Natural logic and natural language inference

Bill MacCartney and Christopher D. Manning

Abstract We propose a model of natural language inference which identifies valid inferences by their lexical and syntactic features, without full semantic interpretation. We extend past work in *natural logic*, which has focused on semantic containment and monotonicity, by incorporating both semantic exclusion and implicativity. Our model decomposes an inference problem into a sequence of atomic edits linking premise to hypothesis; predicts a lexical entailment relation for each edit; propagates these relations upward through a semantic composition tree according to properties of intermediate nodes; and joins the resulting entailment relations across the edit sequence. A computational implementation of the model achieves 70% accuracy and 89% precision on the FraCaS test suite. Moreover, including this model as a component in an existing system yields significant performance gains on the Recognizing Textual Entailment challenge.

1 Introduction

Natural language inference (NLI) is the problem of determining whether a natural language hypothesis h can reasonably be inferred from a given premise p. For example:

(1) p: *Every firm polled saw costs grow more than expected, even after adjusting for inflation.*
h: *Every big company in the poll reported cost increases.*

A capacity for open-domain NLI is clearly necessary for full natural language understanding, and NLI can also enable more immediate applications, such as se-

Bill MacCartney
Stanford University, Stanford, California, e-mail: wcmac@cs.stanford.edu

Christopher D. Manning
Stanford University, Stanford, California, e-mail: manning@cs.stanford.edu

mantic search and question answering. Consequently, NLI has been the focus of intense research effort in recent years, centered around the annual Recognizing Textual Entailment (RTE) competition [6].

For a semanticist, the most obvious approach to NLI relies on full semantic interpretation: first, translate p and h into some formal meaning representation, such as first-order logic (FOL), and then apply automated reasoning tools to determine inferential validity. While the formal approach can succeed in restricted domains, it struggles with open-domain NLI tasks such as RTE. For example, the FOL-based system of [1] was able to find a proof for less than 4% of the problems in the RTE1 test set. The difficulty is plain: truly *natural* language is fiendishly complex. The formal approach faces countless thorny problems: idioms, ellipsis, paraphrase, ambiguity, vagueness, lexical semantics, the impact of pragmatics, and so on. Consider for a moment the difficulty of fully and accurately translating (1) to a formal meaning representation.

Yet (1) also demonstrates that full semantic interpretation is often not necessary to determining inferential validity. To date, the most successful NLI systems have relied on surface representations and approximate measures of lexical and syntactic similarity to ascertain whether p subsumes h [9, 13, 10]. However, these approaches face a different problem: they lack the precision needed to properly handle such commonplace phenomena as negation, antonymy, downward-monotone quantifiers, non-factive contexts, and the like. For example, if *every* were replaced by *some* or *most* throughout (1), the lexical and syntactic similarity of h to p would be unaffected, yet the inference would be rendered invalid.

In this paper, we explore a middle way, by developing a model of what Lakoff [11] called *natural logic*, which characterizes valid patterns of inference in terms of syntactic forms which are as close as possible to surface forms. For example, the natural logic approach might sanction (1) by observing that: in ordinary *upward monotone* contexts, deleting modifiers preserves truth; in *downward monotone* contexts, inserting modifiers preserves truth; and *every* is downward monotone in its restrictor NP. Natural logic thus achieves the semantic precision needed to handle inferences like (1), while sidestepping the difficulties of full semantic interpretation.

The natural logic approach has a very long history,[1] originating in the syllogisms of Aristotle and continuing through the medieval scholastics and the work of Leibniz. It was revived in recent times by van Benthem [19, 20] and Sánchez Valencia [17], whose *monotonicity calculus* explains inferences involving semantic containment and inversions of monotonicity, even when nested, as in *Nobody can enter without a valid passport* $\models$ *Nobody can enter without a passport*. However, because the monotonicity calculus lacks any representation of semantic exclusion, it fails to license many simple inferences, such as *Stimpy is a cat* $\models$ *Stimpy is not a poodle*.

Another model which arguably belongs to the natural logic tradition (though not presented as such) was developed by [15] to explain inferences involving implicatives and factives, even when negated or nested, as in *Ed did not forget to force Dave to leave* $\models$ *Dave left*. While the model bears some resemblance to the monotonic-

[1] For a useful overview of the history of natural logic, see [21]. For recent work on theoretical aspects of natural logic, see [7, 18, 23].

ity calculus, it does not incorporate semantic containment or explain interactions between implicatives and monotonicity, and thus fails to license inferences such as *John refused to dance* $\models$ *John didn't tango*.

In this paper, we propose a new model of natural logic which extends the monotonicity calculus to incorporate semantic exclusion, and partly unifies it with Nairn et al.'s account of implicatives. We first define an inventory of *basic entailment relations* which includes representations of both containment and exclusion (section 2). We then describe a general method for establishing the entailment relation between a premise p and a hypothesis h. Given a sequence of *atomic edits* which transforms p into h, we determine the *lexical entailment relation* generated by each edit (section 4); project each lexical entailment relation into an *atomic entailment relation*, according to properties of the context in which the edit occurs (section 5); and join atomic entailment relations across the edit sequence (section 3). We have previously presented an implemented system based on this model [14]; here we offer a detailed account of its theoretical foundations.

2 An inventory of entailment relations

The simplest formulation of the NLI task is as a binary decision problem: the relation between p and h is to be classified as either *entailment* ($p \models h$) or *non-entailment* ($p \not\models h$). The *three-way* formulation refines this by dividing non-entailment into *contradiction* ($p \models \neg h$) and *compatibility* ($p \not\models h \wedge p \not\models \neg h$).[2] The monotonicity calculus carves things up differently: it interprets entailment as a *semantic containment* relation $\sqsubseteq$ analogous to the set containment relation $\subseteq$, and thus permits us to distinguish forward entailment ($p \sqsubseteq h$) from reverse entailment ($p \sqsupseteq h$). Moreover, it defines $\sqsubseteq$ for expressions of every semantic type, including not only complete sentences but also individual words and phrases. Unlike the three-way formulation, however, it lacks any way to represent contradiction (semantic exclusion). For our model, we want the best of both worlds: a comprehensive inventory of entailment relations that includes representations of both semantic containment and semantic exclusion.

Following Sánchez Valencia, we proceed by analogy with set relations. In a universe U, the set of ordered pairs $\langle x, y\rangle$ of subsets of U can be partitioned into 16 equivalence classes, according to whether each of the four sets $x \cap y$, $x \cap \bar{y}$, $\bar{x} \cap y$, and $\bar{x} \cap \bar{y}$ is empty or non-empty.[3] Of these 16 classes, nine represent degenerate cases in which either x or y is either empty or universal. Since expressions having empty denotations (e.g., *round square cupola*) or universal denotations (e.g., *exists*) fail to divide the world into meaningful categories, they can be regarded as semantically vacuous. Contradictions and tautologies may be common in logic textbooks,

[2] The first three RTE competitions used the binary formulation, while the three-way formulation was adopted for RTE4. The three-way formulation was also employed in the FraCaS test suite [5] and has been investigated in depth by [4].

[3] We use $\bar{x}$ to denote the complement of set x in universe U; thus $x \cap \bar{x} = \emptyset$ and $x \cup \bar{x} = U$.

but they are rare in everyday speech. Thus, in a practical model of informal natural language inference, we will rarely go wrong by assuming the *non-vacuity* of the expressions we encounter.[4] We therefore focus on the remaining seven classes, which we designate as the set $\mathfrak{B}$ of *basic entailment relations*, shown in Table 1.

Table 1 The set $\mathfrak{B}$ of seven basic entailment relations

symbol[a]	name	example	set theoretic definition[b]
$x \equiv y$	equivalence	*couch* $\equiv$ *sofa*	$x = y$
$x \sqsubset y$	forward entailment	*crow* $\sqsubset$ *bird*	$x \subset y$
$x \sqsupset y$	reverse entailment	*European* $\sqsupset$ *French*	$x \supset y$
$x \ {}^\wedge\ y$	negation	*human* ${}^\wedge$ *nonhuman*	$x \cap y = \emptyset \wedge x \cup y = U$
$x \mid y$	alternation	*cat* $\mid$ *dog*	$x \cap y = \emptyset \wedge x \cup y \neq U$
$x \smile y$	cover	*animal* $\smile$ *nonhuman*	$x \cap y \neq \emptyset \wedge x \cup y = U$
$x \# y$	independence	*hungry* # *hippo*	(all other cases)

[a] Selecting an appropriate symbol to represent each relation is a vexed problem. We sought symbols which (a) are easily approximated by a single ASCII character, (b) are graphically symmetric iff the relations they represent are symmetric, and (c) do not excessively abuse accepted conventions. The ${}^\wedge$ symbol was chosen to evoke the logically similar bitwise `XOR` operator of the C programming language family; regrettably, it may also evoke the Boolean `AND` function. The $\mid$ symbol was chosen to evoke the Sheffer stroke commonly used to represent the logically similar Boolean `NAND` function; regrettably, it may also evoke the Boolean `OR` function. The $\sqsubset$ and $\sqsupset$ symbols were obviously chosen to resemble their set-theoretic analogs, but a potential confusion arises because some logicians use the horseshoe $\supset$ (with the *opposite* orientation) to represent material implication.
[b] Each relation in $\mathfrak{B}$ obeys the additional constraints that $\emptyset \subset x \subset U$ and $\emptyset \subset y \subset U$ (i.e., x and y are non-vacuous).

First, the semantic containment relations ($\sqsubseteq$ and $\sqsupseteq$) of the monotonicity calculus are preserved, but are factored into three mutually exclusive relations: equivalence ($\equiv$), (strict) forward entailment ($\sqsubset$), and (strict) reverse entailment ($\sqsupset$). Next, we have two relations expressing semantic exclusion: negation (${}^\wedge$), or exhaustive exclusion, which is analogous to set complement; and alternation ($\mid$), or non-exhaustive exclusion. The next relation is cover ($\smile$), or non-exclusive exhaustion. Though its utility is not immediately obvious, it is the *dual under negation* of the alternation relation.[5] Finally, the independence relation (#) covers all other cases: it expresses non-equivalence, non-containment, non-exclusion, and non-exhaustion. Note that # is the least informative relation, in that it places the fewest constraints on its arguments.[6]

[4] Our model can easily be revised to accommodate vacuous expressions and relations between them, but then becomes somewhat unwieldy. The assumption of non-vacuity is closely related to the assumption of *existential import* in traditional logic. For a defense of existential import in natural language semantics, see [2].

[5] We describe relations R and S as *duals under negation* iff $\forall x, y : \langle x, y \rangle \in R \Leftrightarrow \langle \overline{x}, \overline{y} \rangle \in S$. Thus $\sqsubset$ and $\sqsupset$ are dual; $\mid$ and $\smile$ are dual; and $\equiv$, ${}^\wedge$, and # are self-dual. The significance of this duality will become apparent in section 5.

[6] Two sets selected uniformly at random from 2^U are overwhelmingly likely to belong to # (for large $|U|$).

Following Sánchez Valencia, we define the relations in $\mathfrak{B}$ for all semantic types. For semantic types which can be interpreted as characteristic functions of sets,[7] the set-theoretic definitions can be applied directly. The definitions can then be extended to other types by interpreting each type as if it were a type of set. For example, propositions can be understood (per Montague) as denoting sets of possible worlds. Thus two propositions stand in the $|$ relation iff there is no world where both hold (but there is some world where neither holds). Likewise, names can be interpreted as denoting singleton sets, with the result that two names stand in the $\equiv$ relation iff they refer to the same entity, or the $|$ relation otherwise.

By design, the relations in $\mathfrak{B}$ are mutually exclusive, so that we can define a function $\beta(x,y)$ which maps every ordered pair of expressions[8] to the unique relation in $\mathfrak{B}$ to which it belongs.

3 Joining entailment relations

If we know that entailment relation R holds between x and y, and that entailment relation S holds between y and z, then what is the entailment relation between x and z? The *join* of entailment relations R and S, which we denote $R \bowtie S$,[9] is defined by:

$$R \bowtie S \overset{\text{def}}{=} \{\langle x,z \rangle : \exists y \; (\langle x,y \rangle \in R \land \langle y,z \rangle \in S)\}$$

Some joins are quite intuitive. For example, it is immediately clear that $\sqsubset \bowtie \sqsubset = \sqsubset$, $\sqsupset \bowtie \sqsupset = \sqsupset$, $^\wedge \bowtie {}^\wedge = \equiv$, and for any R, $(R \bowtie \equiv) = (\equiv \bowtie R) = R$. Other joins are less obvious, but still accessible to intuition. For example, $| \bowtie {}^\wedge = \sqsubset$. This can be seen with the aid of Venn diagrams, or by considering simple examples: *fish* $|$ *human* and *human* $^\wedge$ *nonhuman*, thus *fish* $\sqsubset$ *nonhuman*.

But we soon stumble upon an inconvenient truth: not every join yields a relation in $\mathfrak{B}$. For example, if $x \,|\, y$ and $y \,|\, z$, the relation between x and z is not determined. They could be equivalent, or one might contain the other. They might be independent or alternative. All we can say for sure is that they are not exhaustive (since both are disjoint from y). Thus, the result of joining $|$ and $|$ is not a relation in $\mathfrak{B}$, but a *union* of such relations, specifically $\bigcup\{\equiv, \sqsubset, \sqsupset, |, \#\}$.[10]

[7] That is, all functional types whose final output is a truth value. If we assume a type system whose basic types are e (entities) and t (truth values), then this includes most of the functional types encountered in semantic analysis: $e \to t$ (common nouns, adjectives, and intransitive verbs), $e \to e \to t$ (transitive verbs), $(e \to t) \to (e \to t)$ (adverbs), $(e \to t) \to (e \to t) \to t$ (binary generalized quantifiers), and so on.

[8] Assuming the expressions are non-vacuous, and belong to the same semantic type.

[9] In Tarskian relation algebra, this operation is known as *relation composition*, and is often represented by a semi-colon: $R\,;S$. To avoid confusion with semantic composition (section 5), we prefer to use the term *join* for this operation, by analogy to the database `JOIN` operation (also commonly represented by $\bowtie$).

[10] We use this notation as shorthand for the union $\equiv \cup \sqsubset \cup \sqsupset \cup \,|\, \cup \,\#$. To be precise, the result of this join is not identical with this union, but is a subset of it, since the union contains some pairs

We will refer to (non-trivial) unions of relations in $\mathfrak{B}$ as *union relations*.[11] Of the 49 possible joins of relations in $\mathfrak{B}$, 32 yield a relation in $\mathfrak{B}$, while 17 yield a union relation, with larger unions conveying less information. Union relations can be further joined, and we can establish that the smallest set of relations which contains $\mathfrak{B}$ and is closed under joining contains just 16 relations.[12] One of these is the total relation, which contains all pairs of (non-vacuous) expressions. This relation, which we denote $\bullet$, is the black hole of entailment relations, in the sense that (a) it conveys zero information about pairs of expressions which belong to it, and (b) joining a chain of entailment relations will, if it contains any noise and is of sufficient length, lead inescapably to $\bullet$.[13] This tendency of joining to devolve toward less-informative entailment relations places an important limitation on the power of the inference method described in section 7.

A complete join table for relations in $\mathfrak{B}$ is shown in Table 2.[14]

Table 2 The join table for the basic entailment relations

$\bowtie$	$\equiv$	$\sqsubset$	$\sqsupset$	$^$	$\mid$	$\smile$	$\#$
$\equiv$	$\equiv$	$\sqsubset$	$\sqsupset$	$^$	$\mid$	$\smile$	$\#$
$\sqsubset$	$\sqsubset$	$\sqsubset$	$\equiv\sqsubset\sqsupset\mid\#$	$\mid$	$\mid$	$\sqsubset{^}\mid\smile\#$	$\sqsubset\mid\#$
$\sqsupset$	$\sqsupset$	$\equiv\sqsubset\sqsupset\smile\#$	$\sqsupset$	$\smile$	$\sqsupset{^}\mid\smile\#$	$\smile$	$\sqsupset\smile\#$
$^$	$^$	$\smile$	$\mid$	$\equiv$	$\sqsupset$	$\sqsubset$	$\#$
$\mid$	$\mid$	$\sqsubset{^}\mid\smile\#$	$\mid$	$\sqsubset$	$\equiv\sqsubset\sqsupset\mid\#$	$\sqsubset$	$\sqsubset\mid\#$
$\smile$	$\smile$	$\smile$	$\sqsupset{^}\mid\smile\#$	$\sqsupset$	$\sqsupset$	$\equiv\sqsubset\sqsupset\smile\#$	$\sqsupset\smile\#$
$\#$	$\#$	$\sqsubset\smile\#$	$\sqsupset\mid\#$	$\#$	$\sqsupset\mid\#$	$\sqsubset\smile\#$	$\bullet$

In an implemented model, the complexity introduced by union relations is easily tamed. Every union relation which results from joining relations in $\mathfrak{B}$ contains #, and thus can safely be approximated by #. After all, # is already the least informative relation in $\mathfrak{B}$—loosely speaking, it indicates ignorance of the relationship between two expressions—and further joining will never serve to strengthen it. Our implemented model therefore has no need to represent union relations.

of sets (e.g. $\langle U \setminus a, U \setminus a\rangle$, for any $|a| = 1$) which cannot participate in the $|$ relation. However, the approximation makes little practical difference.

[11] Some union relations hold intrinsic interest. For example, in the three-way formulation of the NLI task described in section 2, the three classes can be identified as $\bigcup\{\equiv,\sqsubset\}$, $\bigcup\{^,|\}$, and $\bigcup\{\sqsupset,\smile,\#\}$.

[12] That is, the relations in $\mathfrak{B}$ plus 9 union relations. Note that this closure fails to include most of the 120 possible union relations. Perhaps surprisingly, the unions $\bigcup\{\equiv,\sqsubset\}$ and $\bigcup\{^,|\}$ mentioned in footnote 11 do not appear.

[13] In fact, computer experiments show that if relations are selected uniformly at random from $\mathfrak{B}$, it requires on average just five joins to reach $\bullet$.

[14] For compactness, we omit the union notation here; thus $\sqsubset|\#$ stands for $\bigcup\{\sqsubset,|,\#\}$.

4 Lexical entailment relations

Suppose x is a compound linguistic expression, and let $e(x)$ be the result of applying an *atomic edit e* (the deletion, insertion, or substitution of a subexpression) to x. The entailment relation $\beta(x, e(x))$ will depend on (1) the *lexical entailment relation* generated by e, which we label $\beta(e)$, and (2) other properties of the context x in which e is applied (to be discussed in section 5). For example, suppose x is *red car*. If e is SUB(*car*, *convertible*), then $\beta(e)$ is $\sqsupset$ (because *convertible* is a hyponym of *car*). On the other hand, if e is DEL(*red*), then $\beta(e)$ is $\sqsubset$ (because *red* is an intersective modifier). Crucially, $\beta(e)$ depends solely on the lexical items involved in e, independent of context.

How are lexical entailment relations determined? Ultimately, this is the province of lexical semantics, which lies outside the scope of this work. However, the answers are fairly intuitive in most cases, and we can make a number of useful observations.

Substitutions. The entailment relation generated by a substitution edit is simply the relation between the substituted terms: $\beta(\text{SUB}(x, y)) = \beta(x, y)$. For open-class terms such as nouns, adjectives, and verbs, we can often determine the appropriate relation by consulting a lexical resource such as WordNet. Synonyms belong to the $\equiv$ relation (*sofa* $\equiv$ *couch*, *forbid* $\equiv$ *prohibit*); hyponym-hypernym pairs belong to the $\sqsubset$ relation (*crow* $\sqsubset$ *bird*, *frigid* $\sqsubset$ *cold*, *soar* $\sqsubset$ *rise*); and antonyms and coordinate terms generally belong to the $|$ relation (*hot* $|$ *cold*, *cat* $|$ *dog*).[15] Proper nouns, which denote individual entities or events, will stand in the $\equiv$ relation if they denote the same entity (*USA* $\equiv$ *United States*), or the $|$ relation otherwise (*JFK* $|$ *FDR*). Pairs which cannot reliably be assigned to another entailment relation will be assigned to the # relation (*hungry* # *hippo*). Of course, there are many difficult cases, where the most appropriate relation will depend on subjective judgments about word sense, topical context, and so on—consider, for example, the pair *system* and *approach*. And some judgments may depend on world knowledge not readily available to an automatic system. For example, plausibly *skiing* $|$ *sleeping*, but *skiing* # *talking*.

Closed-class terms may require special handling. Substitutions involving generalized quantifiers generate a rich variety of entailment relations: *all* $\equiv$ *every*, *every* $\sqsubset$ *some*, *some* $^\wedge$ *no*, *no* $|$ *every*, *at least four* $\smile$ *at most six*, and *most* # *ten or more*.[16] Two pronouns, or a pronoun and a noun, should ideally be assigned to the $\equiv$ relation if it can determined from context that they refer to the same entity, though this may be difficult for an automatic system to establish reliably. Prepositions are somewhat problematic. Some pairs of prepositions can be interpreted as antonyms, and thus assigned to the $|$ relation (*above* $|$ *below*), but many prepositions are used so flexibly in natural language that they are best assigned to the $\equiv$ relation (*on* [*a plane*] $\equiv$ *in* [*a plane*] $\equiv$ *by* [*plane*]).

[15] Note that most antonym pairs do *not* belong to the $^\wedge$ relation, since they typically do not exclude the middle.

[16] Some of these assertions assume the non-vacuity (section 2) of the predicates to which the quantifiers are applied.

Generic deletions and insertions. For deletion edits, the default behavior is to generate the $\sqsubset$ relation (thus *red car* $\sqsubset$ *car*). Insertion edits are symmetric: by default, they generate the $\sqsupset$ relation (*sing* $\sqsupset$ *sing off-key*). This heuristic can safely be applied whenever the affected phrase is an intersective modifier, and can usefully be applied to phrases much longer than a single word (*car which has been parked outside since last week* $\sqsubset$ *car*). Indeed, this principle underlies most current approaches the RTE task, in which the premise p often contains much extraneous content not found in the hypothesis h. Most RTE systems try to determine whether p subsumes h: they penalize new content inserted into h, but do not penalize content deleted from p.

Special deletions and insertions. However, some lexical items exhibit special behavior upon deletion or insertion. The most obvious example is negation, which generates the $^\wedge$ relation (*didn't sleep* $^\wedge$ *did sleep*). Implicatives and factives (such as *refuse to* and *admit that*) constitute another important class of exceptions, but we postpone discussion of them to section 6. Then there are non-intersective adjectives such as *former* and *alleged*. These have various behavior: deleting *former* seems to generate the $|$ relation (*former student* $|$ *student*), while deleting *alleged* seems to generate the # relation (*alleged spy* # *spy*). We lack a complete typology of such cases, but consider this an interesting problem for lexical semantics. Finally, for pragmatic reasons, we typically assume that auxiliary verbs and punctuation marks are semantically vacuous, and thus generate the $\equiv$ relation upon deletion or insertion. When combined with the assumption that morphology matters little in inference,[17] this allows us to establish, e.g., that *is sleeping* $\equiv$ *sleeps* and *did sleep* $\equiv$ *slept*.

5 Entailment relations and semantic composition

How are entailment relations affected by semantic composition? In other words, how do the entailment relations between compound expressions depend on the entailment relations between their parts? Say we have established the value of $\beta(x,y)$, and let f be an expression which can take x or y as an argument. What is the value of $\beta(f(x), f(y))$, and how does it depend on the properties of f?

The monotonicity calculus of Sánchez Valencia provides a partial answer. It explains the impact of semantic composition on entailment relations $\equiv$, $\sqsubset$, $\sqsupset$, and # by assigning semantic functions to one of three monotonicity classes: UP, DOWN, and NON. If f has monotonicity UP (the default), then the entailment relation between x and y is projected through f without change: $\beta(f(x), f(y)) = \beta(x,y)$. Thus *some parrots talk* $\sqsubset$ *some birds talk*. If f has monotonicity DOWN, then $\sqsubset$ and $\sqsupset$ are swapped. Thus *no carp talk* $\sqsupset$ *no fish talk*. Finally, if f has monotonicity NON, then $\sqsubset$ and $\sqsupset$ are projected as #. Thus *most humans talk* # *most animals talk*.

[17] Indeed, the official definition of the RTE task explicitly specifies that tense be ignored.

The monotonicity calculus also provides an algorithm for computing the effect on entailment relations of multiple levels of semantic composition. Although Sánchez Valencia's presentation of this algorithm uses a complex scheme for annotating nodes in a categorial grammar parse, the central idea can be recast in simple terms: propagate a lexical entailment relation upward through a semantic composition tree, from leaf to root, while respecting the monotonicity properties of each node along the path. Consider the sentence *Nobody can enter without pants*. A plausible semantic composition tree for this sentence could be rendered as (*nobody* (*can* ((*without pants*) *enter*))). Now consider replacing *pants* with *clothes*. We begin with the lexical entailment relation: *pants* $\sqsubset$ *clothes*. The semantic function *without* has monotonicity DOWN, so *without pants* $\sqsupset$ *without clothes*. Continuing up the semantic composition tree, *can* has monotonicity UP, but *nobody* has monotonicity DOWN, so we get another reversal, and find that *nobody can enter without pants* $\sqsubset$ *nobody can enter without clothes*.

While the monotonicity calculus elegantly explains the impact of semantic composition on the containment relations (chiefly, $\sqsubset$ and $\sqsupset$), it lacks any account of the exclusion relations ($^\wedge$ and $|$, and, indirectly, $\smallsmile$). To remedy this lack, we propose to generalize the concept of monotonicity to a concept of *projectivity*. We categorize semantic functions into a number of *projectivity signatures*, which can be seen as generalizations of both the three monotonicity classes of Sánchez Valencia and the nine implication signatures of Nairn et al. (see section 6). Each projectivity signature is defined by a map $\mathfrak{B} \mapsto \mathfrak{B}$ which specifies how each entailment relation is projected by the function. (Binary functions can have different signatures for each argument.) In principle, there are up to 7^7 possible signatures; in practice, probably no more than a handful are realized by natural language expressions. Though we lack a complete inventory of projectivity signatures, we can describe a few important cases.

Negation. We begin with simple negation (*not*). Like most functions, it projects $\equiv$ and # without change (*not happy* $\equiv$ *not glad* and *isn't swimming* # *isn't hungry*). As a downward monotone function, it swaps $\sqsubset$ and $\sqsupset$ (*didn't kiss* $\sqsupset$ *didn't touch*). But we can also establish that it projects $^\wedge$ without change (*not human* $^\wedge$ *not nonhuman*) and swaps $|$ and $\smallsmile$ (*not French* $\smallsmile$ *not German* and *not more than 4* $|$ *not less than 6*). Its projectivity signature is therefore $\{\equiv{:}\equiv, \sqsubset{:}\sqsupset, \sqsupset{:}\sqsubset, {}^\wedge{:}{}^\wedge, |{:}\smallsmile, \smallsmile{:}|, \#{:}\#\}$.

Intersective modification. Intersective modification has monotonicity UP, but projects both $^\wedge$ and $|$ as $|$ (*living human* $|$ *living nonhuman* and *French wine* $|$ *Spanish wine*), and projects $\smallsmile$ as # (*metallic pipe* # *nonferrous pipe*). It therefore has signature $\{\equiv{:}\equiv, \sqsubset{:}\sqsubset, \sqsupset{:}\sqsupset, {}^\wedge{:}|, |{:}|, \smallsmile{:}\#, \#{:}\#\}$.[18]

Quantifiers. While semanticists are well acquainted with the monotonicity properties of common quantifiers, how they project the exclusion relations may be less

[18] At least for practical purposes. The projection of $^\wedge$ and $|$ as $|$ depends on the assumption of non-vacuity, and $\smallsmile$ is actually projected as $\bigcup\{\equiv, \sqsubset, \sqsupset, |, \#\}$, which we approximate by #, as described in section 3.

familiar. Table 3 summarizes the projectivity signatures of the most common binary generalized quantifiers for each argument position.

Table 3 Projectivity signatures for various quantifiers

	projectivity for 1st argument							projectivity for 2nd argument						
quantifier	$\equiv$	$\sqsubset$	$\sqsupset$	$\wedge$	$\mid$	$\smile$	#	$\equiv$	$\sqsubset$	$\sqsupset$	$\wedge$	$\mid$	$\smile$	#
some	$\equiv$	$\sqsubset$	$\sqsupset$	$\smile^{\dagger}$	#	$\smile^{\dagger}$	#	$\equiv$	$\sqsubset$	$\sqsupset$	$\smile^{\dagger}$	#	$\smile^{\dagger}$	#
no	$\equiv$	$\sqsupset$	$\sqsubset$	$\mid^{\dagger}$	#	$\mid^{\dagger}$	#	$\equiv$	$\sqsupset$	$\sqsubset$	$\mid^{\dagger}$	#	$\mid^{\dagger}$	#
every	$\equiv$	$\sqsupset$	$\sqsubset$	$\mid^{\ddagger}$	#	$\mid^{\ddagger}$	#	$\equiv$	$\sqsubset$	$\sqsupset$	$\mid^{\dagger}$	$\mid^{\dagger}$	#	#
not every	$\equiv$	$\sqsubset$	$\sqsupset$	$\smile^{\ddagger}$	#	$\smile^{\ddagger}$	#	$\equiv$	$\sqsupset$	$\sqsubset$	$\smile^{\dagger}$	$\smile^{\dagger}$	#	#

A few observations:

- All quantifiers (like most other semantic functions) project $\equiv$ and # without change.
- The table confirms well-known monotonicity properties: *no* is downward-monotone in both arguments, *every* in its first argument, and *not every* in its second argument.
- Relation $|$ is frequently "blocked" by quantifiers (i.e., projected as #). Thus *no fish talk* # *no birds talk* and *someone was early* # *someone was late*. A notable exception is *every* in its second argument, where $|$ is preserved: *everyone was early* $|$ *everyone was late*. (Note the similarity to intersective modification.)
- Because *no* is the negation of *some*, its projectivity signature can be found by projecting the signature of *some* through the signature of *not*. Likewise for *not every* and *every*.
- Some results depend on assuming the non-vacuity of the other argument to the quantifier: those marked with † assume it to be non-empty, while those marked with ‡ assume it to be non-universal. Without these assumptions, # is projected.

Verbs. Verbs (and verb-like constructions) exhibit diverse behavior. Most verbs are upward-monotone (though not all—see section 6), and many verbs project $\wedge$, $|$, and $\smile$ as # (*eats humans* # *eats nonhumans*, *eats cats* # *eats dogs*, and *eats mammals* # *eats nonhumans*). However, verbs which encode functional relations seem to exhibit the same projectivity as intersective modifiers, projecting $\wedge$ and $|$ as $|$, and $\smile$ as #.[19] Categorizing verbs according to projectivity is an interesting problem for lexical semantics, which may involve codifying some amount of world knowledge.

[19] Consider the verbal construct *is married to*: *is married to a German* $|$ *is married to a non-German*, *is married to a German* $|$ *is married to an Italian*, *is married to a European* # *is married to a non-German*. The AuContraire system [16] includes an intriguing approach to identifying such *functional phrases* automatically.

6 Implicatives and factives

[15] offer an elegant account of inferences involving implicatives and factives[20] such as *manage to*, *refuse to*, and *admit that*. Their model classifies such operators into nine *implication signatures*, according to their implications—positive (+), negative (–), or null (○)—in both positive and negative contexts. Thus *refuse to* has implication signature –/○, because it carries a negative implication in a positive context (*refused to dance* implies *didn't dance*), and no implication in a negative context (*didn't refuse to dance* implies neither *danced* nor *didn't dance*).

Most of the phenomena observed by Nairn et al. can be explained within our framework by specifying, for each implication signature, the relation generated when an operator of that signature is deleted from (or inserted into) a compound expression, as shown in Table 4.

Table 4 Implicatives and factives

	signature	$\beta(\text{DEL}(\cdot))$	$\beta(\text{INS}(\cdot))$	example
implicatives (UP)	+/–	$\equiv$	$\equiv$	*he managed to escape* $\equiv$ *he escaped*
	+/○	$\sqsubset$	$\sqsupset$	*he was forced to sell* $\sqsubset$ *he sold*
	○/–	$\sqsupset$	$\sqsubset$	*he was permitted to live* $\sqsupset$ *he lived*
implicatives (DOWN)	–/+	$^\wedge$	$^\wedge$	*he forgot to pay* $^\wedge$ *he paid*
	–/○	$\mid$	$\mid$	*he refused to fight* $\mid$ *he fought*
	○/+	$\smile$	$\smile$	*he hesitated to ask* $\smile$ *he asked*
factives (NON)	+/+	$\sqsubset$	$\sqsupset$	*he admitted that he knew* $\sqsubset$ *he knew*
	–/–	$\mid$	$\mid$	*he pretended he was sick* $\mid$ *he was sick*
	○/○	#	#	*he wanted to fly* # *he flew*

This table invites several observations. First, as the examples make clear, there is room for variation regarding the appearance of infinitive arguments, complementizers, passivization, and morphology. An implemented model must tolerate such diversity.

Second, some of the examples may seem more intuitive when one considers their negations. For example, deleting signature ○/– generates $\sqsupset$; under negation, this is projected as $\sqsubset$ (*he wasn't permitted to live* $\sqsubset$ *he didn't live*). Likewise, deleting signature ○/+ generates $\smile$; under negation, this is projected as $\mid$ (*he didn't hesitate to ask* $\mid$ *he didn't ask*).

Third, a fully satisfactory treatment of the factives (signatures +/+, –/–, and ○/○) would require an extension to our present theory. For example, deleting signature +/+ generates $\sqsubset$; yet under negation, this is projected not as $\sqsupset$, but as $\mid$ (*he didn't admit that he knew* $\mid$ *he didn't know*). The problem arises because the implication

[20] We use "factives" as an umbrella term embracing counterfactives and nonfactives along with factives proper.

carried by a factive is not an entailment, but a presupposition.[21] As is well known, the projection behavior of presuppositions differs from that of entailments [22]. It seems likely that our model could be elaborated to account for projection of presuppositions as well as entailments, but we leave this for future work.

We can further cement implicatives and factives within our model by specifying the monotonicity class for each implication signature: signatures +/–, +/○, and ○/– have monotonicity UP (*force to tango* $\sqsubset$ *force to dance*); signatures –/+, –/○, and ○/+ have monotonicity DOWN (*refuse to tango* $\sqsupset$ *refuse to dance*); and signatures +/+, –/–, and ○/○ (the propositional attitudes) have monotonicity NON (*think tangoing is fun* # *think dancing is fun*). We are not yet able to specify the complete projectivity signature corresponding to each implication signature, but we can describe a few specific cases. For example, implication signature –/○ seems to project $^\wedge$ as $|$ (*refuse to stay* $|$ *refuse to go*) and both $|$ and $\smile$ as # (*refuse to tango* # *refuse to waltz*).

7 Putting it all together

We now have the building blocks of a general method to establish the entailment relation between a premise p and a hypothesis h. The steps are as follows:

1. Find a sequence of atomic edits $\langle e_1, \ldots, e_n \rangle$ which transforms p into h: thus $h = (e_n \circ \ldots \circ e_1)(p)$. For convenience, let us define $x_0 = p$, $x_n = h$, and $x_i = e_i(x_{i-1})$ for $i \in [1, n]$.
2. For each atomic edit e_i:
   a. Determine the lexical entailment relation $\beta(e_i)$, as in section 4.
   b. Project $\beta(e_i)$ upward through the semantic composition tree of expression x_{i-1} to find an *atomic entailment relation* $\beta(x_{i-1}, e_i) = \beta(x_{i-1}, x_i)$, as in section 5.
3. Join atomic entailment relations across the sequence of edits, as in section 3: $\beta(p,h) = \beta(x_0,x_n) = \beta(x_0,e_1) \bowtie \ldots \bowtie \beta(x_{i-1},e_i) \bowtie \ldots \bowtie \beta(x_{n-1},e_n)$

However, this inference method has several important limitations, including the need to find an appropriate edit sequence connecting p and h;[22] the tendency of the join operation toward less informative entailment relations, as described in section 3; and the lack of a general mechanism for combining information from multi-

[21] Of course, the implicatives may carry presuppositions as well (*he managed to escape* → *it was hard to escape*), but these implications are not activated by a simple deletion, as with the factives.

[22] The order of edits can be significant, if one edit affects the projectivity properties of the context for another edit. In practice, we typically find that different edit orders lead to the same final result (albeit via different intermediate steps), or at worst to a result which is compatible with, though less informative than, the desired result. But in principle, edit sequences involving lexical items with unusual properties—not exhibited, so far as we are aware, by any natural language expressions—could lead to incompatible results. Thus we lack any formal guarantee of soundness.

ple premises.[23] Consequently, the method has less deductive power than first-order logic, and fails to sanction some fairly simple inferences, including de Morgan's laws for quantifiers. But the method neatly explains many inferences not handled by the monotonicity calculus.

For example, while the monotonicity calculus notably fails to explain even the simplest inferences involving semantic exclusion, such examples are easily accommodated in our framework. We encountered an example of such an inference in section 1: *Stimpy is a cat* $\models$ *Stimpy is not a poodle*. Clearly, this is a valid natural language inference. To establish this using our inference method, we must begin by selecting a sequence of atomic edits which transforms the premise p into the hypothesis h. While there are several possibilities, one obvious choice is first to replace *cat* with *dog*, then to insert *not*, and finally to replace *dog* with *poodle*. An analysis of this edit sequence is shown in Table 5. In this representation (of which we will see several more examples in the following pages), we show three entailment relations associated with each edit e_i, namely:

- $\beta(e_i)$, the lexical entailment relation generated by e_i,
- $\beta(x_{i-1}, e_i)$, the atomic entailment relation which holds across e_i, and
- $\beta(x_0, x_i)$, the cumulative join of all atomic entailment relations up through e_i. This can be calculated in the table as $\beta(x_0, x_{i-1}) \bowtie \beta(x_{i-1}, e_i)$.

Table 5 An example inference involving semantic exclusion

i	x_i	e_i	$\beta(e_i)$	$\beta(x_{i-1}, e_i)$	$\beta(x_0, x_i)$
0	*Stimpy is a cat*				
1	*Stimpy is a dog*	SUB(*cat*, *dog*)	$\mid$	$\mid$	$\mid$
2	*Stimpy is not a dog*	INS(*not*)	$\wedge$	$\wedge$	$\sqsubset$
3	*Stimpy is not a poodle*	SUB(*dog*, *poodle*)	$\sqsupset$	$\sqsubset$	$\sqsubset$

In Table 5, x_0 is transformed into x_3 by a sequence of three edits. First, replacing *cat* with its coordinate term *dog* generates $\mid$. Next, inserting *not* generates $\wedge$, and $\mid$ joined with $\wedge$ yields $\sqsubset$. Finally, replacing *dog* with its hyponym *poodle* generates $\sqsupset$. Because of the downward-monotone context created by *not*, this is projected as $\sqsubset$, and $\sqsubset$ joined with $\sqsubset$ yields $\sqsubset$. Therefore, x_0 entails x_3.

For an example involving an implicative, consider the inference in Table 6. Again, x_0 is transformed into x_3 by a sequence of three edits.[24] First, deleting *permitted to* generates $\sqsupset$, according to its implication signature; but because *not* is downward-monotone, this is projected as $\sqsubset$. Next, deleting *not* generates $\wedge$, and $\sqsubset$ joined with $\wedge$ yields $\mid$. Finally, inserting *Cuban cigars* restricts the meaning of *smoked*, generating $\sqsupset$, and $\mid$ joined with $\sqsupset$ yields $\mid$. So x_3 contradicts x_0.

[23] However, some inferences can be enabled by auxiliary premises encoded as lexical entailment relations. For example, *men* $\sqsubset$ *mortal* can enable the classic syllogism *Socrates is a man* $\sqsubset$ *Socrates is mortal*.

[24] We neglect edits involving auxiliaries and morphology, which simply yield the $\equiv$ relation.

Table 6 An example inference involving an implicative

i	x_i	e_i	$\beta(e_i)$	$\beta(x_{i-1}, e_i)$	$\beta(x_0, x_i)$
0	*We were not permitted to smoke*				
1	*We did not smoke*	DEL(*permitted to*)	$\sqsupset$	$\sqsubset$	$\sqsubset$
2	*We smoked*	DEL(*not*)	$\wedge$	$\wedge$	$\mid$
3	*We smoked Cuban cigars*	INS(*Cuban cigars*)	$\sqsupset$	$\sqsupset$	$\mid$

Let's now look at a more complex example (first presented in [14]) that demonstrates the interaction of a number of aspects of the model we've presented. The inference is:

> p: *Jimmy Dean refused to move without blue jeans.*
> h: *James Dean didn't dance without pants.*

Of course, the example is quite contrived, but it has the advantage that it compactly exhibits several phenomena of interest: semantic containment (between *move* and *dance*, and between *pants* and *jeans*); semantic exclusion (in the form of negation); an implicative (namely, *refuse to*); and nested inversions of monotonicity (created by *refuse to* and *without*). In this example, the premise p can be transformed into the hypothesis h by a sequence of seven edits, as shown in Table 7. This time we include even "light" edits yielding $\equiv$ for the sake of completeness.

Table 7 Analysis of a more complex inference

i	x_i	e_i	$\beta(e_i)$	$\beta(x_{i-1}, e_i)$	$\beta(x_0, x_i)$
	Jimmy Dean refused to move without blue jeans				
1		SUB(*Jimmy*, *James*)	$\equiv$	$\equiv$	$\equiv$
	James Dean refused to move without blue jeans				
2		DEL(*refused to*)	$\mid$	$\mid$	$\mid$
	James Dean moved without blue jeans				
3		INS(*did*)	$\equiv$	$\equiv$	$\mid$
	James Dean did move without blue jeans				
4		INS(*n't*)	$\wedge$	$\wedge$	$\sqsubset$
	James Dean didn't move without blue jeans				
5		SUB(*move*, *dance*)	$\sqsupset$	$\sqsubset$	$\sqsubset$
	James Dean didn't dance without blue jeans				
6		DEL(*blue*)	$\sqsubset$	$\sqsubset$	$\sqsubset$
	James Dean didn't dance without jeans				
7		SUB(*jeans*, *pants*)	$\sqsubset$	$\sqsubset$	$\sqsubset$
	James Dean didn't dance without pants				

We analyze these edits as follows. The first edit simply substitutes one variant of a name for another; since both substituends denote the same entity, the edit generates the $\equiv$ relation. The second edit deletes an implicative (*refuse to*) with implication signature –/○. As described in section 6, deletions of this signature generate the $\mid$

relation, and $\equiv$ joined with $\mid$ yields $\mid$. The third edit inserts an auxiliary verb (*did*); since auxiliaries are more or less semantically vacuous, this generates the $\equiv$ relation, and $\mid$ joined with $\equiv$ yields $\mid$ again. The fourth edit inserts a negation, generating the $^\wedge$ relation. Here we encounter the first interesting join: as explained in section 3, $\mid$ joined with $^\wedge$ yields $\sqsubset$. The fifth edit substitutes *move* with its hyponym *dance*, generating the $\sqsupset$ relation. However, because the edit occurs within the scope of the newly-introduced negation, $\sqsupset$ is projected as $\sqsubset$, and $\sqsubset$ joined with $\sqsubset$ yields $\sqsubset$. The sixth edit deletes a generic modifier (*blue*), which generates the $\sqsubset$ relation by default. This time the edit occurs within the scope of *two* downward-monotone operators (*without* and negation), so we have two inversions of monotonocity, and $\sqsubset$ is projected as $\sqsubset$. Again, $\sqsubset$ joined with $\sqsubset$ yields $\sqsubset$. Finally, the seventh edit substitutes *jeans* with its hypernym *pants*, generating the $\sqsubset$ relation. Again, the edit occurs within the scope of two downward-monotone operators, so $\sqsubset$ is projected as $\sqsubset$, and $\sqsubset$ joined with $\sqsubset$ yields $\sqsubset$. Thus p entails h.

Table 8 Analysis of a more complex inference, second try

i	e_i	$x_i = e_i(x_{i-1})$	$\beta(e_i)$	$\beta(x_{i-1}, e_i)$	$\beta(x_0, x_i)$
		Jimmy Dean refused to move without blue jeans			
1	INS(*did*)		$\equiv$	$\equiv$	$\equiv$
		Jimmy Dean did refuse to move without blue jeans			
2	INS(*n't*)		$^\wedge$	$^\wedge$	$^\wedge$
		Jimmy Dean didn't refuse to move without blue jeans			
3	DEL(*blue*)		$\sqsubset$	$\sqsupset$	$\mid$
		Jimmy Dean didn't refuse to move without jeans			
4	SUB(*jeans*, *pants*)		$\sqsubset$	$\sqsupset$	$\mid$
		Jimmy Dean didn't refuse to move without pants			
5	SUB(*move*, *dance*)		$\sqsupset$	$\sqsupset$	$\mid$
		Jimmy Dean didn't refuse to dance without pants			
6	DEL(*refuse to*)		$\mid$	$\smile$	$\sqsubset$
		Jimmy Dean didn't dance without pants			
7	SUB(*Jimmy*, *James*)		$\equiv$	$\equiv$	$\sqsubset$
		James Dean didn't dance without pants			

Of course, the edit sequence shown is not the only sequence which can transform p into h. A different edit sequence might yield a different sequence of intermediate steps, but the same final result. Consider, for example, the edit sequence shown in Table 8. Note that the lexical entailment relation $\beta(e_i)$ generated by each edit is the same as before. But because the edits involving downward-monotone operators (namely, INS(*n't*) and DEL(*refused to*)) now occur at different points in the edit sequence, many of the atomic entailment relations $\beta(x_{i-1}, e_i)$ have changed, and thus the sequence of joins has changed as well. In particular, edits 3 and 4 occur within the scope of *three* downward-monotone operators (negation, *refuse*, and *without*), with the consequence that the $\sqsubset$ relation generated by each of these lexical edits is projected as $\sqsupset$. Likewise, edit 5 occurs within the scope of two downward-monotone operators (negation and *refuse*), and edit 6 occurs within the scope of one downward-monotone operator (negation), so that $|$ is projected as $\smile$. Nevertheless, the ultimate result is still $\sqsubset$.

8 Implementation and evaluation

The model of natural logic described here has been implemented in software as the NatLog system. In previous work [14], we have presented a description and evaluation of NatLog; this section summarizes the main results. Natlog faces three primary challenges:

1. *Finding an appropriate sequence of atomic edits connecting premise and hypothesis.* NatLog does not address this problem directly, but relies instead on edit sequences from other sources. We have investigated this problem separately in [12].
2. *Determining the lexical entailment relation for each edit.* NatLog learns to predict lexical entailment relations by using machine learning techniques and exploiting a variety of manually and automatically constructed sources of information on lexical relations.
3. *Computing the projection of each lexical entailment relation.* NatLog identifies expressions with non-default projectivity and computes the likely extent of their arguments in a syntactic parse using hand-crafted tree patterns.

We have evaluated NatLog on two different test suites. The first is the FraCaS test suite [5], which contains 346 NLI problems, divided into nine sections, each focused on a specific category of semantic phenomena. The goal is three-way entailment classification, as described in section 2. On this task, NatLog achieves an average accuracy of 70%.[25] In the section concerning quantifiers, which is both the largest and the most amenable to natural logic, the system answers all problems but one correctly. Unsurprisingly, performance is mediocre in four sections concerning semantic phenomena (e.g., ellipsis) not relevant to natural logic and not modeled by the system. But in the other five sections (representing about 60% of the problems), NatLog achieves accuracy of 87%. What's more, precision is uniformly high, averaging 89% over all sections. Thus, even outside its areas of expertise, the system rarely predicts entailment when none exists.

The RTE3 test suite [8] differs from FraCaS in several important ways: the goal is binary entailment classification; the problems have much longer premises and are more "natural"; and the problems employ a diversity of types of inference—including paraphrase, temporal reasoning, and relation extraction—which NatLog is not designed to address. Consequently, the NatLog system by itself achieves mediocre accuracy (59%) on RTE3 problems. However, its precision is comparatively high, which suggests a strategy of hybridizing with a broad-coverage RTE

[25] Our evaluation excluded multi-premise problems, which constitute about 44% of the test suite.

system. We were able to show that adding NatLog as a component in the Stanford RTE system [3] led to accuracy gains of 4%.

9 Conclusion

The model of natural logic presented here is by no means a universal solution to the problem of natural language inference. Many NLI problems hinge on types of inference not addressed by natural logic, and the inference method we describe faces a number of limitations on its deductive power (discussed in section 7). Moreover, there is further work to be done in fleshing out our account, particularly in establishing the proper projectivity signatures for a broader range of quantifiers, verbal constructs, implicatives and factives, logical connectives, and other semantic functions.

Nevertheless, we believe our model of natural logic fills an important niche. While approximate methods based on lexical and syntactic similarity can handle many NLI problems, they are easily confounded by inferences involving negation, antonymy, quantifiers, implicatives, and many other phenomena. Our model achieves the logical precision needed to handle such inferences without resorting to full semantic interpretation, which is in any case rarely possible. The practical value of the model is demonstrated by its success in evaluations on the FraCaS and RTE3 test suites.

References

1. J. Bos and K. Markert. Recognising textual entailment with logical inference. In *Proceedings of EMNLP-05*, 2005.
2. M. Böttner. A note on existential import. *Studia Logica*, 47(1):35–40, 1988.
3. N. Chambers, D. Cer, T. Grenager, D. Hall, C. Kiddon, B. MacCartney, M. C. de Marneffe, D. Ramage, E. Yeh, and C. D. Manning. Learning Alignments and Leveraging Natural Logic. In *Proceedings of the ACL-07 Workshop on Textual Entailment and Paraphrasing*, 2007.
4. C. Condoravdi, D. Crouch, V. de Paiva, R. Stolle, and D.G. Bobrow. Entailment, Intensionality and Text Understanding. In *Proceedings of the HLT-NAACL 2003 Workshop on Text Meaning*, Morristown, NJ, USA, 2003.
5. R. Cooper et al. Using the framework. Technical Report LRE 62-051 D-16, The FraCaS Consortium, 1996.
6. I. Dagan, O. Glickman, and B. Magnini. The PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, 2005.
7. Y. Fyodorov, Y. Winter, and N. Francez. A Natural Logic Inference System. In *Proceedings of the 2nd Workshop on Inference in Computational Semantics (ICoS-2)*, 2000.
8. D. Giampiccolo, B. Magnini, I. Dagan, and B. Dolan. The Third PASCAL Recognizing Textual Entailment Challenge. In *Proceedings of the ACL-07 Workshop on Textual Entailment and Paraphrasing*, 2007.
9. O. Glickman, I. Dagan, and M. Koppel. Web based probabilistic textual entailment. In *Proceedings of the PASCAL Challenges Workshop on Recognizing Textual Entailment*, 2005.

10. A. Hickl, J. Williams, J. Bensley, K. Roberts, B. Rink, and Y. Shi. Recognizing Textual Entailment with LCC's GROUNDHOG System. In *Proceedings of the Second PASCAL Challenges Workshop on Recognizing Textual Entailment*, 2006.
11. G. Lakoff. Linguistics and natural logic. *Synthese*, 22:151–271, 1970.
12. B. MacCartney, M. Galley, and C. D. Manning. A phrase-based alignment model for natural language inference. In *Proceedings of EMNLP-08*, Honolulu, HI, 2008.
13. B. MacCartney, T. Grenager, M. C. de Marneffe, D. Cer, and C. D. Manning. Learning to Recognize Features of Valid Textual Entailments. In *Proceedings of NAACL-06*, New York, 2006.
14. B. MacCartney and C. D. Manning. Modeling semantic containment and exclusion in natural language inference. In *Proceedings of Coling-08*, Manchester, UK, 2008.
15. R. Nairn, C. Condoravdi, and L. Karttunen. Computing relative polarity for textual inference. In *Proceedings of ICoS-5*, Buxton, UK, 2006.
16. A. Ritter, D. Downey, S. Soderland, and O. Etzioni. It's a Contradiction—No, it's Not: A Case Study using Functional Relations. In *Proceedings of EMNLP-08*, 2008.
17. V. Sánchez Valencia. *Studies on Natural Logic and Categorial Grammar*. PhD thesis, Univ. Amsterdam, 1991.
18. J. Sukkarieh. Quasi-NL Knowledge Representation for Structurally-Based Inferences. In *Proceedings of the 3rd Workshop on Inference in Computational Semantics (ICoS-3)*, 2001.
19. J. van Benthem. The semantics of variety in categorial grammars. In W. Buszkowski, W. Marciszewski, and J. van Benthem, editors, *Categorial grammar*, pages 33–55. John Benjamins, Amsterdam, 1988.
20. J. van Benthem. *Language in Action: categories, lambdas and dynamic logic*, volume 130 of *Studies in Logic*. North-Holland, Amsterdam, 1991.
21. J. van Benthem. A brief history of natural logic. Technical Report PP-2008-05, Institute for Logic, Language & Computation, 2008.
22. R. A. van der Sandt. Presupposition projection as anaphora resolution. *Journal of Semantics*, 9(4), 1992.
23. J. van Eijck. Natural logic for natural language. http://homepages.cwi.nl/˜jve/papers/05/nlnl/NLNL.pdf, 2005.