UCLA Summary for i2b2/VA 2010 NLP Shared-Task Challenge

Corey W. Arnold, PhD, William Speier, MS, William Hsu, PhD,
Jean Garcia-Gathright, Juan Wu, Maurine Tong, Ricky Taira, PhD
UCLA Medical Imaging Informatics, Los Angeles, CA

ABSTRACT

We describe the system we developed for the 2010 i2b2/VA NLP Shared-Task Challenge. Our system is designed for the first tier of the challenge: extracting medical problems, tests, and treatments. Two technical strategies were pursued in parallel and merged for the final system. The first strategy was a knowledge-rich approach using specialized lexical resources, grammatical rules, and a conditional random field model. The second approach was entirely data-driven and used a hidden Markov model for initialization followed by a Markov chain Monte Carlo process of refinement. Features from the knowledge-rich system were integrated into the data-driven approach to extract a final set of concepts. Independently, results from each system were comparable and in combination a small improvement was observed.

INTRODUCTION

We developed two medical concept extraction systems for the first tier of the 2010 i2b2/VA NLP Shared-Task The first system utilized many of our existing NLP resources, including custom semantic terminologies and grammar rules, to perform concept extraction. Our goal was to train this system on the i2b2/VA dataset and then integrate it with our second stochastic sampling method as a language model in a data-driven Markov Chain Monte Carlo (DDMCMC) process.

METHODS

For this challenge we defined three major areas of work: 1) generating features from the training set; 2) implementing the knowledge-rich method and; 3) implementing the stochastic sampling method. Each area is now described in detail.

First-Level Features

We utilized three freely-available tools for first-level feature extraction and representation: Stanford Parser [Klein03], MMTx [Aronson01], and MALLET [McCallum02].

Stanford Parser. We used the Stanford Parser to produce part-of-speech tags for each word in the input sentences. We recorded unigram and bigram parts-of-speech for each input word and its parents in the parse tree.

MMTx. MMTx was used to associate input phrases with UMLS semantic types. By doing so, we were able to estimate the likelihood of certain semantic types belonging to a conceptual class.

MALLET. MALLET's Pipe architecture provided the framework for basic input/output and feature representation. Using MALLET's *Pipe*, *Token*, and *Instance* structures, we transformed the data from a raw character stream to a list of Instances with associated feature vectors and class labels (if known).

MALLET's built-in *Pipes* were used to form a sequence of *Tokens* from the annotation files. We then implemented our own *Pipe* which uses regular expressions to extract the phrase, concept type, start line, end line, start token, and end token, forming the gold-standard of known positive example concepts. Pipes for extracting Stanford Parser and MMTx features over the raw text of the discharge summaries were also implemented. The Stanford Parser Pipe produces a parse tree for each sentence, then stores part-of-speech tags for input words and their parents in the parse tree. The MMTx Pipe creates a sparse vector representing the UMLS semantic types the phrase maps to, filtering semantic types based on match score. Features were then counted and stored in a database to be used by the classification methods in Method 2 below. However, because our feature generation efforts occurred in parallel with model development, not all features generated were utilized.

Method 1 - CRF Model with Knowledge Sources

Our first approach used a conditional random field (CRF) statistical model that integrated features derived from grammatical patterns and from various lexical resources.

Problem Formulation

The concept identification problem is formalized in terms of a sequence labeling problem. Various tagging schemes for segmenting conceptually coherent phrases in text have been used in the literature (e.g., BIO [Ramshaw95]). We chose the 5-class BEISO coding scheme due to performance advantages described in [Kudo01]. Briefly, given a concept type (e.g., Problem),), the goal of the CRF

classifier is to tag each word in the sentence with one of the following five outcomes: a) Begin [B], b) End [E], c) Inside [I], d) Single [S], and e) Outside [O]. For example, in the sentence: "The patient is a 47 year-old man with a history of coronary artery disease," the correct i2b2 tag would be to tag all tokens (~words) with the labels 'O' except for the last 3 words. The last 3 words have labels B (for the word 'coronary'), I (for the word artery), and E (for the word 'disease'). Note that in Method 1, we use a 5-class tagging scheme, treating each concept type separately. Dependencies can be better modeled at the cost of an increase in state space (sparseness of samples) if a 13-tag scheme (B-problem, B-test, B-treatment, E-problem, E-test, … , Outside) is used as in Method 2 below.

The concept identification task is thus posed as a sequence prediction problem, where we predict the BEISO tag t_i associated with every token w_i in a sequence of tokens. If the input token sequence is denoted by $\{w_1, w_2, \ldots w_n\}$ which we abbreviate by w_1^n and the BEISO tag sequence for this input by t_1^n, then the problem is to solve for the sequence $\hat{t}_1^n$ that maximizes the following probability:

$$\hat{t}_1^n = arg\, max_{t_1^n} P(t_1^n | w_1^n) \qquad (1)$$

Modeling the evidence associated with a correct tag sequence proceeds in a standard manner, that is, we re-represent the context for this conditional probability in terms of a context vector $\vec{x}_i$ that captures the relevant context associated with the sentence word sequence in order to estimate the statistics of the conditional probability. The elements of the context vector are implemented as feature functions that can include simple predicates or complex predicates (conjunctive joins of a number of simple features).

Note that the tokenization scheme can have an effect on contextual modeling (e.g., dates, measurements idioms, co-locations (e.g., white matter)) and it is often advantageous to treat such multi-word phrases as a single token.

Context Modeling

The input of the phrase chunking problem is a sequence of tokenized text. Word-level tokenization is based simply on white space. We use a lexical analyzer that includes a medical lexicon to assign a semantic class and basic part of speech tag to each word token. A 3-tuple consisting of the following features are thus associated with each word token: 1) the string itself, 2) a basic part-of-speech tag, and 3) a semantic class assignment. The lexicon for this mapping is publically available as part of the UCLA NLP toolkit (http://www.mii.ucla.edu/nlp). This lexicon assigns words into one of 12 part-of-speech categories and about 450 semantic classes. Word sense disambiguation (WSD) is handled using a rule-based system that uses evidence from local surrounding context. Tools for adding new entries into the lexicon from the training i2b2 corpus of reports are part of our environment [Taira01]. A high-level interface for adding WSD rules have also previously been developed. The lexical analysis step also includes regular expression pattern matchers for dates, measurements, and other tokens containing special symbols (e.g., disease TNM stages). A list of drug names supplements the general medical lexicon and is derived mainly from the FDA data source.

Three classes of features are used to estimate the context in Eq. 1 (w_1^n):

Class 1. The first class of features was the semantic class labels derived in the lexical analysis step for tokens within a sliding fixed window of width five centered about each token of interest. The semantic class assignments for tokens was used as opposed to the word string to combat sparse statistics and served as a reasonable surrogate to the word (450 unique semantic classes versus ~10,000 word strings). Semantic information can help pool word statistics given that semantic labeling of words represents some clustering of the domain vocabulary. Thus for example, all anatomy terms are pooled under a single semantic class as are chemical substances.

Class 2. The second class of features included variable length token pattern rules. Tokens within a contextual sequence were represented either as exact strings, a vector of morphological features, part-of-speech tags, or semantic tags. These variable length rules help to provide more specific evidence when the surrounding fix window context is insufficient. These patterns were added incrementally after analyzing errors at the end of each iteration of our classifier development cycle.

Class 3. The third class of features was the section class information assigned to each sentence within an i2b2 document. A simple classifier based on section headings was developed to assign each sentence to a section class. This classifier was based mainly using a memory based system that recorded the section heading labels from labels seen in the training reports. The section class features helped to disambiguate drug names (e.g., treatments) from non-therapeutic chemicals (contrast agents, etc.).

Classifier Design

We use a conditional random field (undirected graphic chain graph) for modeling the mapping between the (overlapping) evidence and the output label assignments. Each concept type was modeled independently. The CRF model has various advantages over HMM's or MEMM as described in [Lafferty01] and include improved modeling of context (flexibility of modeling context, dealing with the label bias problem). CRF models have been widely used for successfully modeling sequence labeling problems [Peng06, Wang09]. We use the JAVA implementation from the MALLET-2.0-RC4 toolkit developed by the University of Mass, Amherst. The number of features for the problem, treatment and test classes were 12226, 10211, and 9378 respectively. Training time for parameter estimation for the individual CRF models was between 3 and 5 minutes on a 2.5 GHz Quad-Core PC.

Method 2 - Stochastic Sampling

In this method the concept extraction problem was formulated as a one dimensional image segmentation problem. The annotation of a sentence is an array of labels denoting the concept class $c \in \{\text{problem}, \text{test}, \text{treatment}, \text{outside}\}$ of each word. Words were also labeled based on where the occurred within a segment $l \in \{\text{begin}, \text{inside}, \text{end}, \text{single}\}$. The solution space is then all of the possible arrays $\underline{x}$ of labels.

An initial configuration is obtained using a simple Hidden Markov Model (HMM) [Baum66]. For this model it is assumed that sentences are generated by an underlying structure of tags which can be modeled using the Markov assumption. The bigram counts for the training data are recorded to form transition probabilities for a Markov model and the counts for word tag pairs are recorded for the emission probabilities. The Viterbi algorithm then efficiently provides an initial tag configuration. Since the Markov assumption is too strong to adequately model language, these results can be improved with the use of a more general optimization technique.

Markov Chain Monte Carlo is a method which samples from this solution space based on a probability distribution over the possible states [Metropolis53]. The Markov chain is proven to converge to the desired probability distribution if it's transition function is chosen such that the chain is ergodic and aperiodic and the transition probabilities follow the detailed balance equations.

The transition function chosen consists of five types of transitions called *dynamics*. The first dynamic randomly *creates* a new concept in the sentence, the second randomly *destroys* one, the third *changes* the class of an existing concept, the fourth *splits* an existing concept, and the fifth *merges* two adjacent concepts. For each step of the Markov chain, one of these dynamics is chosen at random, making the Markov chain aperiodic. Since each state can reach any other state in a finite number of steps using these transitions, the Markov chain is ergodic.

Once a transition is chosen, the move is accepted from state $\underline{x}$ to state $\underline{x}'$ with probability

$$\alpha(\underline{x}, \underline{x}') = \min\left(1, \frac{p(\underline{x}')\mathrm{T}(\underline{x}', \underline{x})}{p(\underline{x})\mathrm{T}(\underline{x}, \underline{x}')}\right)$$

where $\mathrm{T}(\underline{x}, \underline{x}')$ is the probability of choosing the transition from state $\underline{x}$ to state $\underline{x}'$. The probability of being in state $\underline{x}$ and accepting a move to state $\underline{x}'$ is then

$$p(\underline{x})\mathrm{T}(\underline{x}, \underline{x}')\alpha(\underline{x}, \underline{x}') = \min\left(p(\underline{x})\mathrm{T}(\underline{x}, \underline{x}'), p(\underline{x}')\mathrm{T}(\underline{x}', \underline{x})\right)$$

which is symmetric in $\underline{x}$ and $\underline{x}'$, so it satisfies detailed balance and converges to the stationary distribution.

The probability distribution used for this method consisted of a prior probability of a tagging sequence multiplied by a posterior distribution of probabilities of words given a tag. The probability of the tagging sequence was obtained by taking the trigram histogram of the training documents. The posterior distribution was obtained by recording the counts for tag word pairs. Since these histograms are sparse, smoothing was necessary to avoid overtraining.

Smoothing

The smoothing method used was a modified Witten-Bell smoothing method with backoff [Moffat90, Katz87]. Here, the probability of a word given a concept and label, $p(w|c, l)$, is discounted by the number of possible words to occur with the same concept and label, $T(c, l)$. The remaining probability is then distributed among the words that did not occur with that concept and label in the training data based on the probability of the word given the concept, $p(w|c)$.

Task	TP	FN	FP	Recall	Precision	F-Score
1.1 Concept Exact Span (All)	32746	12263	12361	72.8	70.1	71.3
Class Exact Span	31980	13029	13127	71.0	70.9	71.0
1.2 Concept Exact Span (problem)	13455	5095	5728	72.5	70.1	71.3
Concept Exact Span (Treatment)	9405	4155	3324	69.3	73.9	71.6
Concept Exact Span (Test)	9886	3013	3309	76.6	74.9	75.8
Class Exact Span (Problem)	13270	5280	6015	71.5	68.8	70.1
Class Exact Span (Treatment)	9128	4432	3571	67.3	71.9	69.5
Class Exact Span (Test)	9582	3317	3541	74.3	73.0	73.6
1.3 Inexact Span – All Concepts Together	39559	5450	5548	87.9	87.7	87.8
Class Inexact Span	39559	5450	5548	87.5	84.3	85.8
1.4 Inexact Span – Separate Classes (Problem)	16733	1817	2425	90.2	87.3	88.8
Inexact Span – Separate Classes (Treatment)	11520	2040	1301	85.0	90.0	87.3
Inexact Span – Separate Classes (Test)	11306	1593	1822	87.7	86.1	86.9
Inexact Span, Class (Problem)	16257	2293	3028	87.7	84.3	85.9
Inexact Span, Class (Treatment)	11011	2549	1688	81.2	86.7	83.8
Inexact Span, Class (Test)	10746	2153	2377	83.0	81.9	82.6

Table 1. Results of combined classifier.

$$p(w|c,l) = \begin{cases} \dfrac{c(w,c,l)}{c(c,l)+T(c,l)} & if\ c(w,c,l) > 0 \\ \alpha(c,l)p(w|c) & otherwise \end{cases}$$

Likewise, the probability of the word given a concept is discounted by the number of possible words to occur with the same concept and the remaining probability is distributed among words that did not occur with that concept in the training data.

$$p(w|c) = \begin{cases} \dfrac{c(w,c)}{c(c)+T(c)} & if\ c(w,c) > 0 \\ \alpha(c)p(w) & otherwise \end{cases}$$

Similar smoothing methods were used for the bigram and trigram probabilities for the tag sequences.

Integration

Since the MCMC algorithm is a general framework with flexibility in the transition function and probability distributions used, the two methods could be integrated to try to take advantage of the benefits of each approach. This was done by using the output from the CRF method to modify the probabilities in the DDMCMC method.

RESULTS AND DISCUSSION

Table 1 shows the results from the final, combined classifier. For inexact matching, a ~2-5% increase in F-score was observed for all tasks compared to each method individually. However, F-scores for the extraction of exact spans showed Method 1 outperformed the combined classifier by ~1-3% due to higher precision likely resulting from the window size and use of semantic classes. In comparison, the combined classifier showed better recall for inexact matches indicating Method 1 may generalize less well.

CONCLUSION

For the i2b2/VA 2010 NLP Shared Task Challenge we implemented and combined two methods with mixed results. We considered the inexact extraction task a better evaluation of a system we would use in our research and therefore decided the combined classifier outperformed either method alone. However, Method 1 outperformed the combined classifier for exact concept matching.

There are several directions for future work that this preliminary work suggests. Firstly, as argued in [Banko01], the emphasis of the number of training samples as likely the easiest route to improving performance will be pursued. Active learning methods will be explored for *selectively sampling* (as opposed to *randomly sampling*) a large test corpus for tagging utilizing various entropy-based scores [Tang02]. Second, the symbolic rule-based portion of the method uses all hand-crafted rules to insure that each make linguistic sense or are consistent with various types of semantic constraints. Thus, the system tries to emphasize accuracy over timely development. This however brings up issues related

to scalability. Future experiments will include methods for automatically discovering variable length candidate rules following the methodology, for example, of [Soderland99]. Finally, an exploration of how syntactic parser analysis can improve concept extraction, especially for long sequences is planned.

Further integrating the probability distributions and also utilizing the CRF output in the transition function in order to better choose boundaries in *split* and *create* dynamics is also a point of future work.

References

1. Klein, D, Manning, C. "Fast exact inference with a factored model for natural language parsing," *Advances in Neural Information Processing Systems*. 2003:3-10.
2. Aronson, A. "Effective mapping of biomedical text to the UMLS Metathesaurus: the MetaMap program," *Proceedings of the American Medical Informatics Association*; 2001; 2001. p. 17-21.
3. McCallum, AK. MALLET: A Machine Learning for Language Toolkit.: http://mallet.cs.umass.edu; 2002.
4. Ramshaw, LA and Marcus, MP. "Text chunking using transformation-based learning," *In Proc. of third workshop on Very Large Corpora*, pages 82–94, June 1995.
5. Kudo, T and Matsumoto, Y. "Chunking with Support Vector Machines," in *Proc. of the 2nd Meeting of the North American Chapter of the Association of Computational Linguistics*, Carnegie Mellon University, Pittsburgh, Pennsylvania, pp. 192-199, June 2-7, 2001.
6. Taira, RK, Soderland, SG, Jakobovits, RM. "Automatic structuring of radiology free text reports," *Radiographics* 21:237-245, 2001.
7. Peng, P and McCallum, A. "Information extraction from research papers using conditional random fields," *IPM*, 42(4):963-979, 2006.
8. Lafferty, J, McCallum, A, and Pereira, F. "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," In *Proc. of the ICML*, pages 282-289, 2001.
9. Wang, Y. "Annotating and recognizing named entities in clinical notes," *Proc. of the ACL-IJCNLP 2009 Student Research Workshop*, pp 18-26, Suntec, Singapore, August 2009.
10. Baum, LE and Petrie, T. "Statistical inference for probabilistic functions of finite state Markov chains," *Ann. Math. Statist*. 37 1554-1563, 1966.
11. Katz, SM. "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3), 400–401, 1987.
12. Metropolis, N, Rosenbluth, AW, Rosenbluth, MN, Teller, AH and Teller, E. "Equations of state calculations by fast computing machines," *J. Chem. Phys*. 21, 1087-91, 1953.
13. Moffat, A. Implementing the PPM data compression scheme. *IEEE Trans. Commun*. 38, 11 (Nov.), 1917-1921, 1990.
14. Tu, Z, Zhu, SC. "Image Segmentation by Data-Driven Markov Chain Monte Carlo," *IEEE Transactions on Pattern analysis and Machine Intelligence*, vol. 24, no. 5, pp. 657-673, May 2002, doi:10.1109/34.1000239.
15. Banko, M and Brill, E. "Mitigating the paucity-of-data problem: exploring the effect of training corpus size on classifier performance for natural language processing," *Proc of the Conference on Human Language Technology*, San Diego, CA. 2001.
16. Tang, M, Luo, X, and Roukos, S. "Active learning for statistical natural language parsing," in Proc. of the 40th Annual Meeting of the Association for Computational Linguistics pp 120-127 July 7-12 Philadelphia, PA 2002.
17. Soderland, S. "Learning information extraction rules for semi-structured and free text," *Machine Learning*, 34: 233-272, 1999.