NaturaLI: Natural Logic Inference for Common Sense Reasoning

Abstract

Common-sense reasoning is important for a range of AI applications, both in NLP but also in semantically motivated vision or robotics tasks. We propose NaturaLI: a system for inferring factoid style common sense facts – for instance, *cats have tails* or *tomatoes are fragile* – from a large database of candidate justifications. The system provides logically valid derivations, while also being flexible to backing off to derivations which are only likely valid, with 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.

1 Introduction

Common-sense reasoning if prevalent in a number of AI tasks; although such reasoning is difficult in general, we hope to capture a subset of useful phenomena. For instance, in computer vision it may be useful to provide priors for cars generally being found on roads, or computer mice being next to monitors. Similarly, for robotics (e.g., for illustration, a robot to help in the kitchen) it may be useful to know that milk is found in a refrigerator, or that tomatoes are soft and easy to crush.

As a step in this direction, we created NaturaLI: a reasoning engine based around Natural Logic to infer the truth of arbitrary query facts given a large database of known facts. The system is intended to follow valid logical derivations when possible, but also back off gracefully to dubious inferences with an associated confidence of validity.

Prior work has largely focused on freebase-style factoids: for instance, *Barack Obama is married to Michelle*, or *Natasha is the daughter of Michelle Obama*. In this framework, inference tasks are generally cast as expanding a partial knowledge base with additional high-probability facts (e.g., Knowledge Base Population). While this approach is viable for these sorts of factoid entries, the number of common-sense facts – the vast majority of which are never explicitly mentioned – is unlikely to be amenable to this strategy.

We therefore re-cast the problem as querying whether an arbitrary fact is true or not, given the entire knowledge base as the antecedent set. We then approach the problem as finding a correct reverse derivation from the consequent to any antecedent in our knowledge base.

When possible, these derivations should be valid Natural Logic derivations; however, the search is robust to proposing plausible, if not strictly correct, inferences at a discount proportional to the likelihood of validity. For instance, our approach generalizes similarity metrics, suggesting that if two entities are similar to each other (e.g., cats and dogs) then they are likely to share properties (e.g., have tails). This discount on incorrect inferences can be set to enforce strict validity, but can as easily be trained on a dataset of facts labeled with their truth, to calibrate the resulting probability returned from the system.

We evaluate the system on the FraCaS entailment suite to motivate its validity as an inference engine, and evaluate in a more realistic setting of predicting the truth of ReVerb extractions (Fader et al., 2011) annotated with ground truth correctness.

2 Natural Logic

Much of the underpinnings of this work rely on the theoretical framework laid by Natural Logic (Van Benthem, 1986; Valencia, 1991). Broadly, Natural Logic aims to capture a subset of valid logical inferences by appealing directly to the structure of language, as opposed to running deduction on an abstract logical form. In part, this lends itself to computationally efficient inference; in another part, it frees the system from parsing to an abstract logic – this is particularly relevant in our case, where the set of potential antecedents is very large.

A working approximation for Natural Logic is as a principled formalism for and extension of syllogistic reasoning. In this analogy, the minor premise (e.g., *All Greeks are men*) is encoded implicitly, and an inference is made from the major premise to the consequent: (e.g., *All men are mortal* $\Rightarrow$ *All Greeks are mortal*). This style of reasoning is warranted from an analysis of the monotonicity of quantifiers, reviewed in Section 2.1.

The additional expressive power of Natural Logic is derived from two sources: in part, the logic does not rely on template matching, allowing inferences to be made over patterns nested within a larger context, and allowing inferences to be chained. In another part, we can make use of MacCartney's expanded entailment relations, described in Section 2.2. We conclude the section with a novel result showing that the state space of the MacCartney-style derivations can be collapsed to a simpler representation (Section 2.3), and by elaborating on the limitations of Natural Logic (Section 2.4).

Icard III and Moss (2014) offers a more thorough summary of the field, and provides the source for much of the notation and exposition presented here.

2.1 Monotonicity in Natural Logic

Monotonicity in Natural Logic may be viewed analogously to monotonicity in calculus. More precisely, a function can be one of *monotone*, *antitone*, or *non-monotone* in each of its arguments.[1] Intuitively, the function $f(x,y,z) = x - y + (z-2)^2$ is monotone in x, antitone in y, and non-monotone in z. At its heart, Natural Logic makes use of this monotonicity information and the partial ordering over the domain and range of the function to draw inferences over the formula for f without evaluating any part of the function explicitly. For instance, we can safely say that $f(1,1,1) \leq f(2,1,1)$, whereas $f(1,1,1) \geq f(1,2,1)$ by appealing directly to surface form of the expression (The substitution of '2' in place of '1').

[1] *Monotone* and *antitone* are often referenced as *upwards monotone* and *downwards monotone*.

Formally, we define a function $f : \mathcal{D}_1 \rightarrow \mathcal{D}_2$ and a partial ordering $\leq_1$ over $\mathcal{D}_1$ and $\leq_2$ over $\mathcal{D}_2$. If we set $\mathcal{D}_1 = \mathcal{D}_2 = \mathcal{R}$, and take $\leq_1 = \leq_2 = \leq$ to be the conventional "less than or equal to" over real numbers, we can derive the monotonicity of the functions in our example above.

However, we can equally well define a partial ordering over sets: $x \leq_e y \Leftrightarrow x \cap y = x$. Similarly, we can define a partial ordering over functions $f : \mathcal{D}_1 \rightarrow \mathcal{D}_2$: $f \leq_1 g \Leftrightarrow \forall x \;\; f(x) \leq_2 g(x)$ and truth values: $x \leq_t y \Leftrightarrow x = F \vee y = T$. This allows us to adapt our definition of monotonicity to denotational semantics in natural language.

Let the denotation of a phrase, marked $[\![\cdot]\!]$, be the set of entities in the universe to which that phrase refers. Equivalently, we may consider each phrase as a predicate, and the denotation of a phrase is the set of entities for which the predicate holds. We define the set e to be the set of entities in our discourse space, with the partial ordering over sets: $\leq_e$. We define the set t to be the set of truth values: $\{T, F\}$. It should not go unnoticed that Modus Ponens ($x \Rightarrow y$) and $x \leq_t y$ are equivalent. Note that functions $e \rightarrow t$ follow the partial ordering over functions.

We define the denotation of quantifiers in the conventional fashion as functions $e \rightarrow (e \rightarrow t)$. However, we can further mark the arguments to these quantifiers as potentially monotone or antitone. For instance, the quantifier *all* is antitone in its first argument and monotone in its second: $e \xrightarrow{-} (e \xrightarrow{+} t)$.[2] These markings confirm our intuition that, e.g., *all **cats** have tails* implies *all **tabby cats** have tails*. Furthermore, as the default context is taken to be monotone axiomatically, we see that we can appeal to the partial order over quantifiers (functions) to conclude that ***all** cats have tails* implies ***some** cats have tails*.

Proof Theory We can produce proofs in Natural Logic by composing function application from atomic facts. Returning to the arithmetic cast in the beginning of the section, we can construct a proof that $2^{-4} < 2^{-3}$ via:

$$\dfrac{\dfrac{3 \leq 4}{-4 \leq -3}\ {\scriptstyle -x \text{ is antitone}}}{2^{-4} \leq 2^{-3}}\ {\scriptstyle 2^x \text{ is monotone}}$$

[2] These markings can, in fact, be elegantly incorporated into the type system (Icard III and Moss, 2014) of a CCG.

We can then appeal implicitly to this proof to ask whether what relation a substitution of 3 for 4 will produce in 2^{-4} – in this case one of $\leq$, $\geq$, or unknown – without evaluating either expression.

Returning to natural language, we can provide a proof for *all tabby cats have tails*, given in Figure 1. That is to say, on the given parse of the respective sentences, $[\![$All cats have tails$]\!] \leq_t [\![$All tabby cats have tails$]\!]$. We again note that $\leq_t$ is precisely Modus Ponens ($\Rightarrow$) – thus, we have arrived at our expected entailment relation.

2.2 Reasoning with Exclusion

The Natural Logic of Section 2.1 has dealt only with the entailment relation. Work by MacCartney and Manning (2009), building off of MacCartney and Manning (2007) and MacCartney and Manning (2008), has focused on extending the set of atomic relations beyond $\leq$ (subset; entailment) and $\geq$ (superset; reverse entailment).

Icard III and Moss (2014) noted that the partial ordering of $\leq_e$ can be formulated as a *bounded distributive lattice* to encompass the original relations of MacCartney and Manning (2007). In particular, for every domain we can define a set of possible elements $\mathcal{X}$, binary operators $\wedge$ and $\vee$, and a minimum and maximum element $\bot$ and $\top$ respectively. By example, for the domain of entities we can define $\mathcal{X}$ to be the power set of our domain of entities, $\wedge = \cap$, $\vee = \cup$, $\bot = \{\}$, and $\top = E$, where E is the universe of possible entities. We then define the set of MacCartney relations, noting that $\sqsubseteq$ and $\sqsupseteq$ are identical to the definitions of $\leq_e$ and $\geq_e$ from the previous section respectively:[3]

$$\begin{array}{lcl}
x \sqsubseteq y & \Leftrightarrow & x \wedge y = x \\
x \sqsupseteq y & \Leftrightarrow & x \vee y = x \\
x \downharpoonleft\upharpoonright y & \Leftrightarrow & x \wedge y = \bot \\
x \smile y & \Leftrightarrow & x \vee y = \top \\
 & & \\
x \equiv y & \Leftrightarrow & x \sqsubseteq y \text{ and } x \sqsupseteq y \\
x \curlywedge y & \Leftrightarrow & x \downharpoonleft\upharpoonright y \text{ and } x \smile y \\
x \# y & & \text{Always true}
\end{array}$$

Making use of this extended set of relations, we are faced with two questions: we must define how monotonicity projects these relations up the proof tree, and we must define how a sequence of edits compose to determine the relation between the first and last sentence in the sequence.

Relation	**Projection in Context**	
	Monotone	Antitone
$e_1 \sqsubseteq e_2$	$t_1 \sqsubseteq t_2$	$t_1 \sqsupseteq t_2$
$e_1 \sqsupseteq e_2$	$t_1 \sqsupseteq t_2$	$t_1 \sqsubseteq t_2$
$e_1 \downharpoonleft\upharpoonright e_2$	$t_1 \downharpoonleft\upharpoonright t_2$	$t_1 \smile t_2$
$e_1 \smile e_2$	$t_1 \smile t_2$	$t_1 \downharpoonleft\upharpoonright t_2$
$e_1 \equiv e_2$	$t_1 \equiv t_2$	$t_1 \equiv t_2$
$e_1 \curlywedge e_2$	$t_1 \curlywedge t_2$	$t_1 \curlywedge t_2$
$e_1 \# e_2$	$t_1 \# t_2$	$t_1 \# t_2$

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 truth value t_1 and the new truth value t_2. Values are given for when the entities are in a monotone and antitone context.

$\bowtie$	$\sqsubseteq$	$\sqsupseteq$	$\curlywedge$	$\downharpoonleft\upharpoonright$	$\smile$
$\sqsubseteq$	$\sqsubseteq$	$\#$	$\downharpoonleft\upharpoonright$	$\downharpoonleft\upharpoonright$	$\#$
$\sqsupseteq$	$\#$	$\sqsupseteq$	$\smile$	$\#$	$\smile$
$\curlywedge$	$\smile$	$\downharpoonleft\upharpoonright$	$\equiv$	$\sqsupseteq$	$\sqsubseteq$
$\downharpoonleft\upharpoonright$	$\#$	$\downharpoonleft\upharpoonright$	$\sqsubseteq$	$\#$	$\sqsubseteq$
$\smile$	$\smile$	$\#$	$\sqsupseteq$	$\sqsupseteq$	$\#$

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.

Table 1 outlines the (simplified) table used for projecting relations between entities in monotone and antitone contexts. Note that in monotone contexts, the relation projects up without change; in antitone contexts, forward and reverse entailment ($\sqsubseteq$ and $\sqsupseteq$) are flipped, as are cover and alternation ($\smile$ and $\downharpoonleft\upharpoonright$). In full generality, quantifiers are tagged not only with monotonicity but also *additivity* and *multiplicativity*; for simplicity we assume all quantifiers are both additive and multiplicative. See Lemma 2.4 in Icard III (2012) for a more careful analysis.

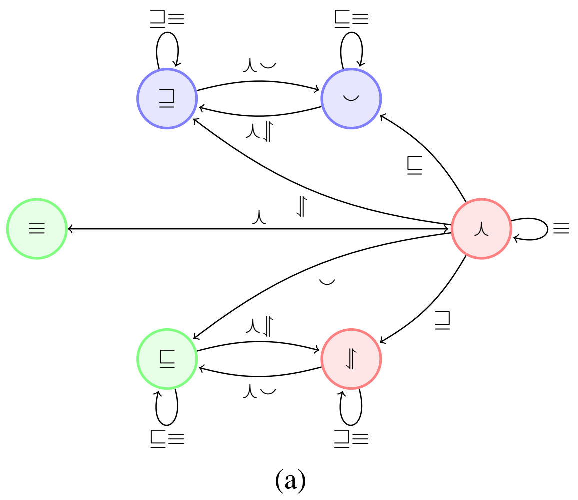
The composition of relations through an inference chain is given in a *join table*, copied in Table 2.

Figure 2a visualizes the join table as a finite state automata. The edges in the automata denote a step in a logical derivation; for instance, attempting to derive *all cats have tails* from *all felines have tails*. The label of the edge denotes the relation between the mutated **entity** in the derivation. In our example, *felines* $\sqsubseteq_e$ *cats*, and therefore we would take the edge labeled with $\sqsubseteq$.

[3]We adopt the semantics of Icard III (2012), in which the relations are not mutually exclusive.

$$\dfrac{\dfrac{\dfrac{[\![\text{tabby cats}]\!] \;\le_e\; [\![\text{cats}]\!]}{[\![\text{all}]\!]\,([\![\text{cats}]\!])\,(\cdot) \;\le_{e\to t}\; [\![\text{all}]\!]\,([\![\text{tabby cats}]\!])\,(\cdot)}\;{\scriptstyle [\![\text{All}]\!]\text{ antitone in subject}} \qquad [\![\text{have tails}]\!] \;\le_e\; [\![\text{have tails}]\!]}{[\![\text{all}]\!]\,([\![\text{cats}]\!])\,([\![\text{have tails}]\!]) \;\le_t\; [\![\text{all}]\!]\,([\![\text{tabby cats}]\!])\,([\![\text{have tails}]\!])}}{[\![\text{all cats have tails}]\!] \;\le_t\; [\![\text{all tabby cats have tails}]\!]}\;{\scriptstyle \text{By axiom}}$$

Figure 1: A proof in Natural Logic for *all tabby cats have tails* from the premise *all cats have tails*. Note that $\le_t$ is equivalent to entailment ($\Rightarrow$). The last step, marked valid axiomatically, is elaborated on in Definition 8 of Icard III and Moss (2014).



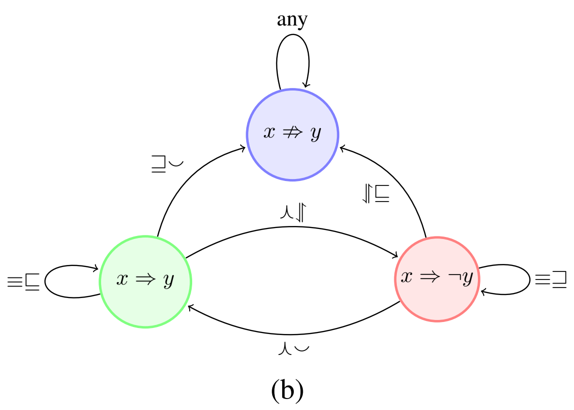


Figure 2: (a) The join table in Table 2 expressed as a finite state automata. Omitted edges go to the unknown state ($\#$), with the exception of omitted edges from $\equiv$, which go to the state of the edge type. Green states ($\equiv$, $\sqsubseteq$) denote valid inferences; red states ($⥮$, $\curlywedge$) denote invalid inferences; blue states ($\sqsupseteq$, $\smile$) denote inferences of unknown validity. (b) The join table collapsed into the three meaningful states over truth values.

States in the automata denote the relation between the **truth value** of the antecedent, and the truth value of the consequent given the mutations so far. We may consider as a motivating example the inference from *all felines have tails* to *no cats have tails*:

Mutation	**Fact**	**Edge**	**State**
	all felines have tails		$\equiv$
feline $\sqsupseteq$ *cat*	*all cats have tails*	$\sqsubseteq$	$\sqsubseteq$
all ⅄ *no*	*no cats have tails*	⅄	⥯

The second column shows the surface form of the fact as the derivation progresses. The first column shows the mutation applied to the previous fact to get the new surface form. The third column shows the edge to take in the FSA – this is derived from the projections in Table 1. For example, since *all* is antitone in its first argument (*felines*), a mutation of type $\sqsupseteq$ projects to an edge type of $\sqsubseteq$. The first three columns can be viewed as shorthand for a formal proof tree like that provided in Figure 1. The last column shows the relationship between the first fact and the fact on this row of the derivation. In our example derivation, we conclude that:

all felines have tails ⥯ *no cats have tails.*

2.3 Collapsed Join Table

Although the full join table is useful when we would like fine-grained semantic comparisons, for our application we care only whether an inference is *valid*, *invalid*, and *unknown*. To illustrate, we do not care that *all felines have tails* is in any particular set theoretic relation with *no cats have tails*; we only care that this is a verifiably invalid inference.

To this end, we make two novel contributions to the theory of Natural Logic: first, we show that the join table can be simplified in the case where we only care about the validity of an inference; and, second, we propose weighting the edges in this finite state automata in such a way as to collapse the *unknown* state into one of either *valid* or *invalid*, with low probability.

The first of these contributions, illustrated in Figure 2b, makes use of two observations. First, if we cluster the seven MacCartney relations[4], the transitions into and out of each class is identical independent of which specific state one is in. This clustering – trivial intuitively – is also the clustering used by MacCartney and Manning (2008) for evaluating inference tasks.

Furthermore, although there are transitions between the states clustered into the unknown category, at no point is there a valid transition from a node in the cluster to a node outside of it. This is a generalization of the well-known observation that long derivations tend to the unknown state (#), from which it is impossible to escape. Here, we observe that even if a derivation enters a seemingly-informative state ($\sqsupseteq$, $\smallsmile$) it is nonetheless doomed to produce a final judgment of unknown.

The second observation – that we can run probabilistic inference and collapse the unknown state into a penalty on the probability of validity/invalidity – is discussed in the next section.

2.4 Limitations and Future Work

Natural Logic is not intended to be a complete solution for natural language inference. Although the formalism covers a reasonable range of intuitive inference patterns, the current theory is unable to capture the full range of logical inferences. For example, DeMorgan's laws cannot be inferred in the framework, including their extensions to quantification (e.g., *not all students study* $\models$ *some students don't study*). Furthermore, Natural Logic does not elegantly handle inferences making use of multiple antecedents; for instance, the law of excluded middle is not derivable according to current theory.

MacCartney and Manning (2008) point out that Natural Logic is not amenable to reasoning over relational implication (*Eve was let go* $\models$ *Eve lost her job*; *Aho, a trader at UBS* $\models$ *Aho works for UBS*; etc.). We contest that these are special cases of monotone reasoning, where relations are treated as functions and ordered in the same way as quantifiers. That is, if a relation r_1 entails another relation r_2 then $r_1 \leq_r r_2$, where $\leq_r$ is an ordering of functions from a list of entities to truth values: $\mathbf{e}^n \rightarrow \{T, F\}$. In fact, under certain conditions Natural Logic can warrant inference over meronymy (*Obama was born in Hawaii* $\models$ *Obama was born in the US*), and potentially other orderings as well. We leave this for future work.

[4]The seventh relation (#) is not shown in the FSA for clarity, but clusters into the unknown class.

3 Inference As Search

In a classical inference setting, a system is provided both the query consequent, and a set of antecedents from which to derive the validity of the consequent. Prior work on inference with Natural Logic has assumed this scenario (MacCartney and Manning, 2008), and taken the two-step of approach of aligning the antecedent and consequent, and classifying each aligned segment into a mutation relation. For common-sense reasoning, however, we are not given a well-defined antecedent, but rather have at our disposal a large database of true facts from which to derive the query. This makes the align-and-classify approach of previous work substantially less appealing, as the antecedents are not readily available and would need to be retrieved from the database.

We therefore approach the problem as a search task: given the query consequent, we search over the space of possible facts for a valid antecedent in our database. The nodes in our search problem correspond to candidate facts; the edges are candidate mutations; the costs over these edges encode the confidence (or likelihood) that the mutation over this edge maintains the truth of the antecedent.

We define the problem by specifying the state space, the valid transitions, along with the weights of the transitions. We then describe how these weights translate to the confidence of truth of a fact, to be used in learning, and a generalization of JC similarity (Jiang and Conrath, 1997).

3.1 States

A state in the graph is defined as a partial path from the consequent to a valid antecedent. That is, a state is primarily parametrized primarily by the surface form of the candidate fact at this point in the (reverse) derivation.

This surface form is augmented with both sense and polarity information. Each word is allowed a sense (up to 31 possible senses), or a default sense; in addition, 2 bits are reserved for marking polarity (monotone, antitone, non-monotone). This leaves sufficient space to store each augmented word in a 32 bit integer. Note also that we refer to potentially arbitrary lexical items in WordNet as words – for instance, *fire truck* is treated as a single word.

A number of important details deserve attention in our definition of a search state:

Lexical mutations As per our definition, transitions between states are transitions between facts. However, the lexical resources for the mutations are over words (e.g., *feline* $\rightarrow$ *cat*) rather than entire facts. This motivates the inclusion of an index in our state, denoting the index of the word which may be mutated.

Importantly, this also imposes a natural order in the construction of potential antecedents during search. The mutation index is allowed to shift forwards through the fact (at no cost), but not backwards. This helps make search more tractable, at the expense of rare esoteric search errors.

Predicting deletions Although inserting words in a derivation (deleting words from the reversed derivation) is trivial, the other direction is not. naïvely, at every state in our search, we must consider every word in the vocabulary as a possible antecedent from which the word was deleted to produce the current state.

This is largely handled by storing the database of known facts as a Trie. Since we are constructing antecedents left-to-right, as per above, we can propose candidate deletable words from the Trie, looking up the partial derivation up to the mutation index. As a special case, when proposing words for the beginning of the fact, we take all facts whose second word matches the first word of the search state.

Tracking inference state The cost of an inference at any given state depends not only on the polarity of the lexical item, but also the state of the derivation so far. We therefore augment a fact with whether the derivation is in a *valid* or *invalid* state

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

3.2 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 – just as a theorem prover would not store all possible derivation steps. 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*.[5] We mitigate this imprecision by introducing a cost for each transition, and learning the appropriate value for this cost (see Section 4).

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 $\uparrow_{w \to w'}$ as transitioning from word w to its hypernym w' (and visa versa for $\downarrow_{w \to w'}$), and set:

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

$$f_{\uparrow_{w \to w'}} = \log \frac{p(w')}{p(w)} \quad = \log p(w') - \log p(w)$$
$$f_{\downarrow_{w \to w'}} = \log \frac{p(w)}{p(w')} \quad = \log p(w) - \log p(w')$$

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 $f_{NN_{w \to w'}}$ to be the arc cosine of the cosine similarity between word vectors associated with lexical items w and w':

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

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

3.3 Generalizing Similarities

An elegant property of our definitions of f_{t_i} is its ability to generalize JC similarity, 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 $\text{dist}_{\text{jc}}(w_1, w_2)$ is:

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

For simplicity, we simplify $\theta_{\uparrow,v,p}$ and $\theta_{\uparrow,v,p}$ as simply $\theta_\uparrow$ and $\theta_\downarrow$. The derivation generalizes trivially to the 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, $\boldsymbol{\theta}^T\mathbf{f}$, along the path from w_1 to w_2 through its lowest common

Template	**Relation**	**Example edge**
WordNet hypernym	$\sqsubseteq$	*some cats like milk* $\sqsubseteq$ *some felines like milk*
WordNet hyponym	$\sqsupseteq$	*some cats like milk* $\sqsupseteq$ *some felines like milk*
WordNet antonym†	$\downharpoonleft\upharpoonright$	*all cats like milk* $\downharpoonleft\upharpoonright$ *all cats hate milk*
WordNet synonym/pertainym†	$\equiv$	*some cats like milk* $\equiv$ *some cats enjoy milk*
Distributional nearest neighbor	$\equiv$	*some cats like milk* $\equiv$ *some cats like dairy*
Delete word†	$\sqsubseteq$	*some tabby cats like milk* $\sqsubseteq$ *some cats like milk*
Add word†	$\sqsupseteq$	*some cats like milk* $\sqsupseteq$ *some tabby cats like milk*
Quantifier weaken	$\sqsubseteq$	*all cats like milk* $\sqsubseteq$ *some cats like milk*
Quantifier strengthen	$\sqsupseteq$	*some cats like milk* $\sqsupseteq$ *all felines like milk*
Quantifier negate	$\curlywedge$	*some cats like milk* $\curlywedge$ *no felines like milk*
Quantifier synonym	$\equiv$	*some cats like milk* $\equiv$ *a few cats like milk*
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.

subsumer (lcs), $[w_1, w_1^{(1)}, \ldots, \text{lcs}, \ldots, w_2^{(1)}, w_2]$, as follows:

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

Note that setting both $\theta_{\uparrow}$ and $\theta_{\downarrow}$ 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.

3.4 Confidence Estimation

The last component in inference is translating a search path into a probability of truth. We notice from Section 3.2 that the *cost* of a path can be represented as $\boldsymbol{\theta}^T\mathbf{f}$. We can normalize this value by negating every element of the weight vector and passing it through a sigmoid:

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

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 visa versa. For completeness, if no path is given we can set $v = 0$. The probability of validity then becomes:

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

Note that in the case where $v = -1$, the above expression reduces to $\frac{1}{2}$ − 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.

4 Learning Transition Weights

The learning task can be viewed as a constrained optimization problem. Subject to the constraint that all elements of the cost vector $\boldsymbol{\theta}$ must be nonnegative, 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 the derivation search over every example, providing a predicted probability of truth for each query fact. Optimizing the likelihood of the training data according to Equation (2) is impractical as the objective is nonconvex[6], therefore we opt to train a simple logistic regression model as a proxy.

In particular, we construct a dataset of (x, y) pairs, where x is the featurized path for a particular example, and y is whether the path ended in a correct state. A correct state is defined trivially between *valid* and *invalid*. In cases where the gold annotation contains a state of *unknown validity*, any prediction is marked incorrect.

The resulting weights – in one-to-one correspondence with the costs for our search – are then negated and normalized via a bilinear transform to fall between 0 and -5, with a mean of -1. A default weight of -2 is assigned for any feature which did not appear in the training data, excepting the case where a previous iteration of the algorithm had set a value for that weight, in which case the previous value is used.

5 Experiments

We evaluate our system on two tasks: the FraCaS test suite used by MacCartney and Manning (2007) and MacCartney and Manning (2008), as well as for filtering valid and invalid extractions from the ReVerb OpenIE system (Fader et al., 2011), as per Angeli and Manning (2013).

[6]Though maybe it's worthwhile to try anyways?

§	Category	Count	P		R		A		
			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).

5.1 FraCaS Entailment Corpus

The FraCaS textual entailment 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.

Following MacCartney and Manning (2007), we take only a subset of the corpus as applicable for Natural Logic. 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.

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 are generally promising. 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 as a new class of error. Second the derivation process itself does not have access to the full parse tree of the candidate fact. Therefore, it is unable

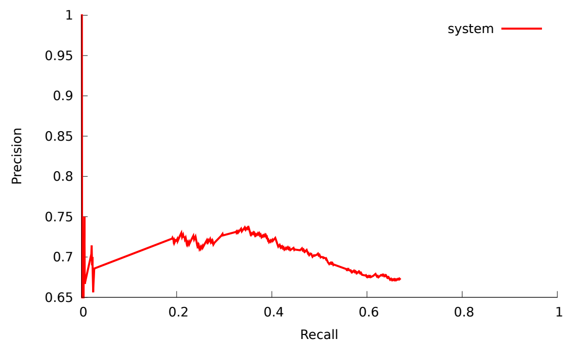


Figure 3: A PR curve showing the accuracy of extraction, given the proportion of examples we provide an estimate for. Imagine the curve keeps going up left of 0.4 – hopefully that's what'll happen by the deadline.

to make large edits over constituents, and is unable to appeal to the parse in its mutation decisions. For instance, our system has no notion of prepositional attachment.

Despite this, the system performs comparable to NatLog on the sections it intends to capture. Furthermore, precision is consistently high among all sections, although recall is usually significantly lower. It should also be noted that MacCartney and Manning (2008) intend to capture the inferences in sections 2 (plurals) and 9 (attitudes), which we omit from our system.

5.2 ReVerb Filtering

To evaluate our system in a more real-world setting, we adopt the task of predicting the truth of ReVerb extractions as introduced in Angeli and Manning (2013). Equivalently, this can be viewed as filtering out semantically implausible extractions.

For this data, 4000 randomly selected ReVerb extractions were shown to Mechanical Turk workers, who ranked each extraction as either *correct*, *plausible*, or *implausible*. Extractions which obtained more *correct* votes than *implausible* votes were considered true, and visa versa. Ties were discarded from the corpus. The examples are broken up into 1796 training examples (70% true) and 1975 test examples (65% true).

Results are given in Figure 3. They're not very good yet. Hopefully they'll get better.

6 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. 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 such as 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.

7 Conclusion

TODO: write me!

References

Gabor Angeli and Christopher Manning. 2013. Philosophers are mortal: Inferring the truth of unseen facts. In *CoNLL*.

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*.

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, Technical Report LRE 62-051 D-16, The FraCaS Consortium.

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 heterogenous evidence. In *ACL*.

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*.

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*.