Using Domain Knowledge for Ontology-Guided Entity Extraction from Noisy, Unstructured Text Data

Sergey Bratus
Dept. of Computer Science
Dartmouth College
Hanover, NH USA
sergey@cs.dartmouth.edu

Anna Rumshisky
Dept. of Computer Science
Brandeis University
Waltham, MA USA
arum@cs.brandeis.edu

Rajendra Magar
Dept. of Computer Science
Dartmouth College
Hanover, NH USA
Rajendra.J.Magar@cs.dartmouth.edu

Paul Thompson
Dept. of Computer Science
Dartmouth College
Hanover, NH USA
Paul.Thompson@cs.dartmouth.edu

ABSTRACT

Domain-specific knowledge is often recorded by experts in the form of unstructured text. For example, in the medical domain, clinical notes from electronic health records contain a wealth of information. Similar practices are found in other domains. The challenge we discuss in this paper is how to identify and extract part names from technicians repair notes, a noisy unstructured text data source from General Motors' archives of solved vehicle repair problems, with the goal to develop a robust and dynamic reasoning system to be used as a repair adviser by service technicians.

In the present work, we discuss two approaches to this problem. We present an algorithm for ontology-guided entity disambiguation that uses existing knowledge sources such as domain-specific ontologies and other structured data. We illustrate its use in automotive domain, using GM parts ontology and the unit structure of repair manuals text to build context models, which are then used to disambiguate mentions of part-related entities in the text. We also describe extraction of part names with a small amount of annotated data using Hidden Markov Models (HMM) with shrinkage, achieving an f-score of approximately 80%. Next we used linear-chain Conditional Random Fields (CRF) in order to model observation dependencies present in the repair notes. Using CRF did not lead to improved performance, but a slight improvement over the HMM results was obtained by using a weighted combination of the HMM and CRF models.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval; I.2.6 [**Computing Methodologies**]: Learning

General Terms

Algorithms

Keywords

Text Analysis, Language Models, Information Extraction, Ontology-Guided Search

1. INTRODUCTION

In many specialized areas, domain knowledge is often codified in the form of specialized lexicons and ontologies. At the same time, working domain experts often keep records of their actions in the form of unstructured notes. Using domain knowledge recorded in such form presents a serious problem, since annotating large amounts of unstructured text for machine learning algorithms with domain concepts and semantic types requires extensive human labor.

For example, such is the situation with clinical notes from the electronic patient records. Despite some recent efforts [12], there is a distinct lack of annotated corpora for unstructured medical text. At the same time, a number of lexicons and ontologies are available in the medical domain. For example, the National Library of Medicine maintains a Unified Medical Language System (UMLS) which provides both taxonomy in the form of network of semantic types, and metathesaurus defining and linking relevant concept structures; a SNOMED nomenclature of clinical terms is available from the College of American Pathologists.

A similar situation exists in many manufacturing companies which often maintain relevant ontologies and thesauri for their products. The challenge we discuss in this paper is how to identify and extract part names from technicians repair notes, a noisy unstructured text data source from General Motors' archives of solved vehicle repair problems, with the goal to develop a robust and dynamic reasoning system to be used as a repair adviser by service technicians.

This extraction problem is alleviated by the presence of a comprehensive and actively maintained ontology, which contains a lot of semantic and lexical clues to disambiguating the said notes. *We proceed from the assumption that*

the writer abbreviates (and the reader therefore interprets) the notes based on a shared conceptual context, and that the same context is expressed in the ontology. This allows us to build and compare context models derived from both text units and the ontology, and base semantic disambiguation on an information-theoretic basis, as described in Section 3.

General Motors has numerous structured knowledge sources that are maintained for various purposes; in particular, a taxonomy of part names organized by functional subsumption, which we use as the ontology. The unstructured text data we use comes from GM's massive archives of service technicians' notes on solved repair problems. The overall goal of the project described in this paper is to use these archives and resources to develop robust, dynamic Textual Case-Based Reasoning (TCBR) systems [1, 5] that can be used as repair advisers by service technicians and others. A core challenge is in the taxonomic indexing of repair text archives, so that smart search, e.g., ontology-guided search (OGS), can be used to match the description of the symptoms of a new problem with those of solved problems in the archive. Since classifying and disambiguating text are key elements of the indexing process, it includes a strong element of natural language processing (NLP) and information extraction. For example, "GAS CAP" and "FUEL CAP" are synonyms and should be classified together, but "GAS" by itself requires disambiguation, because it has three distinct meanings in the context of three distinct subsystems: the *powertrain* (as fuel for the engine), the *heating and cooling system* (as refrigerant for the air conditioning), and the *fuel cell power system*, where hydrogen gas is used to generate electricity. Clearly, two kinds of methods must be developed: (1) methods to classify text by locating key phrases on lexical taxonomies, and (2) methods to disambiguate the text by using context to determine which regions of the taxonomies are more likely to be most relevant.

The types of solved-problem archives we have in mind consist of a few structured attributes along with technicians' notes in free text. Existing collections of text annotated by parts of speech, such as the Penn TreeBank, are of little use in analyzing this kind of text. It has a specialized vocabulary and abbreviations. The text blocks are brief, and commonly they do no obey the rules of standard English spelling or grammar. Therefore, certain standard NLP techniques have not been fruitful, and domain- specific alternatives seem necessary. However, we do not have the resources to hand-annotate domain-specific text in some domain-specific way. Since the text consists of ungrammatical fragments with no sentence boundaries for each fragment, it is doubtful if annotating parts of speech would be the right approach. For example, consider the following actual sample from these notes:

```
CUST STATES THE RIGHT SIDE OF
THE HOOD IS SITTING
HIGHER THENA FOUND RT SIDE
OF HOOD LOOSE TIGHTEN
HOOD BOLT
```

For a domain expert, this segment yields the following sentences:

```
1. CUST STATES THE RIGHT SIDE OF THE HOOD
   IS SITTING HIGHER
2. THENA FOUND RT SIDE OF HOOD LOOSE
3. TIGHTEN HOOD BOLT
```

There is a number of anomalies in this segment, in addition to the absence of sentence boundaries. Notice that unstructured notes in different domains carry their own peculiarities. Consider the following excerpt from a free-text note taken from an electronic medical record:

```
At 8:30am SBP 80 and dopamine gtt increased
to 9.5mg/kg. PA catheter placed under fluro
with opening # PA 70/38 PWP 40 with V waves
to 50, CVP 28 with V waves to 38, CO/CI/SVR
7.5/3.5/437
```

While sentensification may be less difficult, syntactic anomalies remain a problem, as does extensive use of domain-specific acronyms and abbreviations. It is clear, however, that the same kind of semantic abbreviation takes place which depends on a shared concept system. By considering such examples, we have been led to consider approaches that do not depend on standard grammar or spelling, but do exploit the domain-specific structures.

The challenge is to use the existing (and maintained) knowledge resources, in lieu of annotated data that needs to be created. For example, there are several structured lexicons of part names. (They are actually names of categories of parts, but we will refer to them as part names.) Indexing with part names may be sufficient for many smart search applications. In other words, if we can find solved problems involving the parts referenced in the new problem, then we have reduced the number of relevant "cases" significantly.

These various domain-specific knowledge resources have good coverage, but they lack the structure and precision needed for TCBR and OGS. Therefore, we have looked at techniques for refining these knowledge resources to make them more consistent, less redundant, and more powerfully structured. We have also developed robust indexing (and search) techniques that are not destabilized by noisy data.

2. OVERVIEW

For knowledge to function as an asset in a corporation, it must be explicit and machine-processible. "Explicit" means that it is "written down," as opposed to carried in people's heads or derivable via data analysis. Knowledge must be machine-processible because the volume of data makes human processing impossible. It is commonly said that 80% of corporate data exists as text: program notebooks; problem summaries in warranty records, technical-assistance center logs, customer surveys, and various other archives of records, logs, and diaries. Most of this text is machine-processible only in a limited sense. For example, key-word search can find items by exact matching, but the slightest differences in expression can cause an item to be missed. Ideally, one would like to ask questions such as "Has this issue come up before?" or "How many times has this issue come up before?"

and get the answer by searching and cross- comparing these various archives. Current tools and methods cannot do this.

An archive of text records of solved repair problems is especially frustrating: One can easily imagine using such an archive to avoid repeating time-consuming diagnostic and repair experiences, but in practice it is difficult to use. This is because of

(1) assumed contextual information that is not written into the record,

(2) the paraphrase problem, and

(3) ambiguous language.

Let us consider our model TCBR application in more detail: A technician with an unsolved repair problem is searching the Warranty Data Archive to determine if his unsolved problem or a similar problem has arisen before. The technician enters the symptoms and wants the records returned in "most relevant symptom" order. After some analysis of such repair records, we have determined that most symptoms have the form: "PART NAME is broken." We have focused our efforts on identifying nd classifying the part-names references in a repair record. These part names have become our surrogate for symptoms.

In Section 3, we describe the algorithm for ontology-guided entity disambiguation. In Section 4, we describe the overall TCBR application In Section 5, we discuss part name extraction, which supports the TCBR application, and present experimental results. Section 6 provides a discussion of these results.

3. ONTOLOGY-GUIDED ENTITY DISAMBIGUATION

In unstructured expert-generated text, domain-specific entities (e.g. parts in automotive domain; diagnoses, test results, therapy protocols in patient records) are referred to by inherently ambiguous short noun phrase mentions that are disambiguated by human readers based both on textual context and on their extra-textual domain knowledge. Automatic indexing and efficient searching of unstructured noisy text corpora require that this disambiguation be performed automatically so that part mentions are extracted and annotated with the respective part identifiers.

In each domain, this knowledge is partly expressed in domain-specific taxonomies and concept systems. For example, medical domain ontologies may classify entities as body structures, clinical findings, procedures and treatments, and so on. In the GM automotive domain, structural ontologies of automobile parts classify parts by functionality, as well as by systems, subsystems and assemblies. Our method uses lexical features derived from these ontologies, together with lexical features from the context surrounding the automotive text mentions, to perform disambiguation.

The principal scheme of our algorithm is as follows. For a noun phrase NP, we define its context C as a collection of features derived from the unit(s) of text containing NP(the sentence, the paragraph, or other structural units if defined; for instance, the headings of the manual section and chapter for NPs in a manual).

The ontology tree T is composed of nodes corresponding to automotive systems, subsystems, assemblies and individual parts. Each node of T has names and descriptions associated with it, which are composed of lexical units (words and stems). For each node N of the ontology tree T we likewise define a collection of features C', derived from the lexical contents of N and its ancestors on the path to the root of the tree, as well as of its siblings and additional lexical units attached to the nodes of the ontology tree T.

Then for all candidate nodes N in T we compute the score $Q(NP, C, N, C')$ and select the node N with the highest score. The function Q is based on information-theoretic measures associated with the lexical units in the tree T, based in their occurrence in the nodes throughout T. Essentially, the measure associated with a single unit (word or stem) expresses the *uncertainty* about the identity of the node N containing that unit once the unit is known. For example, a word or stem that occurs in the names or descriptions of many nodes throughout T has a higher measure of uncertainty than a word or stem that occurs only in a few leaves or branches of T. These uncertainty scores of participating units are then combined to form Q. For further discussion of how this function is defined, see Appendix A.

4. OVERALL TCBR APPROACH

Repair records are usually partly structured, with important information captured as free text. We want a TCBR system that accepts a "query" consisting of the symptoms of a problem and responds with a prioritized list of repair records that have similar symptoms. We want to use the knowledge structures that are available and maintained by the manufacturer. This is a very rich source of already-available knowledge. We do not want to need a significant knowledge-acquisition effort specifically for the TCBR application. We do not want to need to generate annotated text for training ML methods.

The approach we imagine is to use information extraction to create a structured index of the blocks of text included with the repair records. After a preliminary analysis, we decided that we could focus (for now) on "part names." We want to identify and classify part names in text. It is through the classification of part names that both indexing and similarity are defined. The indexing process would include cleaning the text, extracting the part names, and mapping the part names into existing taxonomies of part names. Thus, the values for the slots in the index would be taken from one or more existing parts taxonomies. Similarity would be defined from the taxonomic structures. A characteristic of the domain is that these part names and their structures are constantly changing. Thus, the system would require a certain robustness.

Knowledge resources created for day-to-day purposes in a company can often be adapted for decision support. Uschold noted that a company's glossary or thesaurus could be adapted to become a semantic net [11]. Any manufacturing company, such as General Motors, has numerous lists and lexicons re-

lated to the names associated with the design, engineering, manufacturing, and servicing of its products. Our key example are the lists of automotive parts that are referenced in servicing vehicles. We might have focused on lexicons for design, engineering, or manufacturing, in developing decision-support tools for those areas. An example for manufacturing is the GM Variation-Reduction Adviser [6, 7].

One important consideration for TCBR is that the objects of interest be structured into ISA and PART-OF taxonomies (and perhaps other relations). This is because indexing text leads to a desire to generalize and specialize to solve the paraphrase problem and the disambiguation problem. One natural context for an object in a taxonomy is the path from the node where the object is named to the root, as well as the descendants of the node. If the same name is used in several nodes (e.g., "GAS"), then the choice of node is the disambiguation of the name, and matching the textual context with the taxonomic context is one way to choose the most relevant node.

The kinds of lexicons and lists we used were:

- T, a taxonomy of part name categories. The relations defined by the links has varying meaning, usually ISA or PART-OF.
- a list of standard abbreviations,
- an engineering glossary,
- L, a list of labor code descriptions, and
- a mapping of the elements of L to the elements of T.

5. PART NAME EXTRACTION FROM NOISY TEXT

First, let us clarify what we mean by part names. We are interested in categories of parts, such as "OIL PAN." There are many kinds of "actual" oil pans, and each is identified by a unique name (or rather a combination of data, including a part number, supplier, date of manufacture, and other identifying information). We find it convenient in this paper to refer to OIL PAN and other part-name categories as "part names," but we are always referring to categories. We want to be able to identify part names in text. We want to classify the discovered part names by mapping them to T. In order to accomplish this, we need to consider which words and two-word phrases of the name are semantically "informative" and which are not. In this paper we focus on identifying part names. In the case that we have grammatical text, we would expect to be able to extract noun phrases (NPs) from the text using NLP techniques as a starting point for identifying part names. Then we could determine by attempting to map each NP to T whether the NP is a part name or some other object (e.g., CUSTOMER).

However, the structure of part names can help us identify them, even when the text is fairly ungrammatical. For concreteness, imagine that a technician is having trouble with the "left outside rear view mirror." Naturally, he might look for this exact string, but a verbatim in a warranty record might describe the "LEFT OUTSIDE REAR VIEW MIRROR" in different ways. The most obvious variants are created simply by omitting some of the qualifying adjectives, which is often done when the context is assumed. Further, vehicles have a bilateral symmetry, so that many parts come in a left (driver's side, etc.) version and a right (passenger's side, etc.) version. Sometimes the difference will be important to the problem-solving potential of a warranty record, but often it is not.

The semantic head of the above phrase is "MIRROR," which in this case is the syntactic head. Mirrors in vehicles are either "rear view" or "vanity," but vanity mirrors can be only inside. Thus, technicians might neglect to add the qualifying phrase "rear view" if they have already notes "outside." It might not even make sense for the search to favor these kinds of qualifying adjectives, because such qualifiers are often omitted in technician's notes. Thus, while if they are present, they add to the information gain of the phrase, if they are not present, there is no information loss. In other words, it means nothing; it is a consequence of the "assumed context," which is so common in such notes. A hurried technician might simply have written "OUTSIDE MIRROR" knowing that vanity mirrors are never outside and the left-right distinction is not important for the particular write up. The essential structure of the NP is "MIRROR," which is essentially a concept class, along with enough qualifiers to distinguish it from all other mirrors. There is (at least one) taxonomic tree implicit in this analysis.

The training data for part name identification consists of repair notes where part names have been labeled. For the evaluation, we hand-labeled 1,000 randomly sampled repair notes. We divided the repair notes into two sets of 500 each, one for evaluation during development and the other for final testing. We also used approximately 30,000 part name phrases and a lexicon of 1,600 part name words collected from these phrases during training. For testing we used five-fold cross validation.

The structure of our HMM consists of "target" and "non-target" states, s_i. Part names correspond to the "target" state of the HMM. The non-target states are Start, Prefix, Suffix, Background, and End. The Prefix state corresponds to a fixed-length sequence of words before the target words. Similarly, the Suffix state corresponds to a fixed-length sequence of words following the target words. The remaining words are thought of as being emitted by the Background state. The probabilities are estimated as ratios of counts. The transition probability $P(s_j|s_i)$ is calculated as the total number of (s_i, s_j) label pairs divided by the total number of s_i labels in the training data. The emission probability $P(w|s_i)$ is calculated as the number of w labeled as s_i divided by the total number of s_i labels. One of the issues we have had to face is getting sufficient labeled data for training. This has had implications for the effectiveness of HMM structure used to model our data [2, 3]. Our HMM is complex with as many states as possible and is able to capture the intricate structure of the data in use; however, it results in poor (high variance) parameter estimation because of the sparseness of training data. In contrast, simpler models with fewer states, while giving robust parameter estimates, are not expressive enough for data modeling. In

order to strike a balance, we used a statistical smoothing technique called "shrinkage" to combine the estimates from these models of differing complexity. Freitag and McCallum [3] report positive results using "shrinkage" to perform information extraction using HMMs. Some states from a complex model are shrunk to a common state to form a new HMM structure – hence the term shrinkage. To further improve parameter estimates, states from the new HMM can be further shrunk to form another HMM with even fewer states, thus forming a shrinkage hierarchy. In our case, we shrunk the Prefix and Suffix states to a common Context state. We then employed another level of shrinkage, in which all the states were shrunk to a single state.

The recall and precision scores from the five-fold cross validation along with the F-score are presented below. Table 1 shows results for the fully expressive model alone, as well as for the optimal shrinkage mixture of three HMM models.

	fully expressive model	optimal shrinkage mixture
Average Recall	12.26	81.64
Average Precision	79.85	79.45
F-Score	21.26	80.53

Table 1: Five fold cross-validation results for (1) the fully expressive model and (2) optimal shrinkage mixture, with a context width of two.

The fully expressive model has poor recall though precision is good. This indicates that the model by itself is not sufficient to cover all part names. The F-score of the fully expressive model is low. On the other hand, using shrinkage with optimal mixture weights improves recall values substantially while maintaining high precision. The substantial improvement in recall indicates that the shrinkage mixture helps to smooth parameter estimates to expand coverage of part names not handled by the fully expressive model alone.

We have investigated the effectiveness of using HMMs with shrinkage for part name extraction and found that HMMs do well modeling the repair notes as shown by an F-score that hovers around 80%. Next, we sought to improve performance on the remaining 20% of the part names that were missed or incorrectly labeled by HMMs by introducing a more flexible model called Linear-Chain Conditional Random Field [9, 10]. We hypothesized that the errors not handled by HMMs could be handled using the observation dependencies found in repair notes. It is desirable to integrate these dependencies into the models to improve overall classification accuracy.

We used unigram and bigram features. Unigram features are obtained using the identity of the current word. For each word seen in the training data, the unigram feature value is assigned to 1. Bigram features use previous and current word.

Contrary to our original hypothesis, extraction using a CRF did not outperform HMM with Shrinkage, although CRF does perform substantially better than the HMM without shrinkage.

Average Recall	85.41
Average Precision	74.86
F-Score	79.78

Table 2: Five fold cross-validation results for CRF with a context width of two.

Analyzing the misclassifications of the HMM and the CRF, we noticed that there was a fair amount of difference in which items were misclassified. This suggested a weighted combination of the two techniques might improve performance. To combine the two models, we merged Viterbi search [8] in both the HMM and the CRF using a weighted combination.

Our results from the weighted combination of the HMM and the CRF are shown in Table 3. There is some improvement in overall score, but there is little improvement over either model alone.

Average Recall	84.35
Average Precision	79.64
F-Score	81.93

Table 3: Five fold cross-validation results for HMM+CRF with a context width of two.

6. DISCUSSION

When we used fully expressive HMMs performance was poor, because of an insufficient amount of training data. With shrinkage performance dramatically improved. We thought that we could make further improvement using CRFs, since, unlike HMMs, CRFs would allow us to model observation dependencies. However, CRF did not outperform HMM with shrinkage. Since the two approaches misclassified different part names, it seemed that it might be possible to find an optimal way to combine the two approaches to obtain better performance than with either approach by itself. We performed this combination by merging the Viterbi search used by both HMMs and CRFs. We did this by using the Expectation Maximization algorithm to estimate the optimal weights. We achieved some improvement in overall score, but did not make a substantial improvement over either model alone. The reason is that we use the same pair of weights for each token position during the Viterbi search. Ideally during the Viterbi search it would be desirable to provide a weight of 1 to the model that correctly labeled the token in question and a weight of 0 to the model that did not provide the correct label. If we had an algorithm that could correctly select the appropriate model at each token position, the best f-score we would get is 86.62%. (We found this score by tallying against the labeled version of the test set.) Finally, it is likely that all of the approaches that we tried would have done better had more training data been available.

7. CONCLUSIONS

Textual Case-Based Reasoning is a promising technology for organizations with large amounts of textual information and taxonomies, such as the part name taxonomy described here, which can be used to aid the case-based reasoning application. When, as is the case here, that extraction is from noisy

data, effective techniques must be found to extract information, here part names, to support TCBR.

While corpora linguistic techniques can perform well with large amounts of labeled training data, many organizations with textual data are not in a position to develop these large training sets. HMMs with shrinkage and CRF approaches, as described in this paper show how extraction can be successful with much smaller training sets. The benchmarks obtained with TCBR using relatively small amounts of annotated data should be used in a comparison against the algorithm that uses exclusively domain ontology and structured data resources to guide part name extraction and identification.

APPENDIX

A. ONTOLOGY-GUIDED DISAMBIGUATION ALGORITHM

Let tokenization of paragraph P produce a sequence of tokens $\{t_i\}$. If sentence boundaries are available, we further subdivide this sequence into groups by their respective sentences. Let $\{u_i\}$ be the sequence of lexical units derived from $\{t_i\}$ after tokenization. The lexical units are derived by a transformation according to a dictionary D of abbreviations and multi-token terms; some tokens are expanded to several lexical units, some are stemmed, the stem replacing the token in its place in the sequence, some multi-token groups collapsed to a single unit. This transformation from sequences of tokens to sequences of standardized lexical units will be referred to as $U : \{t_i\} \rightarrow \{u_i\}$.

For a noun phrase NPthat spans lexical units $u_k, \ldots, u_{k+l}$, we define the *primary context* C_{NP} as the collection of units $u_1, \ldots, u_{k-1}, u_{k+l+1}, \ldots$. If the phrase NP occurs in structured text such as a manual, we also add to this primary context the units of text elements structurally related to P, such as headings under which P occurs. As an extension of this approach, we assign to each lexical unit u in C_{NP} a *weight* depending on u's position relative to NP(e.g., decreasing with the distance from u to NP, counted within P in the number of lexical units separating them, and outside P in the number of structural elements separating the current element from P). Denote this distance $d(u, NP)$ and the corresponding weight $w_{d(u,NP)}$.

For a candidate node N of T, define the *ontology context* C'_N of N as consisting of lexical units derived according to the rules of the transformation U from the names and descriptions attached with the ancestors and siblings of node N. Each unit u coming from a node M_u is associated with a weight based on the distance $d'_T(N, M_u)$, counted in the number of nodes separating N and M on the path to the root of T, with a special distance d_s fixed for siblings of N. Denote the corresponding weight $w_{d'_T(N,M_u)}$.

We define the score $Q(NP, C_{NP}, N, C'_N)$ as follows:

$$Q(NP, C_{NP}, N, C'_N) =$$

$$= \sum_{u \in \mathfrak{C}(NP,N)} w_{d(u,NP)} \cdot w'_{d_T(N,M_u)} \cdot (H(T) - H(T_{U(\{NP\})}))$$

where $\mathfrak{C}(NP, N) = (U(\{NP\}) \cup C_{NP}) \cap C'_N$, $W = U(\{NP\})$ is the set of lexical units derived from NP, T_W is the pruned subtree of T as described above, and $H(T_W)$ is the entropy measure associated with it as described above.

B. REFERENCES

[1] Bruninghaus, S. and Ashley, K. D. 2005. Reasoning with Textual Cases Proceedings of the International Conference on Case-Based Reasoning (ICCBR), 137-151.

[2] Freitag, D. and McCallum, A. 2000. Information Extraction with HMM Structures Learned by Stochastic Optimization. In Proceedings of the Seventeenth National Conference on Artificial Intelligence, AAAI, 584-589.

[3] Freitag, D. and McCallum, A. 1999. Information Extraction with HMMs and Shrinkage. In Papers from the AAAI-99 Workshop on Machine Learning for Information Extraction, 31-36, July. AAAI Technical Report WS-99-11.

[4] Lafferty, J., McCallum, A., and Pereira, F. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In Proc. 18th International Conference on Machine Learning.

[5] Lenz, M. 1998. Textual CBR and Information Retreival: A Comparison. In Gierl, L. and Lenz, M. (eds.) Proceedings of the 6th German Workshop on Case-Based Reasoning, IMIB Series vol. 7, Inst. fuer Medizinische Informatik und Biometrie, University of Rostock.

[6] Morgan, A. P., Cafeo, J. A., Gibbons, D. I., Lesperance, R. M., Sengir, G. H., and Simon, A. M. 2003. The General Motors Variation-Reduction Adviser: Evolution of a CBR System. ICCBR 2003, 306-318.

[7] Morgan, A. P., Cafeo, J. A., Godden, K., Lesperance, R. M., Simon, A. M, McGuinness, D. L., and Benedict, J. L. 2005. The General Motors Variation-Reduction Adviser. AI Magazine 26,3, 18-28.

[8] Rabiner, L. R. 1989. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In Proceedings of the IEEE, 77, 2.

[9] Sha, F. and F. Pereira. Shallow Parsing with Conditional Random Fields. Technical Report MS-CIS-02-35, University of Pennsylvania (2003)

[10] Sutton, C. and McCallum, A. 2006. An Introduction to Conditional RandomFields for Relational Learning. In Introduction to Statistical Relational Learning. Getoor, L. and BenTaskar, B. (eds.) MIT Press.

[11] Uschold, M. 2000. Creating, Integrating and Maintaining Local and Global Ontologies. Proceedings of the 14th European Conference on Artificial Intelligence ECAI 2000, Berlin, Germany.

[12] Roberts, A., R. Gaizauskas, M. Hepple, N. Davis, G. Demetriou, Y. Guo, J. Kola, I. Roberts, A. Setzer, A. Tapuria, et al. 2007. The CLEF corpus: Semantic annotation of clinical text. In AMIA Annu Symp Proc, volume 625.