Natural Logic in AI and Cognitive Science

Lawrence S. Moss
Indiana University
lmoss@indiana.edu

Michael Wollowski
Rose-Hulman Institute of Technology
wollowsk@rose-hulman.edu

Abstract

This paper presents an ongoing research project called "natural logic" and makes the case that it is relevant to AI, Computational Linguistics, and Cognitive Science. We propose to add some of the natural logic modules which have already been developed to existing NLP systems. We see our approach as complementing and augmenting data-driven approaches exemplified by IBM's Watson. We give a brief introduction to natural logic and present examples of proofs that can be given in a working system. We furthermore introduce monotonic logic, another promising approach for extracting information from sentences that contain quantifiers. We finish the paper by presenting some early work that integrates syllogistic reasoning into exsiting NLP systems.

Introduction

The history of logic and AI is a checkered one. Starting with huge optimism, the idea of applying logic in AI and NLP is very much a minority one today. The 2015 report of the *One Hundred Year Study on Artificial Intelligence* states that "The resounding success of the data-driven paradigm has displaced the traditional paradigms of AI. Procedures such as theorem proving and logic-based knowledge representation and reasoning are receiving reduced attention, in part because of the ongoing challenge of connecting with real world groundings." (Stone *et al.* 2016). At the same time, there were important contributions in previous parts of the AI literature on the topic of reasoning with fragments of natural language. Important for our story are (Nishihara *et al.* 1990; McAllester and Givan 1992; Purdy 2006).

We propose to add syllogistic reasoning to NLP systems. The sort of system we have in mind is exemplified by IBM's Watson system. Jim Hendler and his research group (Hendler and Ellis 2014) proposed that Watson's accomplishments are due to a combination of natural language processing, search technologies, semantic typing, scoring heuristics and machine learning. Conspicuously absent from this list are reasoning and logic. We know that several kinds of reasoning took place behind the scenes, such as in the context of semantic typing and in the context of reasoning about time and place (Kalyanpur *et al.* 2012; Ferrucci 2012; Gliozzo *et al.* 2013). While some of the reasoning in Watson was in support of finding answers to questions and to establish the confidence of potential answers, the geospatial and temporal reasoning were designed to make explicit some of the information that was implicitly contained in the given information. In this context, our project is aimed at adding addition reasoning components to NLP systems such as Watson.

We see our project as a support system that is designed to extract yet more information from natural language texts. It is decidedly not a free standing reasoner. As such, we see our project as part of a research agenda implicitly defined by IBM Watson, i.e. to do better than the Jeopardy! system. Watson used some fairly off-the-shelf IR techniques, including passage term matching, textual alignment and skip bigrams (Murdock *et al.* 2012). The designers of Watson additionally extracted information from sources that they knew would be valuable, such as anchor tag links and the contents of title tags in the HTML sources of Wikipedia pages. We believe that by adding syllogistic reasoning, NLP systems will be able to extract further information from documents or will be able to find further evidence for given information. In this context, we see our approach as supporting a data-driven approach through reasoning. We believe that reasoning adds additional, valuable resources to processing natural language texts.

Natural Logic: Extended Syllogistic Reasoning

We describe here the research program of *extended syllogistic reasoning* as it appears in papers such as (van Benthem 1986; Sánchez-Valencia 1991; van Eijck 2007; Fyodorov *et al.* 2003; Moss 2015; van Benthem 2008; Pratt-Hartmann and Moss 2009). The basic idea is to take very small fragments of language, fragments where one can find "reasoning"' of some kind, and to find *complete logical systems for those fragments*. Ideally, the logical systems would not use "extra syntax" of any kind, and thus not involve translation into first-order logic. Even more, they should be efficiently decidable. That means that there should be algorithms that carry out the decision procedure (unlike what we see for very strong logical systems), and these algorithms should be very fast for the very simplest fragments.

Here is a summary of the field. There are many sound and complete logical systems for small fragments of language.

The most basic is the one with sentences All x are y. In more detail, the syntax is minimal (and will be extended quite soon): it consists only of sentences of the form All x are y, where x and y are taken from a pre-assigned set of *nouns*. The semantics of the language is very simple. A *model* $\mathcal{M}$ is a set M together with interpretations $[\![x]\!]$ for all nouns. Then we say that $\mathcal{M} \models$ All x are y if $[\![x]\!] \subseteq [\![y]\!]$. Further, for a set Γ of sentences of this language, we say that $\mathcal{M} \models \Gamma$ if $\mathcal{M} \models \psi$ for all $\psi \in \Gamma$. Given Γ and another sentence φ, we say that $\Gamma \models \varphi$ if every $\mathcal{M} \models \Gamma$ has $\mathcal{M} \models \varphi$.

All of this is semantic; that is, having to do with models. There is a matching proof system, generated by the following two rules:

$$\frac{}{\text{All } p \text{ are } p} \text{ AXIOM}$$

$$\frac{\text{All } p \text{ are } n \quad \text{All } n \text{ are } q}{\text{All } p \text{ are } q} \text{ BARBARA}$$

Then we have the following result.

Theorem 1 *The proof system is sound and complete:* $\Gamma \models \varphi$ *iff* $\Gamma \vdash \varphi$*. Moreover, the problem of deciding whether, given a finite* $\Gamma \cup \varphi$*, we have* $\Gamma \models \varphi$ *or not is decidable in polynomial time.*

This is the simplest result in the area. (In fact, it is the simplest completeness theorem in all of logic.)

Figure 1 is a picture of a few of the logical system which have been treated in this way. For a description of the logical fragments depicted in this figure, please see (Moss 2015). Each has a precise syntax, semantics, proof theory, and algorithmic support. For example, $\mathcal{R}$ allows one to use verbs and relative clause. Other fragments add comparative adjectives. Especially noteworthy are some fragments which allow one to compare the sizes of sets. That is, one can say there are more dogs than cats, and perform reasoning with this. This is important because cardinality comparison goes beyond first-order logic; so it cannot be treated by any contemporary theorem provers. The reason that we can incorporate this kind of cardinality comparison into natural logic is that the systems are not built on top of first-order logic, or even on top of propositional logic. As such, working with a relatively weak "base", we are able to do things like cardinality comparison. And in a sense, this is the point.

Haskell Implementation

There are several implementations of natural logic fragments. Some are in Sage, an open source mathematics software package implemented in Python. The work on Sage is available at `www.sagemath.com`. Many of the systems have been implemented in Haskell. We discuss here aspects of the implementation (Moss 2017), primarily to demonstrate that such implementations exist.

The design features of the implementation are as follows:

1. A user can enter a finite set of syllogistic rules (a logic L) and a finite set Γ of assumptions, and generate all sentences which are provable from Γ in L.
2. If sentence φ is provable from Γ in L, we can generate and display the proof.
3. The logic L may contain negated nouns and also verbs and relative clauses.
4. L must consist of rules which are entirely syllogistic. Rules that infer information from contradictions, also called "explosion" rules, are fine. But *reductio ad absurdum* is not allowed in full generality.
5. Up until now, the items on this list are all entirely proof-theoretic. We have also built some tools for the intended semantics. For most of the standard systems in the literature, the work here also builds *counter-models*. That is, if φ does not follow from Γ in L, then we may ask for a counter-model showing this. This is only possible when we have a semantics for which the proof system is sound and complete.
6. As a particular case of point 5 just above, we can handle logics which are not expressible in first-order logic, such as the logic of *most*, *all*, and *some*.
7. For most of points 1–6 above, the system here is the first implementation which can do that point. That is, the implementation here goes well beyond what has been done so far.

We should emphasize the last point, the one about counter-model generation. It is standard to use a theorem prover like Prover9 together with a model-finder like Mace4 [https://www.cs.unm.edu/~mccune/mace4/]. The two programs are very different. In our setting, the prover and the counter-model generator are basically two branches of the same tree. They are not so different, actually. The reason for this seems to be (again) that the logics are rather simpler than full first-order logic.

To give a sense of the flavor of this implementation, we show in figure 2 part of a proof session. The session was edited to fit on one page. The user typed in a set Γ of assumptions. In this case, Γ consists of the sentences all skunks mammals, some skunks skunks (i.e., skunks exist), all skunks chordates. Then the user asks whether it follows that some mammals mammals. That is, the user wants to know whether the assumptions in Γ entail that some mammals exist. This sentence follows from the assumptions and a derivation is given in English. The derivation is shown in the top half of figure 2. While somewhat terse, this proof is much more readable than output from a theorem prover.

Next, the user wants to know whether all mammals skunks. In this case the does not follow from the assumptions and the system returns with a counter-model. This countermodel is shown in the bottom half of figure 2.

Monotonic Reasoning

Let us recapitulate where we are in this paper. We began with a discussion of Watson, mainly to motivate the idea of incorporating reasoning into a real life AI/NLP systems. Then we gave a brief discussion of the logical systems of natural logic, to present logical systems which are sound and complete, have low computational overhead, and which are capable of representing a fair amount of natural language inference. However, the logics from the prior section have some limitations from the point of view of NLP: they are

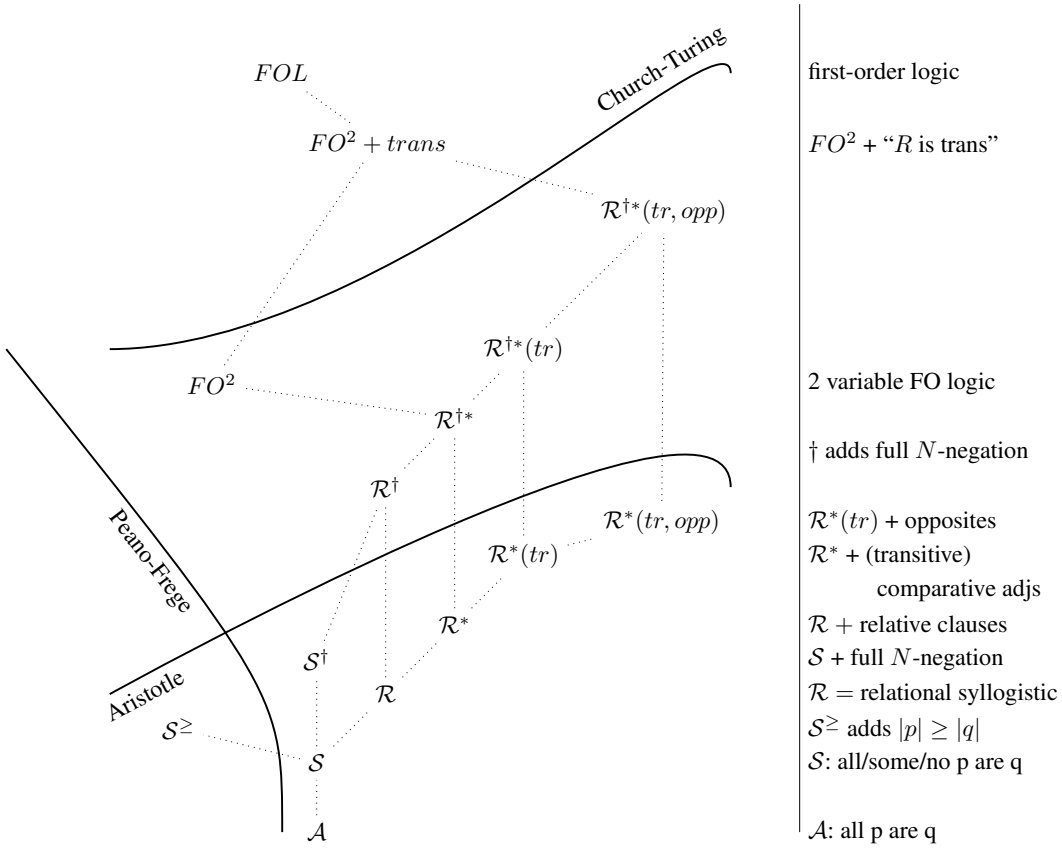

Figure 1: Natural logics. The fragments described on the right are explained in a number of papers, along with sound and complete proof systems for them. In many cases, proof search has been automated.

based on toy grammars and as such have limited applications in text as it comes. In other words, it would be difficult to see how to use those logical systems directly in an recognizing textual entailment task (Sammons 2015).

In this section, we present work that aims to overcome this issue. The work in this section is more speculative than that of the previous one, but we believe that it is more promising for future developments. To set the stage, consider the following sentences.

every dog$^{\downarrow}$ barks$^{\uparrow}$
no dog$^{\downarrow}$ barks$^{\downarrow}$
not every dog$^{\uparrow}$ barks$^{\downarrow}$
some dog$^{\uparrow}$ barks$^{\uparrow}$
most dogs$^{\times}$ bark$^{\uparrow}$

The arrows indicate *upward* or *downward* inference. For an introduction to this notation, see (van Benthem 1986). In the first sentence, if we know (of a scene, or a story, or a situation) that every dog barks, then it follows that every old dog barks and that every beagle barks. In both of those cases, we go "down" from dog to some subset of it. And the inference on barks goes the other way. If we know that every dog barks, then it follows that every dog vociferates.

Work on monotonicity merges the theory of grammar with the theory of inference. The idea is to enrich the lexical entries (words) with various forms of "up and down arrow" information. Then this information is propagated in deductions, as shown above. Deductions in systems like Combinatory categorial grammar (CCG) are basically forms of parsing. Current work on monotonicity (Icard and Moss 2014; Icard 2012; MacCartney and Manning 2009; Nairn *et al.* 2006) merges the monotonicity calculus from natural logic with linguistic frameworks like CCG. The idea is to have something that can automatically infer the correct "up and down arrow" information from real text, and then to use this in connection with NLP inference tasks.

```
ghci
GHCi, version 7.10.2: http://www.haskell.org/ghc/  :? for help
Prelude> :l S_dagger
[1 of 8] Compiling ProofTreeNumbers ( ProofTreeNumbers.hs, interpreted )
[2 of 8] Compiling Syntax2          ( Syntax2.hs, interpreted )
[3 of 8] Compiling ExampleSentences ( ExampleSentences.hs, interpreted )
[4 of 8] Compiling FrontEnd         ( FrontEnd.hs, interpreted )
[5 of 8] Compiling ExampleRules     ( ExampleRules.hs, interpreted )
[6 of 8] Compiling SyllogisticInference ( SyllogisticInference.hs, interpreted )
[7 of 8] Compiling Models           ( Models.hs, interpreted )
[8 of 8] Compiling S_dagger         ( S_dagger.hs, interpreted )
Ok, modules loaded: S_dagger, Syntax2,..., Models

*S_dagger>  let gamma=["all skunks mammals","some skunks skunks","all skunks chordates"]
*S_dagger> followsInSdagger "some mammals mammals" gamma

The sentence follows, and here is a derivation in S-dagger from the assumptions:

(1,all skunks are mammals,"A",[])
(2,some skunks are skunks,"A",[])
(3,some skunks are mammals,"darii",[1,2])
(4,some mammals are skunks,"some2",[3])
(5,some mammals are mammals,"some1",[4])

*S_dagger> followsInSdagger "all mammals skunks" gamma

The given sentence is not provable in S-dagger from the assumptions.
Here is a counter-model:

The universe is the set of numbers in [0,1].

The nouns are interpreted as follows:

  Noun    | Interpretation
----------+---------------
skunks    | 0
mammals   | 0, 1
chordates | 0, 1


Here is how the assumptions and purported conclusion fare in this model:

        Sentence          | Truth Value
--------------------------+------------
all skunks are mammals    | True
some skunks are skunks    | True
all skunks are chordates  | True
all mammals are skunks    | False
```

Figure 2: A Haskell proof session in which one sentence is shown to follow from the assumptions (top) and one is shown as not provable (bottom.)

Integration into NLP Systems

We began work on integrating naural logic into an NLP parser so as to increase the amount of information that can be extracted from text. Consider the following example, which outlines the basic approach we took. At the most basic, we plan to identify passages in natural language text that are instances of universally qualified statements. Consider the statement:

```
All dogs are mortal.
```

An off-the-shelf parser such as the Stanford NLP kit [http://nlp.stanford.edu/software/] produces an internal graph of the parsed sentence that is displayed as follows:

```
word All ,pos: DT ,ne: O
word dogs ,pos: NNS ,ne: O
word are ,pos: VBP ,ne: O
word mortal ,pos: JJ ,ne: O
word o ,pos: o ,ne: O

(ROOT (S (NP (DT All) (NNS dogs))
  (VP (VBP are) (ADJP (JJ mortal))) (. o)))

→ mortal-JJ (root)
  → dogs-NNS (nsubj)
    → All-DT (det)
  → are-VBP (cop)
```

We modified the Stanford NLP kit to process the internal representation of the parsed sentence to (i) identify a subset of simple universally quantified sentences and (ii) if successfully identified, to return a representation of that sentence for further processing. It should be noted that (i) and (ii) are accomplished in one step. If a sentence can be successfully parsed, we return the representation of it. If it cannot be successfully parsed, we ignore it. We would like to point out that this is very much in keeping with the data-intensive approach that a system like Watson uses. We do not expect our system to parse each universally qualified sentence. We will certainly continue to revise our system so that it can process increasingly complex sentences, but expect that this will be a drawn out process. The sentence returned by our system is as follows, but could be returned in other formats, suitable for further processing.

```
mortal(All, dogs)
```

We are now in a position to combine this information with other information we may find in the same document or that is known as part of background information. Suppose that the document states that Fido is a dog. In this case, we are able to confirm that Fido is mortal. This may be sufficient to answer a given question. However, it can also be supporting evidence to already known information. Suppose that either through a different text or from another passage in the given text, we learn that Fido died. In this case, we may use the inferred information that Fido is mortal as supporting evidence for the information that Fido died, thereby increasing the overall confidence in a statement. Again this is a tool in support of a data-driven approach such as found in IBM's Watson.

Conclusions

In this paper, we motivated the idea of incorporating reasoning into AI/NLP systems. We gave a brief discussion of the logical systems of natural logic, to present logical systems which are sound and complete, have low computational overhead, and which are capable of representing a fair amount of natural language inference. We furthermore presentd an approach from monotonic reasoning which enriches words with various forms of "up and down arrow" information, so that this information can be propagated in deductions. We finished this paper by presenting some initial work that aims to integrate natural logics into NLP systems.

The work described here is very much in the beginning stages. Undoubtedly, challenges will need to be addressed. In the near future, we plan to expand and integrate the existing systems. We additionally plan to improve the number and complexity of statements that can be recognized. Later this year, we plan to integrate a syllogistic reasoning component into a multi-year research project aimed at human-robot collaborative problem solving in a shared space. It will be designed to process statements that contain quantifiers. It will be one component of many; however, it will enable us to extend the functionality in a crucial way.

References

D. A. Ferrucci. Introduction to "This is Watson". *IBM Journal of Research and Development*, 56(3/4):1;1 – 1;15, 2012.

Yaroslav Fyodorov, Yoad Winter, and Nissim Fyodorov. Order-based inference in natural logic. *Log. J. IGPL*, 11(4):385–417, 2003. Inference in computational semantics: the Dagstuhl Workshop 2000.

Alfio Gliozzo, Or Biran, Siddharth Patwardhan, and Kathleen McKeown. Semantic technologies in IBM Watson. *ACL 2013, http://www.aclweb.org/anthology/W13-3413*, page 85, 2013.

Jim Hendler and Simon Ellis. Why Watson Won: A cognitive perspective. Retrieved September 03 2014 from http://www.slideshare.net/jahendler/why-watson-won-a-cognitive-perspective, 2014.

Thomas F. Icard and Lawrence S. Moss. Recent progress on monotonicity. *Linguistic Issues in Language Technology*, 9(7):167–194, 2014.

Thomas F. Icard. Inclusion and exclusion in natural language. *Studia Logica*, 100(4):705–725, 2012.

A. Kalyanpur, B. K. Boguraev, S. Patwardhan, J. W. Murdock, A. Lally, C. Welty, J. M. Prager, B. Coppola, A. Fokoue-Nkoutche, L. Zhang, Y. Pan, and Z. M. Qiu. Structured data and inference in DeepQA. *IBM Journal of Research and Development*, 56(3/4):10;1 – 10;14, 2012.

Bill MacCartney and Christopher D. Manning. An extended model of natural logic. In *Proceedings of the Eighth International Conference on Computational Semantics (IWCS-8)*, 2009.

David A. McAllester and Robert Givan. Natural language syntax and first-order inference. *Artificial Intelligence*, 56:1–20, 1992.

Lawrence S. Moss. Natural logic. In *Handbook of Contemporary Semantic Theory, Second Edition*, pages 646–681. Wiley-Blackwell, 2015.

Lawrence S. Moss. Extended syllogistic inference: Progress notes on a Haskell implementation. unpublished ms., Indiana University, 2017.

J. W. Murdock, J. Fan, A. Lally, H. Shima, and B. K. Boguraev. Textual evidence gathering and analysis. *IBM J. RES. and DEV*, 56(3.4), 2012.

Rowan Nairn, Cleo Condoravdi, and Lauri Karttunen. Computing relative polarity for textual inference. In *Proceedings of ICoS-5 (Inference in Computational Semantics)*, Buxton, UK, 2006.

Noritaka Nishihara, Kenichi Morita, and Shigenori Iwata. An extended syllogistic system with verbs and proper nouns, and its completeness proof. *Systems and Computers in Japan*, 21(1):760–771, 1990.

Ian Pratt-Hartmann and Lawrence S. Moss. Logics for the relational syllogistic. *Review of Symbolic Logic*, 2(4):647–683, 2009.

William C. Purdy. Inexpressiveness of first-order fragments. *Australasian Journal of Logic 4*, pages 1–12, 2006.

Mark Sammons. Recognizing textual entailment. In *Handbook of Contemporary Semantic Theory, Second Edition*, pages 711–757. Wiley, 2015.

Victor Sánchez-Valencia. *Studies on Natural Logic and Categorial Grammar*. PhD thesis, Universiteit van Amsterdam, 1991.

Peter Stone, Rodney Brooks, Erik Brynjolfsson, Ryan Calo, Oren Etzioni, Greg Hager, Julia Hirschberg, Shivaram Kalyanakrishnan, Ece Kamar, Sarit Kraus, Kevin Leyton-Brown, David Parkes, William Press, AnnaLee Saxenian, Julie Shah, Milind Tambe, and Astro Teller. "artificial intelligence and life in 2030." one hundred year study on artificial intelligence: Report of the 2015-2016 study panel. Retrieved September 6, 2016 from http://ai100.stanford.edu/2016-report, 2016.

Johan van Benthem. *Essays in Logical Semantics*. Reidel, Dordrecht, 1986.

Johan van Benthem. A brief history of natural logic. In M. Nath Mitra M. Chakraborty, B. Löwe and S. Sarukkai, editors, *Logic, Navya-Nyaya and Applications, Homage to Bimal Krishna Matilal*. College Publications, London, 2008.

Jan van Eijck. Natural logic for natural language. In Balder ten Cate and Henk Zeevat, editors, *6th International Tbilisi Symposium on Logic, Language, and Computation*. Springer, 2007.