Recognition of Patient-Related Named Entities in Noisy Tele-Health Texts

MI-YOUNG KIM, YING XU, OSMAR R. ZAIANE, and RANDY GOEBEL,
University of Alberta, Canada

We explore methods for effectively extracting information from clinical narratives that are captured in a public health consulting phone service called HealthLink. Our research investigates the application of state-of-the-art natural language processing and machine learning to clinical narratives to extract information of interest. The currently available data consist of dialogues constructed by nurses while consulting patients by phone. Since the data are interviews transcribed by nurses during phone conversations, they include a significant volume and variety of noise. When we extract the patient-related information from the noisy data, we have to remove or correct at least two kinds of noise: *explicit noise*, which includes spelling errors, unfinished sentences, omission of sentence delimiters, and variants of terms, and *implicit noise*, which includes non-patient information and patient's untrustworthy information. To filter explicit noise, we propose our own biomedical term detection/normalization method: it resolves misspelling, term variations, and arbitrary abbreviation of terms by nurses. In detecting temporal terms, temperature, and other types of named entities (which show patients' personal information such as age and sex), we propose a bootstrapping-based pattern learning process to detect a variety of arbitrary variations of named entities. To address implicit noise, we propose a dependency path-based filtering method. The result of our denoising is the extraction of normalized patient information, and we visualize the named entities by constructing a graph that shows the relations between named entities. The objective of this knowledge discovery task is to identify associations between biomedical terms and to clearly expose the trends of patients' symptoms and concern; the experimental results show that we achieve reasonable performance with our noise reduction methods.

Categories and Subject Descriptors: I.2.7 [**Natural Language Processing**]: Text Analysis

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Tele-health mining, named entity recognition, biomedical text mining, effective information retrieval

ACM Reference Format:
Mi-Young Kim, Ying Xu, Osmar R. Zaiane, and Randy Goebel. 2015. Recognition of patient-related named entities in noisy tele-health texts. ACM Trans. Intell. Syst. Technol. 6, 4, Article 59 (July 2015), 23 pages.
DOI: http://dx.doi.org/10.1145/2651444

1. INTRODUCTION

Extraction of clinical information such as medications, symptoms, diseases, and patient's personal information from clinical text is an important task of clinical Natural Language Processing (NLP). Most clinical data consist of unstructured natural sentences, and there are few standard templates for the description format. Even existing

This work is supported by the Alberta Innovates Centre for Machine Learning (AICML) and the iCORE division of Alberta Innovates Technology Futures.
Authors' addresses: M.-Y. Kim, Y. Xu, O. R. Zaiane, and R. Goebel, Department of Computing Science, University of Alberta, Edmonton, AB Canada T6G 2E8; emails: {miyoung2, yx2, zaiane, rgoebel}@ualbeta.ca.

DOI: http://dx.doi.org/10.1145/2651444

"CHIEF COMPLAINT:
calling for 14 yr old with stomach pain and feels chils. wants to know the symptoms of swine
PRIORITY SYMPTOMS:
Alert. No difficulty breathing. No vomitng.
HISTORY/ASSESSMENT OF PRESENT ILLNESS OR DETAILS OF CC:
Writer talked to patient
What is child saying about abdominl pain? hurts mid section,below the belly button in the middle, feels really bad nausa
How long having abdominal pain? 30 min ago
Describe the nature of the pain (constant, intermittent, etc): constant
When did the child last stool? 3 hrs ago,normal
Underlying medical problems? no
Current medications? Nyquil
Allergies? Cats
DID CALLER AGREE WITH RECOMMENDATION:YES.rellevant emergent and 24 hrs symptoms reviewed with caller. Aware to call Link back with new symptoms or concerns (vomiting).Interim care given on all situations.Info on swine given"

Fig. 1. Example of a HealthLink call record.

templates are limited in scope, brittle in structure, and often adjusted so that calibration is difficult. It is also difficult to automatically extract that information which health care professionals believe is important to improve health care because there is much noise in most directly captured health data.

Our data come from captured tele-health dialogues in Alberta, Canada, from a publicly accessible system called HealthLink. With HealthLink, the public can access health advice and information by calling a phone line and discussing their complaints in real time with a registered nurse who simultaneously transcribes the conversation into text. The data are complex, highly heterogeneous, and generally not convenient for querying or extracting trends. HealthLink transcripts contain unstructured details regarding a patient's concerns, symptoms, personal information, previous diseases, and the like. An example is provided in Figure 1.

The development of a tool to parse patient records in order to automatically detect signs of a possible health issue would be a tremendous help for epidemiologists and other health professionals, and it could allow them to react more rapidly to a variety of trends. Recent advances in a variety of Artificial Intelligence (AI) NLP techniques, such as information extraction, named entity recognition, and factual assessment, support the development of such tools. As an integral part of Electronic Health Records (EHR), clinical notes pose special challenges for analyzing EHRs due to their unstructured nature and substantial noise because they are written by health practitioners in real time, while talking with patients.

The noise can be divided into two types: First is *explicit noise*, such as spelling errors, abbreviations, unspecified acronyms, unfinished sentences, term variants, and omission of sentence delimiters. The second is *implicit noise,* which is revealed only by a variety of inference methods: We have to figure out the intended meaning and linguistic structure of sentences to detect implicit noise. Written information that is not about a patient and untrustworthy information that may not be true are examples of implicit noise. The following are example sentences that show both types of noise. The spelling errors, such as "feve," "somone," and "concerend" were written by a nurse.

Example sentence: "Son has feve, and not feeling well, caller has flu like symptoms, he has been in contact with somone from Mexico. Mom concerend about H1N1."
→ **Explicit noise** : feve (fever)
→ **Implicit noise** : flu (disease not for the patient),

Mexico (travel history not for the patient),
H1N1 (virus which is not confirmed for the patient).

As part of our noise identification process, we have to detect biomedically named entities—such as disease, virus, drug, and symptom—as well as temporality, temperature, travel history, and other kinds of patient personal named entities (e.g., age and sex). Biomedical named entity recognition is the recognition of technical terms in the biomedical fields such as diseases, drugs, symptoms, and the like. Examples of biomedical named entity recognition systems include extracting clinical information from radiology reports [Fiszman et al. 2000], identifying diseases and drug names in discharge summaries [Uzuner et al. 2011], and detecting gene and protein mentions in biomedical paper abstracts [Yeh et al. 2005].

To identify explicit noise, we embed a misspelling correction module in our unsupervised language model-based biomedical term detection method. For temporality and other types of named entities, we set up seed patterns and run our own bootstrapping method: It detects variants of the seed patterns in the data using Damerau-Levenshtein distance [Damerau 1964]. To identify implicit noise, we use more detailed NLP method employing syntactic analysis and filter out untrustworthy information.

The contents of this article are as follows. In Section 2, we explain how to identify explicit noise. In Section 3, we describe our own method for identifying implicit noise by classifying named entities into facts, nonfacts, and concerns for a patient. Section 4 shows some experimental results, and related work is described in Section 5. Section 6 concludes with a summary and future work.

2. REMOVING EXPLICIT NOISE

2.1. Language Model-Based Biomedical Named Entity Recognition

Here, we describe our method to recognize biomedical terms such as symptom, disease, drug, virus, and the like. Because nurses may arbitrarily write different forms for the same term, we also perform normalization of terms. For literature mining in medical records, the medical ontology known as Unified Medical Language System (UMLS) [Dai et al. 2008] enables physicians to classify signs, symptoms, and diseases using accepted medical concepts. The UMLS integrates more than 2 million names for some 900,000 concepts from more than 60 families of biomedical vocabularies and includes 12 million relations among these concepts. Vocabularies integrated in the UMLS Meta-thesaurus include the National Centre for Biotechnology Information (NCBI) taxonomy, Gene Ontology, the Medical Subject Headings (MeSH), Online Mendelian Inheritance in Man (OMIM), and the Digital Anatomist Symbolic Knowledge Base. Our hypothesis is that, combined with an integrated information retrieval method, the UMLS is a powerful and appropriate tool to use as the basis for automatically mapping biomedical names with variant forms into one concept. UMLS contains more than a million medical and general English concepts, which are further organized under a hierarchy of 134 semantic types. Each concept is assigned one or more high-level semantic types. More specifically, we only keep concepts belonging to the following semantic types as biomedical concepts: *disease or syndrome, finding, sign or symptom, virus, pharmacologic*, and we exclude other concepts because those are not relevant to biomedical terms (e.g., animal, plant, chemical). We chose these five concepts as relevant to the biomedical concepts according to the example entities in UMLS.

A central architectural aspect of our processing of medical documents is based on treating sentences within those documents as queries and UMLS entries as documents. In this Information Retrieval (IR) model, we infer a language model for each UMLS concept entry and rank each related entry according to how likely it generates the input sentence based on its language model. However, our model is different from

traditional IR models such as language model-based IR [Ponte and Croft 1998] and BM25 [Robertson Walker 1994] in three ways. First, the characteristics of a query and a document in our experiments are different from those in traditional IR because a query is longer than a document in our model, and the frequency of most words in a document is uniform.

Second, our purpose is to retrieve all and only matched biomedical concepts for a query sentence whereas the purpose of traditional document retrieval is to retrieve a list of documents ranked by their relevance to a query.

In traditional IR formulation, for a word w that is included in a document but not included in a query, one assigns a penalty by computing the probability that the language model does not generate w based on the term frequency.

There is some risk of assigning a penalty based on only term frequency in a document for words that have different information content as measured by document frequency. To assign a small penalty to those words that are common to the domain but do not have a large amount of information content (e.g., the words "disease," "disorder," and "symptom"), we add a document frequency measure. The document frequency of a word helps determine its information content: The smaller the document frequency of a word, the bigger information content it has.

Finally, our purpose is to detect medical terms in a sentence, but medical terms are often formulated with multiple words. Traditional IR does not consider distance information. In term detection, the distance between words of a multiword term t provides a clue on the likelihood that those words belong to a common term. Therefore, we add a distance measure. In the following subsections, we will explain our method in detail.

2.1.1. Information Retrieval Based on a Language Model. We consider each input sentence as a query and UMLS entries as documents. Then, we would like to estimate $\hat{p}(Q \mid M_d)$, the probability of the query Q given the language model of document d as follows:

$$\hat{p}(Q \mid M_d) = \prod_{w \in Q} \hat{p}(w \mid M_d) \times \prod_{w \notin Q} (1.0 - \hat{p}(w \mid M_d)).$$

The first term is the probability of generating words in the query, and the second term is the probability of not generating other terms. The specific probabilities for $\hat{p}(Q \mid M_d)$ are defined as follows:

$$\hat{P}(w \mid M_d) = \begin{cases} \hat{p}_{ml}(w,d)^{(1.0-\hat{R}_{w,d})} \times \hat{p}_{avg}(w)^{\hat{R}_{w,d}} & \text{if } wf_{(w,d)} > 0 \\ \dfrac{c_w t}{cs} & \text{otherwise} \end{cases},$$

$$\hat{R}_{w,d} = \left(\frac{1.0}{(1.0 + \bar{f}_w)}\right) \times \left(\frac{\bar{f}_w}{(1.0 + \bar{f}_w)}\right)^{wf_{(w,d)}},$$

$$\hat{p}_{ml}(w,d) = \frac{wf_{(w,d)}}{dl_d},$$

$$\hat{p}_{avg}(w) = \frac{\left(\sum_{d(w \in d)} \hat{p}_{ml}(w,d)\right)}{df_w}.$$

where $\hat{p}_{ml}(w,d)$ is the maximum likelihood estimate of the probability of term w under the term distribution of document d, where $wf_{(w,d)}$ is the raw term frequency of term w in document d, and dl_d is the total number of tokens in document d. We use Damerau-Levenshtein distance to identify explicit noise such as misspellings and arbitrary abbreviations. When we compute term frequency, we include the term variants of which the Damerau-Levenshtein distance is less than a threshold. We constructed a

small development set and determined the threshold number on the set, which is the maximum Damerau-Levenshtein distance between term variants.

By using the Damerau-Levenshtein distance measure, we compute term frequency $wf(w, d)$ as follows:

$$wf(w, d) = \sum_{t \in DL_w} count(t, d) \times \left(1 - \frac{DL\_dist(t, w)}{length(w)}\right),$$

where DL_w is a group of a variant t for word w where $DL\_dist(t, w) \leq$ threshold. $DL\_dist(t, w)$ is a Damerau-Levenshtein distance between t and w. $count(t, d)$ is the count of word t in document d. We add up the counts of all variants of w in document d after assigning a penalty based on Damerau-Levenshtein distance.

$c_w t/cs$ is the background probability for the document that is missing one or more of the query terms since we do not want to assign 0 for $\hat{p}(w|M_d)$ of this document, where $c_w t$ is the raw count of term w in the collection and cs is the total number of tokens in the collection. $\hat{p}_{avg}(w)$ is the estimate of the probability of the word w from a larger volume of data. $\hat{R}_{w,d}$ is a risk function based on a geometric distribution, selected to benefit from the robustness of the estimator $\hat{p}_{avg}(w)$ and to minimize the risk of using the estimator. $\bar{f}_w$ is the mean term frequency of term w in documents where w occurs. For more details on each probability, see Ponte and Croft [1998].

2.1.2. Penalty Information for Domain-Specific Common Terms. In the IR formulation of Section 2.1.1, for the word w that is included in a document but not included in a query, we assign a penalty by computing the probability that M_d does not generate w based on the term frequency. However, in our task, the term frequency of most terms in a document is uniform. So, there is some risk of assigning a probability measured by only term frequency for words with different information content. To assign a small penalty to the domain-specific common words, we add a document frequency measure, which is:

$$DF(d, Q) = \prod_{w \text{ in } d, w \notin Q} \frac{df(w)}{|D|},$$

where $df(w)$ is the frequency of documents that contain w, and $|D|$ is the number of all documents.

2.1.3. Distance Information. We need to consider distance among words to detect whether the words in a query indicate one common term. We modify the distance measure of Gaudan et al. [2008], as follows:

$$dist(d, Q) = \frac{current\_dist(d, Q)}{min\_dist(d, Q)}.$$

Let W be the set of words of a document d found by a query sentence Q. Then, W is:

$$W = tok(Q) \cap tok(d),$$

where n is the number of words in W. Then,

$$min\_dist(d, Q) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} |i - j|,$$

$$current\_dist(d, Q) = \sum_{w_i \in W, w_j \in W} |position(w_i, Q) - position(w_j, Q)|,$$

where $position(w_i, Q)$ is the position index of w_i in the query sentence Q.

The $p(Q \mid M_d)$ (probability of generating Q based on the language model M_d), $DF(d, Q)$ (document frequency measure), and $dist(d, Q)$ (distance measure) are three factors that are combined to score the mention of d in a query Q. The three criteria may be of various importance and must be weighted accordingly. Finally, the three criteria are combined by the product of the functions, and the integrated formula is:

$$score(Q, d) = \hat{p}(Q \mid M_d) \times DF(d, Q)^{\theta 1} \times \left\{ \frac{1}{dist(d, Q)} \right\}^{\theta 2},$$

where document frequency and distance measure are weighted by the parameter θ_1, θ_2. How to estimate each parameter is explained in Section 2.1.4.

2.1.4. Parameter Estimation. Since our approach is unsupervised, we need to set a loss function $f(\theta)$ from the retrieved results, and we estimate parameter θ through iterative scaling that minimizes the difference between loss functions $|f_{t-1}(\theta) - f_t(\theta)|$. We set $f(\theta)$ as the sum of frequencies of the words that are included in the relevant top-K ranked documents but not included in the query. The intuition is that the smaller $f(\theta)$ becomes, the better performance we have. In other words, $f(\theta)$ is:

$$f(\theta) = \sum_{m}^{|U|} \sum_{i}^{K} \sum_{w \notin U_m} tf(w, doc(\theta, i, U_m)),$$

where $tf(w, doc(\theta, i, U_m))$ is term frequency of w in the document $doc(\theta, i, U_m)$ which is retrieved with the ith rank for the m$^{\text{th}}$ query U_m using parameter θ. In our experiments, K is 10 and $|U|$ is the number of input query sentences. We update the parameter using

$$\theta_{n+1,i} = \theta_{n,i} + \eta_{n,i} \left\{ f_{n-1}(\theta) - f_n(\theta) \right\},$$

where $f_0(\theta) = \sum_{m}^{|U|} \sum_{1}^{K} \sum_{w \notin U_m} 1$, and $\{\theta_{0,1}, \theta_{0,2}\} = \{1, 1\}$.

The initial value $f_0(\theta)$ means that the frequency of the words that are not included in the query in each document is 1, and we assign 1 for the initial parameter values.

For the update of parameter value θ_1 of the document frequency measure, we set $\eta_{n,1}$ to be proportional to $(avg\_df_n)/(avg\_df_{n-1})$. avg_df_n is the average document frequency of a word that is included in the top-K ranked documents but is not included in the query at the n^{th} iteration. We perform an update of θ_1 using the following $\eta_{n,1}$:

$$\eta_{n,1} = \alpha \times \frac{avg\_df_n}{avg\_df_{n-1}},$$

$$avg\_df_n = \frac{\sum_{m}^{|U|} \sum_{i}^{K} \sum_{w \text{ in } d_{n,i}, w \notin U_m} df(w)}{\sum_{m}^{|U|} \sum_{i}^{K} \sum_{w \text{ in } d_{n,i}, w \notin U_m} 1}.$$

Similarly, for the update of parameter value θ_2 of the distance measure, we set $\eta_{n,2}$ to be proportional to $(avg\_dist_{n-1})/(avg\_dist_n)$. avg_dist_n is the average distance of words per document in n^{th} iteration. We perform an update of θ_2 using the following $\eta_{n,2}$:

$$\eta_{n,2} = \beta \times \frac{avg\_dist_{n-1}}{avg\_dist_n},$$

$$avg\_dist_n = \frac{\sum_{m}^{|U|} \sum_{i}^{K} dist(d_{n,i}, U_m)}{\left\{ \sum_{m}^{|U|} \sum_{i}^{K} 1 \right\}}.$$

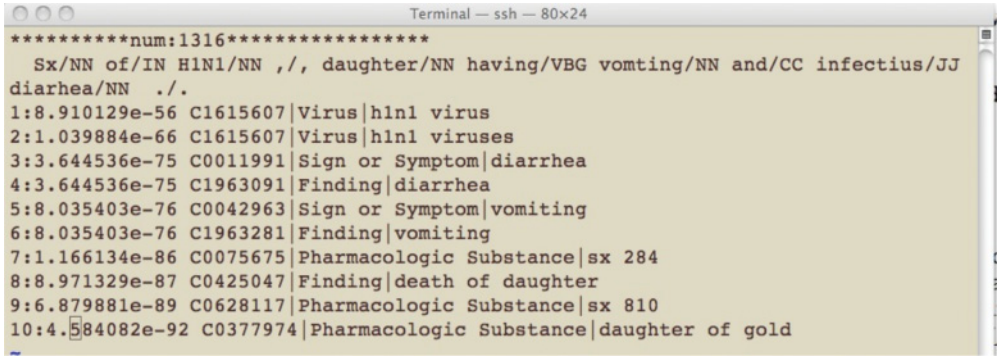

Fig. 2. Top-10 biomedical term candidates example.

HAC(d_1, ..., d_N)
1 **for** $n \leftarrow 1$ to N
2 **do for** $i \leftarrow 1$ to N
3 **do** $C[n][i] \leftarrow \text{SIM}(d_n, d_i)$
4 $I[n] \leftarrow 1$ *(keeps track of active clusters)*
5 A $\leftarrow$ [ ] *(assembles clustering as a sequence of merges)*
6 **for** $k \leftarrow 1$ to N-1
7 **do** $\langle i,m \rangle \leftarrow \text{argmax}_{\{\langle i,m \rangle : i \neq m \wedge I[i]=1 \wedge I[m]=1\}} C[i][m]$
8 **if** $C[i][m] = 0$
9 return A
10 A.APPEND($\langle i,m \rangle$) *(store merge)*
11 $I[m] \leftarrow 0$ *(deactivate cluster)*
12 return A

where $\text{SIM}(A,B) = \text{cosine\_similarity}(A,B)$ *if* $((A \cdot B) = \{\min(\|A\|, \|B\|)\}^2)$,
$\text{SIM}(A,B) = 0$, otherwise;

Fig. 3. Modified HAC algorithm for our task.

The α and ß are constants that control the convergence speed of the iterations. In our experiments, we set α and β to 0.01 and the initial values of $\{\eta_{n,1}\eta_{n,2}\}$ to {0.01,0.01}. We stop iterating when $|f_{n-1}(\theta) - f_n(\theta)| \leq 10$. The threshold 10 was determined on a small development set.

We obtain ranked relevant concepts according to the integrated IR model. Figure 2 shows one retrieval example using this model.

2.2. Step 2: Clustering Retrieved Concepts

We assume there is only one concept ID corresponding to a medical term. But since it is typical that more than one concept is retrieved for each medical term mentioned in a sentence, we need to cluster the concepts according to their shared words.

To do so, we apply a Hierarchical Agglomerative Clustering (HAC) algorithm that is commonly used for document clustering [Willet 1988] and that does not require a prespecified number of clusters. This algorithm begins with each document as a cluster of its own (Lines 1–4 in Figure 3), iterates by merging the two most similar clusters (Lines 6–7), and then terminates when there are no more non-overlapping sets to merge (Lines 8–9). This HAC algorithm requires the definition of a similarity function

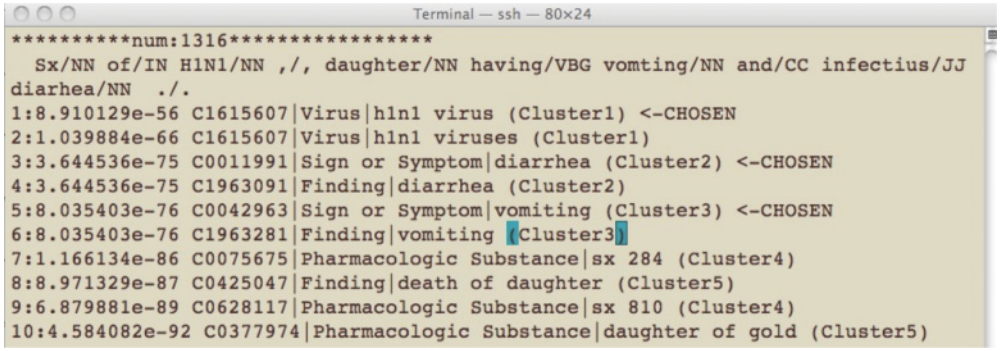

Fig. 4. Example of clustering and retrieved examples.

between documents and between sets of documents. Each document (UMLS concept entry) is represented as an attribute vector, with each word in the input sentence being an attribute in this vector. If a word in the input sentence occurs in a concept entry, the corresponding attribute value of the vector is 1. Otherwise, it is 0.

The similarity of two documents is often taken as a normalized function of the dot product of their attribute vectors.

The HAC algorithm that we use is shown in Figure 3. This algorithm groups the two most similar clusters at each iteration and then recalculates the similarity between clusters. It terminates if the iteration is performed N (number of documents) $-$ 1 times or the maximum similarity between clusters becomes 0. We use cosine similarity between vectors as a similarity measure. We assume that there is a hierarchical relation between concept entries A and B only if all the common words between A and the input sentence occur in the entry B. When A and B are represented as vectors, this property can be described as follows: There is a hierarchical relation between concept entries A and B only if $(A \cdot B)$ equals to $\{\min(\|A\|, \|B\|)\}^2$). Finally, we assign the similarity value as shown in Figure 3.

A clustering example using the top K-ranked list (K = 10) is shown in Figure 4.

2.3. Step 3: Choosing Answer Concepts from Clusters

Once the clustering has been completed, we select a concept that shows the highest rank in each cluster and is within a given threshold. We select the threshold dynamically based on the ranking score distribution, specifically choosing the point at which there is a significant drop in ranking scores, which means the ratio of score[i]/score[i+1] is biggest.

Figure 4 shows clustering results for our example and the selected biomedical concept IDs. In this case, five clusters are constructed according to the HAC algorithm. If we choose a concept that shows the highest rank in each cluster, then we get "h1n1 virus," "diarrhea," and "vomiting." Even though there were misspellings such as "vomting," and "diarhea" in the input query, we obtained the correct terms. The 7th–10th results are dropped because, at this point, there is a significant difference in the ranking scores.

2.4. Recognition of Nonbiomedical Named Entities Based on Bootstrapping

Temporality, temperature, location, and other personal named entities—such as age, and sex—also have various surface forms including misspellings and arbitrary abbreviations. Given these considerations, we address the following question: How can

the named entities having arbitrarily different surface forms be automatically learned from the data with minimal effort using lexical and part-of-speech (POS) patterns?

Ling and Weld [2010] detect temporal named entities using the semantic role labeling tool of Koomen; however, this tool is not publicly available. Wang et al. [2010] added the concept of temporal fact, in addition to the YAGO ontology using regular expression, to extract temporal information from Wikipedia Infobox. Li and Patrick [2012] use surrounding word features to extract temporal information. They show that the contexts that surround the temporal expressions are not sufficient because the same temporal expressions can be mentioned in various contexts.

Previous work on temporal information detection manually constructs regular expressions for the anticipated patterns. We want to avoid human labor to collect all variations of patterns, so we propose a bootstrapping algorithm with a single seed and nonrecursive lexical pattern learning using the WordNet lexicographic dictionary.

Many successful methods have used an unsupervised iterative bootstrapping framework [Riloff and Shepherd 1997]. Bootstrapping has since been effectively applied to extract general semantic lexicons [Riloff and Jones 1999] and facts [Carlson et al. 2010]. This kind of bootstrapping is considered to be minimally supervised because it is initialized with a small set of seed terms of the target category to extract. These seeds are used to identify patterns that can match the target category, which in turn can be used to extract new patterns [McIntosh 2010].

Starting from the original seed, each new pattern produced by the Damerau-Levenshtein distance algorithm can be considered an input seed for another instance of the algorithm. This procedure can be iterated over all the new patterns. This approach increases the number of retrieved patterns but can create unwanted noise. At each bootstrapping step, the produced patterns can diverge from the original seed. We tackle this problem by introducing a stop criterion in the bootstrapping framework whose goal is to select only those new patterns that are semantically similar to the original seed. Our measure of semantic similarity between a surface pattern and the seed is approximated by using Damerau-Levenshtein distance, the WordNet dictionary, and POS tagging. We assume that the first obtained new patterns produced by the expansion of the original seed are the most semantically similar. Therefore, we stop after one iteration. The stop criterion reduces the number of computations and guarantees a semantic similarity between the original seed and the new patterns. The final output of the bootstrapping process is the union, without duplicates, of all the new patterns that are evaluated as correct by the stop criterion.

The more specific description of our method is in the following: There are many types of temporal words (e.g., "this morning," "last Wednesday," "April 24th," etc.). Since manually constructing seed patterns cannot cover all types, we use WordNet to retrieve all words related to time. We collect all the words of which the semantic category is <noun.time> from WordNet, and then we annotate the words in our data as "temporal noun (TEM)" if they are included in the <noun.time> category. There are 930 words in the <noun.time> category in WordNet 3.0. We regard each temporal noun as a seed. The examples of seed patterns are "April," "evening," "Friday," "minute," "yesterday," "year," and so on. We input each seed s to the bootstrapping algorithm and get the output of the seed variants s' if Damerau-Levenshtein distance(s, s$'$) does not exceed a threshold. We set the threshold value as 2, except for the short words; where the length is the same as or less than 4, we set the threshold value as 1, according to the development set.

We have to check if new patterns are semantically similar to the original seed. To do that, we use WordNet dictionary and POS tagging. If an obtained pattern word occurs in the WordNet dictionary with a different meaning or has a different POS tag from that of the seed word, then we consider that the new pattern has a different meaning,

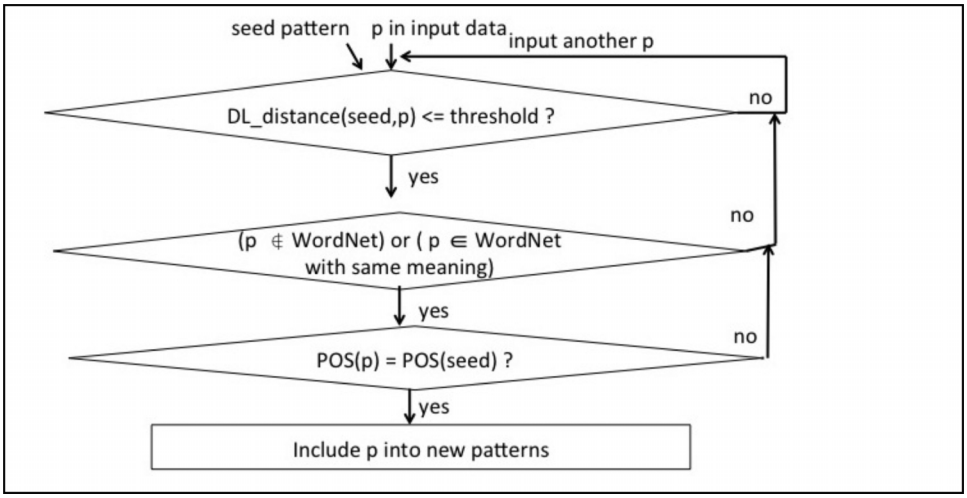

Fig. 5. Bootstrapping algorithm.

1. Temporality
   ($NUM)* ($temporal_noun)$^{+}$ ($NUM)*
   ($determiner) ($temporal_noun)$^{+}$
   [when|time] [$any_word]* ($NUM)
2. Age
   [age] [$any_word]* ($NUM) (year|month)*
   ($NUM) (year|month) (old)
3. Temperature
   [temperature|fever] [$any_word]* ($NUM)
   where 35<=($NUM)<=41 or 92<=($NUM)<=106
4. Sex
   Choose the more frequent category between 'woman' and 'man' category words
5. Location
   ($location_noun)

Fig. 6. Regular expressions.

and we filter such words from the obtained pattern set. We use the POS-tag results of the Stanford parser (http://nlp.stanford.edu/software/lex-parser.shtml). For example, we can obtain the following new variations by running bootstrapping:

days -->says, april -->advil, year-->ear,
today-->body, today-->okay.

We remove these from the obtained new patterns for two reasons. First, for the cases of "advil," "ear," "body," and "okay," each meaning is different from its original word according to the WordNet dictionary. The words of the same meaning share the same database location number in WordNet. Second, for the case of "says," the word is not shown in the WordNet dictionary, but the POS tag of the Stanford parser for "says(verb)" is different from "days(noun)." This process of bootstrapping is shown in Figure 5.

We extract temporal NP, temperature, sex, age, and location using the regular expressions of Figure 6. In the Figure, [ ] surrounds context information and ( ) is the

corresponding named entity. For the age, we use the pattern 'age+num' and 'num-year-old.' For the temperature, there should be words such as "temperature" and "fever" in the surrounding context within the sentence, and we also limit the boundary of numbers between 35 and 41, or between 92 and 106, which indicates the boundary of human fever as Centigrade and Fahrenheit, respectively. For the sex, we choose the more frequently appearing category between "man" and "woman" in the data. For the pronouns, we simply used "he," "him," and "his" as the male information, and "she" and "her" as the female information. We can also get the noun's sex information from the definition of the noun in WordNet using the words "male" and "female." For travel information, we use the city information of the <noun.location> category of WordNet.

The following are examples of patterns obtained from the regular expressions.

<Age>19 yr old, three year old, 6 yrs old, aged 4 onth,
<temporality> this orning, three day, 10days, tonigt,
<sex> husband (male)
<location> Mexico
<temperature> 39c

Even though these examples include spelling errors ("orning," "onth," "tonigt," "39c"), abbreviations ("yr," "yrs"), grammatical errors ("year," "day"), and space error ("10days"), our regular expressions successfully obtained all the patterns. We also use the abbreviation forms provided by WordNet such as Mon (Monday), Sat (Saturday), eve (evening), Mar (March), yr (year), and the like. Since an abbreviated term is already a variation of the standard word, we do not use the bootstrapping algorithm for confirmed abbreviated terms.

3. REMOVING IMPLICIT NOISE

We want to extract facts on patients by removing implicit noise. We need to do two tasks: Extract the information only for patients, not for any other person mentioned in text such as friends, or family members, and identify facts from the patients' information. Details are given in the following sections.

3.1. Extracting Information Only for Patients

We need to know the subject of each named entity to distinguish and remove named entities not associated with a particular patient. Based on the syntactic analysis, we determine the subject of a named entity and then filter it if the subject is not the patient. The following is an example:

Example. Fiance works with someone who returned from mexico with symptoms. fiance has symptoms of fever, cough and sore throat . . . . . .

We perform syntactic analysis using the Stanford parser and detect the subject role of each named entity. To improve the syntactic analysis results, we replace all misspelled words with the corrected results according to Section 2. Figure 7 shows the syntactic dependency result of the first sentence in this example. We make the following two assumptions:

1. The patient is the most frequent person noun.
2. If the parser does not explicitly indicate the subject of a named entity, then we regard the nearest person noun or pronoun as the subject.

Person noun means the noun that has a semantic category <noun.person> in WordNet. The subject of "Mexico" in Figure 7 becomes "someone" since "someone" is nearest person noun from "Mexico," and we know the patient is "fiancé" because it is the

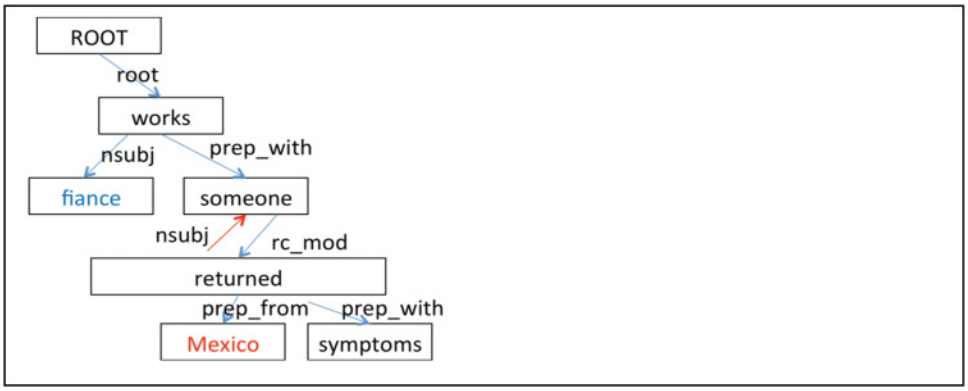

Fig. 7. Syntactic dependency example.

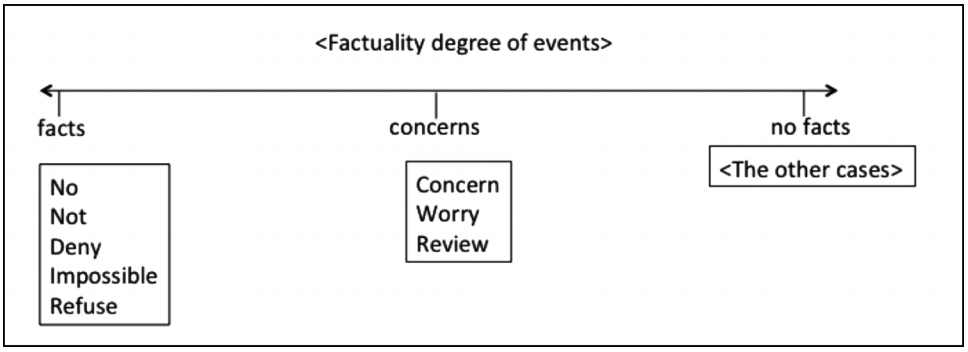

Fig. 8. Heuristic patterns for factuality degree of events.

most frequent person noun in this example paragraph. Since "fiancé" is different from "someone," we remove Mexico from the patient-related named entities.

3.2. Removing Untrustworthy Information of Patients

Identifying the factuality status of mentioned events is fundamental for reasoning about events in discourse [Sauri and Pustejovsky 2012]. Inferences derived from events judged as not having happened or as being only possible are different from those derived from events evaluated as factual. Karttunen and Zaenen [2005] discuss its relevance for information extraction. In the area of textual entailment, factuality-related information (modality, intensional contexts, etc.) has been considered a basic feature. The need for this type of information is also acknowledged in the annotation schemes of corpora devoted to event information, such as the ACE corpus for the Event and Relation recognition task (e.g., ACE 2008) or TimeBank, a corpus annotated with event and temporal information [Pustejovsky et al. 2006].

Significantly, in the past few years this kind of information has been the focus of much NLP research dedicated to the biomedical domain. Factuality-related information was the main focus in the CoNLL-2010 shared task on Learning to Detect Hedges and their Scope in Natural Language Text [Farkas et al. 2010]. The topic in a subtask of the BioNLP '09 and BioNLP '11 shared task editions on Event Extraction [Kim et al. 2009] was dedicated to predict whether the biological event is under negation or speculation.

We assess the factuality degree of events (whether they correspond to facts, counter-facts, or only concerns) based on heuristic patterns as shown in Figure 8. We detect

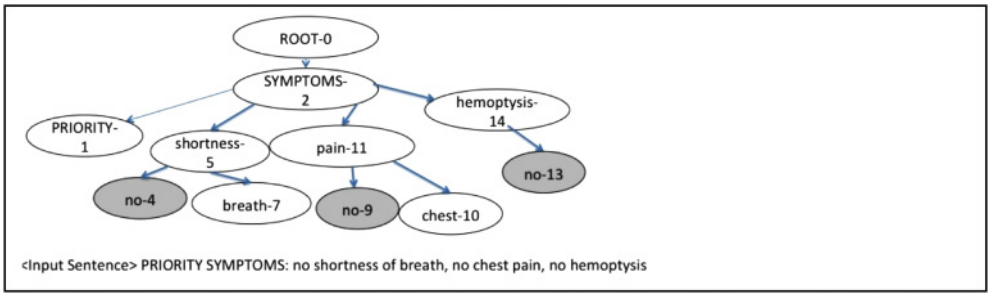

Fig. 9. Example parse tree 1 for negation.

polarity based on the negation words such as "no," "deny," "not," "impossible," and "refuse." The scope of polarity is determined based on syntactic analysis.

As an example, note that the presence of a medical term in a clinical note does not necessarily imply its presence in a patient. Our negation annotator looks at the surrounding text of each medical term annotation and filters term mentions appearing in negated contexts based on simple heuristics, such as the presence of negation-related words just noted. In principle, negative medical term findings can also indicate interesting relationships. However, for this work, we focus only on positive medical term occurrences. For the detection of "concerns," we simply use the following word patterns: "concern," "worry," and "review." We regard obtained named entities as "concerns" if their surrounding context words include these word patterns. We collect negation words and then prune negations based on the negation words to determine the boundaries of negation based on parsing information. Negation words can either be adjectives or verbs. The two cases show different kinds of syntactic graphs, and we need different rules for each case.

First, if the POS-tag of a negation word is an adjective, its main syntactic function is to modify the following noun. Therefore, the boundary of negation includes its governor, which is the following noun, and all the children/descendant nodes of the governor. For example, for the sentence "PRIORITY SYMPTOMS: no shortness of breath, no chest pain, no hemoptysis-14.," the Stanford parsing result is shown in Figure 9. This figure includes an adjective "no" as a negation word. In the figure, the governor of "no-4" is "shortness-5." The governor "shortness-5" and all the children/descendant nodes of the governor are included in the boundary of negation word "no-4." In a similar way, "pain-11" and all children/descendant nodes are included in the boundary of negation word "no-9." For the negation word "no-13," only "hemoptysis" is included in the negation boundary.

Second, if the negation word is a verb, its main function is to govern its dependents. In this case, we just include its children/descendant nodes within a negation boundary. Figure 10 is the parse tree of the sentence "PRIORITY SYMPTOMS: denies respiratory distress, denies chest pain, denies fever.." The example in Figure 10 shows how the verb "deny" is used as a negation word. Since the negation word is a verb, all children/descendant nodes are included within the boundary of negation. In this figure, the negation boundary of "denies-4" includes "distress-6" and "respiratory-5," and the negation boundary of "denies-8" includes "pain-10" and "chest-9." For the negation word "denies-12," only "fever-13" is included.

4. EXPERIMENTAL RESULTS AND DISCUSSION

4.1. Performance of Our System

We constructed a small development set consisting of 100 sentences, and the set was used to determine threshold values in our experiments. We evaluated our system's

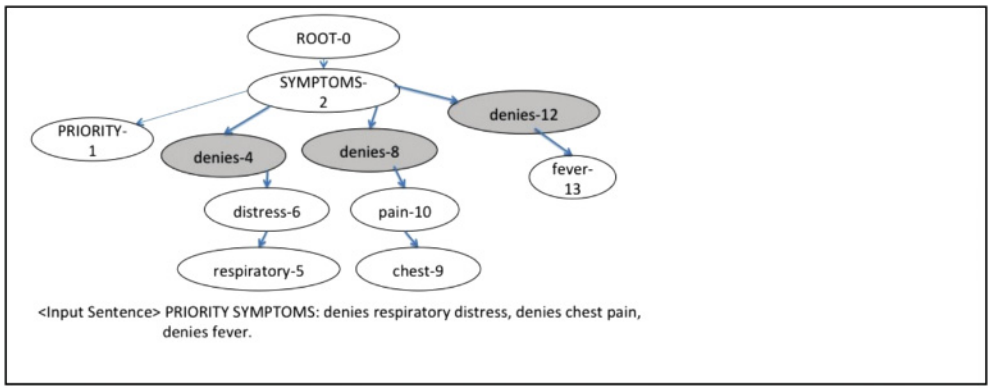

Fig. 10. Example parse tree 2 for negation.

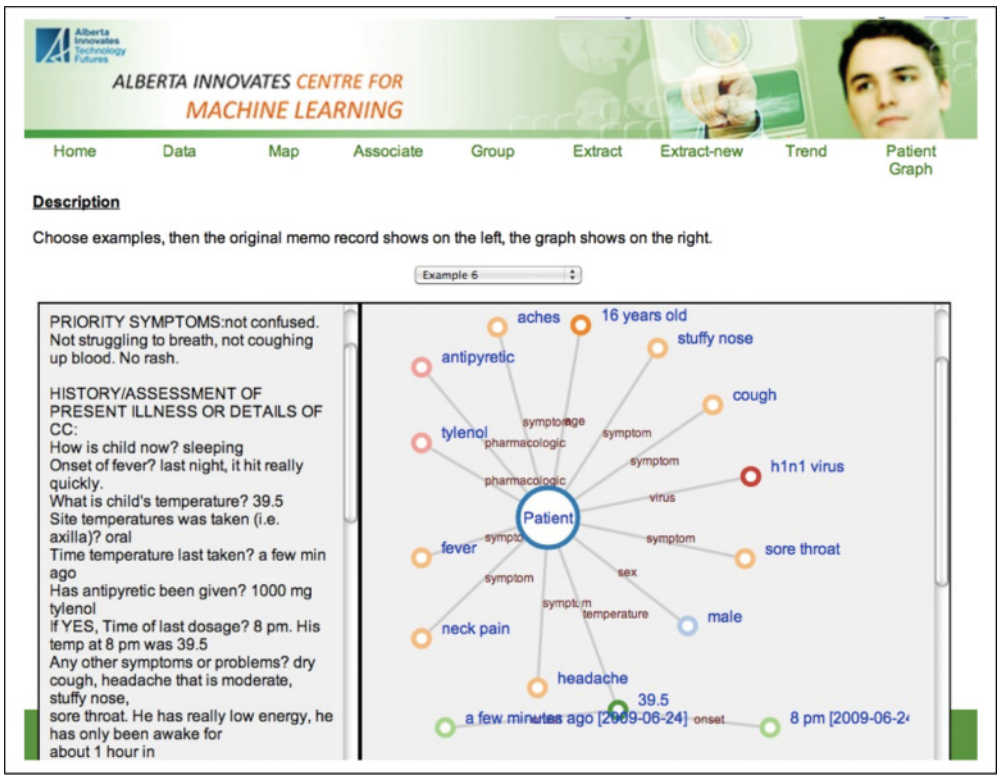

Fig. 11. Visualization example of Tele-health data.

performance on our HealthLink data, which consists of 5,297 sentences for 300 patients and average 7.42 words/sentence. The anonymized data were provided by Alberta Health Services. We annotated the data with 10 named entities such as "disease," "finding," "symptom," "virus," "pharmacologic," "temporality (onset)," "age," "sex," "temperature," and "travel." We can visualize the HealthLink data as shown in Figure 11. In the graph, a node name indicates a detected named entity term, and an edge tag indicates the type of the corresponding named entity. We linked a temporal named

Table I. Performance of Our System

	Precision	Recall	F-measure
Biomedical term detection	0.6820	0.8929	0.7733
Temporality	0.8905	0.9035	0.8970
Age	0.9304	0.8602	0.8939
Sex	0.9484	0.8743	0.9098
Temperature	0.9203	0.9038	0.9120
Travel info.	0.7392	0.9553	0.8335
Factuality Assessment	0.9520	0.8930	0.9216

Table II. Comparison of Our Biomedical Term Detection System with Others

	Precision	Recall	F-measure
Our Biomedical term detection in EBI [Jimeno et al. 2008] data	0.7303	0.7769	0.7529
Performance of EBI's statistical method	0.6617	0.6710	0.6663
Performance of EBI's dictionary lookup method	0.7940	0.6006	0.6839
MetaMap	0.8390	0.5357	0.6539

Table III. Comparison of Our Temporality System with Others

	Precision	Recall	F-measure
Our temporality method	0.8905	0.9035	0.8970
HeldelTime	0.8726	0.7538	0.8089
SUTIME	0.9463	0.7792	0.8547

entity node with the related medical named entity node that occurred nearest in the same sentence. We obtained the following experimental results:

1. Our proposed method in biomedical term detection achieved 77.33% in F-measure (see Table I).
2. Our method significantly outperformed MetaMap [Aronson 2001] by 9.9% and EBI's system [Jimeno et al. 2008] by 8.6% (see Table II).
3. Temporality achieved 89.70% in terms of F-measure, and other named entity detection approaches also showed reasonable performance (see Tables I and III).
4. Factuality for patients showed the F-measure of 92.16% (see Table I).
5. We can see the convergence of the precision/recall points for higher ranks (see Figure 12).

Figure 12 shows each precision/recall of our system when we retrieve top K-ranked biomedical entries in Step 1 of Section 2.1, with K = 1 to 10. We can note the convergence of the precision/recall points for higher ranks, and we see that K = 10 is enough for the convergence of precision/recall. In all the experimental results of this paper, we set K = 10.

We measure the performance of our system based on precision and recall as follows: Precision = (the number of correctly detected terms)/(the number of all detected terms), and Recall = (the number of correctly detected terms)/(the total number of existing terms in the data). F-measure is the harmonic mean of precision and recall and is computed by (2∗precision∗recall)/(precision+recall).

As shown in Table I, among those systems using the same evaluation data, our system outperformed all previous reported systems with a precision of 68.20%, recall of 89.29%,

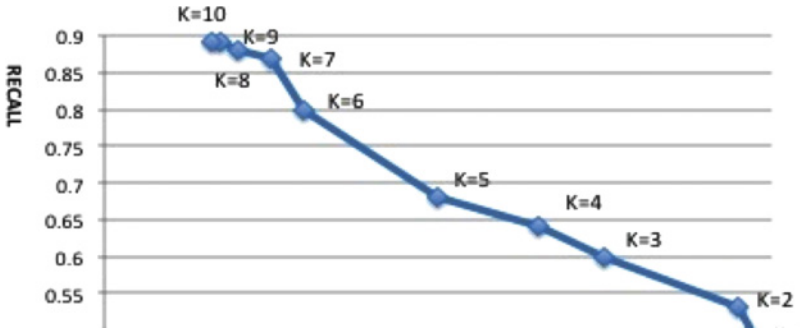

Fig. 12. Precision/recall when top-K ranked biomedical entries are retrieved.

and F-measure of 77.33%. We compare our method with MetaMap [Aronson 2001] (http://metamap.nlm.nih.gov/), a tool developed at the National Library of Medicine, for mapping raw English text to standardized medical concepts in the UMLS metathesaurus, and EBI (European Bioinformatics Institute)'s system [Jimeno et al. 2008]. In our experiments of Table II, all the systems used the data that EBI provided, and the EBI's system and MetaMap program trained and tested the same set of data. The precision, recall, and F-measure represent proportions of populations. In trying to determine the difference in performance of two systems, we employ the z-test on two proportions. We test the significance of differences in F-measures among three kinds of system pairs: {our system, EBI's statistical system}, {our system, EBI's dictionary-based system}, and {our system, MetaMap}. Given two system outputs, the null hypothesis is that there is no difference between the two proportions (i.e., H0: p1 = p2). The alternative hypothesis states that there is a difference between the two proportions (i.e., p1 =/ p2). A z-statistic of ± 1.96 means that the difference between the two proportions is significant at $\alpha = 0.05$. Z-values in all three significance tests are bigger than 2.58, and this shows that the null hypothesis of no difference in the two proportions is rejected.

We also compared our performance of temporality extraction with the methods of two previous systems: HeidelTime [Strötgen and Gertz 2010] and SUTIME [Chang and Manning 2012]. HeidelTime is the best performing system from SemEval-2, and SUTIME is a rule-based system that outperformed HeidelTime. Even though SUTIME outperformed HeidelTime, as shown in Table III, it did not outperform our system.

Of course, our performance is partly explained because our system is optimized for our data and the other two systems are not. In addition, in detecting temporality, both previous systems show lower recall than ours because our data have many variations and misspelled words. Those previous systems do not have rules to deal with those arbitrary variations.

4.2. Significance of Each Step

We now summarize the significance of each step introduced in Section 2.1. As shown in Table IV, each step has a significant impact on the system's performance based on a z-test at alpha = 0.05.

Without using Damerau-Levenshtein distance, the recall was significantly reduced because we missed all named entities with explicit noise. Without using the document frequency and distance measures for the model in Step 1, the recall is significantly reduced because the retrieved concept entries can include words that have high information content but are not related to the query, and we can miss the true answers.

Table IV. Change of Performance When Each Step Is Removed

		Precision	Recall	F-measure
Using all steps		**0.6820**	**0.8929**	**0.7733**
Change in 1st step	Without Damerau-Levenstein distance	0.6967	0.8302	0.7576
	Without distance measure (IR formula + DF)	0.7763	0.7286	0.7517
	Without document frequency +distance (only IR formula)	0.8822	0.4729	0.6157
Change in 2nd step	Without clustering	0.5472	0.9157	0.6850
Change in 3rd step	Without cutting threshold	0.6398	0.9172	0.7538
When we input noun phrases, not a whole sentence		0.7036	0.7618	0.7315

Without the clustering of Step 2, the precision is significantly reduced since more than one concept can be chosen for each disease-related term in the input sentence. The performance without clustering shows reduced performance compared to the method with clustering.

Likewise, without the cutoff threshold restriction in Step 3, the recall improves, which we attribute to the selection of more concepts. However, the precision is reduced.

One might consider the addition of noun phrase detection and then use each noun phrase as an input, rather than the whole sentence. However, the experiment after noun phrase detection shows reduced performance compared with the method using a whole sentence as an input. That indicates that, in many cases, a disease term is not embedded in one noun phrase.

In conclusion, the experiments demonstrate that the second and third steps contribute to the improvement of precision, and the first step to the improvement of recall. We conclude that all three steps are important for mapping sentences into an ontology.

4.3. Discussion

In the process of identifying biomedical terms, our system shows lower precision than recall. This is because some frequently occurring words are treated as biomedical terms because they are included in the UMLS meta-thesaurus. For example, "be," "problem," "other," "is," and "cc" are considered biomedical terms. Our system shows good recall because we exploit our variation dictionary, and our method includes an arbitrary lexical word variation detection module.

Of course, we also have some incorrect detection examples in temporality. In the example "chill will stop for a second," "for a second" is recognized as temporality as a duration for the symptom "chill," which is not correct.

Note that we consider POS tags when detecting variations of patterns. As a result, precision was improved. For example, "eeting" (verb) was recognized by our system as a variation for "evening" (noun), and then we filtered it because the POS tags are different. In the sentence, "eeting" was actually used as the misspelling for "eating." In similar examples, there are "hours(noun)→shouls(verb)," "march(noun)→much(adverb)," and the like. In this example, "shouls" is a misspelled word for "should," not for "hours." In addition, the method could filter wrong variants if they occurred in the WordNet dictionary with another meaning; for example, "hour→thur (which means thursday)," "moment→movement," etc.

As incorrectly obtained examples, there are "hours→hoarsy," "march→ach," "minute→sinue," "month→outh," "noon→noo" (in reality, it was actually a misspelling of "no"), and "Today→onday (which was misspelled for Monday)."

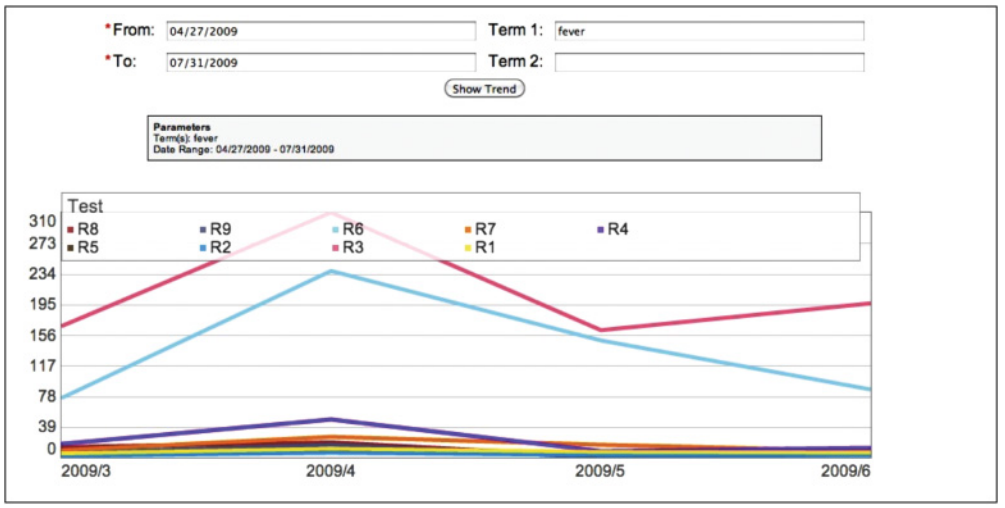

Fig. 13. Trend analysis example of a symptom "fever".

For age detection, the reason for recall loss is those patterns that do not include "old." In our data, there are cases that include only "num" + "year." For example, "caller has one son, 11 year, school sent home. . ."

To identify and remove implicit noise with high performance, we need information about the verbs that indicate some speculation on "thinking" (e.g., "concern," "worry," "review," etc.). In addition, we are now using the parser for factuality assessment, so parsing performance also affects our system. Since our data are imperfect, they create low accuracy of parsing results, and this also impacts system performance. We first need to remove sentence noise by detecting sentence boundaries and recovering sentence marks before applying a parser.

In our system, when we identify misspellings, we partly solve space errors. For example, "10days," and "2day" are detected as variants of "days," so they can be recognized as items in the temporality category.

As future work, we can investigate the trend of a symptom/disease for a specific region and time period based on our method, and we can also retrieve the associations between symptoms/diseases. Figures 13–15 show preliminary results of future research. Figure 13 is the trend of "fever" for each region for a specific period of time after applying our named entity recognition method. In the Figure, R1 to R9 are regions. Figure 14 compares the trends of two symptoms. Figure 15 shows the associated symptoms for a medical term "pneumonia" based on the chi-square test.

5. RELATED WORK

Biomedical term detection has been extensively studied in recent years, including the mapping of text phrases to UMLS concepts [Aronson 2001]. Most of these approaches focus on automatic indexing of biomedical literature and have proved inadequate for processing annotations of high-throughput datasets [Butte and Chen 2006]. It has also been shown that for the task of identifying concepts from annotations of high-throughput datasets, simple methods perform as well or better than MetaMap [Aronson 2001]. In previous work, some approaches of Gaudan et al. [2008] and Jimeno et al. [2008] are based on the identification of weighted words that compose terms denoting ontology concepts. They integrate two new aspects in their scoring method: The proximity between words in text and the amount of information carried

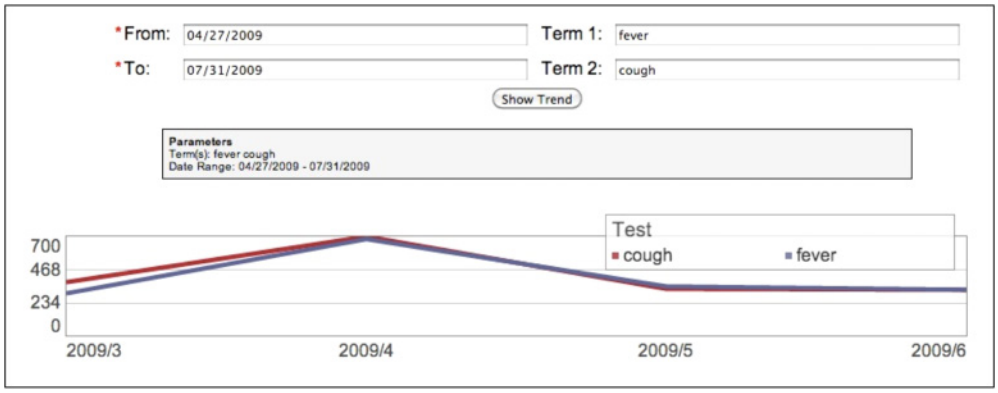

Fig. 14. Trend comparison between two symptoms "fever" and "cough."

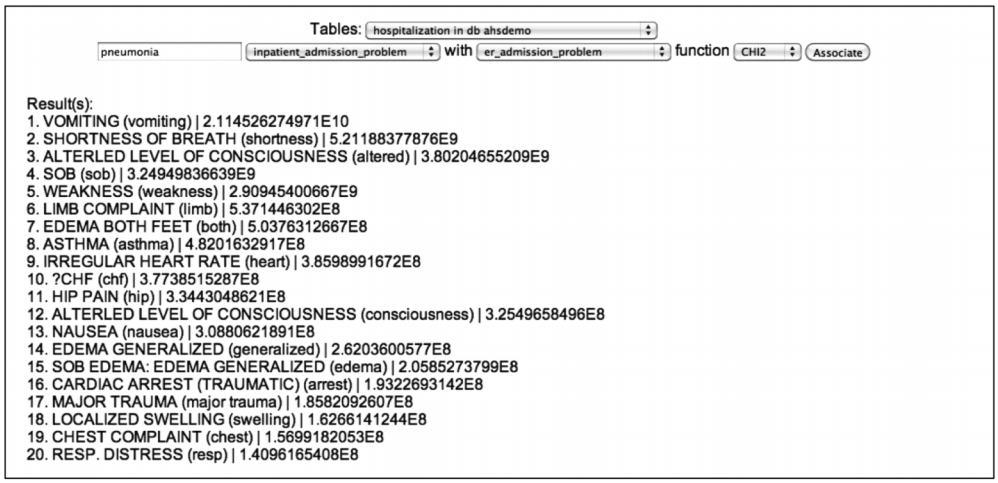

Fig. 15. Retrieval of associated symptoms of a disease "pneumonia."

by each individual word. Their method is a statistical method based on specificity, evidence, and proximity. Specificity and evidence are based on the frequency of a word in a corpus, and proximity is based on the distance between words in a corpus. They adopt TFxIDF, commonly used in information retrieval, to measure the evidence and specificity. But the performance is worse than dictionary-based simple matching methods, and they do not use any threshold methods to choose relevant concepts among the ranked concepts. They also do not consider noise in the data.

Holzinger et al. [2008, 2013] used text mining techniques to analyze medical diagnoses, and they also studied disease–disease relationships using web-based biomedical text mining techniques [Holzinger et al. 2012].

Ruch et al. [2003] considered misspelling correction in electronic patient records. Their system has three modules and tries to correct spelling for all types of words by considering surrounding words and syntactic function. Since our purpose is only retrieving patient-related named entities, we do not need to correct other nonvaluable words such as prepositions. We just adopt the idea that edit distance is useful in correcting spelling errors.

Mougin et al. [2006] mapped gene terms into UMLS based on normalization of words using the UMLSKS API and exact match. Mottaz et al. [2007] tried to map disease names into MeSH terminology. They used manually curated disease annotations to extract disease names and applied exact and partial matches to map the disease names into MeSH. To consider the information content of each word, they applied a TFxIDF weighting schema.

Some other machine learning approaches have also been investigated. Chun et al. [2006] used a maximum entropy-based method to filter candidate disease names found by dictionary-based methods. Various features are selected (e.g., context words, part of speech tag, word affix, etc.). Bundschus et al. [2008] tried cascaded Conditional Random Fields (CRF) using various features based on contexts, dictionary, and orthogonal forms to detect disease terms and the functional relations between them, but they need annotated data for training. In the methods of Neveol et al. [2009], a priority model [Tanabe and Wilbur 2006] was used to find noun phrases that are possibly disease names. However, the mapping process is still done with the MetaMap program.

We conclude that most previous work uses simple exact or partial matching based on a dictionary and sometimes performs deep preprocessing, such as noun phrase detection and normalization of variant words. Some statistical methods try TFxIDF to measure the information content of a word as used in information retrieval but show poorer performance than the dictionary-based matching methods.

Among the few systems in the medical domain that treat time expressions, the study by Denny et al. [2010] is most relevant to our work. They propose timing and status descriptors for colonoscopy testing data. They use the KnowledgeMap concept identifier to extract colonoscopy concepts and have developed a rule-based method with regular expressions to extract and normalize time descriptors. However, they rely on meticulous manual rule writing. Since our ultimate system needs to have more explicit understanding of temporal information, in the future we need to integrate our approach with other machine learning-based approaches.

For pattern learning, previous studies have suggested bootstrap-based pattern learning [Hao 2012; Nakashole et al. 2010; Riloff and Shepherd 1997; Riloff and Jones 1999; Yu and Agichtein 2003; Carlson et al. 2010; Mcintosh 2010; Kozareva and Hovy 2010]. Kozareva and Hovy [2010] use graphs to obtain patterns, define a vertex and an edge in the graph, and then choose the patterns (vertices) that have large edge values. Hao [2012] used bootstrapping for pattern learning for collaborative question answering, and Skeppstedt et al. [2012] also use Levenshtein distance for misspelling. We also used bootstrapping-based pattern extension, and this showed reasonable performance, like the previous work. The difference in our method is that we iterated only once. Because the length of a word in our pattern is short, we cannot allow much variation.

6. CONCLUSION

We propose a method and system for patient information extraction from noisy health records written down during phone conversations. When we extract the patient-related information from the noisy data, there are two kinds of noise that we have identified for removal: Explicit noise (which includes spelling errors, unfinished sentences, omission of sentence mark, etc.) and implicit noise, which includes nonpatient information and a patient's untrustworthy information. To remove explicit noise, we propose our biomedical term detection/normalization method, which deals with misspelling, imperfections, and arbitrary abbreviation by nurses. In detecting temporal named entity, temperature, and other types of named entities that convey patients' personal information, such as age and sex, we propose a bootstrapping-based pattern learning to detect all kinds of arbitrary variations of the named entities. To identify and remove implicit noise, we propose a dependency path-based filtering method. Finally, we obtain normalized

patient information and visualize the patient-related named entities by constructing a graph that indicates detected named entity terms, named entity types, and dependency between named entities.

For biomedical term detection, we use our own unsupervised method using a simple language model coupled with a measure based on Damerau-Levenshtein distance. We also presented a temporality detection system that provides a practical and extensible state-of-the-art system for extracting time expressions. It can be used as a basic component for building temporally aware systems and for investigating problems requiring temporal information, such as event extraction, temporal ordering of events, and question answering even for noisy data. In addition, we exploit our regular expression patterns to detect other types of named entities. Our system includes a factuality assessment component, used to distinguish between fact and nonfact, as well as to remove nonpatient information. Our proposed method in biomedical term detection outperformed previous methods. In the temporality and factuality assessment, the proposed system showed reasonable performance. Our system is useful for experts to mine patient information and to analyze trends in patients' concerns/symptoms.

REFERENCES

ACE. 2008. Automatic Content Extraction. English annotation guidelines for relations. *Linguistic Data Consortium*, version 6.0–2008.01.07 edition. Retrieved from http: //www.ldc.upenn.edu/Projects/ACE/.

A. R. Aronson. 2001. Effective mapping of biomedical text to the UMLS Metathesaurus: The MetaMap program. In *Proceedings of AMIA Symposium*. 17–21.

M. Bundschus, M. Dejori, M. Stetter, V. Tresp, and H. P. Kriegel. 2008. Extraction of semantic biomedical relations from text using conditional random fields. *BMC Bioinformatics* 23, 9, 207.

A. J. Butte and R. Chen. 2006. Finding disease-related genomic experiments within an international repository: First steps in translational bioinformatics. In *Proceedings of the AMIA Annual Symposium.* 106–110.

A. Carlson, J. Betteridge, R. C. Wang, E. R. Hruschka Jr., and T. M. Mitchell. 2010. Coupled semi-supervised learning for information extraction. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*. New York, NY, 101–110.

A. X. Chang and C. D. Manning. 2012. SUTIME: A library for recognizing and normalizing time expressions. In *Proceedings of the Eight International Conference on Language Resources and Evaluation*. Istanbul, Turkey, 3735–3740.

H. W. Chun, Y. Tsuruoka, J. D. Kim, R. Shiba, N. Nagata, T. Hishiki, and J. Tsujii. 2006. Extraction of gene-disease relations from Medline using domain dictionaries and machine learning. In *Proceedings of the Pacific Symposium on Biocomputing.* 4–15.

M. Dai, N. H. Shah, W. Xuan, M. A. Musen, S. J. Watson, B. D. Athey, and F. Meng. 2008. An efficient solution for mapping free text to ontology terms. In *Proceedings of the AMIA Summit on Translational Bioinformatics*. 21.

F. J. Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM* 7, 3, 171–176.

J. C. Denny, J. F. Peterson, N. N. Choma, H. Xu, R. A. Miller, L. Bastarache, and N. B. Peterson. 2010. Extracting timing and status descriptors for colonoscopy testing from electronic medical records. *Journal of the American Medical Information Association* 17, 4, 383–8.

R. Farkas, V. Vincze, G. Móra, J. Csirik, and G. Szarvas. 2010. The CoNLL-2010 shared task: Learning to detect hedges and their scope in natural language text. In *Proceedings of the 14th CoNLL Conference – Shared Task*. 1–12.

M. Fiszman, W. Chapman, D. Aronsky, R. Evans, and P. Haug. 2000. Automatic detection of acute bacterial pneumonia from chest X-ray reports. *Journal of the American Medical Information Association* 7, 6, 593–604.

S. Gaudan, A. Jimeno Yepes, V. Lee, and D. Rebholz-Schuhmann. 2008. Combining evidence, specificity, and proximity towards the normalization of gene ontology terms in text. *EURASIP Journal on Bioinformatics and Systems Biology* 8, 1, 1–9.

T. Hao. 2012. Bootstrap-based equivalent pattern learning for collaborative question answering. *LNCS*, 318–329.

A. Holzinger, R. Geierhofer, F. Modritscher, and R. Tatzl. 2008. Semantic information in medical information systems: Utilization of text mining techniques to analyze medical diagnoses. *Journal of Universal Computer Science* 14, 22, 3781–3795.

A. Holzinger, K. M. Simonic, and P. Yildirim. 2012. Disease-disease relationships for rheumatic diseases: Web-based biomedical textmining an knowledge discovery to assist medical decision making. In *Proceedings of the IEEE 36th Annual Computer Software and Applications Conference (COMPSAC)*. 573–580.

A. Holzinger, P. Yildirim, M. Geier, and K.-M. Simonic. 2013. Quality-based knowledge discovery from medical text on the web. In *Quality Issues in the Management of Web Information, Intelligent Systems Reference Library, ISRL 50*. Springer, Berlin, 145–158.

Jay M. Ponte and W. Bruce Croft. 1998. A language modeling approach to information retrieval. In *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval*. 206–214.

A. Jimeno, E. Jimenez-Ruiz, V. Lee, S. Gaudan, R. Berlanga, and D. Rebholz-Schuhmann. 2008. Assessment of disease named entity recognition on a corpus of annotated sentences. *BMC Bioinformatics* 9, Suppl 3, S3.

L. Karttunen and A. Zaenen. 2005. Veridicity. In *Proceedings of the Dagstuhl Seminar*. Retrieved from http://drops.dagstuhl.de/opus/volltexte/2005/314/pdf/05151.KarttunenLauri.Paper.314.pdf.

J. Kim, T. Ohta, S. Pyysalo, Y. Kano, and J. Tsujii. 2009. Overview of BioNLP'09 shared task on event extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*. 1–9.

Z. Kozareva and E. Hovy. 2010. Learning arguments and supertypes of semantic relations using recursive patterns. In *Proceedings of the ACL*. 1482–1491.

M. Li and J. Patrick. 2012. Extracting temporal information from electronic patient records. In *Proceedings of the AMIA Annual Symposium*. 542–551.

X. Ling and D. S. Weld. 2010. Temporal information extraction. In *Proceedings of the 24th Conference on Artificial Intelligence (AAAI)*. 1385–1390.

T. McIntosh. 2010. Unsupervised discovery of negative categories in lexicon bootstrapping. *EMNLP* 356–365.

A. Mottaz, Y. L. Yip, P. Ruch, and A. Veuthey. 2007. Mapping protein information to disease terminologies. *Journal of Integrative Bioinformatics* 4, 3, 79.

F. Mougin, A. Burgun, and O. Bodenreider. 2006. Mapping data elements to terminological resources for integrating biomedical data sources. *BMC Bioinformatics* 7, S3.

N. Nakashole, M. Theobald, and G. Weikum. 2010. Find your advisor: Robust knowledge gathering from the web. In *Proceedings of the 13th International Workshop on the Web and Databases*. 6.

A. Névéol, W. Kim, John W. Wilbur, and Z. Lu. 2009. Exploring two biomedical text genres for disease recognition. In *Proceedings of the Workshop on BioNLP*. 144–152.

J. Pustejovsky, M. Verhagen, R. Saurí, J. Littman, R. Gaizauskas, G. Katz, I. Mani, R. Knippen, and A. Setzer. 2006. *TimeBank 1.2. Linguistic Data Consortium*, LDC2006T08.

E. Riloff and J. Shepherd. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*. Providence, RI, 117–124.

E. Riloff and R. Jones. 1999. Learning dictionaries for information extraction by multilevel bootstrapping. In *Proceedings of the 16th National Conference on Artificial Intelligence and the 11th Innovative Applications of Artificial Intelligence Conference*. 474–479.

S. Robertson and S. Walker. 1994. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th ACM Conference on Research and Development in Information Retrieval (SIGIR'94)*. ACM Press, 232–241.

P. Ruch, R. Baud, and A. Geissbuhler. 2003. Using lexical disambiguation and named entity recognition to improve spelling correction in the electronic patient record. *Artificial Intelligence in Medicine* 29, 12, 169–184.

R. Saurí and J. Pustejovsky. 2012. Are you sure that this happened? Assessing the factuality degree of events in text. *Computational Linguistics* 38, 2, 261–299.

M. Skeppstedt, M. Kvist, and H. Dalianis. 2012. Rule-based entity recognition and coverage of SNOMED-CT in Swedish clinical text. *LREC* 1250–1257.

J. Strötgen and M. Gertz. 2010. HeidelTime: High quality rule-based extraction and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*. 321–324.

L. K. Tanabe and W. J. Wilbur. 2006. A priority model for named entities. In *Proceedings of HLT-NAACL BioNLP Workshop*. 33–40.

Ö. Uzuner, B. South, S. Shen, and S. DuVall. 2010. i2b2/va challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Information Association* 18, 5, 552–556.

Y. Wang, M. Zhu, L. Qu, M. Spaniol, and G. Weikum. 2010. Timely Yago: Harvesting, querying, and visualizing temporal knowledge from Wikipedia. In *EDBT*. 697–700.

P. Willet. 1988. Recent trends in hierarchical document clustering: A critical review. *Information Processing and Management* 24, 577–597.

H. Yu and E. Agichtein. 2003. Extracting synonymous gene and protein terms from biological literature. *Bioinformatics* 19, 1, i340–i349.

A. Yeh, A. Morgan, M. Colosimo, and L. Hirschman. 2005. Biocreative task 1a: Gene mention finding evaluation. *BMC Bioinformatics* 6, Suppl.1, S2.

Received October 2013; revised May 2014; accepted July 2014