Extending the Semantics in Natural Language Understanding

Michael Marlen
Kansas State University
msm6666@ksu.edu

David Gustafson
Kansas State University
dag@ksu.edu

Abstract

Natural Language understanding over a set of sentences or a document is a challenging problem. We approach this problem using semantic extraction and building an ontology for answering questions based on the data. There is more information in a sentence than found by extracting out the visible terms and their obvious relations between one another. Keeping track of inferences, quantities, inheritance, properties, and set related information is what gives this approach the advantage over alternatives. Our methodology was tested against the FraCas Test Suite with near perfect results for the sections: Generalized Quantifiers, Plurals, Adjectives, Comparatives, Verbs, and Attitudes. The results indicate that extracting the visible semantics as well as the unseen semantics and their interrelations using an ontology to logically provide reliable and provable answers to questions validating our methodology.

1 Introduction

There has yet to be a system that is fully capable of understanding English. We define understanding as the ability to reason by successfully mapping an ontology onto a preexisting ontology built from the premises. This is demonstrated using the FraCaS Test Suite problems that are presented in English (Cooper, 1996). Sukkarieh (2003) showed that the FraCas Test Suite is widely regarded as the gold standard for Natural Language Understanding systems. Our research moves closer to solving the problems presented in the FraCas Test Suite by allowing for multiple premises to be presented using an open world framework.

Our system takes multiple premises and attempts to answer a question correctly based on the premises.

We understand that with Natural Language Understanding, appropriate domain knowledge is important. So background knowledge (additional premises) for certain problems are provided as natural language. The assumption for our work is that there is sufficient domain knowledge available to interpret the semantics of the propositions. This would be needed for any test set where the set of premises does not describe some of the relationships that are generally understood to be known by a human reader. We'd like to be able to obtain domain knowledge and general knowledge from reading internet sources, such as Wikipedia. Currently, we just provide background domain information to the system as part of the problem statements such as those contained within the FraCaS Test Suite.

The system currently focuses on the language contained within the FraCaS Test Suite. In addition, our work only considers the subset of natural language (English) from which a parser can produce a valid grammar tree from problems contained within the FraCaS Test Suite. This subset allows us to test what is possible for our methodology while not having to deal with an invalid or incomplete parse. While it is not the focus of this paper, there are ways of ruling out particularly bad parses, such as when the StanfordNLP parser produces an incomplete parse tree. If the premises are unable to be parsed successfully the user could be asked to reword the premises and / or question and try again. Currently, we ignore these problems.

The reason for this is to identify if it is possible to generate sufficient knowledge to reason over to be able to answer the questions contained within the test suite and if so, it could be extended further to be tested against other test suites or even more real world scenarios.

2 Related Work

There is work in many areas in Natural Language Understanding, from statistical analysis of language (Manning et al., 1999), to predicate logic based systems, or natural logic (MacCartney et al. 2007). The first type of system, the statistical based, comes in many different varieties such as feature analysis, Bayesian priors, domain-based features, etc. (Rosario et al., 2001; Pantel et al., 2006; Nastase, 2006; Turney, 2010). There is a problem with the prepositional logic type systems as well. Those systems only work in the realm of true and false and do not leave any room for non-Boolean related queries. Natural

Logic requires both premises and a working hypothesis to try to find an answer through entailment checking the validity of the statement.

Other work on textual entailment includes (Dagan et al. 2005; Giampiccolo et al. 2007).

There is work in understanding the semantic meaning and modeling semantics as shown in (Grefenstette, 2011; Baroni, 2010; Mitchell, 2010).

Additionally, there is work using ontologies to learn from text as shown in (Buitelaar, Cimiano, Magini 2005). Our work draws on the layered cake approach presented in their book.

Other research areas include entailment inference (Schubert et al., 2010) and the use of episodic logics (Schubert et al., 2000), as well as relationship extraction done by Romano (2006).

The FraCaS Test Suite contains 346 NLI problems, divided into nine sections, each focused on a specific category of semantic phenomena (Cooper, 1996; MacCartney et al., 2008). MaCartney and Manning achieve rather good results, however they removed problems with multiple premises as well as those without a hypothesis (MacCartney et al., 2007; MacCartney et al., 2008). MacCartney's work, worked well with single premise statements.

3 Methodology

The best way to understand how the system works is by taking a look at the high level algorithm shown in Figure 1. This depicts the steps the system must take to achieve an understanding.

```
1. For each premise: Parse the premise and generate
   ordered list of grammar trees
   a. For each grammar tree for a given premise[1]
      i. Generate intermediate object by pattern
         matching each set of children for all
         non-leaf nodes[2] //These intermediate ob-
         jects will hold additional generated in-
         formation
      ii. Normalize words; nouns become singular,
         verbs become present tense[3]
      iii.While there are changes to be made
         1. Apply POS/word rules to intermediate
            object.
         2. Push information into temporary ontol-
            ogy
         3. Type match as needed (notably for
            verbs)
         4. Build relationships
         5. Push relationships into temporary on-
            tology
      iv. Merge temporary ontology into main ontol-
         ogy
         1. Find matches
      v. Generate new information based on struc-
         ture of main ontology
      vi. Clear temporary ontology
2. For the question follow steps 1.a.i-iv
3. Find an answer to the question yes/no/unknown by
   matching the temporary ontology to the main on-
   tology
```

Figure 1. High level view of methodology

The first step towards understanding English using our methodology is to acquire an annotated tree parse of the English statements. OpenNLP (Baldridge, 2005) and StanfordNLP (Toutanova, 2009) are used to acquire the annotated parse. Using them together we get a higher number of acceptable parses.

Given a grammar tree from the parsers mentioned above, pattern matching tells us the type of intermediate object we must instantiate. The intermediate object represents a sub-tree within the grammar tree. It holds information for that particular sub tree. Additional information will be added based on pre-determined rules derived from the language contained within the FraCaS Test Suite. The intermediate object specifies how words relate to one another.

Nouns and verbs are normalized, to assist in matching. All nouns become singular and a quantity attribute that indicates the number or range of elements is attributed to it. Depending on whether the noun was a proper noun or not helps indicate if it was an instance or a class as far as the ontology is concerned. All verbs become present tense and gain a time component, indicating if they occurred past, present, future, etc. A time component is attached to verb predicates is to maintain information as well as infer time based semantics.

Intermediate objects for verbs are similar to a predicate logic. Parameters for a predicate tuple can be either a reference to a noun object or a pointer to another predicate. If it is a pointer to another predicate, think of it as a way to link a verb phrase that has a noun with a prepositional phrase where the preposition is the predicate of another tuple. Other predicates are keywords that describe the action the system should take upon further analysis of said predicate. A unique identifier is added to instances and classes when created, to differentiate between similarly named instances and classes.

After a premise has been processed this temporary ontology is merged into the main ontology. If it was a question it stays as a temporary ontology for analysis in step 3 as shown in Figure 1. When there is no information in the main ontology, the temporary ontology becomes the main ontology. In a more interesting case, instances and classes must be matched against preexisting instances and classes that exist in the ontology. When a match is found, all elements that related to the instance or class in the tempo-

[1] A grammar tree is valid when all sub steps are completed successfully

[2] If there is a set of children where there is no match in the grammar tree restart loop starting on next grammar tree

[3] This information is maintained for nouns to keep track of the quantity, the information is needed for verbs to maintain a partial ordering on the information as it is presented

rary ontology is remapped to point to the item in the main ontology instead.

Step 1.a.v checks every element in the ontology to see if additional information can be generated that is factually true about the currently known information. For example; if there is an *instance contract* in the ontology that represents only 1 *contract* then clearly there is the class *contract* that should exist which represents the set of all instances of *contract*. If there exists a *class contract* with a quantity 1 that has no parent class then it would be true that there is another uniquely identified *class contract* that contains the quantity that is set to 'all' which represents all contracts that can exist.

The same process can be done for an instance that contains a property about the instance. Facts are generated similarly for *contract*; in addition there is also the set with the attribute propagating up the hierarchy where each parent that has the attribute is also a child of the same class without the attribute.

When a question is input to the system the previous steps are taken as indicated above except the temporary ontology isn't merged or cleared. The problem then becomes to find a satisfiable mapping from the temporary ontology to the main ontology. Every object in the temporary ontology tries to find the potential matches it has in the main ontology. For some matches, a temporary set of instances may need to be collected e.g. Figure 7. The system looks at each tuple and evaluates it to be true, false, or unknown depending on the information in the main ontology. The temporary ontology from the question is evaluated for every instance and class and all connections are formed to the main ontology. Using these connections, an attempt to replace the temporary ontology instance or class with each specific related term. At least as far as these problems are concerned, there is only one solution that can be found if it is either *true* or *false*.[4] *Unknown* is the case where no such replacement was found to satisfy a particular relation. The process is to evaluate every relation under this assumption. If a result of either true or false is produced then that is the answer to the question and it returns. However, if it returns unknown then it continues to change another term and repeats this process until no more configurations are possible in which case the answer is truly unknown.

Figure 2 shows one of the problems evaluated using the system, based on the methodology mentioned above.

[4] Some premises and questions can have multiple interpretations, our software picks one (has programmed bias).

Smith signed one contract.

Jones signed another contract.

Did Smith and Jones sign two contracts?

Figure 2. Problem 111 from the FraCas Test Suite

Starting with the first premise in Figure 2, the system generates the main ontology shown in Figure 3.

sign<past tense, t+1>(<Instance: QTY 1>SMITH_1, <Instance: QTY 1>CONTRACT_2)

Figure 3. Main Ontology for premise 1

Figure 4 shows new facts that are generated from the main ontology.

instance_of(<Instance: QTY 1>SMITH_1, <Class: QTY 1>SMITH_1)
instance_of(<Instance: QTY 1>CONTRACT_2, <Class: QTY 1>CONTRACT_2)
subset_of(<Class: QTY 1>SMITH_1, <Class: QTY ALL>SMITH_3)
subset_of(<Class: QTY 1>CONTRACT_2, <Class: QTY ALL>CONTRACT_4)

Figure 4. New facts generated from Main Ontology

The second premise from Figure 2 generates the following facts shown in Figure 5.

sign<past tense, t+2>(<Instance: QTY 1>JONES_3, <Instance: QTY 1>CONTRACT_4)

Figure 5. New facts added to main ontology from premise 2 seen from Figure 1

When generating new facts based on the now updated main ontology, everything follows as normal for Jones. However, for *instance contract*, there is a *class contract* in the main ontology with a quantity set to one, a direct match. No additional information is generated as it already exists.

instance_of(<Instance: QTY 1>JONES_3, <Class: QTY 1> JONES_5)
instance_of(<Instance: QTY 1>CONTRACT_4, <Class: QTY 1>CONTRACT_2)
subset_of(<Class: QTY 1>JONES_5>, <Class: QTY ALL>JONES_6)

Figure 6. New facts generated based on main ontology

sign<past tense, t+3>(<Instance: QTY 1>SMITH_5,<Instance: QTY 2>CONTRACT_6)
sign<past tense, t+3>(<Instance: QTY 1>JONES_6,<Instance: QTY 2>CONTRACT_6)

Figure 7. Facts in temporary ontology for question

Answering the question becomes an exercise in mapping ontologies and checking the predicates. Every instance/relation from Figure 7 must map to another instance/relation in the main ontology. In this case, each relation is satisfied if replacements for both instances can be found. Another condition on the relation must be satisfied by looking at the time component. Not only must there exist a relation *sign* that has both instances but, that relation has to hold true before time (t+3). When trying to find a match, it first attempts to match relation name, which it finds, then check to see if the time component matches, which in this case is satisfied. Next, it checks the first term in the tuple, which for both it finds a valid replacement. However, for the *contract* with QTY 2, no match is found. The process is then to generate all sets that contain instances that are subsets of the instance *contract* that add up to QTY 2, and then try to apply each element within the set to see if it is a valid replacement. All elements in this temporary set must be used. Since all relations were successfully evaluated to true the result to the question is therefore yes.

4 Evaluation

The FraCaS Test Suite contains 346 NLI problems, divided into nine sections, each focused on a specific category of semantic phenomena (Cooper, 1996; MacCartney et al., 2008). For comparison to previous work, we will not remove multiple-premise problems, or problems that are missing a hypothesis as done in (MacCartney et al., 2007; MacCartney et al., 2008). No modification to the test set has been made to accommodate my research. However, we do remove problems from the test suite that contain a bad parse on any one of the premises for the problem or the question. We will show a comparison based on percentage of problems that are answered correctly. This research focuses on six sections which represent Generalized Quantifiers, Plurals, Adjectives, Comparatives, Verbs and Attitudes respectively.

Table 1 show that the system performs exceptionally well. The accuracy is calculated based on the correct answer and remaining problems. There is one critical thing to be taken from this, that while this methodology is fully capable of solving these problems, obtaining a valid part of speech tree for each premise and question in each problem is paramount.

Section	Original Problems	Bad Parses	Remaining Problems	Correct Answer	Acc %
1	80	10	70	61[5]	87.00
2	33	11	22	21[6]	95.45
5	23	7	16	16	100.00
6	31	9	22	22	100.00
8	8	4	4	4	100.00
9	13	6	7	7	100.00
Total	188	47	141	131	92.90

Table 1. Results

[5] The system realized that it could not answer the 9 questions out of the 70 remaining problems for section 1 so it produces a null answer; we count null answers as wrong.

[6] It can solve problem 87 from the test suite but due to this it cannot solve problem 88 due to word play.

When our methodology is compared against the semantic containment and exclusion method as seen in (MacCartney et al., 2008) we clearly see that when statements are analyzed in depth we gain greater accuracy overall, as shown in Table 2. With the notable exception to the first section, generalized quantifiers, where the system does not yet support the language contained in 9 of the problems despite it producing a valid parse. In addition to a higher accuracy rate on the FraCaS Test Suite we also are capable of working with problems that contain multiple premises.

	Problems	Acc%
Most common class ‘yes’	178	51.68
MacCartney 07	108	75.00
Natlog	108	87.04
This system	137	92.27

Table 2. Performance on FraCaS problems on sections: 1, 2, 5, 6, 9 compared

In total, the results mean that where our system supports the language, the system works well. The exception is when multiple problems in the test suite are the same but can be interpreted differently.

5 Conclusion

Making machines understand natural language at any level is a challenging problem. We've developed a methodology that converts natural language into ontology while leveraging the ontology to help solve questions posed in natural language about the facts in the ontology. We've shown that our methodology which works around extending the semantics of language, by keeping track of inferences, quantities, inheritance, properties, and set related information, produces a high degree of accuracy. Using more information than is directly seen in the statements allows us to help match terms in a natural way, which allows for questions to be proved correct (yes or no) or unsolvable (unknown).

6 Future Work

The next logical step is to see how well our methodology adapts and performs to the other sections that are not addressed in this paper. Also, there is a maximum of just five premises in the largest problem in this problem set; analyzing a full page document is a direction that needs to be pursued.

References

Baroni, Marco, and Roberto Zamparelli. "Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space." Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2010.

Buitelaar, Paul, Philipp Cimiano, and Bernardo Magnini, eds. Ontology learning from text: methods, evaluation and applications. Vol. 123. IOS press, 2005.

Dagan, Ido, Oren Glickman, and Bernardo Magnini. "The pascal recognising textual entailment challenge." Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment. Springer Berlin Heidelberg, 2006. 177-190.

Giampiccolo, Danilo, et al. "The third pascal recognizing textual entailment challenge." Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing. Association for Computational Linguistics, 2007.

Grefenstette, Edward, and Mehrnoosh Sadrzadeh. "Experimental support for a categorical compositional distributional model of meaning." Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2011.

J. Baldridge. 2005. The OpenNLP project, http://opennlp.sourceforge.net/

J. Sukkarieh. 2003. Mind your Language! Controlled Language for Inference Purposes. Oral presentation, Presented at the Joint Conference of the Eighth International Workshop of the European Association for Machine Translation and the Fourth Controlled Language Applications Workshop, Dublin, Ireland.

K. Toutanova. 2009. The Stanford NLP Group Part-of-Speech Tagger, http://nlp.stanford.edu/software/tagger.shtml

MacCartney, Bill and Christopher D. Manning. 2007. Natural logic for textual inference, In ACL-07 Workshop on Textual Entailment and Paraphrasing, Prague.

MacCartney, B. and Manning, C. D. 2008. Modeling semantic containment and exclusion in natural language inference. In *Proceedings of the 22nd international Conference on Computational Linguistics - Volume 1* (Manchester, United Kingdom, August 18 - 22, 2008): 521-528.

Manning, C. D., and Schütze, H. 1999. Foundations of Statistical Natural Language Processing. Cambridge, MA: The MIT Press

Mitchell, Jeff, and Mirella Lapata. "Composition in distributional models of semantics." Cognitive Science 34.8 (2010): 1388-1429.

Nastase, Vivi, et al. "Learning noun-modifier semantic relations with corpus-based and WordNet-based features." Proceedings of the National Conference on Artificial Intelligence. Vol. 21. No. 1. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

Pantel, Patrick, and Marco Pennacchiotti. "Espresso: Leveraging generic patterns for automatically harvesting semantic relations." Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics. Association for Computational Linguistics, 2006.

R. Cooper, R. Crouch, J. Van Eijck, C. Fox, J. Van Genabith, J. Jaspars, H. Kamp, M. Pinkal, D. Milward, M. Poesio, and S. Pulman. 1996. Using the Framework. Technical report, FraCaS: A Framework for Computational Semantics, FraCaS deliverable D16.

Romano, Lorenza, et al. "Investigating a generic paraphrase-based approach for relation extraction." Proceedings of EACL. Vol. 2006. 2006.

Rosario, Barbara, and Marti Hearst. "Classifying the semantic relations in noun compounds via a domain-specific lexical hierarchy." Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (EMNLP-01). 2001.

Schubert, Lenhart K., and Chung Hee Hwang. "Episodic Logic meets Little Red Riding Hood: A comprehensive, natural representation for language understanding." Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language (2000): 111-174.

Schubert, Lenhart K., Benjamin Van Durme, and Marzieh Bazrafshan. "Entailment inference in a natural logic–like general reasoner." Proc. of the AAAI 2010 Symp. on Commonsense Knowledge. 2010.

Turney, Peter D., and Patrick Pantel. "From frequency to meaning: Vector space models of semantics." Journal of artificial intelligence research 37.1 (2010): 141-188.