Natural Language Inference with Monotonicity

Hai Hu
Indiana University
huhai@indiana.edu

Qi Chen
Indiana University
qc5@indiana.edu

Lawrence S. Moss
Indiana University
lsm@cs.indiana.edu

Abstract

This paper describes a working system which performs natural language inference using polarity-marked parse trees. The system handles all of the instances of monotonicity inference in the FraCaS data set, and can be easily extended to compute inferences in other sections of FraCaS. We achieve perfect precision and an accuracy comparable to previous systems on the first section of FraCaS. Except for the initial parse, it is entirely deterministic. It handles multi-premise arguments. The kind of inference performed is essentially "logical", but it goes beyond what is representable in first-order logic. In any case, the system works on surface forms and CCG parse trees rather than on logical representations of any kind.

1 Introduction

Computational systems which attempt to automate natural language inference (NLI) generally fall into one of the three categories: 1) systems which translate input into first-order logic (FOL) or higher-order logic; 2) systems based on distributional semantics, using word embeddings and then neural networks for learning inference (e.g. Bowman et al., 2015; Cases and Karttunen, 2017); and 3) systems using natural logic.

This paper is a contribution to the third category, as are Abzianidze (2015, 2017); MacCartney and Manning (2009); Angeli and Manning (2014); Angeli et al. (2016); Hu et al. (2018); Mineshima et al. (2015). Specifically, we continue work on order-based approaches to natural language inference going back to Fyodorov et al. (2003) and Zamansky et al. (2006). We make use of the polarity-marking tool due to Hu and Moss (2018). When given as input a derivation tree in CCG, this tool outputs the polarized yield of the tree. For example, when one inputs *Most Europeans live outside of Asia*, the output will be *Most*$^{\uparrow}$ *Europeans*$^{=}$ *live*$^{\uparrow}$ *outside*$^{=}$ *of*$^{=}$ *Asia*$^{\downarrow}$. Indeed, the tool polarizes every constituent. These arrows indicate whether inferences can be made by replacement "upward", replacement "downward", or in neither direction $=$. Iterating this idea of replacement does give a "fair amount" of inference, but to cope with NLI datasets we augment replacement with rules of natural logic (van Benthem, 1986), and with a mechanism for handling contradictions.

Our system is aimed at inference problems such as those in the FraCaS data set (Cooper et al., 1996), and we compare our efforts with the results in other papers. In addition, the ideas in our system can be adapted by others as part of their NLI toolkits.

2 Inference Algorithm

Input A set $\mathcal{P}$ of *premises*, another set $\mathcal{K}$ called the *knowledge base* and a sentence H, the *hypothesis*.

Output whether the relation between $\mathcal{P} \cup \mathcal{K}$ and H is *entailment*, *contradiction*, or *unknown*. The *unknown* relation means that in general, $\mathcal{P} \cup \mathcal{K}$ neither entails nor contradicts H.

Two auxilliary sets $\mathcal{K}$ and $\mathcal{S}$: a knowledge base $\mathcal{K}$ and a set $\mathcal{S}$ of entailed sentences. $\mathcal{K}$ consists of a relation $\leq$ on constituents (that is, a set of ordered pairs of words or multi-word constituents):

$$\textit{cat} \leq \textit{animal} \qquad \textit{kiss} \leq \textit{touch}$$
$$\textit{kissed some cat} \leq \textit{touched some animal}$$

These come from the premise set $\mathcal{P}$, or from a fixed background knowledge base $\mathcal{K}$, or from a lexical source such as WordNet (Miller, 1995). $\mathcal{K}$ also keeps track of all the nouns, verbs, adjectives, adverbs, relative clauses that appear in either $\mathcal{P}$ or H. The second auxilliary set is a *sentence base*, $\mathcal{S}$. This set stores all the inferences and contradictions our system derives, starting from $\mathcal{P}$. Inferences are stored in $\mathcal{S}$.*inferences* whereas contradictions are stored in $\mathcal{S}$.*contradictions*. (Optionally, it might include a subset of $\mathcal{K}$ which is relevant to $\mathcal{P} \cup \{H\}$).

Algorithm The key intuition of our algorithm is that once we have a correctly polarized CCG tree, e.g., *all*$^{\uparrow}$ *animal*$^{\downarrow}$ *sleep*$^{\uparrow}$, we can replace a constituent with some word or phrase from $\mathcal{K}$ and get an inference: *all*$^{\uparrow}$ *cat*$^{\downarrow}$ *sleep*$^{\uparrow}$. This replacement is extremely simple and effective. Along the same lines, we can replace *all* with *no* to obtain a contradiction: *no*$^{\uparrow}$ *animal*$^{\downarrow}$ *sleep*$^{\downarrow}$. (This is not strictly a contradiction: in a logically-possible model, there might not be any animals. In that model, both *all animals sleep* and *no animals sleep*. However, the spirit of work in the RTE area is that we should use the more natural semantics of *all*, the semantics that carries an existential presupposition. And from this in the premise, we indeed have a contradiction.) Since we ignore most of the morphology, the words are all represented by their lemmas. We also manipulate the sentence structures so that there isn't too much variation. For example, there-be structures such as "there are NP who VP" are changed to "some NP VP". Major steps of our algorithm are listed below. (A more complete pseudocode is presented in the Appendix.)

1. Get the polarities of all sentences in $\mathcal{P}$, using the system in Hu and Moss (2018).
2. For each $P \in \mathcal{P}$: (1) Add P to $\mathcal{S}$.*inferences*; (2) If P is of the form "every X is a N", then add $X \leq N$ to $\mathcal{K}$; (3) If P is of the form "every X VP", then add *be* $X \leq$ *VP* to $\mathcal{K}$; (4) If P is of the form "N_{pr} is a N", then add *every* $N \leq N_{pr}$ and $N_{pr} \leq$ *some* N to $\mathcal{K}$.
3. Next, make the following additions to $\mathcal{K}$:
   - For each noun *n* and each subsective adjective a, each prepositional phrase p, and each relative clause r in $\mathcal{P} \cup \{H\}$: add to $\mathcal{K}$ the following inequalities: $a\ n \leq n$, $n\ p \leq n$, and $n\ r \leq n$. For instance, *small dog* $\leq$ *dog*, *dog from France* $\leq$ *dog*, *dog that barks* $\leq$ *dog*.
   - For each verb *v*, and each adverb a to $\mathcal{K}$ the inequality $v\ a \leq v$.
4. Loop over each premise P and do two types of `replacement`; one derives inferences while the other produces contradictions. See Figure 1 for a concrete example.
   - `replacement_infer`: 1) if a constituent is $^{\downarrow}$, then replace it with something "smaller" in $\mathcal{K}$; 2) if a constituent is $^{\uparrow}$, then replace it with something "bigger" in $\mathcal{K}$. Finally, store the new sentences in $\mathcal{S}$.*inferences*.
   - `replacement_contra`: 1) replace "no" with "some" and vice versa if the quantifier is not embedded in a relative clause[1]. 2) negate the sentence by e.g., adding "do not" before the main verb. Finally, store the new sentences in $\mathcal{S}$.*contradictions*.

   Now the inference problem becomes a typical AI search problem. That is, we are searching for an inference that matches *H* string for string. If such an inference can be found before reaching a stopping criterion, then we return `entail`. If not, we turn to the generated contradictions to see if any of them matches *H*; if so, we return `contradict`, otherwise return `unknown`.

[1] "Books that *no* one reads are on the shelf" does *not* contradict "Books that *some* one reads are on the shelf".

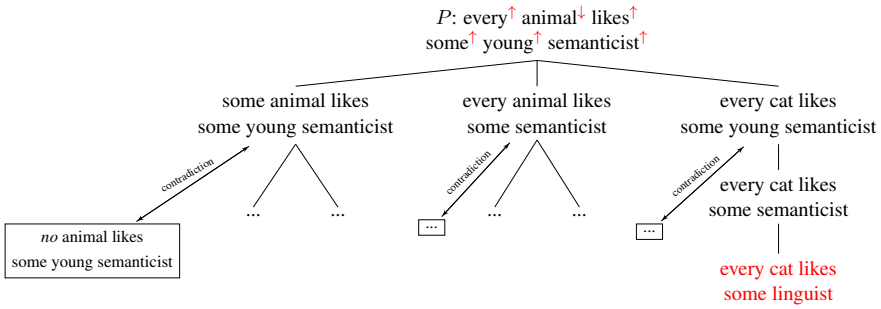

Figure 1: Example search tree where P is *every animal likes some young semanticist*, with the H: *every cat likes some linguist*. Only one `replacement` is allowed at each step. Sentences in rectangular are the generated contradictions. In this case our system will return `entail`.

Note that theoretically we can perform `replacement_infer` indefinitely, on the inferences generated in the last step. On the contrary, `replacement_contra` can only be applied once on each inference, since the contradiction of a contradiction brings us back to the premise again.[2]

This search problem is implemented using depth-first search, w/ default depth $= 2$.

Inferences not handled by `replacement` As discussed in Hu et al. (2018), `replacement` can handle/derive many rules of natural logic, but not all of them. To name just a few the rules below are not covered by `replacement`:

$$\frac{\textit{Some } y \textit{ are } x}{\textit{Some } x \textit{ are } y}\ \text{SOME}_2 \qquad \frac{\textit{All } (r \textit{ all } x)\ (r \textit{ all } y)}{\textit{All } y \textit{ are } x}\ \text{ANTI} \qquad \frac{\textit{Det } x\ y \quad \textit{All } x\ z}{\textit{Det } x\ (y \wedge z)}\ \text{DET}$$

To deal with this, we first convert the premises to a sentence compatible to natural logic syntax, i.e., *quantifier x y*. Then we apply the above rules on these sentences to get inferences. Finally we convert sentences in natural logic to sentence in natural language. This usually only involves minimal editing. For example, *every* cat *(animal* $\wedge$ *meow)* will be converted to "every cat is an animal who meows". As we will show later, DET is useful in solving many of the multi-premise problems in the first section of FraCaS. It is also worth noting that the capacity of our system can be easily expanded by including more rules from natural logic.

Initial knowledge base $\mathcal{K}$ includes the most basic (monotonicity) knowledge that can be utilized for all problems:

- **knowledge from WordNet** (Miller, 1995). dog $\leq$ animal, dog | cat, etc. The first section of the FraCaS dataset does not require world knowledge, so we didn't include WordNet relations for now. However, they can easily be added if need be.
- **knowledge about quantifiers**. Our system treats the following words/phrases as quantifiers:
  - *every* $=$ *all* $=$ *each* $\leq$ *most* $\leq$ *many* $\leq$ *a few* $=$ *several* $\leq$ *some* $=$ *a*; *the* $\leq$ *some* $=$ *a*
  - *at least/most* n.

  Because the parsers do not treat *at least/most* n as quantifiers as we hoped, we need a separate work-around for them.

[2]For example, s_1 = *a man walks*, s_2 = *no man walks*, s_3 = *some man walks*. We see that s_2 contradicts s_1 and that s_3 contradicts s_2, but s_3 is the same as s_1. This is also noted in Angeli et al. (2016).

system	MM08	AM14	LS13	T14	D14	M15	A16	ours
multi-premise?	N	N	Y	Y	Y	Y	Y	Y
# problems	44	44	74	74	74	74	74	74
Acc. (%)	97.73	95	62	80	95	77	93	88

Truth / Pred	E	U	C
Entail	29	7	0
Unknown	0	33	0
Contradict	0	2	3

Table 1: Left: Accuracy of our system and previous ones. Right: confusion matrix of our system. Our system achieves 100% precision and comparable accuracy with others. MM08: MacCartney and Manning (2008). AM14: Angeli and Manning (2014). LS13: Lewis and Steedman (2013). T14: Tian et al. (2014). D14: Dong et al. (2014). M15: Mineshima et al. (2015). A16: Abzianidze (2016).

3 Experiments on section 1 of FraCaS

We run our algorithm on the FraCaS dataset for NLI. This paper reports only on the first section: generalized quantifiers. Extending to other sections of the FraCaS dataset, and to other datasets, is work in progress. Results of our system are shown in Table 1. We have perfect precision and a comparable accuracy with previous systems.

3.1 Choice of parsers and their errors

Parser performance is the biggest bottle-neck of the system. We have tested two commonly used CCG parsers, C&C (Clark and Curran, 2007) and EasyCCG (Lewis and Steedman, 2014). C&C fails to parse four sentences from Sec. 1 of FraCaS. EasyCCG can parse all of them but we still need to semi-automatically modify the trees. Some of these are modifications that transform the tree into a semantically more meaningful form, while others are correcting parse errors. For example, not all quantifiers are super-tagged consistently, e.g., *most*, *few* are sometimes not tagged as `NP/N`. There are parsing errors involving multi-word expressions such as "a lot of", "used to be". We only correct systematic ones.

3.2 An example

The following example shows the actual process solving FraCas-026, which is a multiple-premise problem, and handled not only by replacement, but also with DET rule. Major steps are listed below:

1. Get polarities[3] of all premises $\mathcal{P}$, but not the hypothesis H:
   P1: Most$^{\uparrow}$ Europeans$^{=}$ are$^{\uparrow}$ resident$^{\uparrow}$ in$^{\uparrow}$ Europe$^{\uparrow}$
   P2: All$^{\uparrow}$ Europeans$^{\downarrow}$ are$^{\uparrow}$ people$^{\uparrow}$
   P3: All$^{\uparrow}$ people$^{\downarrow}$ who$^{\downarrow}$ are$^{\downarrow}$ resident$^{\downarrow}$ in$^{\downarrow}$ Europe$^{\downarrow}$ can$^{\uparrow}$ travel$^{\uparrow}$ freely$^{\uparrow}$ within$^{\uparrow}$ Europe$^{\downarrow}$
   H: Most Europeans can travel freely within Europe
2. Update knowledge base $\mathcal{K}$ with the information from $\mathcal{P}$, e.g.:
   Based on the form "every (or equivalent quantifiers, see above) X VP", add $X \leq VP$, which is:
   people who are resident in Europe $\leq$ *can travel freely within Europe*
   We can also get: *be people who are resident in Europe* $\leq$ *can travel freely within Europe*
3. Using the DET rule, the system generates a series of sentences which are also polarized, e.g., applying the DET rule to P1 and P2 we get:
   Most$^{\uparrow}$ *Europeans*$^{=}$ *are*$^{\uparrow}$ *people*$^{\uparrow}$ *who*$^{\uparrow}$ *are*$^{\uparrow}$ *resident*$^{\uparrow}$ *in*$^{\uparrow}$ *Europe*$^{\uparrow}$
4. Then adds generated sentences into sentence base $\mathcal{S}$, and start to do `replacement` on every constituent of each sentence. Therefore, we obtain a series of inferences like the following:
   Many European are people who are resident in Europe
   Most European are people
   Most European can travel freely within Europe
   Several European are people who are resitdent in Europe

[3]The polarity marking in P3 of the second occurrence of *Europe* was corrected from our system's output. The point is that under the scope of a modal *can*, a prepositional phrase headed by *in* or *within* changes polarity.

5. At last, one of the sentences in the list of inferences above matches the given hypothesis H, which means that solution to the original problem is `entail`.

4 Comparison with previous systems

We comment on our system and compare it to several systems mentioned in our Introduction (e.g. MacCartney and Manning, 2008; Angeli and Manning, 2014; Mineshima et al., 2015; Abzianidze, 2015).

Our algorithm is *provably correct* in the following sense. If one has a correctly-parsed set $\mathcal{P} = P_1, \ldots, P_n$ of premises and uses our algorithm, and if the hypothesis H is proved from the premises in our system, then H follows logically from $\mathcal{P}$. So in this sense, our system will have no false positives, i.e., no type I error. Now this requires words of clarification. First, frequently the parsed output does not reflect the logical structure accurately, and in this case, the polarity-marking step of our algorithm might well go wrong. Second, in the case of ambiguous logical words, it is also possible that errors in a parse will lead to errors in our output. For example, the English word "any" means "all" in a downward-entailing environment, but it means "some" in an upward-entailing environment. There are exceptions to this.

MacCartney and Angeli's systems, on the other hand, find downward-entailing environments by pattern-matching using dependency trees, which makes the polarizing algorithm more error-prone. For example, their system (part of Stanford CoreNLP v3.9.1) incorrectly polarizes the following sentences: *no*$^{\uparrow}$ *man*$^{\downarrow}$ *walks*$^{\uparrow}$, *Ed*$^{\uparrow}$ *refused*$^{\uparrow}$ *to*$^{\uparrow}$ *dance*$^{\uparrow}$, *John*$^{\uparrow}$ *loves*$^{\uparrow}$ *dancing*$^{\uparrow}$ *without*$^{\uparrow}$ *shoes*$^{\uparrow}$, *I*$^{\uparrow}$ *like*$^{\uparrow}$ *most*$^{\uparrow}$ *dogs*$^{\uparrow}$, whereas our system correctly polarizes all of them (*walks*, *dance*, *shoes* should all be $^{\downarrow}$ and *dogs* $^{=}$). Another difference is that in our system, `replacement` can happen in any order, and the results are the same, whereas in Angeli's system only certain "mutating" orders lead to the correct inferences (see Section 3.1 of Angeli et al. (2016)). A final point is that their systems only polarize at the word level, but our system computes polarities also on the constituent level, which is important for `replacement` as shown in the example above.

We used the tool in Hu and Moss (2018), but they did not provide a working system for NLI. Such a system was described in Hu et al. (2018), but that paper was programmatic and did not have an implementation, or test data. So it left open the issue of how much NLI can be done with mononotonicity alone, and how much requires natural logic rules. We present initial attempts to address this problem.

The best-performing system in the logic-related area of NLI is the one described in Abzianidze (2014, 2015, 2017). That line of work uses tableau rules rather than deduction, as a more standard theorem-prover from automated reasoning. Our system is arguably simpler than his: we use no lambda logical forms, and we believe that the theoretical basis of our system is also simpler. When our system finds an inference (or contradiction) the derivations in our system are not so far from a natural language proof of the hypothesis (or its negation) from the premises. This would not be possible from a tableau proof.

5 Conclusion and future work

We have provided a generic algorithm for natural language inference based on polarized parses and natural logic rules. The algorithm requires parsed output, but it does not require translation into a logical form, or special alignment steps. It would be possible to extend our system in either or both directions.

For future work, we are currently tuning our system on other sections of the FraCaS dataset (especially Sections 5 and 6) and the larger SICK dataset (Marelli et al., 2014). Another line of work is to incorporate into our system some "light" representation of the sentences, e.g. dependency parses, abstract meaning representation, that allows for more flexible syntactic variation than the current string-for-string match strategy.

References

Abzianidze, L. (2014). Towards a wide-coverage tableau method for natural logic. In *New Frontiers in Artificial Intelligence - JSAI-isAI 2014 Workshops, LENLS, JURISIN, and GABA, Kanagawa, Japan, October 27-28, 2014, Revised Selected Papers*, pp. 66–82.

Abzianidze, L. (2015). A tableau prover for natural logic and language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pp. 2492–2502.

Abzianidze, L. (2016). Natural solution to fracas entailment problems. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pp. 64–74.

Abzianidze, L. (2017). Langpro: Natural language theorem prover. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 115–120.

Angeli, G. and C. Manning (2014). NaturalLI: Natural logic inference for common sense reasoning. In *EMNLP*, pp. 534–545.

Angeli, G., N. Nayak, and C. Manning (2016). Combining natural logic and shallow reasoning for question answering. In *ACL*, pp. 442–452.

Bowman, S. R., C. Potts, and C. D. Manning (2015). Learning distributed word representations for natural logic reasoning. In *AAAI Spring Symposium on Knowledge Representation and Reasoning*, pp. 10–13.

Cases, I. and L. Karttunen (2017). Neural networks and textual inference: How did we get here and where do we go now?

Clark, S. and J. R. Curran (2007). Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics 33*(4), 493–552.

Cooper, R., D. Crouch, J. Van Eijck, C. Fox, J. Van Genabith, J. Jaspars, H. Kamp, D. Milward, M. Pinkal, M. Poesio, et al. (1996). Using the framework. Technical report, Technical Report LRE 62-051 D-16, The FraCaS Consortium.

Dong, Y., R. Tian, and Y. Miyao (2014). Encoding generalized quantifiers in dependency-based compositional semantics. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing*.

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

Hu, H., T. F. Icard, and L. S. Moss (2018). Automated reasoning from polarized parse trees. In *Proceedings of the Fifth Workshop on Natural Language and Computer Science*.

Hu, H. and L. S. Moss (2018). Polarity computations in flexible categorial grammar. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pp. 124–129.

Lewis, M. and M. Steedman (2013). Combined distributional and logical semantics. *Transactions of the Association of Computational Linguistics 1*, 179–192.

Lewis, M. and M. Steedman (2014). A* CCG parsing with a supertag-factored model. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 990–1000.

MacCartney, B. and C. D. Manning (2008). Modeling semantic containment and exclusion in natural language inference. In *Proceedings of COLING*, pp. 521–528. Association for Computational Linguistics.

MacCartney, B. and C. D. Manning (2009). An extended model of natural logic. In *IWCS-8, Proceedings of the Eighth International Conference on Computational Semantics*, pp. 140–156.

Marelli, M., S. Menini, M. Baroni, L. Bentivogli, R. Bernardi, and R. Zamparelli (2014). A SICK cure for the evaluation of compositional distributional semantic models.

Miller, G. A. (1995). Wordnet: a lexical database for English. *Communications of the ACM 38*(11), 39–41.

Mineshima, K., P. Martínez-Gómez, Y. Miyao, and D. Bekki (2015). Higher-order logical inference with compositional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2055–2061.

Tian, R., Y. Miyao, and T. Matsuzaki (2014). Logical inference on dependency-based compositional semantics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Volume 1, pp. 79–89.

van Benthem, J. (1986). *Essays in Logical Semantics*, Volume 29 of *Studies in Linguistics and Philosophy*. Dordrecht: D. Reidel Publishing Co.

Zamansky, A., N. Francez, and Y. Winter (2006). A 'natural logic' inference system using the Lambek calculus. *Journal of Logic, Language, and Information 15*(3), 273–295.

A Pseudocode of our system

Algorithm 1 Infer with Monotonicity

procedure MYPROCEDURE($premises$, H)
build s and k:
    $s \leftarrow$ *SentenceBase*() ▷ initialize s
    $k \leftarrow$ *KnowledgeBase*() ▷ initialize k
    k.*buildQuantifier*() ▷ add to k: all = every = each $\leq$ some = a
    k.*buildMorphTense*() ▷ add to k: man = men, have = has
    for P in *premises* **do**
        P.*fixTree*()
        k.*extractPattern*(P) ▷ add to k: "every x is NP" $\rightarrow$ $x \leq NP$
        k.*updateWordLists*(P) ▷ add to k: all nouns, adjs, rel clauses, etc. in P
        s.*addInference*(P) ▷ add to s: P
    end for
    k.*updateWordLists*(H) ▷ add to k: all nouns, adjs, rel clauses, etc. in H
    s.*addHypothesis*(H) ▷ add to s: H, "there be N who VP" $\rightarrow$ "some N VP"
update k:
    k.*update*() ▷ adj + noun $\leq$ noun, noun + RC/PP $\leq$ noun, verb + PP $\leq$ verb
    k.*updateRules*() ▷ SOME, ANTI, DET rules
polarize premises:
    for P in *premises* **do**
        P.*polarize*()
    end for
infer by replacement:
    infer($s, k, depth$) ▷ depth = how many rounds of replacement; iterative deepening search
predict:
    if H in s.inferences **then return** entail
    else
        if H in s.contradictions **then return** contradiction
        else return unknown
        end if
    end if
end procedure