Structured Term Recognition in Medical Text

Michael Glass *Alfio Gliozzo*
IBM T.J. Watson Research, Yorktown Heights
mrglass@us.ibm.com, gliozzo@us.ibm.com

ABSTRACT
In this work we present a novel approach to bootstrap domain specific terminology, namely Structured Term Recognition, and we apply it to the medical domain. In contrast to previous approaches, based on observing distributional properties of terminology with respect to their contexts, our method analyzes the "internal structure" of multi-word terms by learning patterns of word clusters. Evaluation shows that our method can be used to boost the performances of term recognition systems based on dictionary lookup while extending the coverage of large ontologies like UMLS.

KEYWORDS: terminology recognition, bootstrapping, medical terminology, named entity recognition, joint inference.

Proceedings of COLING 2012: Technical Papers, pages 943–958,
COLING 2012, Mumbai, December 2012.

1 Introduction

Recognizing occurrences of domain specific terms and their types is important for many text processing applications. This problem is not easy, particularly in domains like medicine, where very rich terminology is generated by domain experts on a daily basis.

In spite of the large interest in statistical term recognition in Natural Language Processing (NLP), state of the art approaches for term recognition in the medical domain are still based on dictionary lookup with some heuristics for partial mapping (Aronson, 2001). In fact, very large terminological resources, such as the Unified Medical Language System (UMLS) (Bodenreider, 2004), have been developed in the medical domain. The reason is that medical terminology can not be identified by looking at superficial features only, such us capitalization of words, prefixes and suffixes. In fact, diseases names, symptoms and most medical terms are not proper names, so they are not capitalized. In addition, they are usually characterized by a rather complex internal structure and composed by many words. In addition, distributional similarity metrics, i.e. recognition approaches based on the analysis of the local context where the term is located, work well when applied to single words or very frequent words, which is not the case for most of the medical terms we are interested in.

In the context of the research on adapting a question answering system to the medical domain, we encountered the term recognition problem in many places. For example, recognizing names of diseases, symptoms and treatments is necessary to answer most of the Doctor's Dilemma^{TM}questions (American College of Physicians, 2012), an evaluation benchmark we used to measure the ability of our system to answer medical questions. To assess the validity of the answer "HFE hereditary hemochromatosis" with respect to the question "Metabolic condition that can set off airport metal detector", it is important to know that the answer is a type of metabolic condition. One way to address this problem is to use medical lexica where different terms are associated to semantic types, and then check whether the type of the candidate answer matches the type required by the question.

On the other hand, UMLS is far from complete. Many disease names (especially multi-words) are not recognized by dictionary lookup approaches. In many cases, specific terms are missing (e.g. "HFE hereditary hemochromatosis" in the question above). The problem is particularly evident when examining the answers to the Doctor's Dilemma questions. Analyzing 704 doctor's dilemma questions where the system located the correct answer as candidate, in 16.5% of the cases one of the correct answers was not in UMLS. About half of them were quantities (e.g. 20mg) while the remaining half were medical terms that are not in UMLS. On the latter subset of questions, the end to end accuracy of the full QA system dropped by almost 50% if compared to the performance on questions where the answer was an UMLS term. The explanation for this drop in performance is mostly due to type recognition problems: the system is not able to select the right answer on the basis of its type, for example confusing treatments with diseases.

This observation motivates the work presented in this paper. Our method is able to recognize the type of domain specific terms based on an innovative principle, called Structured Term Recognition (STR). We begin with a simple observation: in many technical domains - and in the medical domain in particular, terms have repetitive patterns. For example, terms describing cancer include "bladder cancer", "brain tumor", "skin cancer" and "uterine sarcoma". These terms are all composed of a word indicating a part of the human anatomy followed by a word similar to "cancer". If the sequence of words "pituitary tumor" appears in text, and "pituitary" is known to be an a word indicating a part of the anatomy, it is reasonable to identify this

as a cancer term even if it is not in the dictionary of cancer terms. A simple baseline where any noun phrase with the word "cancer" as it's head is identified as a cancer term will lead to such misidentifications as "some types of cancer" and "a different cancer". In fact, syntactic information alone is not enough to recognize technical terms, additional semantic information reflecting the type of the term constituents should be considered.

Structured Term Recognition (STR) is a way to address this problem. It works by examining the semantic structure of terms, acquiring semantic patterns describing it, and identifying new terms with similar structure. In this paper we focus on multi-word terms, in the case of single word terms our method becomes indistinguishable from prior work on extending a thesaurus by distributional similarity.

Similar to other existing bootstrapping methods, STR begins with an existing dictionary and uses patterns to extend its coverage. In contrast, it does not use contextual information or shallow features describing the term itself, instead it uses "internal deep semantic" features describing what a domain term should look like, i.e. its semantic pattern. Combinations of STR and techniques based on contextual information are obviously possible. However, in this paper we decided to focus on the contribution of the STR technique in isolation because we want to show that in the medical domain this information is a strong indicator of the entity type if well represented and handled. We present three different approaches to solve the problem, two based on recognizing terms on the basis of the sequences of the word clusters and a third one based on Probabilistic Context Free Grammars (PCFG) and we evaluate them on the task of predicting the semantic group of a held out subset of terms extracted from UMLS.

The paper is structured as follows. Section 2 describe STR into details, while section 3 is about evaluating our method. In section 4 a literature review is done, while section 5 concludes the paper and highlight directions for future work.

2 Structured Term Recognition

The inputs of STR are a set of terms $t \in T$ from a dictionary, each belonging to a set of types $\Gamma(t) = \{\gamma_1, \gamma_2, \ldots\}$, and a domain specific corpus U. Terms are composed of lists of words $t = w_1, w_2, \ldots, w_n$ as identified by a tokenizer. The union of all tokens for all of the terms is the word set W.

The output of STR is a set of pattern-to-type mappings $p \mapsto \gamma$ that can be used to recognize terms and their type in new texts. These terms are not necessarily present in U.

STR combines two key components:

1. Clustering words in W by their distributional and type similarity.
2. Constructing patterns describing terms in the dictionary.
   (a) Sequences of Clusters are patterns.
   (b) Patterns are the left hand side of PCFG rules.

2.1 Word Clustering

The goal of this component is to cluster the words in W into clusters $C_1, C_2, \ldots, C_n$. Clusters are partly determined by distributional similarity, meaning that words in the same cluster

have similar meanings because they are substitutable in context. This follows a large body of work in unsupervised clustering of words. Early work used distributional similarity to perform part-of-speech induction (Schütze, 1993). Later work has developed automatic methods of thesaurus construction using similar techniques (Lin, 1998).

Using a large, domain specific, unannotated corpus U, STR gathers vectors describing the context in which each word is found. Since parsing is generally lower quality in non-newswire domains, we do not use dependency based contexts. Instead the corpus is simply tokenized. Contexts are built from a small window of surrounding words: two words to the left and two words to the right. The information about the direction and distance from the word is retained. For example, in the sentence "Early detection of **cancer** increases the odds of successful treatment." the contexts for the word "cancer" are given in table 1.

cancer	
Direction	Context Word
Left-2	detection
Left-1	of
Right-1	increases
Right-2	the

Table 1: Contexts for "cancer"

For each word we build a vector of contexts. Note here that in order to build the context vector we do not restrict context words to words in W, instead we consider any possible token in the large corpus. Each dimension of the vector is an individual context f and its value is the number of times it occurred with the word. This vector of raw counts must be re-weighted, otherwise very high frequency but uninformative contexts, such as "of" and "the", will dominate. We use Pointwise Mutual Information (PMI) as a re-weighting method. PMI measures the strength of association between two events such as the occurrence of a word and the occurrence of a context.

$$\text{pmi}(w; f) = \log \frac{p(w, f)}{p(w)p(f)} \tag{1}$$

Equation 1 gives the formula for PMI, where $p(w)$ is the frequency of the word in the corpus divided by the number of words in the corpus and $p(f)$ is the frequency of the context divided by the total number of contexts.

Once the context vectors for each word are gathered and re-weighted they can be compared for similarity using the cosine similarity metric.

The domain specific dictionary may have many of the words in W mapped to types. However, in general a word need not be mapped to one type. It may have multiple senses, each with a distinct type. For each word we construct a type vector, where the dimensions are types and the value is either a one or zero depending on whether the word is given as that type in the dictionary. Since the dictionary groups words with similar meanings in the same type, words with overlapping type vectors will have some meanings that are similar.

2.2 Pattern Acquisition

The goal of this step is to learn a set of rules $R = \{p_1 \mapsto \gamma_1, p_2 \mapsto \gamma_2, \ldots, p_k \mapsto \gamma_k\}$ which are able to recognize most terms in T and at the same time can be used to identify new terms in the large corpus. This is done by generating a set of candidate rules and scoring them according to how they perform on the training set. We trained two different models for clusters and patterns.

In the sequence of clusters model, each word is assigned to a single cluster, patterns are sequences of clusters, and rules map a pattern to a type. Each possible sequence of clusters maps to at most one type.

In the probabilistic context free grammar (PCFG) model, words are assigned to multiple clusters, each assignment has some weight corresponding to a probability. The rules are a set of binary PCFG rules where two non-terminals are mapped to a single non-terminal. All non-terminals on the right hand side of such rules are types. Each possible sequence of clusters has a Viterbi (most likely) parse ending in a single type.

2.3 Sequence of Clusters

For the Sequence of Clusters model, the clustering is a mapping function $\Theta : W \rightarrow C$ assigning each word in W to its corresponding cluster. The patterns are the mappings of each term $t \in T$ into its corresponding pattern $P(t) = p = \Theta(w_1), \Theta(w_2), \ldots, \Theta(w_j)$, where t is the sequence of words $t = w_1, w_2, \ldots, w_j$. A term is typed if there is a rule $P(t) \mapsto \gamma \in R$. This typing is correct ($t \in Correct$) if t has γ as a type in the dictionary, $\gamma \in \Gamma(t)$, otherwise it is incorrect ($t \in Incorrect$).

An example sequence of clusters pattern and some of its recognized terms are given in Table 2. The clusters are named according to their most frequent members. So "{sensation, fear, feeling}" is the cluster that includes those words, as well as other distributionally similar words.

Pattern		
{spastic, flaccid, tonic}	{sensation, fear, feeling}	{tenderness, headache, tremor}
Recognized Terms		
horizontal	gaze	paresis
tonic	reflex	spasm
dural	puncture	headache
exertional	heat	exhaustion
charley	horse	cramps

Table 2: Example pattern and recognized terms

The training seeks to create rules and word-to-cluster mappings to maximize intra-cluster similarity and the performance of the rules on the training data. The intra-cluster similarity is measured by the cosine of each word to its cluster's centroid. The performance of the rules is given by $|Correct| - |Incorrect| + \theta_{REG} \cdot |R|$. The θ_{REG} is a regularization parameter to penalize the creation of too many rules.

2.3.1 Non-Joint Optimization

A straightforward way of doing this is to first optimize intra-cluster similarity, which can easily be converted to the objective function in spherical KMeans. We use GMeans (Dhillon et al.,

2001), a variant of KMeans that uses first variation to escape from some local maxima, to cluster. Then, given fixed word-to-cluster assignments, the optimal set of rules may be found in linear time. By mapping each term to its pattern and constructing a hash map from patterns to sets of terms we can determine for any possible θ_{REG} if mapping the pattern to the most frequent type γ_{ts} in its term set will improve rule performance. This method is fast, but it does not perform nearly as well as a joint optimization.

2.3.2 Joint Optimization

Our method of joint optimization is to cast the problem as a MAP (Maximum a posteriori) inference task in probabilistic logic and use a general purpose probabilistic inference engine to do the optimization. Different sets of word-to-cluster mappings and rules are different possible worlds. The probability of a world is given by the objective function and the inference attempts to find the most likely world.

The objective function is precisely described by the formulas in Figure 1, which closely matches the description given to the probabilistic inference engine. In this description, all variables are implicitly universally quantified. The quantity preceding the "#" in the formulas that have them is a weight, every true grounding of the formula on the right adds the quantity on the left to the weight of the possible world. Like Markov Logic (Richardson and Domingos, 2006), the semantics of the weight are that it is the log of an unnormalized probability. Note that unlike Markov Logic, we are not limited to first order logic.

$$\theta_{DC} \cdot \textbf{cosine}(v, cv) \;\#\; inCluster(w, c) \wedge distVect(w, v) \wedge distCentroid(c, cv)$$
$$cluster(c) \Rightarrow distCentroid(c, \textbf{sum}(\{v : inCluster(w, c) \wedge distVect(w, v)\}))$$
$$\theta_{TC} \cdot \textbf{cosine}(v, cv) \;\#\; inCluster(w, c) \wedge typeVect(w, v) \wedge typeCentroid(c, cv)$$
$$cluster(c) \Rightarrow typeCentroid(c, \textbf{sum}(\{tv : inCluster(w, c) \wedge typeVect(w, tv)\}))$$
$$wordAt(t, i, w) \wedge inCluster(w, c) \Rightarrow typeAt(t, i, c)$$
$$term(t) \Rightarrow patternOf(t, \textbf{makePattern}(\{(i, c) : typeAt(t, i, c)\}))$$
$$\theta_A \cdot \textbf{scoreType}(t, \gamma) \;\#\; isRule(p, \gamma) \wedge patternFor(p, t)$$
$$\theta_{REG} \;\#\; isRule(p, \gamma)$$

Figure 1: Logical description of the Sequence of Clusters model

There are three basic parts: the first four lines give the objective function for spherical KMeans, lines 1 and 2 using distributional similarity and lines 3 and 4 using type vector similarity. The next three lines evaluate the performance of the patterns on the training data. The final line is the regularization, penalizing the creation of too many patterns.

The predicates $distVect, typeVect, cluster, term, wordAt$ are given as evidence to the inference. Each word is related to its normalized distribution vector and type vector by $distVect$ and $typeVect$ respectively. A term, position and word at that position are related by $wordAt$. The function **makePattern** takes a set of index/cluster pairs and produces a sequence of clusters as a string so that for every term t, $patternOf(t, P(t))$. The distribution and type centroids are found by summing the corresponding vectors from the words in the clusters. Since cosine is our similarity metric, it is not necessary to renormalize these vectors. The function **scoreType**

implements the logic of returning a 1 for a correct type mapping, -1 for an incorrect type mapping and 0 for a $\emptyset$ type mapping.

The θ parameters control the balance of the different components of the optimization. We selected values of $\theta_{DC} = 1, \theta_{TC} = 1, \theta_A = 1, \theta_{REG} = -3$.

The predicates $inCluster, isRule$ are the proposal predicates. The basic inference algorithm is to make proposals which are then either accepted or rejected. We have three proposals: move a word to a different cluster, create a rule, and remove a rule. A proposal is accepted if and only if it improves the objective function. The proposals are described formally in Table 3. The variables in the proposal formula are bound to a random true grounding of the formula. Then the remove list is set false and the add list is set true.

Proposal Formula	Remove	Add
$inCluster(w, c_1) \wedge cluster(c_2) \wedge c_1 \neq c_2$	$inCluster(w, c_1)$	$inCluster(w, c_2)$
$patternFor(p, t) \wedge \gamma \in \Gamma(t) \wedge \neg isRule(p, \gamma)$	$\emptyset$	$isRule(p, \gamma)$
$isRule(p, \gamma)$	$isRule(p, \gamma)$	$\emptyset$

Table 3: Proposals for the Sequence of Clusters Model

A key optimization is to not recompute the centroid of each cluster on every proposal. Instead, it is recomputed with a probability that decreases as the size of the cluster increases. The intuition behind this idea is that the centroid does not change much if the size of the cluster is large.

A final step in sequence of clusters is to find the optimal pattern set given the word-to-cluster assignments, as in the non-joint approach. This step can also accommodate different precision-recall trade-offs by varying the θ_{REG} parameter.

In the sequence of clusters model we restrict our attention to terms of length two and three so that each sequence of clusters will have many terms that match it.

2.4 Probabilistic Context Free Grammar

In order to address longer terms, and to improve performance we developed a Probabilistic Context Free Grammar (PCFG) for terms. This grammar is binary with non-terminals consisting of types and clusters.

For all words in the dictionary as terms, we fix their cluster assignments to their types, initially with equal weight. Formally, for all single-word terms w and all types $\gamma \in \Gamma(w)$, $inCluster(w, \gamma, 1)$. We used the results of the previous model to assign clusters to the remaining words. This is a soft clustering.

An example PCFG parse is given in Figure 2. The non-terminal labeling a terminal (word) may be either a cluster or a type, but the non-terminal labeling a pair of non-terminals is always a type.

The objective function is to maximize the correctness of the Viterbi parses on the training data. The objective function is precisely described in Figure 3.

As before the predicates $wordAt, term$ are given as evidence. Now $term$ relates a term to its length. The function **maxType** takes a set of type/weight pairs and returns the type with the largest weight or $\emptyset$ if the set is empty. The key predicate is $chart$ which holds the chart parse as a set of ground predicates where $chart(t, i, j, \gamma, x)$ indicates that for term t the type γ spans

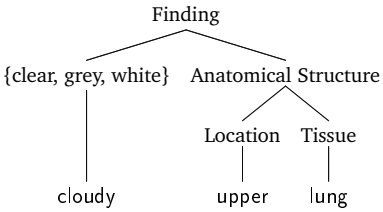

Figure 2: Example PCFG parse

$$wordAt(t,i,w) \wedge inCluster(w,\gamma,x) \Rightarrow chart(t,i,(i+1),\gamma,x)$$
$$isRule(\gamma_1,\gamma_2,\gamma_p,x_p) \wedge chart(t,i,j,\gamma_1,x_1) \wedge chart(t,j,k,\gamma_2,x_2) \Rightarrow$$
$$typeAt(t,i,k,\gamma_p,(x_1+x_2+x_p))$$
$$typeAt(t,i,j,\gamma,z) \Rightarrow chart(t,i,j,\gamma,\max(\{x : typeAt(t,i,j,\gamma,x)\}))$$
$$term(t,l) \Rightarrow vitParse(t,\textbf{maxType}(\{(\gamma,x) : chart(t,0,l,\gamma,x)\}))$$
$$\theta_A \cdot \textbf{scoreType}(t,\gamma) \ \#\ vitParse(t,\gamma)$$
$$\theta_{REG} \ \#\ isRule(\gamma_1,\gamma_2,\gamma_p,x_p)$$

Figure 3: Logical description of the PCFG model

i to j with log likelihood x. The proposal predicates are $isRule, inCluster$. The weights of rules and proposals are the log of their probability. The proposals are: change the weight of a parse rule or cluster assignment, add a parse rule, and remove a parse rule. Table 4 gives these proposals formally. The function $random(x,y)$ produces a random number between x and y.

Proposal Formula	Remove	Add
$inCluster(c,w,x)$	$inCluster(c,w,x)$	$inCluster(c,w,random(-2,2))$
$isRule(\gamma_1,\gamma_2,\gamma_p,x)$	$isRule(\gamma_1,\gamma_2,\gamma_p,x)$	$isRule(\gamma_1,\gamma_2,\gamma_p,random(-2,2))$
$\gamma_1 \in C \cup \Gamma \wedge \gamma_2 \in C \cup \Gamma \wedge \gamma_p \in \Gamma$	$\emptyset$	$isRule(\gamma_1,\gamma_2,\gamma_p,random(-2,2))$
$isRule(\gamma_1,\gamma_2,\gamma_p,x)$	$isRule(\gamma_1,\gamma_2,\gamma_p,x)$	$\emptyset$

Table 4: Proposals for the PCFG Model

Another key optimization is the use of declarative variable coordination inspired by the imperative variable coordination of Factorie (McCallum et al., 2009). The formulas with a "⇒" may be read as infinite weight implications, but unlike other approaches to combining hard and soft formulas we never consider states where the hard formulas are violated. Instead, for every true grounding of the left hand side, we immediately create a grounding of the right hand side. If all the left hand sides supporting a coordinated predicate become false, the corresponding coordinated predicate also becomes false. It may be more clear to consider the predicate on the right hand side as being defined by the disjunction of all its left hand sides.

Because the speed of the training is closely related to the length of the terms we only train on terms up to length four. However, we test the model on all terms.

3 Evaluation

The patterns found by the STR algorithm produce a term recognition function, able to decide the type of a term $t = w_1, w_w, \ldots, w_j$ by simply checking for a rule $P(t) \mapsto \gamma$ in the case of sequence of clusters, or by parsing the term with a chart parser in the PCFG model. We evaluate the quality of this recognition function by testing it on a held out set of terms in the dictionary. As with training, we consider a mapping correct if it maps the term to one of the types given in the dictionary.

3.1 Experimental Setup

For our input dictionary we use the Unified Medical Language System (UMLS). UMLS is an ontology of mostly medical terms called atoms. There is a type hierarchy with 133 semantic types these are collected into 15 semantic groups. The semantic groups are a flat and clear division of semantic types. Each UMLS atom may have multiple semantic types and possibly multiple semantic groups. In our test set 7.2% percent of the atoms had multiple semantic groups.

First we selected the multi-word UMLS atoms that occur in our corpus at least 20 times. Our corpus is 3.3 GB of text from Medline abstracts and medical textbooks. This first step is necessary because many UMLS atoms are not terms that occur in text and therefore there is no benefit to recognizing them. This set of UMLS atoms is then divided into a training and test set with 90 percent in training and the remaining 10 percent in test. We trained STR to recognize the semantic group of the terms in the training set, then evaluated the resulting rules on the test set. This produced a training size of around 72,000 terms.

3.2 Results

For the sequence of clusters model we obtain a precision recall curve by varying the pattern regularization parameter θ_{REG}. For the PCFG model we vary the parse score threshold to obtain a precision-recall trade-off.

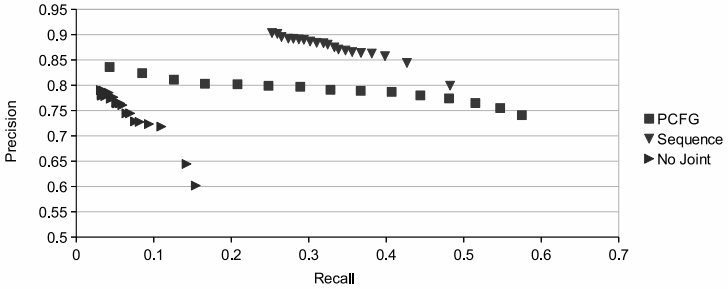

Figure 4: Precision Recall Curves

With a $\theta_{REG} = -20$ we obtain a precision of 90.3% and a recall of 25.3%. Setting the regularization parameter to zero maximizes recall at 48.2% with precision decreasing to 79.9%. With the parse score threshold of 3.5, we obtain a precision of 83.6%, but a very low recall,

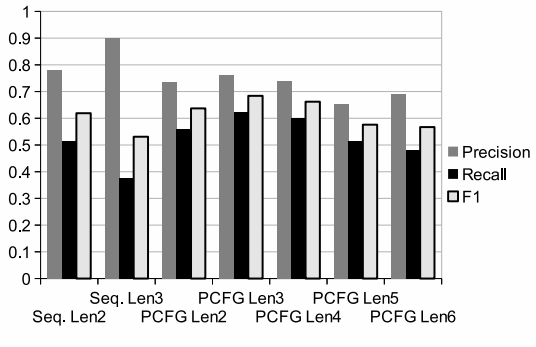

Figure 5: Performance per Term Length

4.3%. Dropping the threshold entirely gives a precision of 74.1% and a recall of 57.5%. Both of these models substantially outperform the non-joint method described in Section 2.3.1. A baseline model of selecting the most frequent semantic group would perform at precision = recall = F1 score = 23.8%. With a test set of approximately 8,000 all of these differences are highly significant.

Note that while the sequence of clusters model is evaluated only on length two and three terms, the PCFG model is evaluated on all terms. Figure 5 compares the performance of each model on different term lengths.

The performance of all models improves as the amount of optimization time increases. The learning curve for the sequence of clusters model is given in Figure 6. Note that the x-axis is the number of proposals, not the number of training instances. Performance grows from an F1 score of 49.4% with 500 thousand proposals to an F1 score of 60.2% after 27 million proposals.

The performance per group is given in Figure 7. The groups are listed in order of their frequency in the dictionary. There is a clear effect of term frequency on performance with all but one of the top performing half of groups in the top half of term frequency.

The performance of the PCFG model is broken down by group in Figure 8. The recall and F1 scores on “Genes and Molecular Sequences” and “Devices” show large gains relative to the sequence of clusters model. There is again a clear effect of term frequency.

Performance is generally lowest on the semantic groups that are not specifically biomedical: Objects, Devices, Phenomena, Activities & Behaviors, Organizations, Occupations and Geographic Areas. Other than a lower frequency in UMLS, this is likely due to a lower amount of regular structure in these less technical terms.

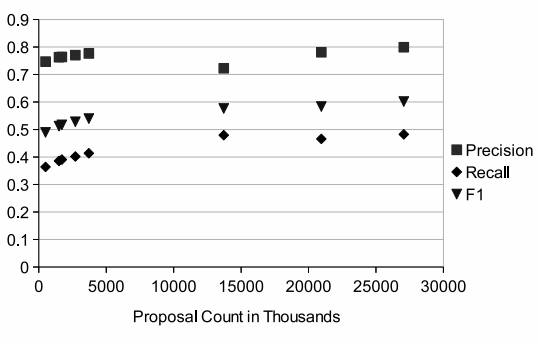

Figure 6: Learning Curve for Sequence of Clusters Model

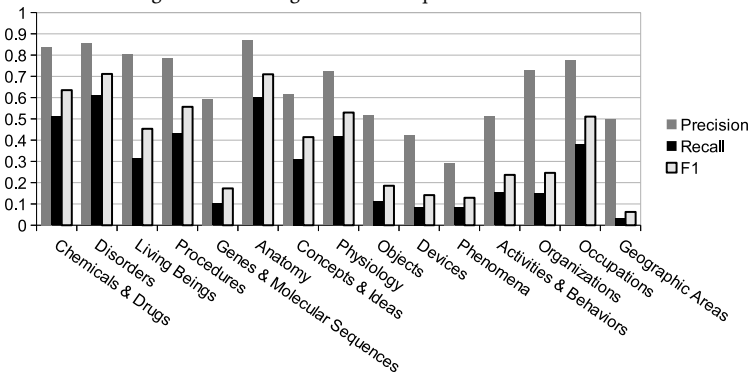

Figure 7: Performance per Group for Sequence of Clusters

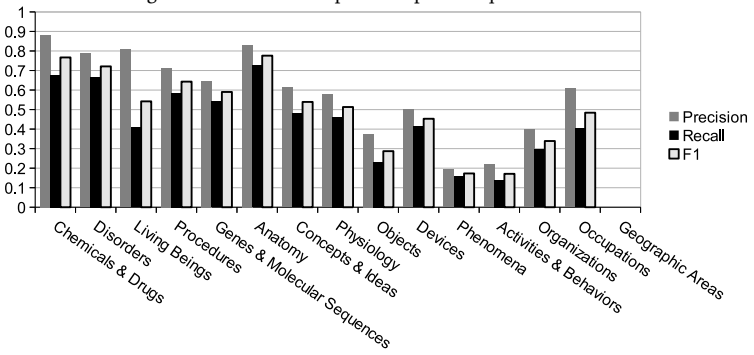

Figure 8: Performance per Group for PCFG

4 Related Work

The goal of our work is to extend UMLS with additional terms mapped to types. Other work has pursued similar goals. This section provides a state of the art about.

4.1 Automatic Term Recognition

There is a large body of work on Automatic Term Recognition ATR and term clustering. In STR we identify that a sequence of words is a term and that it has a particular type in a single process. ATR operates by first gathering strings that are terms and then clustering them, or extending existing clusters where each cluster of terms are terms of a common type.

After terms are identified, some measure of term similarity is applied to cluster the terms or extend existing clusters of terms. Some methods of term similarity are contextual, lexical and pattern-based. Contextual similarity uses information about what words appear before and after the term (Maynard and Ananiadou, 2000), or what words are nearby in a dependency parse (Grefenstette, 1994). The frequency of each of these contexts is a dimension in a vector describing the distribution of the term. These vectors can then be compared for similarity using some vector similarity function. Lexical similarity examines what words are in common between two terms. For example, two terms with the same head word are likely to have the same type. Pattern-based similarity (Nenadić et al., 2002) uses lexico-syntactic patterns like the such-as pattern "X such as Y and Z". These patterns can be used as evidence that the terms appearing as Y and Z have the same type.

ATR deals primarily with frequent words. The term must be frequent enough to be identified as a term and in the case of contextual similarity must be frequent enough to build a meaningful distribution vector. Only the lexical similarity metric examines the words inside the term and this metric only considers lexical matches between the words of the term, not the types of the words or the structure of the words in the term. The main advantage of this type of work is that it can proceed without an existing term dictionary.

4.2 Supervised Named Entity Recognition

Our methods are also similar to Named Entity Recognition (NER). NER is the task of identifying mentions of rigid designators, especially people, places and organizations (Coates-Stephens, 1992). Recently, biological types such as protein, DNA and cell line have gained attention (Settles, 2004). Unlike automatic term recognition, where the goal is to build a dictionary of terms, NER typically proceeds from a corpus annotated with mentions and their types and learns a model for detecting future mentions.

One common model is the linear chain Conditional Random Field (CRF) (McCallum and Li, 2003). This is the discriminatively trained variant of the Hidden Markov Model (HMM) (Bikel et al., 1997). In the simplest version of such models, each word belonging to a mention for some type is tagged with that type's identifier. One weakness of this simple version is that in the case where two mentions of a certain type are contiguous, it is not possible to tell where one begins and the other ends - or even that there are two mentions rather than one. To alleviate this problem, and enrich the model, more complex tag sets are used. A model might have a tag for the beginning, inside and end of each type of entity. This enables the model to learn words that are more likely to begin and end mentions, effectively learning a little of the structure of the entities.

NER systems have also addressed the lexical composition of mentions for each type. For example, organization names often include the name of a person or a place (Wolinski et al., 1995). Even common nouns such as "associates" can indicate that a mention is of type organization (Wolinski et al., 1995).

In the biomedical domain, supervised CRFs have shown success at identifying genes and proteins. Using an annotated corpus from Medline abstracts, state of the art systems reach F-Scores of 78.4% for protien names (Tsai et al., 2006). Unfortunately, over a broader set of medical term types, basic linear chain CRF models cannot generalize beyond the terms in the training data (Zhang et al., 2010), with only a handful of new, correct medical terms identified from thousands of candidate new terms.

Unlike most NER systems our focus is on terms composed primarily of common nouns. While NER systems are trained on a labeled corpus, STR instead uses a term dictionary and an unlabeled corpus.

4.3 Bootstrapping Named Entity Recognition

Other work has identified and typed named entities based on an existing list of entities for each type. Usually, NER bootstrapping focuses on building a gazetteer of relatively common entities given a very small initial set of entities for each type. One approach is to use a small set (4) of example entities and search the web for documents that contain all of the example entities (Nadeau et al., 2006). The resulting documents are likely to contain lists or tables of the entities for the relevant type. By using an HTML context classifier trained with the context of the search entities as positive examples, other entities in the same HTML contexts may be extracted (for example "<td> x </td>"). The extracted entities are added to the list of known entities enabling additional search queries to be generated.

Mutual bootstrapping (Riloff and Jones, 1999) simultaneously learns entities of the selected type and patterns for identifying the entities. From a small set of seed entities, lexico-syntactic patterns are learned that suggest an entity of the appropriate type. For example, the noun phrase following "headquarted in" is likely to be a location. Extraction patterns are scored by their frequency and (estimated) reliability and entities are scored by the weighted sum of patterns that identify them. This scoring helps to alleviate the core problem of bootstrapping: semantic drift and also allows for a precision/recall trade-off.

A related bootstrapping approach (Kozareva, 2006) trains an NER classifier based on the current gazetteer then runs the classifier over text and adds the new terms recognized by the classifier to the gazetteer. This method requires a recognition system that goes beyond dictionary matching and uses context to a significant degree. The features used by the classifier include capitalization, trigger words specific to locations, organizations and people, and whether words in the noun phrase belong to a gazetteer for the type of interest.

Unlike traditional NER bootstrapping, STR assumes an existing large dictionary with hundreds if not thousands of examples per type. STR can identify the long tail of entities that may never occur in an easily interpretable list, table or lexico-syntactic pattern.

4.4 Joint Clustering

Other work has pursued different goals using related methods. The segmentation and de-duplication of citations has received considerable attention as a joint inference task. The

decisions of linking author, title, venue and citation records are all interdependent, since linking two citations implies that the fields composing the citations should also be linked. The task of segmenting the citations and identifying their fields is also influenced by the record linking. An easily segmented citation linked to a difficult case may make the difficult case much easier.

Using Markov Logic a joint entity linking model showed improvement in citation, author and venue linking over an independent model based on the field similarities alone (Singla and Domingos, 2006). Imperatively-Defined Factor Graphs (IDFs) produced still higher performance by incorporating features that could not be tractably expressed in factor graph systems that fully ground the network before inference (Singh et al., 2009).

5 Conclusion and Future Work

In this paper we presented a novel approach to semi-supervised term recognition. Unlike previous approaches we induced structure of the known terms to predict new terms. This enables type recognition for terms that appear only once in the corpus. We compared three different methods. All of them substancially improved over a most frequent baseline. We proposed two methods based on recognizing sequences of clusters. The one based on joint optimization significantly increased both precision and recall over the same method based on static clustering, reaching 90% precision at almost 30% recall. PCFG grammars allow us to achieve better recall (60%) with reasonably high precision (75%). In addition, PCFG allows us to recognize terms of any length.

The results shown by this paper are only partial as they do not take into account the role of context in disambiguating the types of terms. This limitation is intentional because: (i) we are interested in recognizing the type of answers independently of their context in cases when they are generated from databases or other non-textual material (ii) we are interested in exploring to what extent the internal structure of terminology can be used to express the semantics of terms.

In the future we will integrate STR with existing approaches of contextual, lexical and pattern-based term recognition. In addition, we will apply our technique to more fine grained type systems and other domains.

The type of structure learned is currently very simple, either a sequence of word clusters or a binary grammar. Other types of regularities exist, such as reordering. The pair of terms “leg pain” and “pain in leg” as well as many other similar pairs suggest a general alternation rule. We are going to explore this direction in order to learn transformational rules that can lead to identification of synonyms, hypernyms and to a better understanding on the underlying linguistic phenomena characterizing the generation and recognition of domain specific terminology.

Acknowledgments

The authors gratefully acknowledge the support of Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0172. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government. We would like to express our gratitude to the Watson Team and in particular to Chris Welty for inspiring leadership and David Gondek for helping us figuring out goals and technical solutions.

References

American College of Physicians (2012). Doctor's Dilemma^{TM}competition. `www.acponline.org/residents_fellows/competitions/doctors_dilemma`.

Aronson, A. R. (2001). Effective mapping of biomedical text to the umls metathesaurus: The metamap program. In *Proceedings of AMIA Symposium*, pages 17–21.

Bikel, D. M., Miller, S., Schwartz, R., and Weischedel, R. (1997). Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on Applied natural language processing*, ANLC '97, pages 194–201, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bodenreider, O. (2004). The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic acids research*, 32(Database issue):D267–D270.

Coates-Stephens, S. (1992). The analysis and acquisition of proper names for the understanding of free text. *Computers and the Humanities*, 26(5/6):pp. 441–456.

Dhillon, I. S., Fan, J., and Guan, Y. (2001). Efficient clustering of very large document collections. In R. Grossman, C. Kamath, V. K. and Namburu, R., editors, *Data Mining for Scientific and Engineering Applications*, pages 357–381. Kluwer Academic Publishers. Invited book chapter.

Grefenstette, G. (1994). *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers, Norwell, MA, USA.

Kozareva, Z. (2006). Bootstrapping named entity recognition with automatically generated gazetteer lists. In *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, EACL '06, pages 15–21, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lin, D. (1998). Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics - Volume 2*, COLING '98, pages 768–774, Stroudsburg, PA, USA. Association for Computational Linguistics.

Maynard, D. and Ananiadou, S. (2000). Identifying terms by their family and friends. In *Proceedings of COLING 2000*, pages 530–536.

McCallum, A. and Li, W. (2003). Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, pages 188–191, Stroudsburg, PA, USA. Association for Computational Linguistics.

McCallum, A., Schultz, K., and Singh, S. (2009). Factorie: Probabilistic programming via imperatively defined factor graphs. In *Neural Information Processing Systems Conference (NIPS)*.

Nadeau, D., Turney, P. D., and Matwin, S. (2006). Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In Lamontagne, L. and Marchand, M., editors, *Canadian Conference on AI*, volume 4013 of *Lecture Notes in Computer Science*, pages 266–277. Springer.

Nenadić, G., Spasić, I., and Ananiadou, S. (2002). Automatic discovery of term similarities using pattern mining. In *COLING-02 on COMPUTERM 2002: second international workshop on computational terminology - Volume 14*, COMPUTERM '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

Richardson, M. and Domingos, P. (2006). Markov Logic networks. *Machine Learning*, 62:107–136.

Riloff, E. and Jones, R. (1999). Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence*, AAAI '99/IAAI '99, pages 474–479, Menlo Park, CA, USA. American Association for Artificial Intelligence.

Schütze, H. (1993). Part-of-speech induction from scratch. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, ACL '93, pages 251–258, Stroudsburg, PA, USA. Association for Computational Linguistics.

Settles, B. (2004). Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, JNLPBA '04, pages 104–107, Stroudsburg, PA, USA. Association for Computational Linguistics.

Singh, S., Schultz, K., and Mccallum, A. (2009). Bi-directional joint inference for entity resolution and segmentation using imperatively-defined factor graphs. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*, ECML PKDD '09, pages 414–429, Berlin, Heidelberg. Springer-Verlag.

Singla, P. and Domingos, P. (2006). Entity resolution with markov logic. In *Data Mining, 2006. ICDM '06. Sixth International Conference on*, pages 572 –582.

Tsai, T.-h., Chou, W.-C., Wu, S.-H., Sung, T.-Y., Hsiang, J., and Hsu, W.-L. (2006). Integrating linguistic knowledge into a conditional random fieldframework to identify biomedical named entities. *Expert Syst. Appl.*, 30(1):117–128.

Wolinski, F., Vichot, F., and Dillet, B. (1995). Automatic processing of proper names in texts. In *Proceedings of the seventh conference on European chapter of the Association for Computational Linguistics*, EACL '95, pages 23–30, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Zhang, X., Song, Y., and Fang, A. C. (2010). How well conditional random fields can be used in novel term recognition. In *Proceedings of the 24th Pacific Asia Conference on Language, Information and Computation*, pages 583–592, Tohoku University, Sendai, Japan. Institute of Digital Enhancement of Cognitive Processing, Waseda University.