Machine-learned solutions for three stages of clinical information extraction: the state of the art at i2b2 2010

Berry de Bruijn, Colin Cherry, Svetlana Kiritchenko, Joel Martin, Xiaodan Zhu

Institute for Information Technology, National Research Council, Ottawa, Ontario, Canada

Correspondence to
Dr Berry de Bruijn, Institute for Information Technology, National Research Council, Canada, 1200 Montreal Road, Building M50, Ottawa, ON, K1A OR6, Canada; berry.debruijn@nrc-cnrc.gc.ca

Received 11 January 2011
Accepted 13 April 2011
Published Online First 12 May 2011

ABSTRACT

Objective As clinical text mining continues to mature, its potential as an enabling technology for innovations in patient care and clinical research is becoming a reality. A critical part of that process is rigid benchmark testing of natural language processing methods on realistic clinical narrative. In this paper, the authors describe the design and performance of three state-of-the-art text-mining applications from the National Research Council of Canada on evaluations within the 2010 i2b2 challenge.

Design The three systems perform three key steps in clinical information extraction: (1) extraction of medical problems, tests, and treatments, from discharge summaries and progress notes; (2) classification of assertions made on the medical problems; (3) classification of relations between medical concepts. Machine learning systems performed these tasks using large-dimensional bags of features, as derived from both the text itself and from external sources: UMLS, cTAKES, and Medline.

Measurements Performance was measured per subtask, using micro-averaged F-scores, as calculated by comparing system annotations with ground-truth annotations on a test set.

Results The systems ranked high among all submitted systems in the competition, with the following F-scores: concept extraction 0.8523 (ranked first); assertion detection 0.9362 (ranked first); relationship detection 0.7313 (ranked second).

Conclusion For all tasks, we found that the introduction of a wide range of features was crucial to success. Importantly, our choice of machine learning algorithms allowed us to be versatile in our feature design, and to introduce a large number of features without overfitting and without encountering computing-resource bottlenecks.

INTRODUCTION

Clinical text-mining systems have great potential but are hard to properly evaluate. Mostly for privacy reasons, sizeable data collections are rarely shared between studies, and when task objectives, metrics, and gold standards also differ, studies become very hard to compare. One place where rigid system evaluations have been coming together is in the i2b2 natural language processing (NLP) challenges. The fourth i2b2 challenge in 2010 was designed around a three-stage scenario consisting of (1) extraction, from discharge summaries and progress notes, of medical problems, tests, and treatments; (2) classification of assertions made on medical problems; and (3) classification of relations between medical concepts. Combined, these three stages represent a realistic chain in practically applicable text mining. With such chains, valuable structured information can be derived from free-text data sources that otherwise would remain largely untapped. As clinical text mining continues to mature, its potential as an enabling technology for improved patient care as well as more efficient clinical research is increasingly becoming a reality.

BACKGROUND

Uzuner and DuVall[1] describe the 2010 i2b2 NLP challenge in full detail in their article in this issue. To summarize: participants were provided with 349 human annotated documents and 827 non-annotated documents to be used for system development and training. The test data set comprised 477 documents for which human annotations had also been collected. The documents were anonymized hospital discharge summaries and progress reports, collected from four different sources. A typical document had 81 lines, 664 tokens, and 3728 characters (medians over 349+477 documents). Documents were presegmented on sentence and token boundaries.

The challenge consisted of three tasks:

1. Identifying medical concepts of types 'problem,' 'test,' and 'treatment'—which included identifying the concept boundaries and reporting the concept type. The primary evaluation metric was F-score for getting boundary and type right, and the secondary metric ('inexact' F-score) allowed the boundary to be off.
2. Asserting for each 'problem' concept whether the context describes it as 'present,' 'absent,' 'possible,' 'conditional,' 'hypothetical,' or 'associated with someone else.' The second task, as well as the third, was evaluated through micro-averaged F-scores.
3. Establishing the relationship between pairs of concepts, with concepts occurring in the same sentence and one being a 'problem' concept. Types of relationships were from a closed set; examples are 'treatment is given for the problem,' 'treatment is not given because of the problem,' 'treatment worsened the problem,' 'test revealed the problem,' and 'problem indicates another problem.'

Training material was released 2 months before the challenge to allow for system development. The challenge protocol involved three consecutive 24 h time windows to submit system outputs for tasks 1, 2, and 3. Immediately after each deadline, ground truths were released for that task so that the next task would not need to rely on imperfect system output on a previous task. Teams were allowed three system submissions per task.

Methods

We developed systems for each of the three tasks within the challenge. All systems were built around a (semi-) supervised machine learning paradigm where elements of the source texts were represented as bags of features. Patterns between features and desired ('ground truth') output in the training texts were learned by the system, and this allowed it to generate output for observed feature patterns in test texts. While the three systems employed different learning mechanisms, the feature sets were mostly similar.

Feature representation for each token in feature space consisted of:

- token features, including casefolding, punctuation, prefix/stem patterns, word shape information (eg, 'Chem-7' becomes 'Aa-0'), and character n-grams;
- context features, including token features from four tokens before to four tokens after the word, word bi/tri/quad-grams, and skip-n-grams (word sequences with wild-cards, eg, 'onset* pain');
- sentence features, including sentence-length indicators, case-folding patterns, presence of digits, enumeration tokens at the start, a colon at the end, and whether verbs indicate past or future tense;
- section features: including headings, assumed to be the most recently seen all-caps line ending with a colon, and subsection headings, assumed to be the most recently seen mixed-case line ending with a colon;
- document features, including upper-case/lower-case patterns seen across the document, and a document length indicator.

In addition to these surface or text-oriented features, we used concept mapping features, which were mostly derived from existing annotation tools. These were cTAKES,[2] MetaMap[3] (UMLS), and ConText.[4] Inspired by Miller *et al*,[5] we also used the Brown clustering algorithm[6] to create 7-bit hierarchical word clusters from the provided unlabeled data. These clusters tend to be a mixture of semantic concepts and parts-of-speech; for example, some clusters contain mostly past-tense verbs, while others contain mostly drug names. Cluster granularity was optimized by cross-validation at seven hierarchical levels, resulting in 128 clusters. Additionally, a few simple pattern matching expressions from our own libraries were applied to cluster words and terms into more general concepts, including negations, auxiliary verbs, words indicating uncertainty, family members, and prepositions. For Task 3, we used the output from Charniak's syntactic parser with its improved, self-trained biomedical parsing model,[7] after transferring these into Stanford dependencies.[8] Extracted features included words, their tags (eg, POS tags), and arc labels on the dependency path between the minimal trees that cover the two concepts, along with the word and tags of their common ancestor, and the minimal, average, and maximal tree distances to the common ancestor.

In most cases, the features were binary—that is, either 'present' or 'absent'—and represented in a sparse vector—that is, only 'present' features are written out. Throughout the development phase, inclusion of potential groups of features depended on validation within the training data, either using a single hold-out set or through n-fold cross-validation. Post-hoc re-estimates of the relative contribution of feature groups to the test-set scores are reported in the Results.

System design: task 1—Concepts Task

The Concepts Task concerns the identification of key concepts anywhere in the source text, and includes determining the exact boundaries of the concept, as well as the class of the concept ('problem,' 'test,' or 'treatment'). Concepts are non-overlapping and non-nested.

In our system, concept tagging is carried out using a discriminative semi-Markov HMM, trained using passive-aggressive (PA) online updates. Semi-Markov models[9] are Hidden Markov Models that tag multitoken spans of text, as opposed to single tokens. This allows us to conduct information extraction without requiring a Begin/Inside/Outside (BIO) tagging formalism; instead, we need only four tags: *outside*, *problem*, *treatment*, and *test*. *Outside* is constrained to tag only single words, while the others can tag spans up to 30 tokens in length.

The semi-Markov model provides two major advantages over BIO. First, by labeling multitoken spans, labels cohere naturally. This allows the tagger to perform well without tracking the transitions between labels. Second, semi-Markov models allow much greater flexibility in feature construction, as one has access to the entire text of the concept as it is tagged, allowing easy inclusion of whole-concept features such as concept length.

Semi-Markov models are generally trained as Conditional Random Fields. However, we found Conditional Random Field training to be too slow and memory-intensive for our large feature sets. Instead, we carry out training using an online algorithm similar to Collins' structured perceptron,[10] called the PA algorithm.[11] In particular, we use a loss-driven variant with a 0—1 cost. PA learning makes several passes through the training set. Per pass, each training example is visited once and decoded to find the max-loss response. The weight vector is then adjusted with the smallest possible update that will separate the correct tagging from the max-loss response by a margin of 1. During development, PA consistently outperformed a structured perceptron.

Since the PA algorithm has no explicit regularization, it is helpful to average the parameter values over all updates.[10] We report the results using this averaged weight vector. We used cross-validation to discern that 15 passes through the training data were sufficient for good performance. The complete system trained in about 2.5 h on modern hardware.

Throughout development, we explored an effective feature space that spanned 4.2 million features from the feature groups described above; 1.1 million features were assigned non-zero weights. We did not find overfitting to be an issue. All of our standard features were word-level features. In the case of a multiword concept, the features for all of its words were combined in a bag. All features were paired with one of our four tags (problem, treatment, test, outside). We began with a standard part-of-speech tagging feature set, as described by Ratnaparkhi.[12] We then added token features and concept-mapping features for the word being tagged and for adjacent words. For sentences immediately following a section heading, we included each word from the heading as a feature of the concept. We also included word-level features extracted from cTakes and UMLS output.

Our system design included four innovations that substantially deviate from existing best practices:

1. Begin/end annotation. As stated above, our semi-Markov system has only four tags, which are applied to multiple tokens at a time. However, this coarse tagging does not predict concept boundaries well. This may give an advantage to BIO, which crucially singles out the beginning of a concept in its tag set. Fortunately, we can include the same information (without modifying our tag set) by annotating when word-level features are extracted from words that begin or end a concept. When a concept contains only one word, it both begins and ends the concept. Thus, we maintain the

advantages of semi-Markov modeling while also building rich representations of concept boundaries.
2. Semi-supervised learning through clustering. We ran the Brown clustering algorithm[6] on the provided unlabeled clinical data. These clusters generated word-level back-off features, allowing us to accurately tag previously unseen words in the test data, as long as those words were seen in the unlabeled data.
3. Semi-Markov features. In addition to bag-of-token-features, we found several useful features that look at all of the words in a concept at once. These describe (a) whether brackets (mis) matched; (b) which cTAKES and UMLS labels were encountered together inside a concept; (c) sequences of function words and wildcards (eg, *'the * with the *'*); and (d) the number of prepositions in concepts of three words or longer.
4. Annotated outside tags. In general, we found transition features in the form of tag bigrams and trigrams, such as '*<test> <outside> <problem>*' to be harmful. However, by augmenting the outside tags for common words with their lexical items, we created meaningful (and beneficial) tag n-grams, such as '*<test>* <for> *<problem>*.' We annotated outside tags for the 20 most common tokens, which included primarily function words and punctuation.

The contributions of these innovations, as determined through ablation experiments on the official test data, are included in the Results.

For the official test runs, we selected three system variations: (1.1) the complete system; (1.2) a system that did not use UMLS or cTAKES features; and (1.3) a self-training-system, where the complete system tagged the unlabeled data to generate additional tagged training data, and was then retrained.[13]

System design: task 2—Assertions Task

The Assertions Task is phrased as follows: for every 'problem' concept in a text, assert whether that concept was found to be present, absent, possible, conditional, hypothetical, or associated with someone else.

This task, being a multiclass categorization problem, allows us to use machine learning classification methods for generating predictions about the class of a new instance based on features that are shared with instances for which the class is already known. Our system implementation solved this task in two stages. In stage 1, assertion class predictions were generated for every word that was part of a 'problem' concept. In stage 2, a secondary classifier predicted a class for the complete concept based on the separate per-word predictions from the ensemble of stage-1 classifiers.

In stage 1, each word was represented as a large, sparse, binary feature vector, from the feature groups described above. Three classifiers were trained and applied independently: (1) the SVM-multiclass from the SVM-light/SVM-struct library,[14] which outputted one score per class per word; (2) LibSVM,[15] where six classifiers were trained and applied in a one-class-versus-rest set-up, resulting in one score per class per word; and (3) our multi-class PA learner (see Task 1), outputting one score per class per concept. The reason for using the three classifiers was the observation of some independence between respective outputs. This independence was consistent throughout development, although it diminished as the feature representation became more sophisticated. The stage 2 classifier used SVM-multiclass with a linear kernel, default parameter settings and a C-parameter value of 20 000, as selected using a development set.

Given this general architecture, we produced three variants for our official submission:

- System 2.1: the complete two-stage process, as described above.
- System 2.2: a simplified system. Stage 1 consisted of SVM-multiclass alone, predicting word-level classes. Stage 2 remained unchanged.
- System 2.3: designed to improve minority class recall, even if overall performance suffered. The output of System 1 was over-ruled when the LibSVM score on 'associated-with-someone-else'-versus-rest exceeded a manually set threshold for any of the words in a concept. The same was then done for 'hypothetical,' 'conditional,' and 'possible'—mimicking the order specified in the challenge guidelines. If none of the scores over-ruled System 1 output, the original output was retained.

While the general design is straightforward, the specifics of our architecture deviated from common practice on three points: (1) the parallel use of different classifiers, since these consistently showed some level of independence throughout development; (2) the use of millions of features; and (3) generation of predictions on a word-by-word basis, rather than for an entire concept, followed by an aggregation step (the stage-2 classifier). The last design decision allowed for very simple integration of features between resources such as UMLS or cTAKES, even if the exact concept boundaries mismatched.

System design: task 3—Relations Task

The goal of the Relations Task was to determine the relationship between a pair of concepts provided that the two concepts appear in the same sentence and at least one is a 'problem' concept. Including the 'no relationship' classes, this task defines six treatment—problem relations, three test—problem relations, and two problem—problem relations.

We trained three separate classifiers to categorize treatment—problem, test—problem, and problem—problem relations respectively. The classification framework was maximum entropy, as implemented in the OpenNLP maximum entropy toolkit.[16] Relations were classified independently; that is, a decision made on one concept pair did not affect other decisions.

Our baseline feature set was similar to that of Patrick and Li,[17] which was further augmented with features derived directly from the concept and assertion-tagged text, from the external MetaMap and cTAKES taggers, and from the parser output, as described above. Note that the specific feature sets, as optimized during development, were not necessarily the same between the three classifiers.

Additionally, we found benefit in balancing the category distribution. In the training set, some of the relationship types were observed much more often than others—for example, there were about eight times more negative problem—problem relations than positive ones. We addressed this issue by down-sampling the training set to a positive/negative ratio between 1:2 and 1:4, as selected using a development set. This reduced a classifier's bias towards the majority class.

The components outlined up to here, all contributed to the System 3.1 submission. For System 3.2, we added semantic information by using Medline as a semi-structured source of knowledge. For approximating the relatedness of two concepts, we calculated Pointwise Mutual Information between the concepts as they were found in Medline abstracts.

For System 3.3, we added semi-supervised training by applying bootstrapping on the supplied unlabeled data. For this, our system for Task 1 was applied to the unlabeled documents to provide concept span tags and labels. For this system, the

Table 1 Test set performance for the three tasks

	True positive	False negative	False positive	Recall	Precision	F-score
Task 1: Concepts Task						
System 1.1	37 646	7363	5683	0.8364	0.8688	**0.8523**
System 1.2	36 776	8233	6125	0.8170	0.8572	0.8366
System 1.3	37 663	7346	5787	0.8367	0.8668	0.8515
Task 2: Assertions Task						
System 2.1	17 366	1184	1184	0.9362	0.9362	**0.9362**
System 2.2	17 338	1212	1212	0.9347	0.9347	0.9347
System 2.3	17 197	1353	1353	0.9271	0.9271	0.9271
Task 3: Relations Task						
System 3.1	6296	2809	1965	0.6902	0.7611	0.7239
System 3.2	6269	2801	1896	0.6911	0.7677	0.7274
System 3.3	6288	2782	1838	0.6932	0.7738	**0.7313**

inclusion of down sampling was especially important, since bias is amplified when bootstrapping is deployed.

RESULTS AND DISCUSSION

Concepts task

System 1.1 gave an F-score on Exact evaluation of 0.8523, which positioned it as the highest scoring system in the challenge, differing significantly from the second-ranked system.[1] The results for System 1.1 versus System 1.2 (see table 1) demonstrate that the external sources of semantic and syntactic tagging (UMLS and cTAKES) are beneficial; together they improve F-measure by 1.5 percentage points. Bootstrapping on the unlabeled data (System 1.3) showed no improvement. Inexact evaluation, where boundaries were allowed to be off, gave 0.9167 recall, 0.9322 precision, and 0.9244 F-score measurements for System-1.1.

The ablation experiment results (table 2) show that the largest single contributor to the final score is begin/end annotation. When one accounts for the fact that only end annotation provides new information over BIO, the impact decreases. The remaining components as a block are worth another half-point of F-measure, although their impact as individual templates is small. Also, note that the contribution of all four components together is larger than the sum of their parts, indicating that the various features compensate for one another when only one is absent.

Assertions task

System 2.1 achieved an F-score of 0.9362 (table 1), which positioned it as the highest-scoring system in the challenge, while the differences with the three next systems were not statistically significant.[1] Table 3 shows the class-by-class confusion matrix between prediction and truth. The matrix demonstrates that despite efforts to balance type 1 and type 2 errors, the classifier still tended to favor the majority class. Sifting out the 'conditional' cases proved troublesome, with recall only slightly above 15%. The mislabeling of true 'possible' cases as 'present' accounted for 33% of our system's mistakes.

System 2.2, while being much simpler in design, gave by and large comparable results: an accuracy of 0.9347 and a similar contingency table (not shown). Most additional errors were caused by an even stronger preference to label cases as 'present,' the majority class. The higher System 2.1 score indicates that there is still some independency between its stage 1 classifiers.

System 2.3 performed as expected: it increased recall (reflected by higher numbers on the diagonal of table 3) for all five minority classes, even as precision dropped enough to lower micro-averaged accuracy to 0.9271. The average F-score, when calculated per class and then macro-averaged, was 0.774 for System 3, up from 0.753 for System 2.1 and 0.740 for System 2.2.

The post-hoc evaluation of feature group contributions revealed that any one group of features could be removed with only a fairly modest drop in performance; the two strongest contributors were NegEx/ConText and character-n-grams (removal reduced performance with 0.66% point and 0.47% point respectively). Adding token and concept mapping features from the context proved to be of key importance: removal of such context features would reduce the performance to an F-score of 0.9073.

The high scores on this task for all teams in the challenge does not equate to it 'having been solved.' The reason for the high scores can be traced back to class imbalances combined with good performance on the two majority classes. Performance on the classes 'conditional' and 'possible' was far from perfect. It should be noted that optimizing systems on the primary metric (micro-averaged F-score) may cause them to perform suboptimally on minority classes.

During post-challenge analyses, we noticed that one obvious feature, which had been discarded early in the design process, should have been a clear candidate for reintroduction into the feature set but was overlooked. That feature was the ground-truth annotation for *test*, *treatment*, and *problem* concepts from Task 1. By the nature of the task, every token or concept to be classified was a *problem* concept, which by itself would make the feature non-informative. But via the introduction of context features, these labels on neighboring terms *would* contribute information. With that feature group added, system 2.1 was rerun with the system kept exactly as it was in all other respects. This yielded a micro-averaged F-score of 0.9383, or 0.216 percentage-point above the highest reported score.

Table 2 Concepts Task feature contributions

Feature set	Recall	Precision	F	ΔF
All feature sets included	0.8364	0.8688	0.8523	NA
w/o Begin/End, Outside Clusters, Semi-Markov	0.8094	0.8369	0.8229	−0.0294
w/o Begin/End	0.8214	0.8571	0.8389	−0.0134
w/o Outside Annotation	0.8338	0.8660	0.8496	−0.0027
w/o Clusters	0.8348	0.8677	0.8509	−0.0014
w/o Semi-Markov	0.8360	0.8684	0.8519	−0.0004

Table 3 System 2.1 and 2.3 (top/bottom row for each cell) prediction confusion matrix for the Assertions Task, as counts for predictions (rows) and truths (columns)

Prediction	Truth	Absent	Associated with someone else	Conditional	Hypothetical	Possible	Present
Absent		**3370**	20	6	13	14	121
		3409	9	5	12	21	273
Associated with someone else		3	**105**		1		1
		4	**124**		1		2
Conditional		0		**26**	0		1
		1		**44**	2		30
Hypothetical		4			**617**	10	48
		4			**621**	11	53
Possible		14		1	15	**468**	74
		20		0	12	**491**	159
Present		218	20	138	71	391	**12 780**
		171	12	122	69	360	**12 508**

Correct predictions are in bold.

Relations task

System 3.3 achieved an F-score of 0.7313 (table 1) and ranked as the second-highest scoring system in the challenge, although the difference was not statistically significant with the first ranked system.[1] The system improved its exact F-score with each version, with Medline Pointwise Mutual Information providing a 0.3-point gain, and self-training providing an additional 0.4-point gain.

Details about the usefulness of features are shown in table 4. The baseline set of features (a) spans the minimal features among several interpretations of the features used in Patrick and Li.[17] These were enhanced by making the features order- or concept-type sensitive, which identifies, for example, whether a problem appears before a treatment or otherwise (row b). Row (c) shows a further performance improvement when rich text-based features were added. These include sentence-level features such as 'number of concepts in the sentence,' punctuation-related features, word n-grams, the Brown cluster features, and several 'super-features' whose presence nullify all other features. Next, the feature set was augmented with explicit domain knowledge derived from UMLS and MEDLINE (d). Syntactic and dependency parsing further improved the results (e), as did the bootstrapping on the unlabeled data (f).

For all three tasks, this year's i2b2 challenge was found to suit machine learning methods well. The task definitions allowed such methods to be applied in a straightforward uncompromised way. The training data collection was large enough to create stable systems, and the test data collection was adequate to provide reliable performance evaluation. In most cases, the number of classes per decision was small, and the number of examples per class was large enough for training and testing. Finally, imbalances between classes had the potential to have a negative impact on performance. Mostly, the built-in correction mechanisms in the learning algorithms sufficiently dealt with class imbalances; only task 3 required more explicit correction (ie, training data downsampling). As we attempted self-training for all three tasks in the challenge, it is interesting to note that it was successful only for the Relations Task 3. We suspect that this may be because this task had the smallest amount of training data. Ablation tests showed that for all three tasks, the systems could be greatly simplified with only minor and possibly insignificant sacrifices to accuracy.

CONCLUSIONS

This paper addressed three key steps in clinical information extraction: (1) extraction of medical problems, tests, and treatments, from discharge summaries and progress notes; (2) classification of assertions made on medical problems; and (3) classification of relations between medical concepts. For each task, a (semi-)supervised machine learning approach was taken, so that we could leverage a rigorous feature engineering effort. These systems produced a state-of-the-art performance as evaluated in the 2010 i2b2 NLP challenge. Our best-scoring systems ranked highly among the i2b2 participants, with F-scores of 0.8523 on the Concepts Task (ranked first), 0.9362 on the Assertions Task (ranked first), and 0.7313 on the Relations Task (ranked second).

We identified various critical components to which we attribute this strong performance. For medical concept recognition (task 1), we selected a model that enables intraconcept features that cannot be easily formulated in a BIO tagging scheme. The fast and scalable parameter learning that such a model requires was provided here by PA updates. For the concept assertion detection (task 2), an ensemble classifier configuration was still successful in reducing errors, even as performance converged as the feature representation became more sophisticated. In medical-relation detection (task 3), the use of unlabeled data and

Table 4 Performance for feature accumulations in the Relations Task

Feature set	Recall	Precision	F-score
(a) Baseline	0.646	0.718	0.680
(b) +order/type-sensitive	0.672	0.731	0.700
(c) +rich word features	0.681	0.753	0.715
(d) +domain knowledge	0.694	0.750	0.721
(e) +syntax	0.694	0.763	0.727
(f) +unannotated data	0.693	0.773	0.731

syntactic dependency structures was important for improving performance.

There were several common factors between all our systems that are notable. We aimed at broadly expanding the feature representation of the text without applying feature reduction techniques. Rule-based components were used to generate additional features as a preprocess, rather than for post-processing the output of the machine learning system. This architecture allowed the learning systems to self-optimize better. The large feature space dimensionality led us to select machine learning algorithms that scaled up sufficiently and that were not susceptible to overfitting. The strong performance of these learning systems at i2b2 suggests that one should consider these same principles when facing similar tasks on similar datasets.

Acknowledgments L Wei and C Dai provided help with programming system components.

Funding The 2010 i2b2/VA challenge and the workshop were funded in part by the grant number U54-LM008748 on Informatics for Integrating Biology to the Bedside from National Library of Medicine, and facilities of the VA Salt Lake City Health Care System with funding support from the Consortium for Healthcare Informatics Research (CHIR), VA HSR HIR 08-374, and the VA Informatics and Computing Infrastructure (VINCI), VA HSR HIR 08-204. MedQuist co-sponsored the 2010 i2b2/VA challenge meeting at AMIA.

Competing interests None.

Provenance and peer review Not commissioned; externally peer reviewed

REFERENCES

1. **Uzuner O,** South B, Shen S, *et al*. 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. *J Am Med Inform Assoc* 2011;**18**:552—6.
2. **Savova G,** Kipper-Schuler K, Buntrock J, *et al*. UIMA-based clinical information extraction system. In: Proceedings of The Sixth International Conference on Language Resources and Evaluation: Workshop on Towards Enhanced Interoperability for Large HLT Systems: UIMA for NLP. Marrakech, Morocco: Language Resources and Evaluation Conference, 2008:39—42.
3. **Aronson AR,** Lang FM. An overview of MetaMap: historical perspective and recent advances. *J Am Med Inform Assoc* 2010;**17**:229—36.
4. **Harkema H,** Dowling JN, Thornblade T, *et al*. ConText: An algorithm for determining negation, experiencer, and temporal status from clinical reports. *J Biomed Inform* 2009;**42**:839—51.
5. **Miller S,** Guinness J, Zamanian A. Name tagging with word clusters and discriminative training. In: Proceedings of Human Language Technology Conference / North American Chapter of the Association for Computational Linguistics Annual Meeting. Boston, MA, USA: Association for Computational Linguistics, 2004;337—42.
6. **Brown PF,** Della Pietra VJ, deSouza PV, *et al*. Class-based n-gram models of natural language. *Comput Ling* 1992;**18**:467—79.
7. **McClosky D.** *Any Domain Parsing: Automatic Domain Adaptation for Natural Language Parsing*. PhD thesis. Providence, RI, USA: Brown University, 2009.
8. **de Marneffe MC,** MacCartney B, Manning CD. *Generating Typed Dependency Parses from Phrase Structure Parses*. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation. Genoa, Italy: European Language Resources Association, 2006;449—54.
9. **Sarawagi S,** Cohen WW. Semi-Markov Conditional Random Fields for Information Extraction. In: Saul LK, Weiss Y, Bottou L (eds.) *Advances in Neural Information Processing Systems* **Vol 17**. MA, USA: MIT Press Cambridge, 2005.
10. **Collins M.** *Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms*. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Philadelphia, PA, USA: Association for Computational Linguistics, 2002;1—8.
11. **Crammer K,** Dekel O, Keshet J, *et al*. Online passive-aggressive algorithms. *J Mach Learn Res* 2006;**7**:551—85.
12. **Ratnaparkhi A.** *A Maximum Entropy Part-of-Speech Tagger*. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Philadelphia, PA, USA: Association for Computational Linguistics,1996;133—42.
13. **McClosky D,** Charniak E, Johnson M. *Effective Self-Training for Parsing*. In: Proceedings of Human Language Technology conference / North American Chapter of the Association for Computational Linguistics Annual Meeting. New York, NY, USA: Association for Computational Linguistics, 2006:152—9.
14. **Joachims T.** Making large-scale SVM learning practical. In: Schölkopf B, Burges C, Smola A, eds. *Advances in Kernel Method—Support Vector Learning*. Cambridge, MA, USA: MIT Press, 1999.
15. **Chang C-C,** Lin C-J. *LIBSVM: A Library For Support Vector Machines*. 2001. http://www.csie.ntu.edu.tw/~cjlin/libsvm (accessed 21 Jun 2010).
16. http://maxent.sourceforge.net/index.html (accessed 21 Jun 2010).
17. **Patrick J,** Li M. A cascade approach to extracting medication events. In: Proceedings of Australasian Language Technology Workshop. Sydney, Australia: Australasian Language Technology Association, 2009:99—103.