NaturalLI: Natural Logic Inference for Common Sense Reasoning

Gabor Angeli
Stanford University
Stanford, CA 94305
angeli@cs.stanford.edu

Christopher D. Manning
Stanford University
Stanford, CA 94305
manning@cs.stanford.edu

Abstract

Common-sense reasoning is important for AI applications, both in NLP and many vision and robotics tasks. We propose NaturalLI: a Natural Logic inference system for inferring common sense facts – for instance, that *cats have tails* or *tomatoes are round* – from a very large database of known facts. In addition to being able to provide strictly valid derivations, the system is also able to produce derivations which are only *likely* valid, accompanied by an associated confidence. We both show that our system is able to capture strict Natural Logic inferences on the FraCaS test suite, and demonstrate its ability to predict common sense facts with 49% recall and 91% precision.

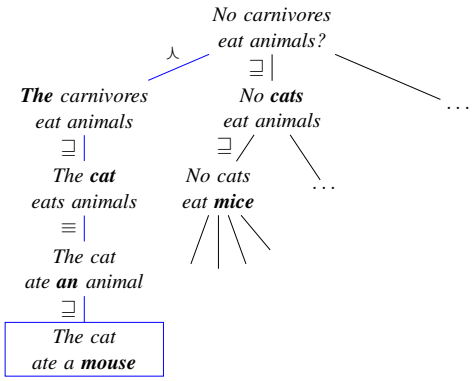


Figure 1: Natural Logic inference cast as search. The path to the boxed premise *the cat ate a mouse* disproves the query *no carnivores eat animals*, as it passes through the negation relation (人). This path is one of many candidates taken; the premise is one of many known facts in the database. The edge labels denote Natural Logic inference steps.

1 Introduction

We approach the task of *database completion*: given a database of true facts, we would like to predict whether an unseen fact is true and should belong in the database. This is intuitively cast as an inference problem from a collection of candidate premises to the truth of the query. For example, we would like 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 in a principled way while maintaining high recall, particularly for large scale open-domain tasks. Learned inference rules are difficult to generalize to arbitrary 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 in providing broad-coverage meaning representations, it remains appealing to use human language as the vessel for inference. Furthermore, OpenIE and similar projects have been very successful at collecting databases 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 – for broad coverage database completion.

Prior work on Natural Logic has focused on inferences from a single relevant premise, making use of only formally valid inferences. We improve upon computational Natural Logic in three ways: (i) our approach operates over a very large set of candidate premises simultaneously; (ii) we do not require explicit alignment between a premise and the query; and (iii) we allow imprecise inferences at an associated cost learned from data.

Our approach casts inference as a single unified search problem from a query to any valid

supporting premise. Each transition along the search denotes a (reverse) inference step in Natural Logic, and incurs a cost reflecting the system's confidence in the validity of that step. This approach offers two 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 for inference, rather than translating to a formal logic syntax. Moreover, the entire pipeline is implemented in a single elegant search framework, which scales easily to large databases.

2 MacCartney's Natural Logic

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. The logic builds upon traditional rather than first-order logic: to a first approximation, Natural Logic can be seen as an enhanced version of Aristotle's syllogistic system (van Benthem, 2008). A working understanding of the logic as syllogistic reasoning is sufficient for understanding the later contributions of the paper. While some inferences of first-order logic are not captured by Natural Logic, it nonetheless allows for a wide range of intuitive inferences in a computationally efficient and conceptually clean way.

We build upon the variant of the logic introduced by the NatLog system (MacCartney and Manning, 2007; 2008; 2009), based on earlier theoretical work on Natural Logic and Monotonicity Calculus (van Benthem, 1986; Valencia, 1991). Later work formalizes many aspects of the logic (Icard, 2012; Djalali, 2013); we adopt the formal semantics of Icard and Moss (2014), along with much of their notation.

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

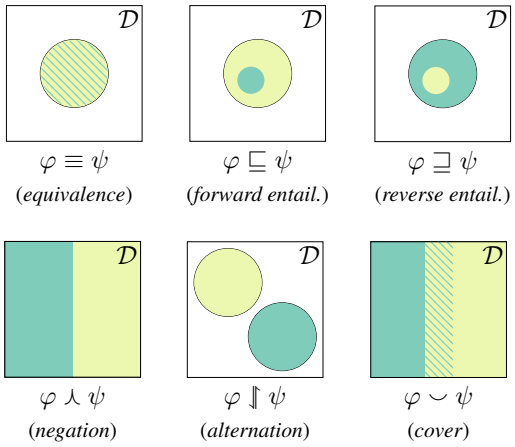


Figure 2: The model-theoretic interpretation of the MacCartney relations. The figure shows the relation between the denotation of φ (dark) and ψ (light). The universe is denoted by $\mathcal{D}$.

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 item is the set of objects in the domain of discourse $\mathcal{D}$ to which that lexical item refers. For instance, the denotation of *cat* would be the set of all cats. Two denotations 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 to the completely uninformative relation. For instance, the example search path in Figure 1 makes use of the following relations:

$$\begin{array}{rcl} \textit{No } x\ y & \curlywedge & \textit{The } x\ y \\ \textit{cat} & \sqsubseteq & \textit{carnivore} \\ \textit{animal} & \equiv & \textit{a animal} \\ \textit{animal} & \sqsupseteq & \textit{mouse} \end{array}$$

Denotations are not required to be in the space of predicates (e.g., *cat*, *animal*). In the first example, the denotations of *No* and *The* are in the space of operators $p \to (p \to t)$: functions from predicates p to truth values t. The $\curlywedge$ relation becomes the conjunction of two claims: $\forall x \forall y\ \neg(\textit{no } x\ y \wedge \textit{the } x\ y)$ and $\forall x \forall y\ (\textit{no } x\ y \vee \textit{the } x\ y)$. This is analogous to the construction of the set-theoretic definition of $\curlywedge$ in Figure 2: $\varphi \cap \psi = \varnothing$ and $\varphi \cup \psi = \mathcal{D}$ (see Icard and Moss (2014)).

Examples of the last two relations ($\Downarrow$ and $\smile$) and the complete independence relation ($\#$) include:

$$\begin{array}{rcl} \textit{cat} & \Downarrow & \textit{dog} \\ \textit{animal} & \smile & \textit{nonhuman} \\ \textit{cat} & \# & \textit{friendly} \end{array}$$

2.2 Monotonicity and Polarity

The previous section details the relation between lexical items; however, we still need a theory for how to "project" the relation induced by a lexical mutation as a relation between the two containing sentences. For example, *cat* $\sqsubseteq$ *animal*, and *some cat meows* $\sqsubseteq$ *some animal meows*, but *no cat barks* $\not\sqsubseteq$ *no animal barks*. Despite differing by the same lexical relation, the first example describes a valid entailment, while the second does not.

We appeal to two important concepts: *monotonicity* as a property of arguments to natural language operators, and *polarity* as a property of lexical items in a sentence. Much like monotone functions in calculus, an [upwards] monotone operator has an output truth value which is non-decreasing (i.e., material implication) as the input "increases" (i.e., the subset relation). 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 operators acting on it. All lexical items have *upward* polarity by default; upwards monotone operators preserve polarity, and downwards monotone operators reverse polarity. For example, *mice* in ***no*** *cats eat mice* has downward polarity, whereas *mice* in ***no*** *cats do**n't** eat mice* has upward polarity (it is in the scope of two downward monotone operators). The relation between two sentences differing by a single lexical relation is then given by the projection function ρ in 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 first generates an alignment between the premise and the query, and then classifies each aligned segment into one of the lexical relations in Figure 2. Inference reduces to projecting each

r	$\equiv$	$\sqsubseteq$	$\sqsupseteq$	$\Downarrow$	$\smile$	$\frown$	$\#$
$\rho(r)$	$\equiv$	$\sqsupseteq$	$\sqsubseteq$	$\smile$	$\Downarrow$	$\frown$	$\#$

Table 1: The projection function ρ, shown for downward polarity contexts only. The input r is the lexical relation between two words in a sentence; the projected relation $\rho(r)$ is the relation between the two sentences differing only by that word. Note that ρ is the identity function in upward polarity contexts.

$\bowtie$	$\equiv$	$\sqsubseteq$	$\sqsupseteq$	$\frown$	$\Uparrow$	$\smile$	$\#$
$\equiv$	$\equiv$	$\sqsubseteq$	$\sqsupseteq$	$\frown$	$\Uparrow$	$\smile$	$\#$
$\sqsubseteq$	$\sqsubseteq$	$\sqsubseteq$	$\#$	$\Uparrow$	$\Uparrow$	$\#$	$\#$
$\sqsupseteq$	$\sqsupseteq$	$\#$	$\sqsupseteq$	$\smile$	$\#$	$\smile$	$\#$
$\frown$	$\frown$	$\smile$	$\Uparrow$	$\equiv$	$\sqsupseteq$	$\sqsubseteq$	$\#$
$\Uparrow$	$\Uparrow$	$\#$	$\Uparrow$	$\sqsubseteq$	$\#$	$\sqsubseteq$	$\#$
$\smile$	$\smile$	$\smile$	$\#$	$\sqsupseteq$	$\sqsupseteq$	$\#$	$\#$
$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$	$\#$

Table 2: The join table as shown in Icard (2012). Entries in the table are the result of joining a row with a column.

of these relations according to the projection function ρ (Table 1) and iteratively *joining* two projected relations together to get the final entailment relation. This join relation, denoted as $\bowtie$, is given in Table 2.

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: $r_1 \coloneqq \textit{cat} \rightarrow \textit{dog}$, $r_2 \coloneqq \cdot \rightarrow \textit{not}$, and $r_3 \coloneqq \textit{dog} \rightarrow \textit{poodle}$. Each of these are then projected with the projection function ρ, and are joined using the join relation:

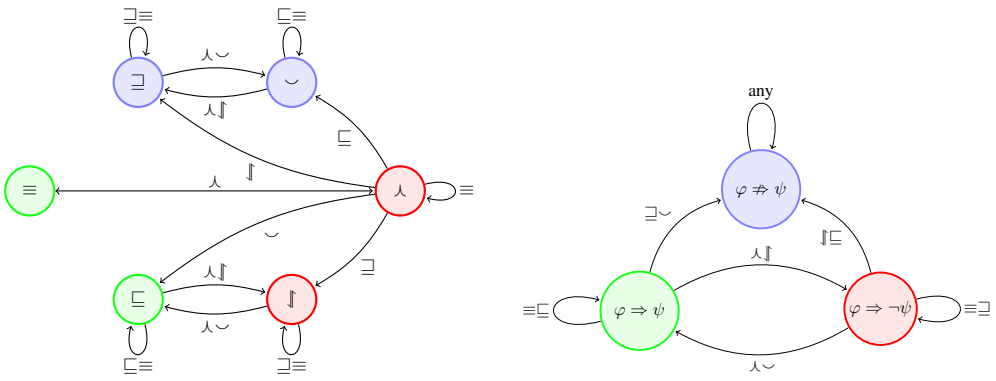
$$r_0 \bowtie \rho(r_1) \bowtie \rho(r_2) \bowtie \rho(r_3),$$

where the initial relation r_0 is axiomatically $\equiv$. In MacCartney's work this style of proof is presented as a table. The last column (s_i) is the relation between the premise and the i^{th} step in the proof, and is constructed inductively as $s_i \coloneqq s_{i-1} \bowtie \rho(r_i)$:

	Mutation	r_i	$\rho(r_i)$	s_i
r_1	*cat*$\rightarrow$*dog*	$\Downarrow$	$\Downarrow$	$\Downarrow$
r_2	$\cdot \rightarrow$*not*	$\frown$	$\frown$	$\sqsubseteq$
r_3	*dog*$\rightarrow$*poodle*	$\sqsupseteq$	$\sqsubseteq$	$\sqsubseteq$

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

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



(a) (b)

Figure 3: (a) Natural logic inference expressed as a finite state automaton. Omitted edges go to the unknown state (#), with the exception of omitted edges from ≡, which go to the state of the edge type. Green states (≡, ⊑) denote valid inferences; red states (⫛, ⅄) denote invalid inferences; blue states (⊒, ⌣) denote inferences of unknown validity. (b) The join table collapsed into the three meaningful states over truth values.

3 Inference as a Finite State Machine

We show that the tabular proof formulation from Section 2.3 can be viewed as a finite state machine, and present a novel observation that we can losslessly collapse this finite state machine into only three intuitive inference states. These observations allow us to formulate our search problem such that a search path corresponds to an input to (i.e., path through) this collapsed state machine.

Taking notation from Section 2.3, we construct a finite state machine over states $s \in \{\sqsubseteq, \sqsupseteq, \dots\}$. A machine in state s_i corresponds to relation s_i holding between the initial premise and the derived fact so far. States therefore correspond to states of *logical validity*. The start state is $\equiv$. Outgoing transitions correspond to *inference steps*. Each transition is labeled with a projected relation $\rho(r) \in \{\sqsubseteq, \sqsupseteq, \dots\}$, and spans from a source state s to a target s' according to the join table. That is, the transition $s \xrightarrow{\rho(r)} s'$ exists iff $s' = s \bowtie \rho(r)$. For example, the path in Figure 1 yields the transitions $\equiv \xrightarrow{⅄} ⅄ \xrightarrow{\sqsupseteq} ⫛ \xrightarrow{\equiv} ⫛ \xrightarrow{\sqsupseteq} ⫛$. Figure 3a shows the automaton, with trivial edges omitted for clarity.

Our second contribution is collapsing this automaton into the three meaningful states we use as output: *valid* ($\varphi \Rightarrow \psi$), *invalid* ($\varphi \Rightarrow \neg\psi$), and *unknown validity* ($\varphi \not\Rightarrow \psi$). We can cluster states in Figure 3a into these three categories. The relations $\equiv$ and $\sqsubseteq$ correspond to valid inferences; ⅄ and ⫛ correspond to invalid inferences; $\sqsupseteq$, $\smile$ and # correspond to unknown validity. This clustering mirrors that used by MacCartney for his textual entailment experiments.

Collapsing the FSA into the form in Figure 3b becomes straightforward from observing the regularities in Figure 3a. Nodes in the valid cluster transition to invalid nodes always and only on the relations ⅄ and ⫛. Symmetrically, invalid nodes transition to valid nodes always and only on ⅄ and $\smile$. A similar pattern holds for the other transitions.

Formally, for every relation r and nodes a_1 and a_2 in the same cluster, if we have transitions $a_1 \xrightarrow{r} b_1$ and $a_2 \xrightarrow{r} b_2$ then b_1 and b_2 are necessarily in the same cluster. As a concrete example, we can take $r = ⅄$ and the two states in the *invalid* cluster: $a_1 = ⅄$, $a_2 = ⫛$. Although $⅄ \xrightarrow{⅄} \equiv$ and $⫛ \xrightarrow{⅄} \sqsubseteq$, both $\equiv$ and $\sqsubseteq$ are in the same cluster (*valid*). It is not trivial *a priori* that the join table should have this regularity, and it certainly simplifies the logic for inference tasks.

A few observations deserve passing remark. First, even though the states $\sqsupseteq$ and $\smile$ appear meaningful, in fact there is no "escaping" these states to either a valid or invalid inference. Second, the hierarchy over relations presented in Icard (2012) becomes apparent – in particular, ⅄ always behaves as negation, whereas its two "weaker" versions (⫛ and $\smile$) only behave as negation in certain contexts. Lastly, with probabilistic inference, transitioning to the unknown state can be replaced with staying in the current state at a (potentially arbitrarily large) cost to the confidence of validity. This allows us to make use of only two states: *valid* and *invalid*.

4 Inference As Search

Natural Logic allows us to formalize our approach elegantly as a single search problem. Given a query, we search over the space of possible facts for a valid premise in our database. The nodes in our search problem correspond to candidate facts (Section 4.1); the edges are mutations of these facts (Section 4.2); the costs over these edges encode the confidence that this edge maintains an informative inference (Section 4.5). This mirrors the automaton defined in Section 3, except importantly we are constructing a reversed derivation, and are therefore "traversing" the FSA backwards.

This approach is efficient over a large database of 270 million entries without making use of explicit queries over the database; nor does the approach make use of any sort of approximate matching against the database, beyond lemmatizing individual lexical items. The motivation in prior work for approximate matches – to improve the recall of candidate premises – is captured elegantly by relaxing Natural Logic itself. We show that allowing invalid transitions with appropriate costs generalizes JC distance (Jiang and Conrath, 1997) – a common thesaurus-based similarity metric (Section 4.3). Importantly, however, the entire inference pipeline is done within the framework of weighted lexical transitions in Natural Logic.

4.1 Nodes

The space of possible nodes in our search is the set of possible partial derivations. To a first approximation, this is a pair (w, s) of a surface form w tagged with word sense and polarity, and an inference state $s \in \{\text{valid}, \text{invalid}\}$ in our collapsed FSA (Figure 3b). For example, the search path in Figure 1 traverses the nodes:

(*No carnivores eat animals*,	valid)
(*The carnivores eat animals*,	invalid)
(*The cat eats animals*,	invalid)
(*The cat eats an animal*,	invalid)
(*The cat ate a mouse*,	invalid)

During search, we assume that the validity states s are reversible – if we know that *the cat ate a mouse* is true, we can infer that *no carnivores eat animals* is false. In addition, our search keeps track of some additional information:

Mutation Index Edges between sentences are most naturally defined to correspond to mutations of individual lexical items. We therefore maintain an index of the next item to mutate at each search state. Importantly, this enforces that each derivation orders mutations left-to-right; this is computationally efficient, at the expense of rare search errors. A similar observation is noted in MacCartney (2009), where prematurely collapsing to # occasionally misses inferences.

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

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 nodes, which are separated by a single transition instance. For example, an edge exists between *some felines have tails* and *some cats have tails*. Transition [instances] are stored statically in memory, whereas edges are constructed on demand.

Transition templates provide a means of defining transitions and subsequently edges in our search space using existing lexical resources (e.g., WordNet, distributional similarity, etc.). We can then define a mapping from these templates to Natural Logic lexical relations. This allows us to map every edge in our search graph back to the Natural Logic relation it instantiates. The full table of transition templates is given in Table 3, along with the Natural Logic relation that instances of the template introduce. We include most relations in WordNet as transitions, and parametrize insertions and deletions by the part of speech of the token being inserted/deleted.

Once we have an edge defining a lexical mutation with an associated Natural Logic relation r, we can construct the corresponding end node (w', s') such that w' is the sentence with the lexical mutation applied, and s' is the validity state obtained from the FSA in Section 3. For instance, if our edge begins at (w, s), and there exists a transition in the FSA from $s' \xrightarrow{r} s$, then we define the end point of the edge to be (w', s'). To illustrate concretely, suppose our search state is:

(*some felines have tails*, valid)

Transition Template	Relation
WordNet hypernym	$\sqsubseteq$
WordNet hyponym	$\sqsupseteq$
WordNet antonym†	$\Downarrow$
WordNet synonym/pertainym†	$\equiv$
Distributional nearest neighbor	$\equiv$
Delete word†	$\sqsubseteq$
Add word†	$\sqsupseteq$
Operator weaken	$\sqsubseteq$
Operator strengthen	$\sqsupseteq$
Operator negate	$\curlywedge$
Operator synonym	$\equiv$
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 transition template for WordNet hypernymy gives us a transition instance from *feline* to *cat*, corresponding to the Natural Logic inference *cat* $\stackrel{\sqsubseteq}{\Longrightarrow}$ *feline*. Recall, we are constructing the inference in reverse, starting from the consequent (query). We then notice that the transition $valid \stackrel{\sqsubseteq}{\Longrightarrow} valid$ in the FSA ends in our current inference state (*valid*), and set our new inference state to be the start state of the FSA transition – in this case, we maintain validity.

Note that negation is somewhat subtle, as the transitions are not symmetric from valid to invalid and visa versa, and we do not know our true inference state with respect to the premise yet. In practice, the search procedure treats all three of $\{\curlywedge, \Downarrow, \smile\}$ as negation, and re-scores complete derivations once their inference states are known.

It should be noted that the mapping from transition templates 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 from fact (w, v) with surface form w and validity v to a new fact (w', v'), using a transition instance t_i of template t and mutating a word with polarity p, is given by $f_{t_i} \cdot \theta_{t,v,p}$. We define this as:

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

$\theta_{t,v,p}$: A learned cost for taking a transition of template t, if the source of the edge is in a inference 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. We set f_{t_i} to be 1 in most cases; the exceptions are the edges over the WordNet hypernym tree and the nearest neighbors edges. In the first case, taking the hypernymy relation from w to w' to be $\uparrow_{w \to w'}$, we set:

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

The value $f_{\downarrow_{w \to w'}}$ is set analogously. We define $p(w)$ to be the "probability" of a concept – that is, the normalized frequency of a word w 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. However, many of these transitions have low weight: for instance $f_{\uparrow_{cat \to feline}}$ is only 0.37.

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 (i.e., the angle) between word vectors associated with lexical items w and w':

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

For instance, $f_{NN_{cat \to dog}} = 0.43$. In practice, we explore the 100 nearest neighbors of each word.

We can express f_{t_i} as a feature vector by representing it as a vector with value f_{t_i} at the index corresponding to (t, v, p) – the transition template, the validity of the inference, and the polarity of the mutated word. Note that the size of this vector mirrors the number of cost parameters $\theta_{t,v,p}$, and

[2] The latter example is actually a consequence of the projection function in Table 1 being overly optimistic.

is in general smaller than the number of transition instances.

A search path can then be parametrized by a sequence of feature vectors $f_1, f_2, \ldots, f_n$, which in turn can be collapsed into a single vector $\mathbf{f} = \sum_i f_i$. The cost of a path is defined as $\theta \cdot \mathbf{f}$, where θ is the vector of $\theta_{t,v,p}$ values. Both $\mathbf{f}$ and θ are constrained to be non-negative, or else the search problem is misspecified.

4.3 Generalizing Similarities

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

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

For simplicity, we simplify $\theta_{\uparrow,v,p}$ and $\theta_{\downarrow,v,p}$ as simply $\theta_\uparrow$ and $\theta_\downarrow$. Without loss of generality, we also assume that a path in our search is only modifying a single lexical item w_1, eventually reaching a mutated form 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), $[w_1, w_1^{(1)}, \ldots, \text{lcs}, \ldots, w_2^{(1)}, w_2]$, as follows:

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

Note that setting both $\theta_\uparrow$ and $\theta_\downarrow$ to 1 exactly yields Formula (1) for JC distance. This, in addition to the inclusion of nearest neighbors as transitions, allows the search to capture the intuition that similar objects have similar properties (e.g., as used in Angeli and Manning (2013)).

4.4 Deletions in Inference

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, since from the perspective of search we are inserting lexical items.

Naïvely, at every node in our search we must consider every item in the vocabulary as a possible insertion. We can limit the number of items we consider by storing the database as a trie. Since the search mutates the fact left-to-right (as per Section 4.1), we can consider children of a trie node as candidate insertions. To illustrate, given a search state with fact $w_0\, w_1 \ldots w_n$ and mutation index i, we would look up completions w_{i+1} for $w_0\, w_1 \ldots w_i$ in our trie of known facts.

Although this approach works well when i is relatively large, there are too many candidate insertions for small i. We special case the most extreme example 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. To illustrate, given a search state with fact $w_0\, w_1 \ldots w_n$, we would propose candidate insertions w_{-1} such that $w_{-1}\, w_0\, w_1' \ldots w_k'$ is a known fact for some $w_1' \ldots w_k'$. More concretely, if we know that *fluffy cats have tails*, and are at a node corresponding to *cats like boxes*, we propose *fluffy* as a possible insertion: *fluffy cats like boxes*.

4.5 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 cost vector θ 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 query is in the *valid* state with respect to the premise, and -1 if the query is in the *invalid* state. For completeness, if no path is given we can set $v = 0$. The

probability of validity 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 $\frac{1}{2}$ − 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.

5 Learning Transition Costs

We describe our procedure for learning the transition costs θ. Our training data $\mathcal{D}$ consists of query facts q and their associated gold truth values y. Equation (2) gives us a probability that a particular inference is *valid*; we axiomatically consider a valid inference from a known premise to be justification for the truth of the query. This is at the expense of the (often incorrect) assumption that our database is clean and only contains true facts.

We optimize the likelihood of our gold annotations according to this probability, subject to the constraint that all elements in our cost vector θ be non-negative. We run the search algorithm described in Section 4 on every query $q_i \in \mathcal{D}$. This produces the highest confidence path x_1, along with its inference state v_i. We now have annotated tuples: $((x_i, v_i), y_i)$ for every element in our training set. Analogous to logistic regression, the log likelihood of our training data $\mathcal{D}$, subject to costs θ, is:

$$l_\theta(\mathcal{D}) = \sum_{0 \le i < |\mathcal{D}|} \Big[y_i \log \left(\frac{v_i}{2} + \frac{1}{1 + e^{v_i \theta \cdot \mathbf{f}(x_i)}} \right) + (1 - y_i) \log \left(\frac{-v_i}{2} + \frac{1}{1 + e^{-v_i \theta \cdot \mathbf{f}(x_i)}} \right) \Big],$$

where y_i is 1 if the example is annotated true and 0 otherwise, and $\mathbf{f}(x_i)$ are the features extracted for path x_i. The objective function is the negative log likelihood with an L_2 regularization term and a log barrier function to prohibit negative costs:

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

We optimize this objective using conjugate gradient descent. Although the objective is non-convex, in practice we can find a good initialization of weights to reduce the risk of arriving at local optima.

An elegant property of this formulation is that the weights we are optimizing correspond directly to the costs used during search. This creates a positive feedback loop – as better weights are learned, the search algorithm is more likely to find confident paths, and more data is available to train from. We therefore run this learning step for multiple epochs, re-running search after each epoch. The weights for the first epoch are initialized to an approximation of valid Natural Logic weights. Subsequent epochs initialize their weights to the output of the previous epoch.

§	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 (although not 2 and 9, which are also included in M08).

6 Experiments

We evaluate our system on two tasks: the FraCaS test suite, used by MacCartney and Manning (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 a system's handling of various entailment patterns. We process the corpus following MacCartney and Manning (2007). 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, and consequently our system as well.

Results on the dataset are given in Table 4.

System	P	R	F_1	Accuracy
Lookup	100.0	12.1	21.6	56.0
NaturalLI Only	88.8	40.1	55.2	67.5
NaturalLI + Lookup	90.6	49.1	63.7	72.0

Table 5: Accuracy inferring common-sense facts on a balanced test set. *Lookup* queries the lemmatized lower-case fact directly in the 270M fact database. *NaturalLI Only* disallows such lookups, and infers every query from only distinct premises in the database. *NaturalLI + Lookup* takes the union of the two systems.

Since the corpus is not a blind test set, the results are presented less as a comparison of performance, but rather to validate the expressive power of our search-based approach against MacCartney's align-and-classify approach. For the experiments, costs were set to express valid Natural Logic inference as a hard constraint.

The results show that the system is able to capture Natural Logic inferences with similar accuracy to the 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 premise when constructing the search, leading to the introduction of a class 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*. In order to err on the side of safety, we disallow deleting the verb *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 on FraCaS shows that our search formulation captures applicable inferences as well as prior work, this evaluation presents a novel use-case for Natural Logic inference.

For our database of facts, we run the Ollie OpenIE system (Mausam et al., 2012) over Wikipedia,[3] Simple Wikipedia,[4] and a random 5% of CommonCrawl. Extractions with confidence below 0.25 or which contained pronouns were discarded. This yielded a total of 305 million unique extractions composed entirely of lexical items which mapped into our vocabulary (186 707 items). Each of these extracted triples (e_1, r, e_2) was then flattened into a plain-text fact $e_1\ r\ e_2$ and lemmatized. This yields 270 million unique lemmatized premises in our database. In general, each fact in the database could be arbitrary unstructured text; our use of Ollie extractions is motivated only by a desire to extract short, 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 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 queries 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 (Angeli and Manning, 2013). This provides a set of plausible but nonetheless incorrect examples, and ensures that our recall is not due to over-zealous search. Search costs are tuned from an additional set of 540 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 short snippets, already lemmatized and lower-cased; therefore, it is not in principle unreasonable to expect a database of 270 million extractions to contain these facts. Nonetheless, only 12% of facts were found via a direct lookup. We show that NaturalLI (allowing lookups) improves this recall four-fold, at only an

[3] `http://wikipedia.org/` (2013-07-03)
[4] `http://simple.wikipedia.org/` (2014-03-25)

9.4% drop in precision. Furthermore, if we prohibit direct lookup, we still recover all but 9% 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.

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), and information extraction systems (Hoffmann et al., 2011; Surdeanu et al., 2012). 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.

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, following a line of work by Bordes et al. (2011) and Jenatton et al. (2012). Yao et al. (2012) and Riedel et al. (2013) present a related line of work, inferring new relations between Freebase entities via inference over both Freebase and OpenIE relations. In contrast, this work runs inference over arbitrary text, without restricting itself to a particular set of relations, or even entities.

The goal of tackling common-sense reasoning is by no means novel in itself. Work by Reiter and McCarthy (Reiter, 1980; McCarthy, 1980) attempts 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 broad-coverage system. More recently, work by Schubert (2002) and Van Durme et al. (2009) approach common sense reasoning with *episodic logic*; we differ in our focus on inferring truth from an arbitrary query, and in making use of longer inferences.

This work is similar in many ways to work on recognizing textual entailment – e.g., Schoenmackers et al. (2010), Berant et al. (2011). Work by Lewis and Steedman (2013) is particularly relevant, as they likewise evaluate on the FraCaS suite (Section 1; 89% accuracy with gold trees). They approach entailment by constructing a CCG parse of the query, while mapping questions which are paraphrases of each other to the same logical form using distributional relation clustering. However, their system is unlikely to scale to either our large database of premises, nor our breadth of relations.

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. Furthermore, our system is able to explore mutations which are only valid in one direction, rather than the bidirectional entailment of paraphrases, and does not require a corpus of such paraphrases for training.

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 premises while maintaining Natural Logic semantics, and 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. For example, extending the approach to handle meronymy and relation entailments. Furthermore, we hope to learn richer lexicalized parameters, and use the syntactic structure of a fact during search.

Acknowledgements We thank the anonymous reviewers for their thoughtful comments. We gratefully acknowledge the support of the Defense Advanced Research Projects Agency (DARPA) Deep Exploration and Filtering of Text (DEFT) Program under Air Force Research Laboratory (AFRL) contract no. FA8750-13-2-0040. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government.

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.

Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, et al. 2011. Learning structured embeddings of knowledge bases. In *AAAI*.

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*, 9:1–18.

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*.

Rodolphe Jenatton, Nicolas L Roux, Antoine Bordes, and Guillaume R Obozinski. 2012. A latent factor model for highly multi-relational data. In *NIPS*.

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.

Mike Lewis and Mark Steedman. 2013. Combined distributional and logical semantics. *TACL*, 1:179–192.

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*.

Bill MacCartney. 2009. *Natural Language Inference*. Ph.D. thesis, Stanford.

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. *Principles of Knowledge Representation and Reasoning*.

Raymond Reiter. 1980. A logic for default reasoning. *Artificial intelligence*, 13(1):81–132.

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*.

Lenhart Schubert. 2002. Can we derive general world knowledge from texts? In *HLT*.

Rion Snow, Daniel Jurafsky, and Andrew Y Ng. 2006. Semantic taxonomy induction from heterogenous evidence. In *ACL*.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. In *NIPS*.

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, University of Amsterdam.

Johan van Benthem. 1986. *Essays in logical semantics*. Springer.

Johan van Benthem. 2008. A brief history of natural logic. Technical Report PP-2008-05, University of Amsterdam.

Benjamin Van Durme, Phillip Michalak, and Lenhart K Schubert. 2009. Deriving generalized knowledge from corpora using wordnet abstraction. In *EACL*.

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*.