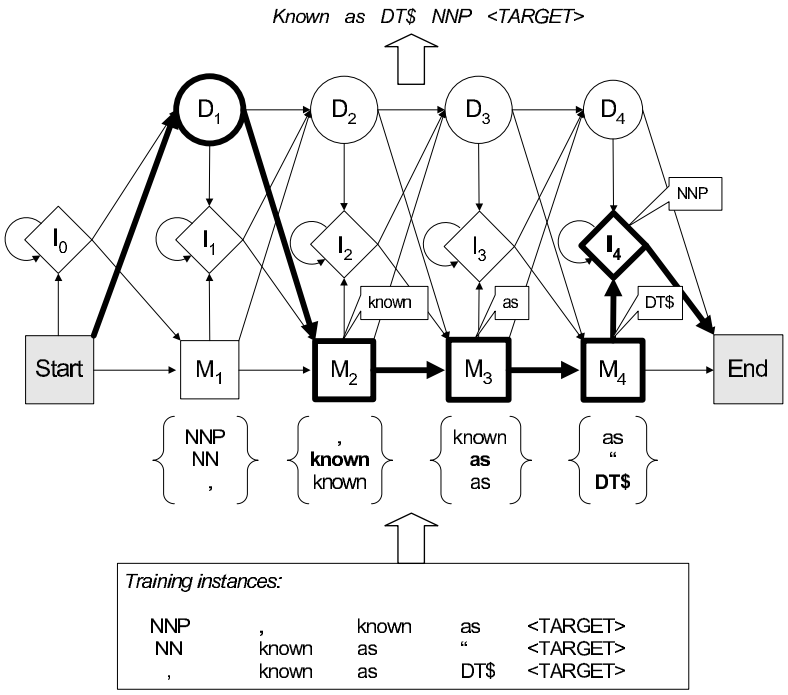

Fig. 7. Illustration of Generating a Test Instance with Gaps Using the PHMM. The optimal path is in bold, and words or tags emitted are shown in callouts.

the path with the highest probability determined by the Viterbi algorithm during the re-estimation process, and not all possible paths used in the standard Baum-Welch algorithm.

4.3.1 *Initialization of the Model.* Although probabilities in a PHMM can be estimated automatically using an iterative EM algorithm starting with random or uniform probabilities, the re-estimation process only guarantees that the model reaches a local maximum. Definition patterns are diverse and sparse in terms of both lexical tokens and POS tags. If we initialize EM with such random or uniform probabilities, we may end up with a suboptimal model that is unable to discriminate between different sequences. To make training manageable given our small training set, we assume that the most probable state path for a sequence should go through as many match states as possible. The reason is that although insertion and deletion states add flexibility, they may hurt generalization of underlying definition patterns if the model assigns high probabilities to them. Specifically, we set the emission probabilities for each match and insertion state using the smoothed maximum likelihood estimate of the emission probabilities (Equation 6). We adjust the initial value of $P(t|I_i)$ such that the probability of emitting a token from match states is always higher than that from insertion states. We set the initial state transition probability for a state as $\frac{1}{n}$, where n is the number of transition links that lead from the state.

5. EVALUATIONS

We have evaluated the proposed methods in an extensive series of experiments using the TREC question answering datasets. For completeness, we recap our previous work [Cui et al. 2004; 2005] in this section. In this paper, we extend our experiments by adopting a larger data set to test soft pattern models' robustness to scalability, and we report soft pattern matching results as measured by recent automated metrics. More specifically, our evaluation goals are: (1) to affirm the conclusion that soft pattern models outperform hard pattern matching on a larger test set which includes the latest TREC-14 data; (2) to verify our hypothesis drawn in [Cui et al. 2005] that the PHMM could yields significantly greater performance than the bigram model, given more training data; and (3) to assess the performance of our soft pattern models using the newly-proposed metric POURPRE, which is specifically designed for automatic evaluation of definitional QA systems.

5.1 Evaluation Setup

5.1.1 *Data Set.* We employ the TREC question answering task dataset for our experiments. It includes the AQUAINT corpus of over one million news articles and over 200 definition questions, each of which is about a search target. For training, we use the TREC-12 and TREC-13 data, consisting of 114 definition question-answer pairs. Based on the answer nuggets (ground truth, manually edited data) provided by TREC for these questions, we manually label 1,769 sentences that cover the nuggets from the corpus as training definition sentences for estimating the soft matching models.

5.1.2 *Evaluation Metrics.* We adopt three metrics to evaluate definitional QA performance: F_3, POURPRE and ROUGE. The F_3 measure is based on a manual examination while the latter two metrics are automatically evaluated. Automatic scores are a good supplement to manual evaluations for two reasons. First, as previous work [Lin and Demner-Fushman 2005a; Xu et al. 2004] suggests, POURPRE and ROUGE are highly correlated with manual counting of nuggets. Second, manual evaluation can often be inconsistent across runs [Voorhees 2003b].

F_3 Measure. We first adopt the evaluation metrics used in the TREC definitional question answering task [Voorhees 2004]. Along with each topic, TREC provides a list of answer keys (nuggets) to evaluate system responses. Answer nuggets are labeled as either *vital* or *okay* (see Figure 1). *Vital* nuggets represent the most important facts about the target and should be included in a definition. *Okay* nuggets contribute to relevant information but are not essential. From TREC-13 onwards, the definitional nuggets only answer questions not covered by answers to the factoid or list questions. Since we do not distinguish between the other two sources, we add the factoid and list nuggets as *vital* nuggets to the standard definitional nuggets. In the manual assessment used in TREC, an official assessor examined the nuggets covered in the returned answer; similarly in our research, one of the authors performed the manual checking of answer nuggets. We tried to preserve the the objectivity of the evaluation process by following the TREC guidelines as closely as possible. Each definition is scored using nugget recall (NR) and an approximation to nugget precision (NP) based on answer length. These

Table III. TREC Definition of NR, NP and F_3 Measure

r	number of vital nuggets in the system response
R	number of vital nuggets in the gold standard
a	number of okay nuggets in the system response
l	length of the system response

$$NR = \frac{r}{R}$$

$$NP = \begin{cases} 1 & if\ l < 100 \times (r+a) \\ 1 - \frac{l - 100 \times (r+a)}{l} & otherwise \end{cases}$$

$$F_3 = \frac{10 \times NR \times NP}{9 \times NP + NR}$$

scores are combined using the F_3 measure with recall being weighted three times as important as precision. We list the official definition of F_3 measure in Table III.

POURPRE. Lin and Demner-Fushman [2005a] proposed the POURPRE metric specifically for evaluating definitional QA systems. POURPRE simulates the process of manual checking of answer nuggets. It counts the answer nuggets that are covered by the system response by examining non-trivial unigrams shared between them. It calculates F_3-like scores by using the automatically determined nuggets. To ensure integrity of the answers, POURPRE counts only the words appearing within the same answer string. POURPRE counts a nugget matched if the system response covers a certain percentage of all the non-trivial unigrams in the gold standard. We set this ratio to 25% according to our experiments.

ROUGE. ROUGE [Lin and Hovy 2003] is a metric originally designed for summarization evaluation and has previously been adapted for definitional QA evaluation by Xu et al. [2004]. We use the metric ROUGE-3, which was adopted by Xu et al. and counts the trigrams shared between the official and system answers.

5.1.3 *Gold Standard for Automatic Scoring.* To perform automatic scoring by POURPRE and ROUGE, we construct a gold standard inventory of sentences that contain answer nuggets provided by TREC. We manually checked such nuggets.

For each search target, we construct two groups of gold standard answers, ALL and VITAL, which are analogous to the TREC answer nuggets but are corresponding judgments at the sentence level. The VITAL group consists of *vital* nuggets, which include answers to the factoid and list questions, as well as *vital* nuggets to the "other" questions. The ALL group is a superset of VITAL group, which adds the *okay* nuggets that address the "other" questions.

For each answer nugget, we retrieve up to five sentences that reflect that nugget to serve as the gold standard. This is because the answer nugget may be embedded in different sentences, possibly realized with different vocabulary. Accordingly, we construct five groups of sentences as gold standard answers for each target. A group of sample gold standard sentences for TREC topic #72 are given in Table IV.

The final scores are the average scores obtained by running the evaluation tools over the five groups of gold standard sentences.

5.1.4 *System Settings.* In our experiments, the base definitional QA system used is illustrated in Figure 2. For comparison, we apply a set of manually constructed hard matching definition patterns which have demonstrated state-of-the-art per-

Table IV. Gold Standard Sentences for the Topic 72 "Bollywood" - These sentences belong to one of the five groups of gold standard sentences. The third column indicates from what kind of question the nugget is constructed.

1	VITAL/ALL	Factoid	Around 800 movies a year come out of India. The center of the film industry is in Bombay, from which the name Bollywood is derived.
2	VITAL/ALL	Factoid	Organizers said they hoped to gain international recognition for Bollywood, the nickname given to Bombay, which boasts the world's second largest film industry after Hollywood.
3	VITAL/ALL	List	Shabana Azmi, Amitabh Bachchan, Bobby Deal, Madhuri Dixit, Sanjay Dutt, Sunil Dutt, Kajol, Anil Kapoor, Aamir Khan, Salmon Khan, Shah Rukh Khan, Amisha Patel, Aishwarya Rai, Lisa Roy, Hrithik Roshan, Sushmita Sen, Sunil Shetty
4	VITAL/ALL	Other	television production houses, such as Sony entertainment and star TV, pay huge sums to buy the rights of Bollywood favorites.
5	VITAL/ALL	Other	There was no shortage of glitz and glamor Saturday as Bombay's biggest films stars were honored at the International India Film Awards, their cinematic equivalent of the Oscars.
6	VITAL/ALL	Other	Any taxi driver picked at random can probably give you a detailed tour of the movie-star homes in Juhu Beach and Malabar Hill _ Bombay's Malibu and Beverly Hills.
7	ALL	Other	Few Americans have even heard of Bollywood.
8	ALL	Other	A new 30-screen cineplex is dedicating six screens to Bollywood. Three Bollywood movies, known as extravagant productions of epic lengths and lavish musical interludes, entered the United Kingdom's top 10 list this year.
9	ALL	Other	Hollywood star Richard Gere was honored by Bollywood at an awards ceremony for some of the top stars in India.
10	ALL	Other	A Bollywood version of JANE AUSTEN'S PRIDE & PREJUDICE received its world premiere on Mon.
11	ALL	Other	For Hollywood, poaching Indian film talent and learning from Bollywood's efficient, low-cost production techniques may become an economic necessity, as American movie-making costs soar.

formance as our baseline. The patterns used are a combination of ones from Cui et al. [2004] and Hildebrandt et al. [2004], which comprise the most complete published list of patterns to our knowledge. We list the hard patterns in Table VIII in the Appendix.

Since most of the parameters are estimated during the training process for soft pattern models, we only need to set model length for both models. According to the sensitivity analysis on model length in our previous work [Cui et al. 2005], we set the model length to the optimal settings: $L = 3$ for the bigram model and $L = 4$ for the PHMM. We also set answer length N to 14 sentences for all systems to approximate the desirable answer length used in successful TREC systems.

5.2 Experimental Results and Discussion

We list complete F_3, POURPRE and ROUGE scores by the systems in Tables V and VI. This comprises our previous evaluation results [Cui et al. 2005], adding results of POURPRE scores, and the new results by larger scale experiments in

Table V. Performance comparison of F_3, POURPRE and ROUGE scores tested on TREC-14 Data Set (trained on TREC-13 and 12 data). Percentage of improvement over the baseline is shown in the brackets; * and ** represent different significance levels, $p < 0.05$ and $p < 0.01$, respectively.

System Setting	Hard Pattern (Baseline)	Bigram SP	PHMM SP
NR (ALL)	0.3042	0.3348 (+10.06%)	0.3527** (+15.97%)
NP (ALL)	0.1391	0.1509 (+8.48%)	0.1508 (+8.41%)
F_3 (ALL)	0.2628	0.2911 (+10.73%)	0.3035** (+15.45%)
POURPRE (ALL)	0.2400	0.2539 (+5.77%)	0.2556* (+6.49%)
POURPRE (VITAL)	0.3410	0.3668* (+7.57%)	0.3730** (+9.38%)
ROUGE-3 (ALL)	0.0984	0.1004 (+2.05%)	0.1096 (+11.42%)
ROUGE-3 (VITAL)	0.1022	0.1080 (+5.69%)	0.1095 (+7.15%)

Table VI. Performance comparison of F_3, POURPRE and ROUGE scores tested on TREC-13 Data Set (trained on TREC-12 data). Percentage of improvement over the baseline is shown in the brackets; * and ** represent $p < 0.05$ and $p < 0.01$, respectively. Largely reproduced from [Cui et al. 2005].

System Setting	Hard Pattern (Baseline)	Bigram SP	PHMM SP
NR (ALL)	0.5027	0.5519* (+9.79%)	0.5420* (+7.82%)
NP (ALL)	0.3159	0.3403 (+7.72%)	0.3264 (+3.32%)
F_3 (ALL)	0.4633	0.5088** (+9.83%)	0.4971** (+7.30%)
POURPRE (ALL)	0.2785	0.2921 (+4.88%)	0.2896 (+3.99%)
POURPRE (VITAL)	0.4238	0.4580** (+8.07%)	0.4528* (+6.84%)
ROUGE-3 (ALL)	0.2106	0.2303 (+9.37%)	0.2234 (+6.08%)
ROUGE-3 (VITAL)	0.2286	0.2553* (+11.67%)	0.2496 (+9.18%)

which we train the soft pattern models using the 1,769 manually labeled sentences, as compared to employing 761 training sentences in [Cui et al. 2005]. To recap our previous work, we were drew three conclusions from our evaluations: (1) Both the PHMM and the bigram model significantly outperform the hard matching system; (2) the PHMM is less sensitive to the variation of model length than the bigram model; (3) given more training data, the PHMM tends to improve more.

However, limited by the small size of training data, we had not been able to demonstrate that the PHMM is able to perform better than the bigram model. We had conjectured that this was so as the PHMM has a more complex topology that could potentially capture more language variations. Given the additional data used in this article, we are now able to verify this hypothesis. We make the following observations from the new results in Table V:

(1) Soft pattern models again outperform hard pattern matching. As Table VI shows, both the bigram model and the PHMM perform significantly better in F_3 scores and automatic POURPRE scores (on vital nuggets) than the baseline system using hard patterns on TREC-13 data. When testing on the TREC-14 data set, the PHMM outperforms the baseline system significantly in both F_3 scores and POURPRE scores. This affirms our conclusion drawn earlier that soft pattern models are more capable of identifying definition sentences and boost the performance of definitional QA systems.

The significance levels of the improvements over the baseline vary across the automatic scores of POURPRE and ROUGE. This may be caused by mismatches

in which automatic evaluation methods incorrectly credit system responses. For instance, the following sentence for the target "DePauw University":

> *... will provide* ***scholarships*** *to* ***DePauw University students*** *from* ***Indiana****, Illinois, Michigan and Ohio.*

is mis-matched to the vital nugget:

> *Some institutions, like Rhodes College in Tennessee,* ***DePauw University*** *in* ***Indiana*** *and Bucknell University in Pennsylvania, say they allow* ***students*** *to keep 100 percent of outside* ***scholarships****.*

due to the match of non-trivial words in bold. Therefore, automatic scores are unable to discern verbose incorrect answers which overlap with the gold-standard sentences. We feel that automatic checking of answer nuggets is a good supplement, but not a substitute, for manual checking.

(2) Given more training data, the PHMM outperforms the bigram model. We use 1,769 training sentences by combining the labeled definition sentences from TREC-12 and -13, compared to 761 training sentences from using only TREC-12 in our previous work. As seen in Table V, the PHMM outperforms the bigram model by 4.26% in F_3 measure and by 9.18% in ROUGE score based on ALL nuggets. Note that the improvements obtained by the PHMM over the bigram model are not statistically significant on our test set. We discuss the possible reason at the end of this section.

(3) Evaluation results are dependent on the determination of *vital* and *okay* nuggets. The evaluation scores by both manual and automatic checking on TREC-14 data are lower across the board compared with those from TREC-13 data. In addition, the statistical significance test values (p-values) on the difference between the evaluation scores obtained by systems using the TREC-14 data are less significant than those using the TREC-13 data. We conjecture that this is due to that there are more targets in TREC-14 that have only a few *vital* nuggets. Lin and Demner-Fushman [2005b] studied the gold standard answer nuggets in TREC-12, -13 and -14. They found that 5 targets (out of 75 targets) have only one *vital* nugget and 16 targets have two *vital* nuggets in TREC-14, whereas the corresponding numbers are 2 and 15 in TREC-13 (out of 64 targets). As only *vital* nuggets count for NR, missing any of the few *vital* nuggets causes scores to drop to zero for some targets. Therefore, we see significant drop in the evaluation scores and a lower level of statistical significance for the results that are obtained from using the TREC-14 test data.

How Much Can PHMM Help?. While we have shown that given more training data, the PHMM performs better than both hard patterns and the bigram model, we have yet to quantitatively measure how much PHMM can improve over other matching models. Is the PHMM's more flexible matching mechanism actually responsible for its improvement over the bigram model?

To answer this question, we analyze the sentences that are retrieved only by the PHMM and not by the bigram model. We rank the sentences for each topic using each soft pattern model alone (without centroid ranking). We take the top ranked 50 sentences per topic and get the sentences that are in the PHMM's resulting set but not in the bigram model's. In total, we obtain 1,187 unique sentences.

We want to find the proportion of these sentences that were retrieved by the PHMM specifically by its edit operations. This can be done by checking whether a sentence retrieved by the PHMM was retrieved due to a non-trivial insertion/deletion operation. Insertion immediately following deletion (or vice versa) is an alternative to matching, and is considered a trivial use of the PHMM states as it can be simulated in the bigram model. Non-trivial uses of isolated and repeated insertion or deletion states in the PHMM cannot be represented in the bigram model. We generate the optimal state transition paths calculated by Viterbi algorithm for the left and the right sequences for each sentence. We obtain 194 sentences (or 16% of the 1,187 sentences) that are exclusively retrieved by the PHMM's non-trivial edit operations. Such a small percentage of non-trivial state sequences partially explains why the margin of difference in performance is small. Given a more noisy data set, such as web pages, the PHMM may perform even better because more sentences that cannot be matched within the training data would benefit from insertion and deletion operations.

6. CONCLUSIONS AND FUTURE WORK

We have proposed two generic soft pattern models – one based on a bigram language model and the other on the PHMM – to identify definition sentences in a definitional question answering system. Both models provide formal probabilistic methods to capture lexico-syntactic patterns represented by token sequences. Our experimental results show that both models significantly outperform the system version that uses carefully constructed hard matching patterns. In particular, we have shown that the PHMM is more capable of dealing with gaps in pattern matching caused by language variations by performing insertion and deletion editing operations. In order to show the effectiveness of the PHMM, we employ more manually labeled data for estimating the models. The evaluation results show that given more training data, the PHMM does perform better than the bigram model. Moreover, we quantitatively analyze the differences between the sentences retrieved by the two models. However, in our data set, we find only a small amount of sentences that benefit from the PHMM's special operations of insertion and deletion, and thus the PHMM does not show a large improvement over the bigram model.

While we experiment on definitional QA with the soft pattern models, we note that both models are generic matching models and can be applied to the following applications in future work:

Information extraction (IE). The pattern matching problem in IE tasks is formally the same as definition sentence retrieval. When applied to free texts, an IE system may also suffer from various unseen instances not being matched by the trained patterns. Xiao et al. [2004] have demonstrated that soft pattern matching greatly improves recall in an IE system. Although some HMM topologies have been employed for IE tasks, our models are more generic and require less configuration and parameter tuning with changing domains.

Factoid question answering. Pattern matching is also utilized to improve precision in factoid QA [Ravichandran and Hovy 2002]. Soft pattern models should be trained on each kind of questions along with the question taxonomy.

Subjective expressions. Riloff and Wiebe [2003] applied an IE system to learn patterns of subjective expressions so that opinions can be identified from news articles. Texts that reflects opinions are more diversified than those reporting facts. We believe the PHMM could significantly improve the system performance in capturing expression patterns on opinions.

APPENDIX

Table VII: Techniques Employed by Recent TREC Systems to Answer Definition Questions

TREC Systems	**Linguistic Constructs**	**Bag-of-Words Ranking**	**Mining External Knowledge**
Amsterdam [Ahn et al. 2004]	Nugget extraction based on dependency parsing trees.	Rank the sentences from the corpus by measuring their lexical and semantic similarity with the facts mined from the external reference web site. The semantic similarity is measured by the distance of words in WordNet or word co-occurrence statistics in large corpus.	Rely heavily on the external reference database - an online encyclopedia. Mine the facts about the targets from the web site.
LCC [Harabagiu et al. 2003]	Utilize 38 definition patterns, out of which 23 find match in TREC questions.		
MIT [Katz et al. 2004]	16 classes of regular expression based patterns. These patterns are used to construct a database of definitions offline.	Retrieve sentences that contain the target and rank the sentences by the overlap of keywords in the sentences and the dictionary definitions.	Dictionary look-up on an online dictionary and use the dictionary definitions to score sentences.

BBN [Xu et al. 2003; Xu et al. 2004]	—Patterns identifying appositive and copulas constructs. —Propositions extracted from parsing trees. —40+ manual rules for structured definition patterns. —Special relations extracted by a specialized information extraction module. Assign weights to linguistic constructs according to the extraction types.	Rank linguistic constructs by their similarity with the question profile. The question profile is constructed by 3 options: (1) Centroid of extracted definitions from online dictionaries, encyclopedias and biographical sites. (2) Centroid of 17,000 short biographies for the profile of a person. (3) Centroid of extracted linguistic constructs from the corpus for the target.	Construct profiles for targets by mining external definitional resources.
Columbia [Blair-Goldensohn et al. 2003]	Extract definitional predicates, which include three types of genus, species and non-specific definitional, based on 23 manual patterns on parsing trees.	Construct a centroid vector for the target by selecting frequent non-trivial words from all extracted constructs. The centroid vector is used to rank the constructs by measuring their similarity with the centroid vector.	
IBM PIQUANT [Chu-Carroll et al. 2004; Prager et al. 2003]	—Appositions and relative clauses. —Surface patterns similar to those by Ravichandran and Hovy [2002].	Establish a profile for each target. The profile is constructed by concepts represented by nouns that occur more with the target than random occurrences. Passages are ranked by the number of concepts they contained.	—Pre-defined auxiliary questions for different types of targets. —Hypernyms from WordNet to define the terms. —Biographical data from particular website.

Korea University [Han et al. 2004]	Extract pre-defined constructs from syntactic parsing trees of sentences. Such constructs include modifying phrases of the target, relative pronoun phrases, copulas, general verb phrases, etc.	Statistical ranking of extracted constructs based on: —Count of the head word of the target being as the head word in the answer constructs. —Count of terms in extracted constructs. —Trained statistics of biographical terms from an encyclopedia, applying only to persons.	Biographies from external encyclopedia for training term statistics.

Table VIII. Hard Definition Patterns Used in the Baseline System

```
<TARGET> , (a|an|the)
<TARGET> (is|are|was|were) (a|an|the)
<TARGET> , (also)* (known as|called)
<TARGET> (is|are) (usually|generally|normally)* (called|known as|defined as)
<TARGET> (refer to|refers to|satisfies|satisfy)
known as <TARGET>
<TARGET> (becomes|become|became)
<TARGET> (.{1,40})
<TARGET> , or
<TARGET> (is|are) (usually|generally|normally)* (being used to|used to|referred
 to|employed to|defined as|formalized as|described as|concerned with|called)
<TARGET> (-|: )
```

ACKNOWLEDGMENTS

The authors would like to thank Wee-Sun Lee for enlightening us on the PHMM. Thanks also go to Jimmy Lin and Dina Demner-Fushman for their help in the POURPRE evaluation toolkit. The authors are also grateful to the editors and anonymous reviewers for their detailed and valuable comments. The first author is supported by the Singapore Millennium Foundation Scholarship in his Ph.D. studies (ref no. 2003-SMS-0230).

REFERENCES

Ahn, D., Jijkoun, V., Mishne, G., Müller, K., de Rijke, M., and Schlobach, S. 2004. Using Wikipedia at the TREC QA Track. In *TREC*.

Blair-Goldensohn, S., McKeown, K., and Schlaikjer, A. H. 2003. A hybrid approach for QA track definitional questions. In *TREC*. 185–192.

Blair-Goldensohn, S., McKeown, K., and Schlaikjer, A. H. 2004. Answering definitional questions: A hybrid approach. In *New Directions in Question Answering*. 47–58.

Carbonell, J. G. and Goldstein, J. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 335–336.

Chu-Carroll, J., Czuba, K., Prager, J., Ittycheriah, A., and Blair-Goldensohn, S. 2004. IBM's PIQUANT II in trec 2004. In *TREC*.

Cui, H., Kan, M.-Y., and Chua, T.-S. 2004. Unsupervised learning of soft patterns for generating definitions from online news. In *WWW*. 90–99.

Cui, H., Kan, M.-Y., and Chua, T.-S. 2005. Generic soft pattern models for definitional question answering. In *SIGIR '05: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press, New York, NY, USA, 384–391.

Cui, H., Kan, M.-Y., Chua, T.-S., and Xiao, J. 2004. A comparative study on sentence retrieval for definitional question answering. In *SIGIR 2005 Workshop IR4QA: Information Retrieval for Question Answering*.

Dempster, A., Laird, N., and Rubin, D. 1977. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society 39*, 1–38.

Gaizauskas, R., Greenwood, M. A., Hepple, M., Roberts, I., and Saggion, H. 2004. The University of Sheffields TREC 2004 Q&A experiments. In *TREC*.

Han, K.-S., Chung, H., Kim, S.-B., Song, Y.-I., Lee, J.-Y., , and Rim, H.-C. 2004. Korea University Question Answering System at TREC 2004. In *TREC*.

Harabagiu, S. M., Moldovan, D. I., Clark, C., Bowden, M., Hickl, A., and Wang, P. 2005. Employing two question answering systems in TREC-2005. In *TREC*.

Harabagiu, S. M., Moldovan, D. I., Clark, C., Bowden, M., Williams, J., and Bensley, J. 2003. Answer mining by combining extraction techniques with abductive reasoning. In *TREC*. 375–382.

Harabagiu, S. M., Moldovan, D. I., Pasca, M., Mihalcea, R., Surdeanu, M., Bunescu, R. C., Girju, R., Rus, V., and Morarescu, P. 2000. FALCON: Boosting knowledge for answer engines. In *TREC*.

Hildebrandt, W., Katz, B., and Lin, J. J. 2004. Answering definition questions with multiple knowledge sources. In *HLT-NAACL*. 49–56.

Jing, H. 2000. Sentence reduction for automatic text summarization. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 310–315.

Katz, B., Bilotti, M., Felshin, S., Fernandes, A., Hildebrandt, W., Katzir, R., Lin, J., Loreto, D., Marton, G., Mora, F., and Uzuner, O. 2004. Answering multiple questions on a topic from heterogeneous resources. In *TREC*.

Klavans, J. and Muresan, S. 2001. Evaluation of DEFINDER: a system to mine definitions from consumer-oriented medical text. In *JCDL*. 201–202.

Lannon, J. M. 1991. *Technical Writing*. HarperCollins Publishers Inc.

Lin, C.-Y. and Hovy, E. H. 2003. Automatic evaluation of summaries using N-gram co-occurrence statistics. In *HLT-NAACL*.

Lin, J. and Demner-Fushman, D. 2005a. Automatically evaluating answers to definition questions. In *HLT/EMNLP*. 931–938.

Lin, J. and Demner-Fushman, D. 2005b. Will pyramids built of nuggets topple over?

Liu, B., Chin, C. W., and Ng, H. T. 2003. Mining topic-specific concepts and definitions on the web. In *WWW*. 251–260.

Mani, I., Pustejovsky, J., and Sundheim, B. 2004. Introduction to the special issue on temporal information processing. *ACM Transactions on Asian Language Information Processing (TALIP) 3,* 1, 1–10.

Manning, C. D. and Schütze, H. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts.

McCallum, A., Freitag, D., and Pereira, F. C. N. 2000. Maximum Entropy Markov Models for information extraction and segmentation. In *ICML*. 591–598.

Muresan, S., Popper, S. D., Davis, P. T., and Klavans, J. L. 2003. Building a terminological database from heterogeneous definitional sources. In *DG.O*.

Muslea, I. 1999. Extraction patterns for information extraction tasks: A survey. In *Proceedings of AAAI-99 Workshop on Machine Learning for Information Extraction*. 1–6.

Peng, F., Weischedel, R., Licuanan, A., and Xu, J. 2005. Combining deep linguistics analysis and surface pattern learning: A hybrid approach to Chinese definitional question answering. In *HLT/EMNLP*. 307–314.

Prager, J., Radev, D., and Czuba, K. 2001. Answering what-is questions by virtual annotation. In *HLT '01: Proceedings of the first international conference on Human language technology research*. Association for Computational Linguistics, Morristown, NJ, USA, 1–5.

Prager, J. M., Chu-Carroll, J., Czuba, K., Welty, C. A., Ittycheriah, A., and Mahindru, R. 2003. IBM's PIQUANT in TREC2003. In *TREC*. 283–292.

Radev, D. R., Jing, H., Sty, M., and Tam, D. 2004. Centroid-based summarization of multiple documents. *Inf. Process. Manage. 40,* 6, 919–938.

Ravichandran, D. and Hovy, E. H. 2002. Learning surface text patterns for a question answering system. In *ACL*. 41–47.

Riloff, E. and Wiebe, J. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, M. Collins and M. Steedman, Eds. 105–112.

Rosenfeld, R. 2000. Two decades of statistical language modeling: Where do we go from here. *Proceedings of the IEEE 88,* 8.

Schiffman, B., Mani, I., and Concepcion, K. J. 2001. Producing biographical summaries: Combining linguistic knowledge with corpus statistics. In *ACL*. 450–457.

Schwartz, A. S. and Hearst, M. A. 2003. A simple algorithm for identifying abbreviation definitions in biomedical text. In *Pacific Symposium on Biocomputing*. 451–462.

Skounakis, M., Craven, M., and Ray, S. 2003. Hierarchical Hidden Markov Models for information extraction. In *IJCAI*. 427–433.

Voorhees, E. M. 2001. Overview of the TREC 2001 Question Answering Track. In *TREC*.

Voorhees, E. M. 2003a. Evaluating answers to definition questions. In *HLT-NAACL*.

Voorhees, E. M. 2003b. Overview of the TREC 2003 Question Answering Track. In *TREC*. 54–68.

Voorhees, E. M. 2004. Overview of the TREC 2004 Question Answering Track. In *TREC*.

Xiao, J., Chua, T.-S., and Cui, H. 2004. Cascading use of soft and hard matching pattern rules for weakly supervised information extraction. In *Proceedings of COLING 2004*. COLING, Geneva, Switzerland, 542–548.

Xu, J., Licuanan, A., and Weischedel, R. M. 2003. TREC 2003 QA at BBN: Answering definitional questions. In *TREC*. 98–106.

Xu, J., Weischedel, R. M., and Licuanan, A. 2004. Evaluation of an extraction-based approach to answering definitional questions. In *SIGIR*. 418–424.

Yang, H., Cui, H., Maslennikov, M., Qiu, L., Kan, M.-Y., and Chua, T.-S. 2003. QUALIFIER In TREC-12 QA main task. In *TREC*. 480–488.

Zahariev, M. 2003. Efficient acronym-expansion matching for automatic acronym acquisition. In *IKE*. 32–37.

Received MONTH 9999; accepted MONTH 9999