

NaturalLI: Natural Logic Inference for Common Sense Reasoning

NOTE: intro+natlog rewritten; typos fixed everywhere

Abstract

Common-sense reasoning is important for AI applications, both in NLP and many vision or robotics tasks. We propose NaturalLI: a Natural Logic inference system for inferring common sense facts – for instance, that *cats have tails* or *tomatoes are fragile* – from a very large database of known facts. The system provides logically valid derivations, while also allowing derivations which are only likely valid, accompanied by an associated confidence. We show that our system is able to capture strict Natural Logic inferences on the FraCaS test suite, and demonstrate the system’s ability to infer previously unseen facts with 50% recall and 91% precision.

1 Introduction

We approach the task of *database completion*: given a database of facts, we would like to predict whether an unseen fact should belong to the database. This is naturally cast as an inference problem from a collection of candidate premises to the truth of the query. For example, we would like to be able to infer that *no carnivores eat animals* is false given a database containing *the cat ate a mouse* (see Figure 1).

These inferences are difficult to capture with high recall in a principled way, particularly for open-domain text. Learned inference rules are difficult to generalize to large sets of relations, and standard IR methods easily miss small but semantically important lexical differences. Furthermore, many methods require explicitly modeling either the database, the query, or both in a formal meaning representation (e.g., Freebase tuples).

Although projects like the Abstract Meaning Representation (Banarescu et al., 2013) have made headway providing a broad-coverage meaning

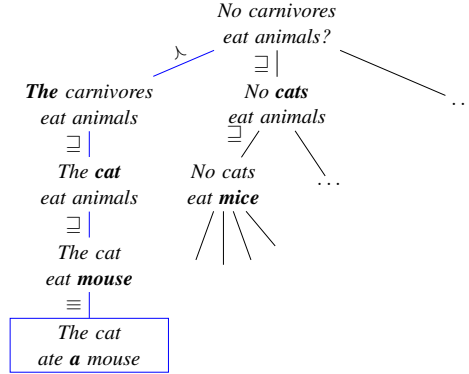


Figure 1: Natural Logic inference cast as search, disproving the query *no carnivores eat animals* given the premise *the cat ate a mouse*. The valid path is one of many candidates taken; the premise is one of many known facts in the database. The edge labels denote Natural Logic inference steps.

representation, it remains appealing to use the human language itself as the vessel for inference. Moreover, OpenIE and similar projects have been very successful at collecting a database of natural language snippets from an ever-increasing corpus of unstructured text. These factors motivate our use of Natural Logic – a proof system built on the syntax of human language – to create a system for broad coverage database completion.

Prior work on Natural Logic has focused on inferences from a single relevant premise to justify a conclusion, making use of only formally valid inferences. We propose three contributions to computational Natural Logic: (i) This work operates over a very large set of (generally irrelevant) candidate premises simultaneously; (ii) this work maintains logically valid inferences without needing to appeal to explicit alignment between the premise and the query; and (iii) we allow imprecise inferences at an associated learned cost to provide a more broad-coverage system over real-world facts.

Our approach casts inference as a search problem from a query to any premise which would support that query. Each transition along the search denotes a (reverse) inference step in Natural Logic, and incurs a cost reflecting the confidence in the validity of that step. This approach offers two big contributions over prior work in database completion: (i) it allows for unstructured text as the input database without any assumptions about the schema or domain of the text, and (ii) it proposes Natural Logic rather than explicit inference rules as the means of proposing inferences.

2 MacCartney’s Natural Logic

Broadly, Natural Logic aims to capture common logical inferences by appealing directly to the structure of language, as opposed to running deduction on an abstract logical form. Natural Logic is inspired largely by the Aristotelian syllogisms and subsequent traditional logics (van Benthem, 2008). While not all inferences in first-order logic are in Natural Logic, it nonetheless allows for a wide range of intuitive inferences in a computationally efficient manner.

We build off of the variant of Natural Logic introduced by the NatLog inference system (MacCartney and Manning, 2007; 2008; 2009), based off of earlier theoretical work on Natural Logic and Monotonicity Calculus (Van Benthem, 1986; Valencia, 1991). Later work has formalized many aspects of the logic (Icard III, 2012; Djalali, 2013); formally we are adopting the semantics of outlined in Icard III and Moss (2014).

At a high level, Natural Logic proofs operate by mutating spans of text in precise ways to ensure that the mutated sentence follows from the original – each step is much like a syllogistic inference. We therefore must introduce three components for a complete proof system: we define the valid atomic relations between lexical entries (Section 2.1), the effect these mutations have on the validity of the inference (Section 2.2), and a practical system for executing these proofs. We introduce MacCartney’s alignment-based approach in Section 2.3, and show that we can generalize and simplify this system in Section 3.

2.1 Lexical Relations

MacCartney and Manning (2007) introduce seven set-theoretic relations between the denotations of any two lexical items. The denotation of a lexical

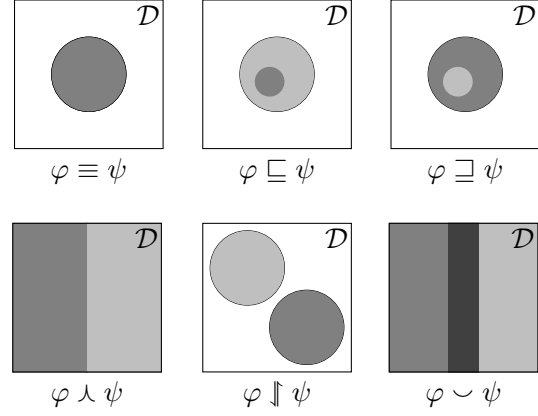


Figure 2: The model-theoretic interpretation of the MacCartney relations. The figure shows the relation between the denotation of φ , in dark gray, compared to the denotation of ψ , in light gray. The universe is denoted by \mathcal{D} .

item is the set of objects in the domain of discourse \mathcal{D} which that lexical item refers to. For instance, the denotation of *cat* would be the set of all cats. The denotations of two lexical items can then be compared in terms of set algebra: if we define the set of cats to be φ and the set of animals to be ψ , we can state that $\varphi \subseteq \psi$.

The six informative relations are summarized in Figure 2; a seventh relation ($\#$) corresponds to the completely uninformative relation. For instance, the example search path in Figure 1 makes use of the following relations:

<i>No x y</i>	\sqcap	<i>The x y</i>
<i>cat</i>	\sqsupseteq	<i>carnivore</i>
<i>animals</i>	\sqsupseteq	<i>mouse</i>
<i>mouse</i>	\equiv	<i>a mouse</i>

Denotations are not required to be in the space of entities. In the first example, the denotations of *No* and *The* are in the space of quantifiers $e \rightarrow (e \rightarrow t)$: functions from entities e to truth values t . Our claim is really the conjunction of two claims: $\forall x \forall y \neg (no\ x\ y \wedge the\ x\ y)$ and $\forall x \forall y (no\ x\ y \vee the\ x\ y)$. This is analogous to the formal construction of the set-theoretic definition of \sqcap in Figure 2: $\varphi \cap \psi = \emptyset$ and $\varphi \cup \psi = \mathcal{D}$ (see Icard III and Moss (2014)).

Examples of the last two relations (\parallel and \sim) and the complete independence relation ($\#$) are given below:

<i>cat</i>	\parallel	<i>dog</i>
<i>animal</i>	\sim	<i>nonhuman</i>
<i>cat</i>	$\#$	<i>friendly</i>

2.2 Monotonicity and Polarity

We can determine the relation between lexical items from the previous section; however, we still need a theory for how to propagate the relation induced by a lexical mutation “up the parse tree” to become a relation between two sentences. For example, $cat \sqsubseteq animal$, and $some\ cat\ meows \sqsubseteq some\ animal\ meows$, but $no\ cat\ barks \sqsupseteq no\ animal\ barks$. Despite differing by the same lexical relation, the first example is entailed, while the second is not.

We appeal two important concepts: *monotonicity* as a property of arguments to natural language quantifiers, and *polarity* as a property of lexical items in a sentence. Much like monotone functions in calculus, an [upwards] monotone quantifier has an output truth value which is non-decreasing (i.e., material implication; true cannot become false) as the input “increases” (i.e., in terms of set containment). From the example above, *some* is upwards monotone in its first argument, and *no* is downwards monotone in its first argument.

Polarity is a property of lexical items in a sentence determined by the quantifiers acting on it. All lexical items have *upward* polarity by default; upwards monotone quantifiers preserve polarity, and downwards monotone quantifiers reverse polarity. For example, *cats* in *all cats eat mice* has downward polarity; or, *mice* in *no cats don’t eat mice* has upward polarity (in the scope of two downward monotone quantifiers). The relation between two sentences differing by a single lexical relation is then given in the projection table (Table 1).¹

2.3 Proof By Alignment

MacCartney and Manning (2007) approach the inference task in the context of inferring whether a single relevant premise entails a query. Their approach is to generate an alignment between the premise and the query, and classify each aligned segment into one of the relations in Figure 2. Inference then reduces to projecting each of these relations according to Table 1 and iteratively *joining* two projected relations together to get the final entailment relation. This *join table*, denoted as \bowtie , is given in Table 2.

¹Note that this table optimistically assumes every quantifier is *additive* and *multiplicative*, as defined in Icard III (2012).

Relation	Polarity of Context	
	Upward	Downward
$e_1 \sqsubseteq e_2$	$s_1 \sqsubseteq s_2$	$s_1 \sqsupseteq s_2$
$e_1 \sqsupseteq e_2$	$s_1 \sqsupseteq s_2$	$s_1 \sqsubseteq s_2$
$e_1 \Downarrow e_2$	$s_1 \Downarrow s_2$	$s_1 \sim s_2$
$e_1 \sim e_2$	$s_1 \sim s_2$	$s_1 \Downarrow s_2$
$e_1 \equiv e_2$	$s_1 \equiv s_2$	$s_1 \equiv s_2$
$e_1 \wedge e_2$	$s_1 \wedge s_2$	$s_1 \wedge s_2$
$e_1 \# e_2$	$s_1 \# s_2$	$s_1 \# s_2$

Table 1: The projection table used for a relation between a phrase e_1 and its candidate mutation e_2 , in terms of the produced relation between sentence s_1 and the new sentence s_2 . Values are given for when the entities are in a monotone and antitone context.

\bowtie	\sqsubseteq	\sqsupseteq	\wedge	\Downarrow	\sim
\sqsubseteq	\sqsubseteq	$\#$	\Downarrow	\Downarrow	$\#$
\sqsupseteq	$\#$	\sqsubseteq	\sim	$\#$	\sim
\wedge	\sim	\Downarrow	\equiv	\sqsubseteq	\sqsubseteq
\Downarrow	$\#$	\Downarrow	\sqsubseteq	$\#$	\sqsubseteq
\sim	\sim	$\#$	\sqsubseteq	\sqsubseteq	$\#$

Table 2: The join table, as copied from Icard III (2012). Note that the $\#$ always joins to yield $\#$, and \equiv always joins to yield the input relation.

To illustrate we can consider MacCartney’s example inference from *Stimpy is a cat* to *Stimpy is not a poodle*. An alignment of the two statements would provide three lexical mutations: $r_1 := cat \rightarrow dog$, $r_2 := \cdot \rightarrow not$, and $r_3 := dog \rightarrow poodle$. The initial relation r_0 is axiomatically \equiv . Each of these then project through the projection function ρ from Table 1, and are joined by the join relation:

$$r_0 \bowtie \rho(r_1) \bowtie \rho(r_2) \bowtie \rho(r_3)$$

In MacCartney’s work this style of proof is presented as a table, as follows. The last column (s_i) is the relation between the premise and the i^{th} step in the proof, and is constructed inductively as $s_i := s_{i-1} \bowtie \rho(r_i)$:

Mutation	r_i	$\rho(r_i)$	s_i
$r_1 \quad cat \rightarrow dog$	\Downarrow	\Downarrow	\Downarrow
$r_2 \quad \cdot \rightarrow not$	\wedge	\wedge	\sqsubseteq
$r_3 \quad dog \rightarrow poodle$	\sqsubseteq	\sqsubseteq	\sqsubseteq

In our example, we would conclude that *Stimpy is a cat* \sqsubseteq *Stimpy is not a poodle* since s_3 is \sqsubseteq , and therefore the inference is valid.

information to handle various details.

Lexical mutations As per our definition, transitions between nodes are transitions between facts. However, the lexical resources for the mutations are over lexical items (e.g., *feline* \sqsupseteq *cat*) rather than entire facts. This motivates the inclusion of an index i in our node, denoting the index of the item which may be mutated. Importantly, this also imposes a natural ordering of items to mutate, making search more efficient at the expense of rare search errors.

Predicting deletions Although inserting lexical items in a derivation (deleting words from the reversed derivation) is trivial, the other direction is not. For brevity, we refer to a deletion in the derivation as an insertion, as from the perspective of the search algorithm constructing a reverse derivation, we are inserting lexical items. naïvely, at every node in our search, we must consider every item in the vocabulary as a possible insertion.

This is largely handled by storing the database of known facts as a trie. Since search mutates the fact left-to-right (as per above), we can look up completions in the trie to use as candidate insertions. To illustrate, given a search state with fact w_0, w_1, \dots, w_n and mutation index i , we would look up completions w_{i+1} for w_0, w_1, \dots, w_i in our trie of known facts.

Although this approach works well when i is relatively large, there are likely many candidate insertions for small i . We special case the most extreme case for this, where $i = 0$; that is, when we are inserting into the beginning of the fact. In this case, rather than taking all possible lexical items that start any fact, we take all items which are followed by the first word of our current fact. For instance, given a search state with fact w_0, w_1, \dots, w_n , we would propose candidate insertions w_{-1} such that $w_{-1}, w_0, w'_1, \dots, w'_k$ is a known fact, for some w'_1, \dots, w'_k .

Polarity tracking Mutating quantifiers can change the polarity information on a span in the fact. Since we do not have the full parse tree at our disposal at search, we track a small amount of metadata to guess the scope of the mutated quantifier.

4.2 Transitions

We begin by introducing some terminology. A *transition template* is a broad class of transitions;

for instance WordNet hypernymy. A *transition* or *transition instance* is a particular instantiation of a transition template. For example, the transition from *cat* to *feline*. Lastly, an *edge* in the search space connects two facts, which are separated by a single transition instance. For example, an edge exists between *some cats have tails* and *some felines have tails*.

Note that the edges in the search are not constructed *a priori*. It is sufficient to store the transitions, and construct particular edges on demand. At a high level, we include most relations in WordNet as transitions, and parametrize insertions and deletions by the part of speech of the token being inserted/deleted. The full table of transitions is given in Table 3, along with the relation that transition introduces, and an example edge in our derivation corresponding to that transition.

It should be noted that the mapping from transitions to relation types is intentionally imprecise. For instance, clearly nearest neighbors do not preserve equivalence (\equiv); more subtly, while *all cats like milk* \Downarrow *all cats hate milk*, it is not the case that *some cats like milk* \Downarrow *some cats hate milk*.² We mitigate this imprecision by introducing a cost for each transition, and learning the appropriate value for this cost (see Section 5).

The cost of an edge is parametrized by two values; the cost of an edge from fact (f, v, p) with surface form f , validity v and polarity p to a new fact (f', v', p') using a transition instance t_i of type t is given by $f_{t_i} \cdot \theta_{t,v,p}$, where:

f_{t_i} : A value associated with every transition instance t_i , intuitively corresponding to how “far” the endpoints of a mutation are.

$\theta_{t,v,p}$: A learned cost for taking a transition of type t , if the source of the edge is in a validity state of v and the word being mutated has polarity p .

The notation for f_{t_i} is chosen to evoke an analogy to features, and we will refer to this quantity as the feature value. We set f_{t_i} to be 1 in most cases; the exception are the edges over the WordNet hypernym tree, and the nearest neighbors edges. In the first case, we take $\uparrow_{w \rightarrow w'}$ as transitioning from word w to its hypernym w' (and vice versa for $\downarrow_{w \rightarrow w'}$), and set:

²The latter example is a consequence of the projection table in Table 1 being overly optimistic.

Template	Relation	Example edge
WordNet hypernym	\sqsubseteq	<i>some cats like milk</i> \sqsubseteq <i>some felines like milk</i>
WordNet hyponym	\sqsupseteq	<i>some felines like milk</i> \sqsupseteq <i>some cats like milk</i>
WordNet antonym [†]	\mathbb{J}	<i>all cats like milk</i> \mathbb{J} <i>all cats hate milk</i>
WordNet synonym/pertainym [†]	\equiv	<i>some cats like milk</i> \equiv <i>some cats enjoy milk</i>
Distributional nearest neighbor	\equiv	<i>some cats like milk</i> \equiv <i>some cats like dairy</i>
Delete word [†]	\sqsubseteq	<i>some tabby cats like milk</i> \sqsubseteq <i>some cats like milk</i>
Add word [†]	\sqsupseteq	<i>some cats like milk</i> \sqsupseteq <i>some tabby cats like milk</i>
Quantifier weaken	\sqsubseteq	<i>all cats like milk</i> \sqsubseteq <i>some cats like milk</i>
Quantifier strengthen	\sqsupseteq	<i>some cats like milk</i> \sqsupseteq <i>all felines like milk</i>
Quantifier negate	\wedge	<i>some cats like milk</i> \wedge <i>no felines like milk</i>
Quantifier synonym	\equiv	<i>some cats like milk</i> \equiv <i>a few cats like milk</i>
Change word sense	\equiv	

Table 3: The edges allowed during inference. Entries with a dagger are parametrized by their part-of-speech tag, from the restricted list of {noun, adjective, verb, other}. The first column describes the type of the transition. The set-theoretic relation introduced by each relation is given in the second column. The third column gives an example of the transition in practice, as an edge in the search graph.

$$f_{\uparrow_{w \rightarrow w'}} = \log \frac{p(w')}{p(w)} = \log p(w') - \log p(w)$$

$$f_{\downarrow_{w \rightarrow w'}} = \log \frac{p(w)}{p(w')} = \log p(w) - \log p(w')$$

We define $p(w)$ to be the probability (normalized frequency) of a word or any of its hyponyms in the Google N-Grams corpus (Brants and Franz, 2006). Intuitively, this ensures that relatively long paths through fine-grained sections of WordNet are not unduly penalized. For instance, the path from *cat* to *animal* traverses six intermediate nodes, naïvely yielding a prohibitive search depth of 6.

For nearest neighbors edges, we take Neural Network embeddings learned in Huang et al. (2012) corresponding to each vocabulary entry. We then define $f_{NN_{w \rightarrow w'}}$ to be the arc cosine of the cosine similarity between word vectors associated with lexical items w and w' :

$$f_{NN_{w \rightarrow w'}} = \arccos \left(\frac{w \cdot w'}{\|w\| \|w'\|} \right)$$

The set of features which fire along a path can be expressed as a vector \mathbf{f} . Each element of \mathbf{f} corresponds to the sum of the values of that feature along the path. Correspondingly, we define the weight vector θ as the weight for every element of \mathbf{f} . The cost of a path can then be expressed as the dot product: $\theta \cdot \mathbf{f}$.

4.3 Generalizing Similarities

An elegant property of our definitions of f_{t_i} is its ability to generalize JC distance, and upper-bound distributional similarity. Let us assume we have words w_1 and w_2 , with a least common subsumer lcs. The JC distance $\text{dist}_{jc}(w_1, w_2)$ is:

$$\text{dist}_{jc}(w_1, w_2) = \log \frac{p(\text{lcs})^2}{p(w_1)p(w_2)} \quad (1)$$

For simplicity, we simplify $\theta_{\uparrow, v, p}$ and $\theta_{\downarrow, v, p}$ as simply θ_{\uparrow} and θ_{\downarrow} . The derivation generalizes trivially to the case where weights are further parametrized. Without loss of generality, we also assume that a path in our search is only modifying a single word w_1 , ending at a mutation of the word w_2 .

We can factorize the cost of a path, $\theta \cdot \mathbf{f}$, along the path from w_1 to w_2 through its lowest common subsumer (lcs), $[w_1, w_1^{(1)}, \dots, \text{lcs}, \dots, w_2^{(1)}, w_2]$, as follows:

$$\begin{aligned} \theta \cdot \phi &= \theta_{\uparrow} \left(\left[\log p(w_1^{(1)}) - \log p(w_1) \right] + \dots \right) + \\ &\quad \theta_{\downarrow} \left(\left[\log p(\text{lcs}) - \log p(w_1^{(n)}) \right] + \dots \right) \\ &= \theta_{\uparrow} \left(\log \frac{p(\text{lcs})}{p(w_1)} \right) + \theta_{\downarrow} \left(\log \frac{p(\text{lcs})}{p(w_2)} \right) \\ &= \log \frac{p(\text{lcs})^{\theta_{\uparrow} + \theta_{\downarrow}}}{p(w_1)^{\theta_{\uparrow}} + p(w_2)^{\theta_{\downarrow}}} \end{aligned}$$

Note that setting both θ_{\uparrow} and θ_{\downarrow} to 1 exactly yield the Formula (1) for JC distance.

It is also worth noting that the nearest neighbors path provides an upper bound on the true similarity between the start and end words in the path. In this way, the search based approach presented here can be thought of as generalizing and formalizing the intuition that similar objects have similar properties (e.g., as presented in Angeli and Manning (2013)) using both common classes of similarity metrics.

4.4 Confidence Estimation

The last component in inference is translating a search path into a probability of truth. We notice from Section 4.2 that the *cost* of a path can be represented as $\theta \cdot \mathbf{f}$. We can normalize this value by negating every element of the weight vector and passing it through a sigmoid:

$$\text{confidence} = \frac{1}{1 + e^{\theta \cdot \mathbf{f}}}$$

Importantly, note that the cost vector must be non-negative for the search to be well-defined, and therefore the confidence value will be constrained to be between 0 and $\frac{1}{2}$.

At this point, we have a confidence that the given path has not violated strict Natural Logic. However, to translate this value into a probability we need to incorporate whether the inference path is confidently valid, or confidently invalid. To illustrate, a fact with a low confidence should translate to a probability of $\frac{1}{2}$, rather than a probability of 0.

We therefore define the probability of validity as follows. We take v to be 1 if the final fact of our derivation is in the *valid* state with respect to the antecedent, and -1 if this fact is in the *invalid* state. In practice, as our search is reversed, we take the state of the antecedent in our database when compared to the consequent, rather than vice versa. For completeness, if no path is given we can set $v = 0$. The probability of validity then becomes:

$$p(\text{valid}) = \frac{v}{2} + \frac{1}{1 + e^{v\theta \cdot \mathbf{f}}} \quad (2)$$

Note that in the case where $v = -1$, the above expression reduces to $\frac{1}{2} - \text{confidence}$; in the case where $v = 0$ it reduces to simply $\frac{1}{2}$. Furthermore, note that the probability of truth makes use of the same parameters as the cost in the search. Thus, as better weights are learned, the search is likewise more likely to produce derivations which would

confidently support or disprove the query. We proceed to describe how these weights are learned.

5 Learning Transition Costs

The learning task can be viewed as a constrained optimization problem. Subject to the constraint that all elements of the cost vector θ must be non-negative, we optimize the probability from Equation (2) compared against the gold annotation. As training data, we are given a number of facts, annotated with a truth value of *true* or *false*. We assume that all facts in our database are true; therefore, $p(\text{valid})$ corresponds directly to $p(\text{true})$.

We learn costs using an iterative algorithm. At each iteration, we take the costs from the previous iteration and run search over every example to obtain candidate paths. At each iteration, analogous to logistic regression, the log-likelihood of the training data becomes:

$$\begin{aligned} ll_{\theta}(\mathcal{D}) = \sum_{0 \leq i < |\mathcal{D}|} & \left[y_i \log \left(\frac{v_i}{2} + \frac{1}{1 + e^{v_i \theta \cdot f(x_i)}} \right) \right. \\ & \left. + (1 - y_i) \log \left(\frac{-v_i}{2} + \frac{1}{1 + e^{-v_i \theta \cdot f(x_i)}} \right) \right], \end{aligned}$$

where y_i is 1 if the example is annotated true, and 0 otherwise, $f(x_i)$ are the features extracted for path i , and v_i is the inference state predicted in search, as in Section 4.4. The objective function becomes our negative log-likelihood, in addition to an l_2 regularization term, and a log barrier function to prohibit negative costs:

$$O(\mathcal{D}) = -ll_{\theta}(\mathcal{D}) + \frac{1}{2\sigma^2} \|\theta\|_2^2 - \epsilon \log(-\theta).$$

We set $\sigma = 100$ and $\epsilon = 1e^{-5}$.

Although this objective is non-convex, there is a natural initialization of costs to match intuitions about Natural Logic. We initialize the first iteration to these costs, and run conjugate gradient descent to convergence on a local minima on each subsequent iteration. Lastly, after each iteration, the cost for any feature which has never fired during training is decreased by half. This is to encourage the search to take these expensive edges, and learn an appropriate cost for them.

6 Experiments

We evaluate our system on two tasks: the FraCaS test suite used by MacCartney and Manning

§	Category	Count	Precision		Recall		Accuracy		
			N	M08	N	M08	N	M07	M08
1	Quantifiers	44	91	95	100	100	95	84	97
2	Plurals	24	80	90	29	64	38	42	75
3	Anaphora	6	100	100	20	60	33	50	50
4	Ellipses	25	100	100	5	5	28	28	24
5	Adjectives	15	80	71	66	83	73	60	80
6	Comparatives	16	90	88	100	89	87	69	81
7	Temporal	36	75	86	53	71	52	61	58
8	Verbs	8	—	80	0	66	25	63	62
9	Attitudes	9	—	100	0	83	22	55	89
Applicable (1,5,6)		75	89	89	94	94	89	76	90

Table 4: Results on the FraCaS textual entailment suite. N is this work; M07 refers to MacCartney and Manning (2007); M08 refers to MacCartney and Manning (2008). The relevant sections of the corpus intended to be handled by this system are sections 1, 5, and 6 (not 2 and 9, which are also included in M08).

(2007; 2008), evaluates the system’s ability to capture Natural Logic inferences even without the explicit alignments of these previous systems. In addition, we evaluate the system’s ability to predict common-sense facts from a large corpus of OpenIE extractions.

6.1 FraCaS Entailment Corpus

The FraCaS corpus (Cooper et al., 1996) is a small corpus of entailment problems, aimed at providing a comprehensive test of an entailment system’s handling of various entailment patterns.

We process the corpus following MacCartney and Manning (2007). 12 problems were discarded as degenerate – lacking either an antecedent or a consequent. 151 problems were discarded as involving multiple antecedents to justify the inference. Lastly, it should be noted that many of the sections of the corpus are not directly applicable to Natural Logic inferences; MacCartney and Manning (2007) identify three sections which are in the scope of their system.

Results on the corpus are given in Table 4. Since the corpus is not a blind test set, the results are presented less as a comparison of performance, but rather as a comparison of the expressive power of our search-based approach compared with MacCartney’s align-and-classify approach. For the experiments, costs were hard-coded to represent a strict logical entailment system – costs corresponding to valid Natural Logic mutations were set to a small constant cost; other costs were set to infinity.

The results validate the system’s ability to capture valid Natural Logic inferences as well as the

System	Precision	Recall	Accuracy
Lookup	100.0	12.1	56.0
NaturalLI Only	91.9	39.5	68.0
NaturalLI + Lookup	93.3	48.3	72.6

Table 5: Accuracy inferring common-sense facts on a balanced test set. Lookup queries the lemmatized lower-case fact directly in the 300M fact database. NaturalLI Only disallows such lookups, and infers every query from only unseen facts. NaturalLI + Lookup takes the union of the two systems.

state-of-the-art system of MacCartney and Manning (2008). Note that our system is comparatively crippled in this framework along at least two dimensions: It cannot appeal to the antecedent when constructing the search, leading to the introduction of search errors which are entirely absent from prior work. Second the derivation process itself does not have access to the full parse tree of the candidate fact.

Although precision is fairly high even on the non-applicable sections of FraCaS, recall is significantly lower than prior work. This is a direct consequence of not having alignments to appeal to. For instance, we can consider two inferences:

Jack saw Jill is playing $\stackrel{?}{\Rightarrow}$ *Jill is playing*

Jill saw Jack is playing $\stackrel{?}{\Rightarrow}$ *Jill is playing*

It is clear from the parse of the sentence that the first is valid and the second is not; however, from the perspective of the search algorithm both make the same two edits: inserting *Jack* and *saw*.

6.2 Common Sense Reasoning

We validate our system’s ability to infer unseen common sense facts from a large database of such facts. Whereas evaluation of FraCaS shows that our search formulation captures applicable inferences as well as prior work, this evaluation presents a new use-case for Natural Logic which is not trivial given prior work.

For our database of facts, we run the Ollie OpenIE system (Mausam et al., 2012) over Wikipedia, Simple Wikipedia,³ and a small subset of CommonCrawl. Extractions with confidence below 0.25 or which contained pronouns were discarded. This yielded in a total of 305 million

³<http://simple.wikipedia.org/>

unique extractions composed entirely of lexical items which mapped into our vocabulary (186 707 words). Each of these extracted triples (e_1, r, e_2) was then flattened into a plain-text fact $e_1 \ r \ e_2$. In general, each fact in the database could be arbitrary unstructured text; our use of Ollie extractions is motivated by a desire to extract and query concise facts.

For our evaluation, we infer the top 689 most confident facts from the ConceptNet project (Tandon et al., 2011). To avoid redundancy with WordNet, we take all facts from eight ConceptNet relations: MemberOf, HasA, UsedFor, CapableOf, Causes, HasProperty, Desires, and CreatedBy. We then treat the *surface text* field of these facts as our candidate query. This yields facts like the following:

not all birds can fly
noses are used to smell
nobody wants to die
music is used for pleasure

For negative examples, we take the 689 ReVerb extractions (Fader et al., 2011) judged as false by Mechanical Turk workers (?). This provides a set of *plausible* queries, similar in many ways to the database of Ollie extractions, and ensures that our recall is not due to an over-zealous search. The search costs are tuned from a balanced set of true ConceptNet and 540 false ReVerb extractions.

Results are shown in Table 5. We compare against the baseline of looking up each fact verbatim in the fact database. Note that both the query and the facts in the database are lemmatized and lower-cased; therefore, it is not in principle unreasonable to expect a database of 300 million extractions to contain these facts. Nonetheless, only 12% of facts were found via a lookup to the database. We show that NaturalLI (allowing lookups) improves this recall four-fold, at only an 8.5% drop in precision. Furthermore, if we prohibit matching the query fact verbatim, we still recover all but 8% of our recall.

7 Related Work

A large body of work is devoted to compiling open-domain knowledge bases. For instance, OpenIE systems (Yates et al., 2007; Fader et al., 2011; Mausam et al., 2012) extract concise facts via surface or dependency patterns. In a similar vein, NELL (Carlson et al., 2010; Gardner et

al., 2013) continuously learns new high-precision facts from the internet. The MIT Media Lab’s CONCEPTNET project (Liu and Singh, 2004) has been working on creating a large knowledge base emphasizing common sense facts.

A natural alternative to the approach taken in this paper is to extend knowledge bases by inferring and adding new facts directly. For instance, Snow et al. (2006) present an approach to enriching the WORDNET taxonomy; Tandon et al. (2011) extend CONCEPTNET with new facts; Soderland et al. (2010) use REVERB extractions to enrich a domain-specific ontology. Chen et al. (2013) and Socher et al. (2013) use Neural Tensor Networks to predict unseen relation triples in WordNet and Freebase. This work runs inference over arbitrary text, without restricting itself to a particular set of relations, or even entities. Yao et al. (2012) and Riedel et al. (2013) present a related line of work, inferring new relations between Freebase entities by appealing to inferences over both Freebase and OpenIE relations. This work, however, focuses primarily on common-sense reasoning rather than inferring relations between named entities.

The goal of tacking common-sense reasoning is by no means novel in itself. Work by Reiter and McCarthy (Reiter, 1980; McCarthy, 1980) attempt to reason about the truth of a consequent in the absence of strict logical entailment. Similarly, Pearl (1989) presents a framework for assigning confidences to inferences which can be reasonably assumed. Our approach differs from these attempts in part in its use of Natural Logic as the underlying inference engine, and more substantially in its attempt at creating a truly broad-coverage system dealing with millions of candidate antecedents.

Many NLP applications query large knowledge bases. Prominent examples include question answering (Voorhees, 2001), semantic parsing (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2007; Kwiatkowski et al., 2013; Berant and Liang, 2014), information extraction (Hoffmann et al., 2011; Surdeanu et al., 2012), and recognizing textual entailment (Schoenmackers et al., 2010; Berant et al., 2011). A long-term goal of this work is to improve accuracy on these downstream tasks by providing a *probabilistic* knowledge base over both explicitly known and likely true facts.

Fader et al. (2014) propose a system for question answering based on a sequence of paraphrase

rewrites followed by a fuzzy query to a structured knowledge base. This work can be thought of as an elegant framework for unifying this two-stage process, while explicitly tracking the “risk” taken with each paraphrase step.

8 Conclusion

We have presented NaturalLI, an inference system over *unstructured text* intended to infer common sense facts. We have shown that we can run inference over a large set of antecedents while maintaining valid Natural Logic semantics, and have shown that we can learn how to infer unseen common sense facts.

Future work will focus on enriching the class of inferences we can make with Natural Logic – in particular, reasoning with meronymy and relational entailment. Furthermore, in the future we hope to learn with lexicalized parameters, and making use of the syntactic structure of a fact during search.

References

- Gabor Angeli and Christopher Manning. 2013. Philosophers are mortal: Inferring the truth of unseen facts. In *CoNLL*.
- Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. *Proc. Linguistic Annotation Workshop*.
- J. Berant and P. Liang. 2014. Semantic parsing via paraphrasing. In *Association for Computational Linguistics (ACL)*.
- Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *Proceedings of ACL*, Portland, OR.
- Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram version 1. *Linguistic Data Consortium*.
- Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *AAAI*.
- Danqi Chen, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2013. Learning new facts from knowledge bases with neural tensor networks and semantic word vectors. *arXiv preprint arXiv:1301.3618*.
- Robin Cooper, Dick Crouch, Jan Van Eijck, Chris Fox, Johan Van Genabith, Jan Jaspars, Hans Kamp, David Milward, Manfred Pinkal, Massimo Poesio, et al. 1996. Using the framework. Technical report, The FraCaS Consortium.
- Alex J Djalali. 2013. Synthetic logic. *Linguistic Issues in Language Technology*.
- Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *EMNLP*.
- Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *KDD*.
- Matt Gardner, Partha Pratim Talukdar, Bryan Kisiel, and Tom Mitchell. 2013. Improving learning and inference in a large knowledge-base using latent syntactic cues. *EMNLP*.
- Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL-HLT*.
- Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2012. Improving word representations via global context and multiple word prototypes. *ACL*.

- Thomas Icard III and Lawrence Moss. 2014. Recent progress on monotonicity. *Linguistic Issues in Language Technology*.
- Thomas Icard III. 2012. Inclusion and exclusion in natural language. *Studia Logica*.
- Jay J Jiang and David W Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. *Proceedings of the 10th International Conference on Research on Computational Linguistics*.
- Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching.
- Hugo Liu and Push Singh. 2004. ConceptNet: a practical commonsense reasoning toolkit. *BT technology journal*.
- Bill MacCartney and Christopher D Manning. 2007. Natural logic for textual inference. In *ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.
- Bill MacCartney and Christopher D Manning. 2008. Modeling semantic containment and exclusion in natural language inference. In *Coling*.
- Bill MacCartney and Christopher D Manning. 2009. An extended model of natural logic. In *Proceedings of the eighth international conference on computational semantics*.
- Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. 2012. Open language learning for information extraction. In *EMNLP*.
- John McCarthy. 1980. Circumscription—a form of non-monotonic reasoning. *Artificial intelligence*.
- Judea Pearl. 1989. *Probabilistic semantics for non-monotonic reasoning: A survey*. Knowledge Representation and Reasoning.
- Raymond Reiter. 1980. A logic for default reasoning. *Artificial intelligence*.
- Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *NAACL-HLT*.
- Stefan Schoenmackers, Oren Etzioni, Daniel S Weld, and Jesse Davis. 2010. Learning first-order horn clauses from web text. In *EMNLP*.
- Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2006. Semantic taxonomy induction from heterogeneous evidence. In *ACL*.
- Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems*, pages 926–934.
- Stephen Soderland, Brendan Roof, Bo Qin, Shi Xu, Oren Etzioni, et al. 2010. Adapting open information extraction to domain-specific relations. *AI Magazine*.
- Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *EMNLP*.
- Niket Tandon, Gerard de Melo, and Gerhard Weikum. 2011. Deriving a web-scale common sense fact database. In *AAAI*.
- Víctor Manuel Sánchez Valencia. 1991. *Studies on natural logic and categorial grammar*. Ph.D Thesis.
- Johan Van Benthem. 1986. *Essays in logical semantics*.
- Johan FAK van Benthem. 2008. A brief history of natural logic.
- Ellen M Voorhees. 2001. Question answering in TREC. In *Proceedings of the tenth international conference on Information and knowledge management*.
- Limin Yao, Sebastian Riedel, and Andrew McCallum. 2012. Probabilistic databases of universal schema. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*.
- Alexander Yates, Michael Cafarella, Michele Banko, Oren Etzioni, Matthew Broadhead, and Stephen Soderland. 2007. TextRunner: Open information extraction on the web. In *ACL-HLT*.
- John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *AAAI/IAAI*, Portland, OR.
- Luke S. Zettlemoyer and Michael Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *EMNLP-CoNLL*.