Leveraging Linguistic Structure For Open Domain Information Extraction

Gabor Angeli **Melvin Johnson Premkumar** **Christopher D. Manning**

Department of Computer Science
Stanford University
{angeli, melvinj, manning}@cs.stanford.edu

Abstract

Relation triples produced by open domain information extraction (open IE) systems are useful for question answering, inference, and other IE tasks. Traditionally these are extracted using a large set of patterns; however, this approach is brittle on out-of-domain text and long-range dependencies, and gives no insight into the substructure of the arguments. We replace this large pattern set with a few patterns for canonically structured sentences, and shift the focus to a classifier which learns to extract self-contained clauses from longer sentences. We then run natural logic inference over these short clauses to determine the maximally specific arguments for each candidate triple. We show that our approach outperforms a state-of-the-art open IE system on the end-to-end TAC-KBP 2013 Slot Filling task.

1 Introduction

Open information extraction (open IE) has been shown to be useful in a number of NLP tasks, such as question answering (Fader et al., 2014), relation extraction (Soderland et al., 2010), and information retrieval (Etzioni, 2011). Conventionally, open IE systems search a collection of patterns over either the surface form or dependency tree of a sentence. Although a small set of patterns covers most simple sentences (e.g., subject verb object constructions), relevant relations are often spread across clauses (see Figure 1) or presented in a non-canonical form.

Systems like Ollie (Mausam et al., 2012) approach this problem by using a bootstrapping method to create a large corpus of broad-coverage partially lexicalized patterns. Although this is effective at capturing many of these patterns, it can lead to unintuitive behavior on out-of-domain text. For instance, while *Obama is president* is extracted correctly by Ollie as *(Obama; is; president)*, replacing *is* with *are* in *cats are felines* produces no extractions. Furthermore, existing systems struggle at producing canonical argument forms – for example, in Figure 1 the argument *Heinz Fischer of Austria* is likely less useful for downstream applications than *Heinz Fischer*.

Born in Honolulu, Hawaii, Obama is a US Citizen.

Our System	**Ollie**
(Obama; is; US citizen)	(Obama; is; a US citizen)
(Obama; born in; Honolulu, Hawaii)	(Obama; be born in; Honolulu)
	(Honolulu; be born in; Hawaii)
	(Obama; is citizen of; US)

Friends give true praise.
Enemies give fake praise.

Our System	**Ollie**
(friends; give; true praise)	(friends; give; true praise)
(friends; give; praise)	
(enemies; give; fake praise)	(enemies; give; fake praise)

Heinz Fischer of Austria visits the US

Our System	**Ollie**
(Heinz Fischer; visits; US)	(Heinz Fischer of Austria; visits; the US)

Figure 1: Open IE extractions produced by the system, alongside extractions from the state-of-the-art Ollie system. Generating coherent clauses before applying patterns helps reduce false matches such as *(Honolulu; be born in; Hawaii)*. Inference over the sub-structure of arguments, in turn, allows us to drop unnecessary information (e.g., *of Austria*), but only when it is warranted (e.g., keep *fake* in *fake praise*).

In this paper, we shift the burden of extracting informative and broad coverage triples away from this large pattern set. Rather, we first pre-process the sentence in linguistically motivated ways to produce coherent clauses which are (1) logically entailed by the original sentence, and (2) easy to segment into open IE triples. Our approach consists of two stages: we first learn a classifier for splitting a sentence into shorter utterances (Section 3), and then appeal to natural logic (Sánchez Valencia, 1991) to maximally shorten these utterances while maintaining necessary context (Section 4.1). A small set of 14 hand-crafted patterns can then be used to segment an utterance into an open IE triple.

We treat the first stage as a greedy search problem: we traverse a dependency parse tree recursively, at each step predicting whether an edge should yield an independent clause. Importantly, in many cases naïvely yielding a clause on a dependency edge produces an incomplete utterance (e.g., *Born in Honolulu, Hawaii*, from Figure 1). These are often attributable to control relationships, where either the subject or object of the governing clause controls the subject of the subordinate clause. We therefore allow the produced clause to sometimes inherit the subject or object of its governor. This allows us to capture a large variety of long range dependencies with a concise classifier.

From these independent clauses, we then extract shorter sentences, which will produce shorter arguments more likely to be useful for downstream applications. A natural framework for solving this problem is natural logic – a proof system built on the syntax of human language (see Section 4.1). We can then observe that *Heinz Fischer of Austria visits China* entails that *Heinz Fischer visits China*. On the other hand, we respect situations where it is incorrect to shorten an argument. For example, *No house cats have rabies* should not entail that *cats have rabies*, or even that *house cats have rabies*.

When careful attention to logical validity is necessary – such as textual entailment – this approach captures even more subtle phenomena. For example, whereas *all rabbits eat fresh vegetables* yields *(rabbits; eat; vegetables)*, the apparently similar sentence *all young rabbits drink milk* does not yield *(rabbits; drink; milk)*.

We show that our new system performs well on a real world evaluation – the TAC KBP Slot Filling challenge (Surdeanu, 2013). We outperform both an official submission on open IE, and a baseline of replacing our extractor with Ollie, a state-of-the-art open IE systems.

2 Related Work

There is a large body of work on open information extraction. One line of work begins with TextRunner (Yates et al., 2007) and ReVerb (Fader et al., 2011), which make use of computationally efficient surface patterns over tokens. With the introduction of fast dependency parsers, Ollie (Mausam et al., 2012) continues in the same spirit but with learned dependency patterns, improving on the earlier WOE system (Wu and Weld, 2010). The Never Ending Language Learning project (Carlson et al., 2010) has a similar aim, iteratively learning more facts from the internet from a seed set of examples. Exemplar (Mesquita et al., 2013) adapts the open IE framework to n-ary relationships similar to semantic role labeling, but without the expensive machinery.

Open IE triples have been used in a number of applications – for example, learning entailment graphs for new triples (Berant et al., 2011), and matrix factorization for unifying open IE and structured relations (Yao et al., 2012; Riedel et al., 2013). In each of these cases, the concise extractions provided by open IE allow for efficient symbolic methods for entailment, such as Markov logic networks or matrix factorization.

Prior work on the KBP challenge can be categorized into a number of approaches. The most common of these are *distantly supervised* relation extractors (Craven and Kumlien, 1999; Wu and Weld, 2007; Mintz et al., 2009; Sun et al., 2011), and rule based systems (Soderland, 1997; Grishman and Min, 2010; Chen et al., 2010). However, both of these approaches require careful tuning to the task, and need to be trained explicitly on the KBP relation schema. Soderland et al. (2013) submitted a system to KBP making use of open IE relations and an easily constructed mapping to KBP relations; we use this as a baseline for our empirical evaluation.

Prior work has used natural logic for RTE-style textual entailment, as a formalism well-suited for formal semantics in neural networks, and as a framework for common-sense reasoning (MacCartney and Manning, 2009; Watanabe et al., 2012; Bowman et al., 2014; Angeli and Manning, 2014). We adopt the precise semantics of Icard and Moss (2014). Our approach of finding short entailments from a longer utterance is similar in spirit to work on textual entailment for information extraction (Romano et al., 2006).

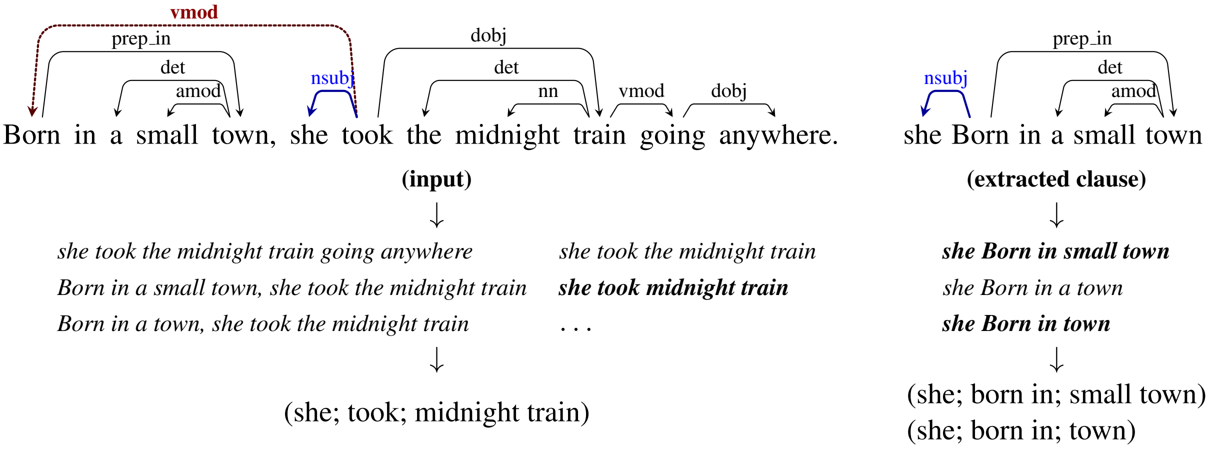
vmod
prep_in
det
amod
nsubj
dobj
det
nn
vmod
dobj
Born in a small town, she took the midnight train going anywhere.
(input)
she took the midnight train going anywhere
Born in a small town, she took the midnight train
Born in a town, she took the midnight train
she took the midnight train
she took midnight train
. . .
(she; took; midnight train)
prep_in
det
amod
nsubj
she Born in a small town
(extracted clause)
she Born in small town
she Born in a town
she Born in town
(she; born in; small town)
(she; born in; town)

Figure 2: An illustration of our approach. From left to right, a sentence yields a number of independent clauses (e.g., *she Born in a small town* – see Section 3). From top to bottom, each clause produces a set of entailed shorter utterances, and segments the ones which match an atomic pattern into a relation triple (see Section 4.1).

3 Inter-Clause Open IE

In the first stage of our method, we produce a set of self-contained clauses from a longer utterance. Our objective is to produce a set of clauses which can stand on their own syntactically and semantically, and are entailed by the original sentence (see Figure 2). Note that this task is not specific to extracting open IE triples. Conventional relation extractors, entailment systems, and other NLP applications may also benefit from such a system.

We frame this task as a search problem. At a given node in the parse tree, we classify each outgoing arc $e = p \xrightarrow{l} c$, from the governor p to a dependent c with [collapsed] Stanford Dependency label l, into an action to perform on that arc. Once we have chosen an action to take on that arc, we can recurse on the dependent node. We decompose the action into two parts: (1) the action to take on the outgoing edge e, and (2) the action to take on the governor p. For example, in our motivating example, we are considering the arc: $e = \textit{took} \xrightarrow{\textit{vmod}} \textit{born}$. In this case, the correct action is to (1) yield a new clause rooted at *born*, and (2) interpret the subject of *born* as the subject of *took*.

We proceed to describe this action space in more detail, followed by an explanation of our training data, and finally our classifier.

3.1 Action Space

The three actions we can perform on a dependency edge are:

Yield Yields a new clause on this dependency arc. A canonical case of this action is the arc $\textit{suggest} \xrightarrow{\textit{ccomp}} \textit{brush}$ in *Dentists suggest that you should brush your teeth*, yielding *you should brush your teeth.*

Recurse Recurse on this dependency arc, but do not yield it as a new clause. For example, in the sentence *faeries are dancing in the field where I lost my bike*, we must recurse through the intermediate constituent *the field where I lost my bike* – which itself is not relevant – to get to the clause of interest: *I lost my bike.*

Stop Do not recurse on this arc, as the subtree under this arc is not entailed by the parent sentence. This is the case, for example, for most leaf nodes (*furry cats are cute* should not entail the clause *furry*), and is an important action for the efficiency of the algorithm.

With these three actions, a search path through the tree becomes a sequence of **Recurse** and **Yield** actions, terminated by a **Stop** action (or leaf node). For example, a search sequence $A \xrightarrow{\textit{Recurse}} B \xrightarrow{\textit{Yield}} C \xrightarrow{\textit{Stop}} D$ would yield a clause rooted at C. A sequence $A \xrightarrow{\textit{Yield}} B \xrightarrow{\textit{Yield}} C \xrightarrow{\textit{Stop}} D$ would yield clauses rooted at both B and C. Finding all such sequences is in general exponential in the size of the tree. In practice, during training we run breadth first search to collect the first 10 000 sequences. During inference we run uniform cost search until our classifier predictions fall below a given threshold.

For the **Stop** action, we do not need to further specify an action to take on the parent node. However, for both of the other actions, it is often the case that we would like to capture a controller in the higher clause. We define three such common actions:

Subject Controller If the arc we are considering is not already a subject arc, we can copy the subject of the parent node and attach it as a subject of the child node. This is the action taken in the example *Born in a small town, she took the midnight train*.

Object Controller Analogous to the subject controller action above, but taking the object instead. This is the intended action for examples like *I persuaded Fred to leave the room.*[1]

Parent Subject If the arc we are taking is the only outgoing arc from a node, we take the parent node as the (passive) subject of the child. This is the action taken in the example *Obama, our 44th president* to yield a clause with the semantics of *Obama [is] our 44th president*.

Although additional actions are easy to imagine, we found empirically that these cover a wide range of applicable cases. We turn our attention to the training data for learning these actions.

3.2 Training

We collect a noisy dataset to train our clause generation model. We leverage the *distant supervision* assumption for relation extraction, which creates a noisy corpus of sentences annotated with relation mentions (subject and object spans in the sentence with a known relation). Then, we take this annotation as itself distant supervision for a correct sequence of actions to take: any sequence which recovers the known relation is correct.

We use a small subset of the KBP source documents for 2010 (Ji et al., 2010) and 2013 (Surdeanu, 2013) as our distantly supervised corpus. To try to maximize the density of known relations in the training sentences, we take all sentences which have at least one known relation for every 10 tokens in the sentence, resulting in 43 155 sentences. In addition, we incorporate the 23 725 manually annotated examples from Angeli et al. (2014).

Once we are given a collection of labeled sentences, we assume that a sequence of actions which leads to a correct extraction of a known relation is a *positive sequence*. A correct extraction is any extraction we produce from our model (see Section 4) which has the same arguments as the known relation. For instance, if we know that Obama was born in Hawaii from the sentence *Born in Hawaii, Obama* . . . , and an action sequence produces the triple *(*Obama, born in, Hawaii), then we take that action sequence as a positive sequence.

Any sequence of actions which results in a clause which produces no relations is in turn considered a *negative sequence*. The third case to consider is a sequence of actions which produces a relation, but it is not one of the annotated relations. This arises from the *incomplete negatives* problem in distantly supervised relation extraction (Min et al., 2013): since our knowledge base is not exhaustive, we cannot be sure if an extracted relation is incorrect or correct but previously unknown. Although many of these unmatched relations are indeed incorrect, the dataset is sufficiently biased towards the STOP action that the occasional false negative hurts end-to-end performance. Therefore, we simply discard such sequences.

Given a set of noisy positive and negative *sequences*, we construct training data for our action classifier. All but the last action in a positive sequence are added to the training set with the label **Recurse**; the last action is added with the label **Split**. Only the last action in a negative sequence is added with the label **Stop**. We partition the feature space of our dataset according to the action applied to the parent node.

3.3 Inference

We train a multinomial logistic regression classifier on our noisy training data, using the features in Table 1. The most salient features are the label of the edge being taken, the incoming edge to the parent of the edge being taken, neighboring edges for both the parent and child of the edge, and the part of speech tag of the endpoints of the edge. The dataset is weighted to give 3× weight to examples in the **Recurse** class, as precision errors in this class are relatively harmless for accuracy, while recall errors are directly harmful to recall.

Inference now reduces to a search problem. Be-

[1] The system currently misses most most such cases due to insufficient support in the training data.

Feature Class	Feature Templates
Edge taken	$\{l, \text{short\_name}(l)\}$
Last edge taken	$\{\text{incoming\_edge}(p)\}$
Neighbors of parent	$\{\text{nbr}(p), (p, \text{nbr}(p))\}$
Grandchild edges	$\{\text{out\_edge}(c),$ $(e, \text{out\_edge}(c))\}$
Grandchild count	$\{\text{count}\,(\text{nbr}(e_{\text{child}}))$ $(e, \text{count}\,(\text{nbr}(e_{\text{child}})))\}$
Has subject/object	$\forall_{e \in \{e, e_{\text{child}}\}} \forall_{l \in \{\mathit{subj}, \mathit{obj}\}}$ $\mathbb{1}(l \in \text{nbr}(e))$
POS tag signature	$\{\text{pos}(p), \text{pos}(c),$ $(\text{pos}(p), \text{pos}(c))\}$
Features at root	$\{\mathbb{1}(p = root), \text{POS}(p)\}$

Table 1: Features for the clause splitter model, deciding to split on the arc $e = p \xrightarrow{l} c$. The feature class is a high level description of features; the feature templates are the particular templates used. For instance, the POS signature contains the tag of the parent, the tag of the child, and both tags joined in a single feature. Note that all features are joined with the action to be taken on the parent.

ginning at the root of the tree, we consider every outgoing edge. For every possible action to be performed on the parent (i.e., clone subject, clone root, no action), we apply our trained classifier to determine whether we (1) split the edge off as a clause, and recurse; (2) do not split the edge, and recurse; or (3) do not recurse. In the first two cases, we recurse on the child of the arc, and continue until either all arcs have been exhausted, or all remaining candidate arcs have been marked as not recursable.

We will use the scores from this classifier to inform the score assigned to our generated open IE extractions (Section 4). The score of a clause is the product of the scores of actions taken to reach the clause. The score of an extraction will be this score multiplied by the score of the extraction given the clause.

4 Intra-Clause Open IE

We now turn to the task of generating a maximally compact sentence which retains the core semantics of the original utterance, and parsing the sentence into a conventional open IE subject verb object triple. This is often a key component in downstream applications, where extractions need to be not only *correct*, but also *informative*. Whereas an argument like *Heinz Fischer of Austria* is often correct, a downstream application must apply further processing to recover information about either *Heinz Fischer*, or *Austria*. Moreover, it must do so without the ability to appeal to the larger context of the sentence.

4.1 Validating Deletions with Natural Logic

We adopt a subset of natural logic semantics dictating contexts in which lexical items can be removed. Natural logic as a formalism captures common logical inferences appealing directly to the form of language, rather than parsing to a specialized logical syntax. It provides a proof theory for lexical mutations to a sentence which either preserve or negate the truth of the premise.

For instance, if *all rabbits eat vegetables* then *all cute rabbits eat vegetables*, since we are allowed to mutate the lexical item *rabbit* to *cute rabbit*. This is done by observing that *rabbit* is in scope of the first argument to the operator *all*. Since *all* induces a *downward polarity* environment for its first argument, we are allowed to replace *rabbit* with an item which is more specific – in this case *cute rabbit*. To contrast, the operator *some* induces an *upward polarity* environment for its first argument, and therefore we may derive the inference from *cute rabbit* to *rabbit* in: *some cute rabbits are small* therefore *some rabbits are small*. For a more comprehensive introduction to natural logic, see van Benthem (2008).

We mark the scopes of all operators (*all*, *no*, *many*, etc.) in a sentence, and from this determine whether every lexical item can be replaced by something more general (has upward polarity), more specific (downward polarity), or neither. In the absence of operators, all items have upwards polarity.

Each dependency arc is then classified into whether deleting the dependent of that arc makes the governing constituent at that node more general, more specific (a rare case), or neither.[2] For example, removing the *amod* edge in *cute* $\xleftarrow{amod}$ *rabbit* yields the more general lexical item *rabbit*. However, removing the *nsubj* edge in *Fido* $\xleftarrow{nsubj}$ *runs* would yield the unentailed (and nonsensical) phrase *runs*. The last, rare, case is an edge that causes the resulting item to be more specific – e.g., *quantmod*: *about* $\xleftarrow{quantmod}$ *200* is more general than *200*.

[2] We use the Stanford Dependencies representation (de Marneffe and Manning, 2008).

For most dependencies, this semantics can be hard-coded with high accuracy. However, there are at least two cases where more attention is warranted. The first of these concerns non-subsective adjectives: for example a *fake gun* is not a gun. For this case, we make use of the list of non-subsective adjectives collected in Nayak et al. (2014), and prohibit their deletion as a hard constraint.

The second concern is with prepositional attachment, and direct object edges. For example, whereas *Alice went to the playground* $\xrightarrow{prep\_with}$ *Bob* entails that *Alice went to the playground*, it is not meaningful to infer that *Alice is friends* $\xrightarrow{prep\_with}$ *Bob* entails *Alice is friends*. Analogously, *Alice played* $\xrightarrow{dobj}$ *baseball on Sunday* entails that *Alice played on Sunday*; but, *Obama signed* $\xrightarrow{dobj}$ *the bill on Sunday* should not entail the awkward phrase **Obama signed on Sunday*.

We learn these attachment affinities empirically from the syntactic n-grams corpus of Goldberg and Orwant (2013). This gives us counts for how often object and preposition edges occur in the context of the governing verb and relevant neighboring edges. We hypothesize that edges which are frequently seen to co-occur are likely to be essential to the meaning of the sentence. To this end, we compute the probability of seeing an arc of a given type, conditioned on the most specific context we have statistics for. These contexts, and the order we back off to more general contexts, is given in Figure 3.

To compute a score s of *deleting* the edge from the affinity probability p collected from the syntactic n-grams, we simply cap the affinity and subtract it from 1:

$$s = 1 - \min(1, \frac{p}{K})$$

where K is a hyperparameter denoting the minimum fraction of the time an edge should occur in a context to be considered entirely unremovable. In our experiments, we set $K = \frac{1}{3}$.

The score of an extraction, then, is the product of the scores of each deletion multiplied by the score from the clause splitting step in Section 3.

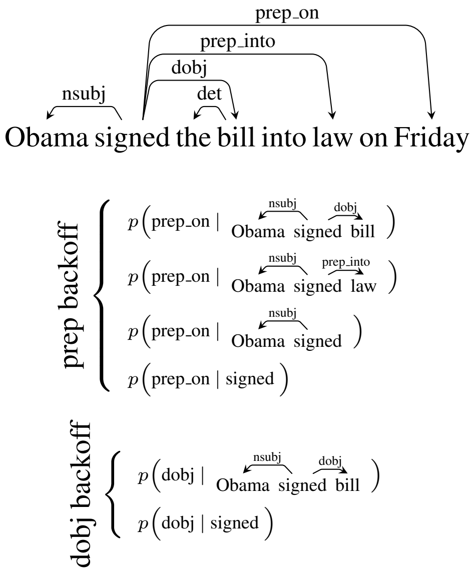
prep_on
prep_into
dobj
det
nsubj
Obama signed the bill into law on Friday
prep backoff
p(prep_on | Obama signed bill)
nsubj
dobj
p(prep_on | Obama signed law)
nsubj
prep_into
p(prep_on | Obama signed)
nsubj
p(prep_on | signed)
dobj backoff
p(dobj | Obama signed bill)
nsubj
dobj
p(dobj | signed)

Figure 3: The ordered list of backoff probabilities when deciding to drop a prepositional phrase or direct object. The most specific context is chosen for which an empirical probability exists; if no context is found then we allow dropping prepositional phrases and disallow dropping direct objects. Note that this backoff arbitrarily orders contexts of the same size.

Input	**Extraction**
cats play with yarn	(cats; play with; yarn)
fish like to swim	(fish; like to; swim)
cats have tails	(cats; have; tails)
cats are cute	(cats; are; cute)
Tom and Jerry are fighting	(Tom; fighting; Jerry)
There are cats with tails	(cats; have; tails)

Table 2: The six dependency patterns used to segment an atomic sentence into an open IE triple.

4.2 Atomic Patterns

Once a set of short entailed sentences is produced, it becomes straightforward to segment them into conventional open IE triples. We employ 6 simple dependency patterns, given in Table 2, which cover the majority of atomic relations we are interested in.

When information is available to disambiguate the substructure of compound nouns (e.g., named entity segmentation), we extract additional relations with 5 dependency and 3 TokensRegex (Chang and Manning, 2014) surface form patterns. These are given in Table 3; we refer to these as *nominal relations*. Note that the constraint of named entity information is by no means required for the system. In other applications – for example, applications in vision – the otherwise trivial nominal relations could be quite useful.

KBP Relation	Open IE Relation	PMI^2	KBP Relation	Open IE Relation	PMI^2
Org:Founded	*found in*	1.17	Per:Date_Of_Birth	*be bear on*	1.83
	be found in	1.15		*bear on*	1.28
Org:Dissolved	**buy Chrysler in*	0.95	Per:Date_Of_Death	*die on*	0.70
	**membership in*	0.60		*be assassinate on*	0.65
Org:LOC_Of_HQ	*in*	2.12	Per:LOC_Of_Birth	*be bear in*	1.21
	base in	1.82	Per:LOC_Of_Death	**elect president of*	2.89
Org:Member_Of	**tough away game in*	1.80	Per:Religion	*speak about*	0.67
	**away game in*	1.80		*popular for*	0.60
Org:Parents	*'s bank*	1.65	Per:Parents	*daughter of*	0.54
	**also add to*	1.52		*son of*	1.52
Org:Founded_By	*invest fund of*	1.48	Per:LOC_Residence	*of*	1.48
	own stake besides	1.18		**independent from*	1.18

Table 4: A selection of the mapping from KBP to lemmatized open IE relations, conditioned on the types of the arguments being correct. The top one or two relations are shown for 7 person and 6 organization relations. Incorrect or dubious mappings are marked with an asterisk.

Input	Extraction
Durin, son of Thorin	(Durin; is son of; Thorin)
Thorin's son, Durin	(Thorin; 's son; Durin)
IBM CEO Rometty	(Rometty; is CEO of; IBM)
President Obama	(Obama; is; President)
Fischer of Austria	(Fischer; is of; Austria)
IBM's research group	(IBM; 's; research group)
US president Obama	(Obama; president of; US)
Our president, Obama,	(Our president; be; Obama)

Table 3: The eight patterns used to segment a noun phrase into an open IE triple. The first five are dependency patterns; the last three are surface patterns.

5 Mapping OpenIE to a Known Relation Schema

A common use case for open IE systems is to map them to a known relation schema. This can either be done manually with minimal annotation effort, or automatically from available training data. We use both methods in our TAC-KBP evaluation. A collection of relation mappings was constructed by a single annotator in approximately a day,[3] and a relation mapping was learned using the procedure described in this section.

We map open IE relations to the KBP schema by searching for co-occurring relations in a large distantly-labeled corpus, and marking open IE and KBP relation pairs which have a high PMI^2 value (Béatrice, 1994; Evert, 2005) conditioned on their type signatures matching. To compute PMI^2, we collect probabilities for the open IE and KBP relation co-occurring, the probability of the open IE relation occurring, and the probability of the KBP relation occurring. Each of these probabilities is conditioned on the type signature of the relation. For example, the joint probability of KBP relation r_k and open IE relation r_o, given a type signature of t_1, t_2, would be

$$p(r_k, r_o \mid t_1, t_2) = \frac{\text{count}(r_k, r_o, t_1, t_2)}{\sum_{r'_k, r'_o} \text{count}(r'_k, r'_o, t_1, t_2)}.$$

Omitting the conditioning on the type signature for notational convenience, and defining $p(r_k)$ and $p(r_o)$ analogously, we can then compute The PMI^2 value between the two relations:

$$\text{PMI}^2(r_k, r_o) = \log\left(\frac{p(r_k, r_o)^2}{p(r_k) \cdot p(r_o)}\right)$$

Note that in addition to being a measure related to PMI, this captures a notion similar to *alignment by agreement* (Liang et al., 2006); the formula can be equivalently written as $\log\left[p(r_k \mid r_o)p(r_o \mid r_k)\right]$. It is also functionally the same as the JC WordNet distance measure (Jiang and Conrath, 1997).

Some sample type checked relation mappings are given in Table 4. In addition to intuitive mappings (e.g., *found in* → Org:Founded), we can note some rare, but high precision pairs (e.g., *invest fund of* → Org:Founded_By). We can also see

[3] The official submission we compare against claimed two weeks for constructing their manual mapping, although a version of their system constructed in only 3 hours performs nearly as well.

the noise in distant supervision occasionally permeate the mapping, e.g., with *elect president of* → Per:LOC_Of_Death – a president is likely to die in his own country.

6 Evaluation

We evaluate our approach in the context of a real-world end-to-end relation extraction task – the TAC KBP Slot Filling challenge. In Slot Filling, we are given a large unlabeled corpus of text, a fixed schema of relations (see Section 5), and a set of query entities. The task is to find all relation triples in the corpus that have as a subject the query entity, and as a relation one of the defined relations. This can be viewed intuitively as populating Wikipedia Infoboxes from a large unstructured corpus of text.

We compare our approach to the University of Washington submission to TAC-KBP 2013 (Soderland et al., 2013). Their system used OpenIE v4.0 (a successor to Ollie) run over the KBP corpus and then they generated a mapping from the extracted relations to the fixed schema. Unlike our system, Open IE v4.0 employs a semantic role component extracting structured SRL frames, alongside a conventional open IE system. Furthermore, the UW submission allows for extracting relations and entities from substrings of an open IE triple argument. For example, from the triple *(Smith; was appointed; acting director of Acme Corporation)*, they extract that Smith is employed by Acme Corporation. We disallow such extractions, passing the burden of finding correct precise extractions to the open IE system itself (see Section 4).

For entity linking, the UW submission uses Tom Lin's entity linker (Lin et al., 2012); our submission uses the Illinois Wikifier (Ratinov et al., 2011) without the relational inference component, for efficiency. For coreference, UW uses the Stanford coreference system (Lee et al., 2011); we employ a variant of the simple coref system described in (Pink et al., 2014).

We report our results in Table 5.[4] UW Official refers to the official submission in the 2013 challenge; we show a 3.1 F_1 improvement (to 22.7 F_1) over this submission, evaluated using a comparable approach. A common technique in KBP systems but not employed by the official UW submission in 2013 is to add alternate names based on entity linking and coreference. Additionally, websites are often extracted using heuristic name-matching as they are hard to capture with traditional relation extraction techniques. If we make use of both of these, our end-to-end accuracy becomes 28.2 F_1.

System	**P**	**R**	**F_1**
UW Official*	**69.8**	11.4	19.6
Ollie†	57.4	4.8	8.9
+ Nominal Rels*	57.7	11.8	19.6
Our System			
- Nominal Rels†	64.3	8.6	15.2
+ Nominal Rels*	61.9	13.9	22.7
+ Alt. Name	57.8	17.8	27.1
+ Alt. Name + Website	58.6	**18.6**	**28.3**

Table 5: A summary of our results on the end-to-end KBP Slot Filling task. UW official is the submission made to the 2013 challenge. The second row is the accuracy of Ollie embedded in our framework, and of Ollie evaluated with nominal relations from our system. Lastly, we report our system, our system with nominal relations removed, and our system combined with an alternate names detector and rule-based website detector. Comparable systems are marked with a dagger† or asterisk*.

We attempt to remove the variance in scores from the influence of other components in an end-to-end KBP system. We ran the Ollie open IE system (Mausam et al., 2012) in an identical framework to ours, and report accuracy in Table 5. Note that when an argument to an Ollie extraction contains a named entity, we take the argument to be that named entity. The low performance of this system can be partially attributed to its inability to extract nominal relations. To normalize for this, we report results when the Ollie extractions are supplemented with the nominal relations produced by our system (Ollie + Nominal Rels in Table 5). Conversely, we can remove the nominal relation extractions from our system; in both cases we outperform Ollie on the task.

[4] All results are reported with the `anydoc` flag set to true in the evaluation script, meaning that only the truth of the extracted knowledge base entry and not the associated provenance is scored. In absence of human evaluators, this is in order to not penalize our system unfairly for extracting a new correct provenance.

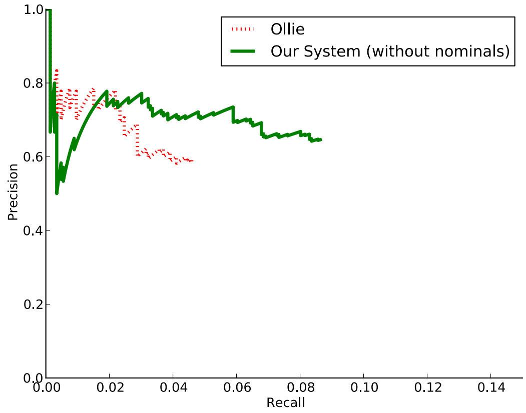

Figure 4: A precision/recall curve for Ollie and our system (without nominals). For clarity, recall is plotted on a range from 0 to 0.15.

6.1 Discussion

We plot a precision/recall curve of our extractions in Figure 4 in order to get an informal sense of the calibration of our confidence estimates. Since confidences only apply to standard extractions, we plot the curves without including any of the nominal relations. The confidence of a KBP extraction in our system is calculated as the sum of the confidences of the open IE extractions that support it. So, for instance, if we find (Obama; be bear in; Hawaii) n times with confidences $c_1 \dots c_n$, the confidence of the KBP extraction would be $\sum_{i=0}^{n} c_i$. It is therefore important to note that the curve in Figure 4 necessarily conflates the confidences of individual extractions, and the frequency of an extraction.

With this in mind, the curves lend some interesting insights. Although our system is very high precision on the most confident extractions, it has a large dip in precision early in the curve. This suggests that the model is extracting multiple instances of a bad relation. Systematic errors in the clause splitter are the likely cause of these errors. While the approach of splitting sentences into clauses generalizes better to out-of-domain text, it is reasonable that the errors made in the clause splitter manifest across a range of sentences more often than the fine-grained patterns of Ollie would.

On the right half of the PR curve, however, our system achieves both higher precision and extends to a higher recall than Ollie. Furthermore, the curve is relatively smooth near the tail, suggesting that indeed we are learning a reasonable estimate of confidence for extractions that have only one supporting instance in the text – empirically, 46% of our extractions.

In total, we extract 42 662 862 open IE triples which link to a pair of entities in the corpus (i.e., are candidate KBP extractions), covering 1 180 770 relation types. 202 797 of these relation types appear in more than 10 extraction instances; 28 782 in more than 100 instances, and 4079 in more than 1000 instances. 308 293 relation types appear only once. Note that our system over-produces extractions when both a general and specific extraction are warranted; therefore these numbers are an overestimate of the number of semantically meaningful facts.

For comparison, Ollie extracted 12 274 319 triples, covering 2 873 239 relation types. 1 983 300 of these appeared only once; 69 010 appeared in more than 10 instances, 7951 in more than 100 instances, and 870 in more than 1000 instances.

7 Conclusion

We have presented a system for extracting open domain relation triples by breaking a long sentence into short, coherent clauses, and then finding the maximally simple relation triples which are warranted given each of these clauses. This allows the system to have a greater awareness of the context of each extraction, and to provide informative triples to downstream applications. We show that our approach performs well on one such downstream application: the KBP Slot Filling task.

Acknowledgments

We thank the anonymous reviewers for their thoughtful feedback. Stanford University gratefully acknowledges the support of a Natural Language Understanding-focused gift from Google Inc. and 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 D. Manning. 2014. Naturalli: Natural logic inference for common sense reasoning. In *EMNLP*.

Gabor Angeli, Julie Tibshirani, Jean Y. Wu, and Christopher D. Manning. 2014. Combining distant and partial supervision for relation extraction. In *EMNLP*.

DAILLE Béatrice. 1994. *Approche mixte pour l'extraction automatique de terminologie: statistique lexicale et filtres linguistiques*. Ph.D. thesis, Thèse de Doctorat. Université de Paris VII.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2011. Global learning of typed entailment rules. In *Proceedings of ACL*, Portland, OR.

Samuel R. Bowman, Christopher Potts, and Christopher D. Manning. 2014. Recursive neural networks can learn logical semantics. *CoRR*, (arXiv:1406.1827).

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*.

Angel X. Chang and Christopher D. Manning. 2014. TokensRegex: Defining cascaded regular expressions over tokens. Technical Report CSTR 2014-02, Department of Computer Science, Stanford University.

Zheng Chen, Suzanne Tamang, Adam Lee, Xiang Li, Wen-Pin Lin, Matthew Snover, Javier Artiles, Marissa Passantino, and Heng Ji. 2010. CUNY-BLENDER. In *TAC-KBP*.

Mark Craven and Johan Kumlien. 1999. Constructing biological knowledge bases by extracting information from text sources. In *AAAI*.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representation. In *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*.

Oren Etzioni. 2011. Search needs a shake-up. *Nature*, 476(7358):25–26.

Stefan Evert. 2005. *The statistics of word cooccurrences: word pairs and collocations*. Ph.D. thesis, Universit at Stuttgart.

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*.

Yoav Goldberg and Jon Orwant. 2013. A dataset of syntactic-ngrams over time from a very large corpus of english books. In **SEM*.

Ralph Grishman and Bonan Min. 2010. New York University KBP 2010 slot-filling system. In *Proc. TAC 2010 Workshop*.

Thomas Icard, III and Lawrence Moss. 2014. Recent progress on monotonicity. *Linguistic Issues in Language Technology*.

Heng Ji, Ralph Grishman, Hoa Trang Dang, Kira Griffitt, and Joe Ellis. 2010. Overview of the tac 2010 knowledge base population track. In *Third Text Analysis Conference*.

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*.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2011. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *CoNLL Shared Task*.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *NAACL-HLT*.

Thomas Lin, Mausam, and Oren Etzioni. 2012. No noun phrase left behind: detecting and typing unlinkable entities. In *EMNLP-CoNLL*.

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*.

Filipe Mesquita, Jordan Schmidek, and Denilson Barbosa. 2013. Effectiveness and efficiency of open relation extraction. In *EMNLP*.

Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. 2013. Distant supervision for relation extraction with an incomplete knowledge base. In *NAACL-HLT*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL*.

Neha Nayak, Mark Kowarsky, Gabor Angeli, and Christopher D. Manning. 2014. A dictionary of nonsubsective adjectives. Technical Report CSTR 2014-04, Department of Computer Science, Stanford University, October.

Glen Pink, Joel Nothman, and R. James Curran. 2014. Analysing recall loss in named entity slot filling. In *EMNLP*.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *ACL*.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *NAACL-HLT*.

Lorenza Romano, Milen Kouylekov, Idan Szpektor, Ido Dagan, and Alberto Lavelli. 2006. Investigating a generic paraphrase-based approach for relation extraction. *EACL*.

Víctor Manuel Sánchez Sánchez Valencia. 1991. *Studies on natural logic and categorial grammar*. Ph.D. thesis, University of Amsterdam.

Stephen Soderland, Brendan Roof, Bo Qin, Shi Xu, Mausam, and Oren Etzioni. 2010. Adapting open information extraction to domain-specific relations. *AI Magazine*, 31(3):93–102.

Stephen Soderland, John Gilmer, Robert Bart, Oren Etzioni, and Daniel S. Weld. 2013. Open information extraction to KBP relations in 3 hours. In *Text Analysis Conference*.

Stephen G Soderland. 1997. *Learning text analysis rules for domain-specific natural language processing*. Ph.D. thesis, University of Massachusetts.

Ang Sun, Ralph Grishman, Wei Xu, and Bonan Min. 2011. New York University 2011 system for KBP slot filling. In *Proceedings of the Text Analytics Conference*.

Mihai Surdeanu. 2013. Overview of the tac2013 knowledge base population evaluation: English slot filling and temporal slot filling. In *Sixth Text Analysis Conference*.

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

Yotaro Watanabe, Junta Mizuno, Eric Nichols, Naoaki Okazaki, and Kentaro Inui. 2012. A latent discriminative model for compositional entailment relation recognition using natural logic. In *COLING*.

Fei Wu and Daniel S Weld. 2007. Autonomously semantifying wikipedia. In *Proceedings of the sixteenth ACM conference on information and knowledge management*. ACM.

Fei Wu and Daniel S Weld. 2010. Open information extraction using wikipedia. In *ACL*. Association for Computational Linguistics.

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*.