Cats have Tails

1 Search Problem

We run a search over mutations over specific words of the input candidate consequent. The search problem is given a consequent as input. This is composed of the following:

- A sentence, tokenized into coherent logical entities. For instance, *dog* and *cat* would be entities; but, so would *George W. Bush* – despite the latter covering multiple words.
- A monotonicity marking for each of these words. This is one of: upward monotone, downward monotone, or flat (no monotonicity information could be gathered).
- The relevant sense of the word. This is a WordNet synset when it's available; if not, this is set to a special sense denoting no synset. If a WordNet synset is available, it will never be set to the 'no synset' category.

The paths from the result of this searches are used as features in the learning problem (see Section 2).

We define the problem by specifying the state space, the valid transitions, along with the weights of the transitions. The algorithm used is a Uniform Cost Search, leading to the additional challenge of ensuring that the cost of each transition is very fast – that is, on the order of microseconds to keep pace with the memory accesses for the search.

1.1 State

A state in the graph is defined as a partial path from the consequent to a valid antecedent. That is, a state is primarily parameterized by its candidate fact which may be in the knowledge base. In addition, some auxilliary information is also tracked in a search state:

- The fact this fact mutated from. This is necessary to reconstruct the final path.

- The monotonicity marking of each word in the fact. This is necessary for the weights, to ensure that monotonicity is respected.
- The synset of each word in the fact. This is necessary to compute the valid mutations of that edge, according to the WordNet hierarchy.
- A bitmask denoting whether a word has been modified in the past. This is set when a word has been modified, and then another word is subsequently modified. This ensures that edits on a word are contiguous. We do not edit a particualr word, and then edit another word, and then perform more edits on the initial word.

1.2 Weights

TODO: This section is now wrong We can denote a state transition as (n_{-1}, n) – corresponding to a state n and the previous state $n-1$. The weight of a transition from (n_{-1}, n) to (n, n_{+1}) is given by appealing to the transitions between n_{-1} and n, and between n and n_{+1}.

We denote the type of transition (e.g., ins(w) or freebase(r)) between n_{-1} and n to be ϕ_{-1}, and the type of transition between n and n_{+1} to be ϕ_0. For both of these, we define weights $w_{\phi_{-1}}$ and w_{ϕ_0}, as well as a weight for the bigram of the two: w_{ϕ_{-1},ϕ_0}. Lastly, we incur a cost for exiting the domain of natural logic entailments: w_{nle}. Our cost is thus given by:

$$\text{cost} = -w_{\phi_0} - w_{\phi_{-1},\phi_0} - \mathbb{1}(\text{broke entailment}) \cdot w_{nle} \tag{1}$$

We constrain $w_\phi \leq 0$ to ensure that the cost is always positive.

Note, furthermore, that this is a natural decomposition of a log-linear model where ϕ denote features and w denote the weights of the features. The cost of a path will become:

$$\sum_i cost_i = -\sum_i \left[w_{\phi_i)} + w_{\phi_{i-1},\phi_i} \right] \tag{2}$$

If we exponentiate the negative of this, we arrive at:

$$\exp(\sum_i cost_i) = \exp\left(\sum_i \left[w_{\phi_i)} + w_{\phi_{i-1},\phi_i} \right] \right) \tag{3}$$

Over two classes: *true* and *false*; and, defining notation where ϕ denotes the vector of active path types taken and w denotes the global weight vector, we arrive at our log-linear model:

$$P(\mathit{true}) \propto e^{w^T \phi} \tag{4}$$

This decomposition is important, as it allows our search to get smarter along with our learning algorithm, and allows us to find better support for facts as we learn what patterns entail "good" support.

1.3 Transitions

A valid transition is one between states (n_{-1}, n) and (n, n_{+1}), such that in relation to n, n_{+1} is one of:

- **Add/Remove a quantifier**. This is a class of transitions denoted by *ins(w)* or *del(w)* where w is the word being added or removed. For example, transitioning from *cat* to *a cat* or *every cat* or visa versa.
- **Add/Remove an adjective**. Similar to above, but with adjectives. These are distinguished from the above in that they have meaning for natural logic entailment. **TODO:** how do we make this efficient?
- **WordNet hypernymy**. This is a class of transitions denoted by *hyper* or *hypo*, For example, we could transition from *cat* to *feline* and eventually to *animal*
- **WordNet relations**. This is a class of transitions related to the other WordNet relations (e.g., antonymy), primarily to capture some of the inferences in natlog.
- **Freebase relations**. This is a class of transitions aimed primarily for proper nouns, traversing Freebase relations. These are denoted by *freebase(r)*, where r denotes the freebase relation we have traversed. For example, *Barack Obama* could transition to *USA* via the *employee_of* relation.
- **ReVerb relation**. As in the Freebase case, we can move along a ReVerb relation, denoted as *reverb(r)*.
- **Acronym and de-acronym**. Expanding or creating an acronym,denoted by *acronym* and *deacronym*.
- **Nearest neighbor similarity**. The fallback is to search for nearest neighbors in similarity space. This is, roughly, the equivalent of the CoNLL paper.
- **Drop sense**. Drop the sense of a word – this is necessary to begin operating in nearest neighbors space, or even with Freebase, etc. relations.
- **Infer sense**. Go from a sense-less word to one of its word senses. For example, chosing a particular sense for *cat* from the initially senseless definition.

2 Learning Problem

The prediction problem is a binary prediction task: is the given fact true or false, given a database of known facts. We divide this section into the model, the objective function, and the data (or lack thereof).

2.1 Model

Given a query fact (a_1, r, a_2) we define the **support** for that fact as the set of facts $\{(a'_1, r', a'_2)\}$ such that we have paths from $a_1 \to a'_1$, $r \to n'$, and $a_2 \to a'_2$. We are thus given a set of triples of paths $\{p_{a_1}, p_r, p_{a_2}\}$. When featurized,

2.2 Objective

3 Misc Snippets

3.1 Generalization of JC 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:

$$\text{dist}_{\text{jc}}(w_1, w_2) = \log \frac{p(\text{lcs})^2}{p(w_1)p(w_2)} \tag{5}$$

We show that our search over the Wordnet hierarchy generalizes this similarity. In particular, let us define two features, $\phi_\uparrow$ and $\phi_\downarrow$, corresponding to going up and down the WordNet hierarchy, respectively. Traversing the Wordnet hierarchy from words $w \to w'$ thus fires the ϕ features with counts:

$$\phi_\uparrow(w \to w') = \log \frac{p(w')}{p(w)} = \log p(w') - \log p(w) \tag{6}$$

$$\phi_\downarrow(w \to w') = \log \frac{p(w)}{p(w')} = \log p(w) - \log p(w') \tag{7}$$

We now introduce weights associated with each of these two operations, denoted $\theta_\uparrow$ and $\theta_\downarrow$, for each pair of words participating in a WordNet edge. The score of a path is then defined as the dot product of the weights and features as described above: $\theta^{\mathrm{T}}\phi$.

We can factorize this along the path $w_1, w_1^{(1)}, w_1^{(2)}, \ldots, \text{lcs}, \ldots, w_2^{(2)}, w_2^{(1)}, w_2$ as follows:

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

Note that setting both $\theta_\uparrow$ and $\theta_\downarrow$ to 1 exactly yield JC similarity.

4 Natural Logic

Much of the underpinnings of this work rely on the theoretical framework laid by Natural Logic Van Benthem (1986); Valencia (1991). **TODO:** motivate NatLog

An interpretation of Natural Logic as syllogistic reasoning, presented in Section 4.1, is sufficient for a working understanding of the contributions of this paper. However, Natural Logic is both more powerful than syllogistic reasoning, and is more general than structural matching to known syllogistic templates. Section 4.2 introduces *monotonicity* in the context of natural language to form the basis of natural logic; Section 4.3 extends this formalism for reasoning about a wider range of phenomena. Icard III & Moss (2014) offers a good introduction and summary of Natural Logic, and provides a source for much of the notation and exposition presented here.

4.1 Syllogistic Reasoning

TODO: write me

4.2 Monotonicity in Natural Logic

Monotonicity in Natural Logic derives its name from and correlates precisely to monotonicity of functions. 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. This fact, as well as the ordering of real numbers, allows us 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 $f(1, 1, 1) \geq f(1, 2, 1)$.

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 functions in our example above. For instance, $f(x) = x + c$ is monotone; $g(x) = -x$ is antitone; $h(x) = (x - 2)^2$ is non-monotone.

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

Let the denotation of a phrase, marked $[\![\cdot]\!]$, be the set of entities 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 a partial ordering $\leq_e$ following the ordering over sets above: for every $x, y \subseteq e$, $x \leq_e y \Leftrightarrow x \cap y = x$. We define the set t to be the set of truth values: $\{T, F\}$. The ordering over truth

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

values is trivially $x \leq_t y \Leftrightarrow x = F \vee y = T$. It should not go unnoticed that $x \Rightarrow y$ and $x \leq_t y$ are equivalent. Note that functions $e \rightarrow t$ follow the partial ordering from above.

We can now 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] This matches our intuition that, e.g., *all cats have tails* implies that *all tabby cats have tails*.

Formally, we can produce proofs of these sorts of facts 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:

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

In a similar vein, we can provide a proof for *all tabby cats have tails* via:

$$\dfrac{\dfrac{\llbracket \textit{tabby cats} \rrbracket \leq_e \llbracket \textit{cats} \rrbracket}{\llbracket \textit{all} \rrbracket\,(\llbracket \textit{tabby cats} \rrbracket)\,(\cdot) \geq_{e \rightarrow t} \llbracket \textit{all} \rrbracket\,(\llbracket \textit{cats} \rrbracket)\,(\cdot)} \qquad \llbracket \textit{have tails} \rrbracket \leq \llbracket \textit{have tails} \rrbracket}{\llbracket \textit{all} \rrbracket\,(\llbracket \textit{tabby cats} \rrbracket)\,(\llbracket \textit{have tails} \rrbracket) \geq_t \llbracket \textit{all} \rrbracket\,(\llbracket \textit{cats} \rrbracket)\,(\llbracket \textit{have tails} \rrbracket)}$$

That is to say, on a given parse of the respective sentences (in this case unambiguous), $\llbracket$*All cats have tails*$\rrbracket \leq_t \llbracket$*All tabby cats have tails*$\rrbracket$; and, as mentioned above, $\leq_t$ and modus ponens ($\Rightarrow$) are defined equivalently.

4.3 Reasoning with Exclusion

Work by MacCartney & Manning (2009), building off of MacCartney & Manning (2007) and MacCartney & Manning (2008), has focused on extending the set of atomic relationships between denotations of objects beyond $\leq_e$ (subset) and $\geq_e$ (superset). We adopt the semantics of Icard III (2012), although in this work for simplicity we do not make use of their extended monotonicity markings. A more thorough treatment of the soundness and completeness of the proof theory described here can be found in Djalali (2013).

Icard III & Moss (2014) noted that the partial ordering of $\leq_e$ can be formulated as a *bounded distributive lattice* to encompass the original relations of **?**. 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 $\perp$ and $\top$ respectively. By example, for the domain of entities we can define $\mathcal{X}$ to be the power set of our

[2]These markings can, in fact, be elegantly incorporated into the type system. Since this work does not make use of categorical grammars requiring strict interpretations of types, we refer the reader to Icard III & Moss (2014) and do not elablorate on this topic further.

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:

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

Note that unlike in the original MacCartney formulation, these relations are not mutually exclusive; in fact, a hierarchy of informativeness emerges where, e.g., $\curlywedge$ subsumes $\Vert$ and $\smile$.

We are now faced with two issues which need resolving to generalize the proofs from Section 4.2. First, we need to resolve how the monotonicity of the function application projects the relation in the argument. Implicitly used in the proofs so far is the notion that antitone contexts will flip $\leq$ and $\geq$, whereas monotone contexts will not. This relation is preserved for $\sqsubseteq$ and $\sqsupseteq$. Icard III (2012) present an enriched tagging of monotonicity which allows the other relations (i.e., $\curlywedge$, $\Vert$, $\smile$) to project meaningfully. This work does not make use of this semantics, although it is not theoretically impossible to do so – practically, this prohibits our system from reasoning about antonyms.

References

Djalali, Alex J. 2013. Synthetic logic. *Linguistic Issues in Language Technology*.

Icard III, Thomas. 2012. Inclusion and exclusion in natural language. *Studia Logica*.

Icard III, Thomas, & Moss, Lawrence. 2014. Recent Progress on Monotonicity. *Linguistic Issues in Language Technology*.

MacCartney, Bill, & Manning, Christopher D. 2007. Natural logic for textual inference. *In: ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.

MacCartney, Bill, & Manning, Christopher D. 2008. Modeling semantic containment and exclusion in natural language inference. *In: Coling*.

MacCartney, Bill, & Manning, Christopher D. 2009. An extended model of natural logic. *In: Proceedings of the eighth international conference on computational semantics*.

Valencia, Víctor Manuel Sánchez. 1991. *Studies on natural logic and categorial grammar*. Ph.D Thesis.

Van Benthem, Johan. 1986. *Essays in logical semantics*.