Grounded Semantic Parsing for Complex Knowledge Extraction

Ankur P. Parikh∗
School of Computer Science
Carnegie Mellon University
apparikh@cs.cmu.edu

Hoifung Poon Kristina Toutanova
Microsoft Research
Redmond, WA, USA
hoifung,kristout@microsoft.com

Abstract

Recently, there has been increasing interest in learning semantic parsers with indirect supervision, but existing work focuses almost exclusively on question answering. Separately, there have been active pursuits in leveraging databases for distant supervision in information extraction, yet such methods are often limited to binary relations and none can handle nested events. In this paper, we generalize distant supervision to complex knowledge extraction, by proposing the first approach to learn a semantic parser for extracting nested event structures without annotated examples, using only a database of such complex events and unannotated text. The key idea is to model the annotations as latent variables, and incorporate a prior that favors semantic parses containing known events. Experiments on the GENIA event extraction dataset show that our approach can learn from and extract complex biological pathway events. Moreover, when supplied with just five example words per event type, it becomes competitive even among supervised systems, outperforming 19 out of 24 teams that participated in the original shared task.

1 Introduction

The goal of semantic parsing is to map text into a complete and detailed meaning representation (Mooney, 2007). Supervised approaches for learning a semantic parser require annotated examples, which are expensive and time-consuming to acquire (Zelle and Mooney, 1993; Zettlemoyer and Collins, 2005; Zettlemoyer and Collins, 2007). As a result, there has been rising interest in learning semantic parsers from indirect supervision. Examples include unsupervised approaches that leverage distributional similarity by recursive clustering (Poon and Domingos, 2009; Poon and Domingos, 2010; Titov and Klementiev, 2011), semi-supervised approaches that learn from dialog context (Artzi and Zettlemoyer, 2011), grounded approaches that learn from annotated question-answer pairs (Clarke et al., 2010; Liang et al., 2011) or virtual worlds (Chen and Mooney, 2011; Artzi and Zettlemoyer, 2013).

Such progress is exciting, but most applications focus on question answering, where the semantic parser is used to convert natural-language questions into formal queries. In contrast, complex knowledge extraction represents a relatively untapped ap-
Figure 2: Grounded semantic parsing for complex knowledge extraction: (a) input database of complex events, without textual annotation; (b) event extraction as semantic parsing; (c) a complex sentence that requires RAISING.

This paper proposes the first approach to learn a semantic parser from a database of complex events and unannotated text, by generalizing distant supervision to complex knowledge extraction. The key idea is to recover the latent annotations via EM, guided by a structured prior that favors semantic parses containing known events in the database, in the form of virtual evidence (Pearl, 1988; Subramanya and Bilmes, 2007). Experiments on the GENIA dataset demonstrate the promise of this direction. Our GUSPEE (GroUnded Semantic Parsing for Event Extraction) system can successfully learn from and extract complex events, without requiring annotation of example sentences.
textual annotations (Figure 2). Moreover, after incor-
porating prototype-driven learning using just five
example words for each event type, GUSPEE be-
comes competitive even among supervised systems,
outperforming 19 out of 24 teams that participated
in the GENIA event extraction shared task. With
significant information loss (skipping event triggers
and, most importantly, the nested event structures),
it is possible to reduce GENIA events to binary re-
lations so that existing distant-supervision methods
are applicable. Yet even in such an evaluation tai-
lored for existing methods, our system still outper-
formed them by a wide margin.

2 Related Work

Existing approaches for GENIA event extraction are
supervised methods that either used a carefully en-
geineered classification pipeline (Bjorne et al., 2009;
Quirk et al., 2011) or applied joint inference (Riedel
et al., 2009; Poon and Vanderwende, 2010; Riedel
and McCallum, 2011). Poon and Vanderwende
(2010) used a dependency-based formulation that re-
sembled our semantic parsing one, but learned from
supervised data. Classification approaches first need
to classify words into event triggers, where distant
supervision is not directly applicable.

In distant supervision (Craven and Kumlien,
1999; Mintz et al., 2009), if two entities are known
to have a binary relation in the database, their co-
ocurrence in a sentence justifies labeling the in-
stance with the relation. This assumption is often
incorrect, and Riedel et al. (2010) introduced latent
variables to model the uncertainty; the model was
later improved by Hoffmann et al. (2011). GUS-
PEE generalizes this idea to structured prediction
where the latent annotations are not simple classifi-
cation decisions, but nested events. Krishnamurthy
and Mitchell (2012) and Reddy et al. (2014) took
an important step toward this direction, by learning
a semantic parser based on combinatorial categorial
grammar (CCG) from Freebase and web sentences.
However, Krishnamurthy and Mitchell (2012) still
learned from binary relations, using only simple sen-
tences (of length ten or less). Reddy et al. (2014)
learned from n-ary relations as well, yet their for-
mulation only allows relations between entities, not
relations between relations. Thus their approach
cannot represent nested events, let alone extract-
ing them. And like Krishnamurthy and Mitchell
(2012), Reddy et al. (2014) focused on simple text
and excluded sentences where entities were not de-
pendency neighbors (i.e., not directly connected in
the ungrounded graph), as well as sentences with un-
known entities. While such restrictions do not im-
pede parsing simple questions in their evaluation,
their approach is not directly applicable to com-
plex knowledge extraction. Reschke et al. (2014)
also generalized distant supervision to n-ary rela-
tions for extracting template-based events, but simi-
lar to Reddy et al. (2014), they did not consider
nested events.

Distant supervision can be viewed as a special
case of the more general paradigm of grounded
learning from a database. Clarke et al. (2010) and
Liang et al. (2011) used the database to determine
if a candidate semantic parse would yield the an-
notated answer, whereas distant supervision uses the
database to determine if a relation instance is con-
tained therein. Our GUSPEE system is inspired
by grounded unsupervised semantic parsing (GUSP)
(Poon, 2013) and shares a similar semantic repre-
sentation. GUSP, like most grounded learning ap-
proaches, applied to question answering and did not
leverage distant supervision. GUSPEE can thus be
viewed as an extension of GUSP to leverage distant
supervision for complex knowledge extraction.

Grounding in GUSPEE is materialized by virtual
evidence favoring semantic structures that conform
with the database. The idea of virtual evidence was
first introduced by Pearl (1988) and later applied in
several applications such as Subramanya and Bilmes
(2007). Unlike in prior work, the virtual evidence
in GUSPEE involves non-local factors (comparing a
semantic parse with complex events in the database)
and presents a major challenge to efficient learning.

Existing semantic parsers often adopt highly ex-
pressive formalisms such as CCG (Steedman, 2000).
Such formalisms are extremely powerful, but also
difficult to learn. We instead adopted a dependency-
based formalism (Poon and Domingos, 2009; Liang
et al., 2011; Poon, 2013). Moreover, following Poon
and Domingos (2009), Krishnamurthy and Mitchell
(2012), Poon (2013), we started with syntactic de-
pendency parses, and focused on annotating nodes
and edges with semantic states.
3 Grounded Semantic Parsing for Event Extraction

We use the GENIA event extraction task (Kim et al., 2009) as a representative example of complex knowledge extraction. The goal is to identify biological events from text, including the trigger words and arguments (Figure 1, bottom). There are nine event types, including simple ones such as Expression and Transcription that can only have one THEME argument, Binding that can have more than one THEME argument, and regulations that can have both THEME and CAUSE arguments. Protein annotations are given as input.

We formulate this task as semantic parsing and present our GUSPEE system (Figure 2). The core of GUSPEE is a tree HMM (Section 3.1), which extracts events from a sentence by annotating its syntactic dependency tree with event and argument states. In training, GUSPEE takes as input unannotated text and a database of complex events, and learns the tree HMM using EM, guided by grounded learning from the database via virtual evidence.

3.1 Problem Formulation

Let \( t \) be a syntactic dependency tree for a sentence, with nodes \( n_i \) and dependency edges \( d_{i,j} \) (\( n_j \) is a child of \( n_i \)). A semantic parse of \( t \) is an assignment \( z \) that maps each node to an event state and each dependency to an argument state. The semantic state of a protein word is fixed to that protein annotation. Basic event states are the nine event types and NULL (signifying a non-event, e.g., “The” in Figure 2(c)). Basic argument states are THEME, CAUSE, and NULL. Additional states will be introduced later in Section 3.2 and 3.4.

GUSPEE models \( z, t \) by a tree HMM:

\[
P_\theta(z, t) = \prod_m P_{\text{EMIT}}(t_m | z_m, \theta) \cdot P_{\text{TRANS}}(z_m | z_{\pi(m)} \cdot \theta)
\]

where \( \theta \) are the emission and transition parameters, \( m \) ranges over the nodes and dependency edges, \( \pi(n_j) = d_{i,j} \) and \( \pi(d_{i,j}) = n_i \). Note that this formulation implicitly assumes a fixed underlying directed tree, while the words and dependencies may vary.

Semantic parsing finds the most probable semantic assignment given the dependency tree:

\[
z^* = \arg \max_z \log P_\theta(z | t) = \arg \max_z \log P_\theta(z, t)
\]

In training, GUSPEE takes as input a set of complex events (database \( K \)) and syntactic dependency trees (unannotated text \( T \)), and maximizes the likelihood of \( T \) augmented by virtual evidence \( \phi_K(z) \).

\[
\theta^* = \arg \max_\theta \log P_\theta(T | K) = \arg \max_\theta \sum_{t \in T} \log \sum_z P_\theta(z, t) \cdot \phi_K(z)
\]

Virtual evidence is analogous to a Bayesian prior, but applies to variable states rather than model parameters (Subramanya and Bilmes, 2007).

3.2 Handling Syntax-Semantics Mismatch

For simple sentences such as the one in Figure 2(b), the complex event can be represented by a semantic parse using only basic states. In general, however, syntax and semantics often diverge. For example, in Figure 2(c), “requires” triggers the top \text{POS–NEG} event that has a THEME argument triggered by “block”, but “ability” stands in between the two; likewise for “block” and “IL-10”. Additionally, mismatch could stem from errors in the syntactic parse. In such cases, the correct semantic parse can no longer be represented by basic states alone. Following GUSP (Poon, 2013), we introduced a new argument state RAISING which, if assigned to a dependency, would require that the parent and child be assigned the same basic event state. We also introduce a corresponding RAISE version for each non-null event state, to signify that the word derives its basic state from RAISING of a child. RAISING is related to but not identical with type raising in CCG and other grammars. For simplicity, we did not use other complex states explored in Poon (2013).

3.3 Virtual Evidence for Grounded Learning

Grounded learning in GUSPEE is attained by incorporating the virtual evidence \( \phi_K(z) \), which favors the \( z \)'s containing known events in \( K \) and penalizes those containing unknown events. Intuitively, this can be accomplished by identifying events in \( z \) and comparing them with events in \( K \). But this is not robust as individual events and mentions may be fragmental and incomplete. Insisting on matching an event in full would miss partial matches that still convey valuable supervision. Proteins are given as
input and can be mapped to event arguments a priori. Matching sub-events with only one protein argument would be too noisy without direct supervision on triggers. We thus consider matching minimum sub-events with two protein arguments.

Specifically, we preprocessed complex events in $K$ to identify minimum logical forms containing two protein arguments from each complex event, where arguments not directly leading to either protein are skipped. For example, the complex event in Figure 1 would generate three sub-events: 

- $(\text{NEG-REG, IL-10, RFLAT})$,
- $(\text{POS-REG, BCL, (NEG-REG, -, RFLAT)})$,
- $(\text{POS-REG, BCL, (NEG-REG, IL-10, -)})$,

where $-$ signifies underspecification. We denote the set of such sub-events as $S(K)$.

Likewise, given a semantic parse $z$, for every protein pair in $z$, we would convert the minimum semantic parse subtree spanning the two proteins into the canonical logical form and compare it with elements in $S(K)$. If the minimum subtree contains NULL, either in an event or argument state, it signifies a non-event and would be ignored. Otherwise, the canonical form is derived by collapsing RAISING states. For example, in both Figure 2 (b) and (c), the minimum subtree spanning the proteins IL-10 and RFLAT is converted into the same logical form of $(\text{NEG-REG, IL-10, RFLAT})$. We denote the set of such logical forms as $E(z)$.

Formally, the virtual evidence in GUSPEE are:

$$\phi_K(z) = \exp \sum_{e \in E(z)} \sigma(e, K)$$

where

$$\sigma(e, K) = \begin{cases} \kappa & : e \in S(K) \\ -\kappa & : e \notin S(K) \end{cases}$$

In distant supervision, where $z$ is simply a binary relation, it is trivial to evaluate $\phi_K(z)$. (In fact, the original distant supervision algorithm is exactly equivalent to this form, with $\kappa = \infty$.) In GUSPEE, however, $z$ is a semantic parse and evaluating $E(z)$ and $\sigma(e, K)$ involves a global factor that does not decompose into local dependencies as the tree HMM $P_{\theta}(z, t)$. The naive way to compute the augmented likelihood (Section 3.1) is thus intractable.

### 3.4 Efficient Learning with Virtual Evidence

To render learning tractable, the key idea is to augment the local event and argument states so that they contain sufficient information for evaluating $\phi_K(z)$. Specifically, the semantic state $z(n_i)$ needs to represent not only the semantic assignment to $n_i$ (e.g., a NEG-REG event trigger), but also the set of (possibly incomplete) sub-events in the subtree under $n_i$. We accomplished this by representing the semantic paths from $n_i$ to proteins in the subtree. For example, in Figure 2 (b), the augmented state of “inhibition” would be $(\text{NEG-REG} \rightarrow \text{THEME} \rightarrow \text{RFLAT}, \text{NEG-REG} \rightarrow \text{CAUSE} \rightarrow \text{IL-10})$. To facilitate canonicalization and sub-event comparison, a path containing NULL will be skipped, and RAISING will be collapsed. E.g., in Figure 2(c), the augmented state of “ability” would become $(\text{NEG-REG} \rightarrow \text{THEME} \rightarrow \text{RFLAT}, \text{NEG-REG} \rightarrow \text{CAUSE} \rightarrow \text{IL-10})$.

With these augmented states, $\phi_K(z)$ decomposes into local factors. The proteins under $n_i$ are known a priori, as well as the children containing them. Semantic paths from $n_i$ to proteins can thus be computed by imposing consistency constraints for each child. Namely, for child $n_{i,j}$ that contains protein $p$, the semantic path from $n_i$ to $p$ should result from combining $z(n_i), z(d_{i,j}),$ and the semantic path from $n_j$ to $p$. The minimum sub-events spanning two proteins under $n_i$, if any, can be derived from the semantic paths in the augmented state. Note that if both proteins come from the same child $n_{i,j}$, the pair needs not be considered at $n_i$, as their minimum spanning sub-event, if any, would be under $n_j$ and already be factored in there.

The number of augmented states is $O(s^p)$, and the number of sub-event evaluations is $O(s \cdot p^2)$, where $s$ is the number of distinct semantic paths, and $p$ is the number of proteins in the subtree. Below, we show how $s, p$ can be constrained to reasonable ranges to make computation efficient.

First, consider $s$. The number of semantic paths is theoretically unbounded since a path can be arbitrarily long. However, semantic paths contained in a database event are bounded in length and can be precomputed from the database (the maximum in GENIA is four). Longer paths can be represented by a special dummy path signifying that they
would not match any database events. Likewise, certain sub-paths would not occur in database events. E.g., in GENIA, simple events cannot take events as arguments, so paths containing sub-paths such as Expression → Transcription are also illegitimate and can be represented same as the above.

We also notice that for regulation events with other regulation events as arguments, the semantics can be compressed into a single regulation event, e.g., POS-REG → NEG-REG is semantically equivalent with NEG-REG, as the collective effect of a positive regulation on top of a negative one is negative. Therefore, when evaluating the semantic path from \( n_i \) to a protein during dynamic programming, we would collapse consecutive regulation events in the child path, if any. This further reduces the length of semantic paths to at most three (regulation - regulation - simple event - protein).

Next, we notice that \( p \) is bounded to begin with, but it could be quite large. When a sentence contains many proteins (i.e., large \( p \)), it often stems from conjunction of proteins, as in “TP53 regulates many downstream targets such as ABCB1, AFP, APC, ATF3, BAX”. All proteins in the conjunct play a similar role in their respective events, such as THEME in the above example among “ABCB1, AFP, APC, ATF3, BAX”, and so share the same semantic paths. Therefore, prior to learning, we preprocessed the sentences to condense each conjunct into a single effective protein node. We identified conjunction by Stanford dependencies (\( \text{conj}^+ \)). In GENIA, this reduces the maximum number of effective protein nodes to two for the vast majority of sentences (over 90%). Both representation and evaluation are now reasonably efficient. To further speed up learning, in our experiments we only trained on sentences with at most two effective protein nodes, as this already performed quite well. Training on GENIA took 1.5 hours and semantic parsing of a sentence took less than a second (with one i7 core at 2.4 GHz).

Unlike RAISING, the augmented states introduced in this section are specific to GENIA events. However, the rules to canonicalize states are general and can potentially be adapted to other domains. An alternative strategy to combat state explosion is by embedding the discrete states in a low-dimensional vector space (Socher et al., 2013), which is a direction for future research.

### 3.5 Features

The GUSPEE model uses log-linear models for the emission and transition probabilities and trains using feature-rich EM (Berg-Kirkpatrick et al., 2010). The features are:

- **Word emission** \( \mathbb{I}[\text{lemma} = l, z_m = n] \);
- **Dependency emission** \( \mathbb{I}[^{\\text{dependency}} = d, z_m = e] \) where \( e \notin \{\text{NULL}, \text{RAISE}\} \);
- **Transition** \( \mathbb{I}[z_m = a, z_{\pi(m)} = b] \) where \( a, b \notin \{\text{NULL}, \text{RAISE}\} \).

To modulate the model complexity, GUSPEE imposes a standard \( L_2 \) prior on the weights, and includes the following features with fixed weights:

- \( W_{\text{NULL}} \): apply to NULL states;
- \( W_{\text{RAISE}^{-\text{P}}} \): apply to protein RAISING;
- \( W_{\text{RAISE}^{-\text{E}}} \): apply to event RAISING.

The advantage of a feature-rich representation is flexibility in feature engineering. Here, we excluded NULL and RAISE in dependency emission and transition features, and regulated them separately to enable parameter tying for better generalization.

### 4 Experiments

#### 4.1 Evaluation on GENIA Event Extraction

In principle, we can learn GUSPEE from any pathway database. However, evaluation is challenging as these databases do not contain textual annotations. Prior work on distant supervision resorted to sampling and annotating new extractions. This is effective for comparing among distant-supervision systems, but it cannot be used to compare them with supervised learning. Moreover, as annotation is conducted by the authors or crowdsourcing, consistency and quality are hard to control.

We thus adopted a novel approach to evaluation by simulating a grounded learning scenario using the GENIA event extraction dataset (Kim et al., 2009). Specifically, we generated a set of complex events from the annotations of training sentences as the database. The annotations were discarded afterwards and GUSPEE learned from the database and unannotated text alone. The learned model was then applied to semantic parsing of test sentences and evaluated on event precision, recall, and F1.
Table 1: GENIA event extraction results of GUSPEE evaluation methodology enables us to assess the true accuracy and compare head-to-head with supervised methods.

GENIA contains 800 abstracts for training and 150 for development. It also has a test set, but its annotation is not made public. Therefore, we used the training set for grounded learning and development, and reserved the development set for testing. The majority events are Regulation (including Positive regulation, Negative regulation). See Kim et al. (2009) for details. We processed all sentences using SPLAT (Quirk et al., 2012), to conduct tokenization, part-of-speech tagging, and constituency parsing. We then postprocessed the parses to obtain Stanford dependencies (de Marneffe et al., 2006). During development on the training data, we found the following parameters (Section 3) to perform quite well and used them in all subsequent experiments: $\kappa = 20$, $W_{\text{NULL}} = 4$, $W_{\text{RAISE-P}} = 2$, $W_{\text{RAISE-E}} = -6$, $L_2$ prior = 0.1. Interestingly, we found that encouraging protein RAISING is beneficial, which probably stems from the fact that proteins are often separated from event triggers by noun modifiers, such as “the BCL gene”, “IL-10 protein”.

Table 1 shows GUSPEE’s results on GENIA event extraction. Note that this event-based evaluation is rather stringent, as it considers an event incorrect if one of its argument events is not completely correct, thus an incorrect event will render all its upstream events incorrect. See Kim et al. (2009) for details. For comparison, Table 2 shows the results of MSR11, a state-of-the-art supervised system. MSR11 also provides a upper bound for the supervised version of GUSPEE, as the latter is much less engineered.

Table 2: GENIA event extraction results of state-of-the-art supervised system MSR11 (Quirk et al., 2011).

Not surprisingly, grounded learning with GUSPEE still lags behind supervised learning. MSR11 used a rich set of features, including POS tags, linear and dependency n-grams, etc. Also, it is expected that indirect supervision do not provide as effective signals as direct supervision. However, the comparison reveals a particularly interesting contrast. Event types such as Expression, Catabolism, Phosphorylation, and Localization are relatively easy, yet GUSPEE performed rather poorly on them. Simple events do not admit multiple arguments, so they appear less often in the virtual evidence, and grounded learning has difficulty learning these event types, especially their triggers. In light of this, it’s actually remarkable that GUSPEE still learned a substantial portion of them.

### 4.2 Prototype-Driven Learning

While full-blown annotations are undoubtedly expensive and time-consuming to generate, it is rather easy for a domain expert to provide a few trigger words per event type, such as “expression”, “expressed” for Expression. This motivates us to explore prototype-driven learning (Haghighi and Klein, 2006) in combination with grounded learning. Specifically, we simulated expert selection by picking the top five most frequent trigger words for each event type from training data. We then augmented grounded learning in GUSPEE by incorporating word emission features for each prototype word and the corresponding event state, e.g., $\llbracket\text{lemma = express}, z_m = \text{Expression}\rrbracket$. The weights are fixed to a large number (five in our case). Table 3 shows the results with prototypes, which improved substantially.
<table>
<thead>
<tr>
<th>Event Type</th>
<th>Rec.</th>
<th>Prec.</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expression</td>
<td>55.3</td>
<td>88.3</td>
<td>68.0</td>
</tr>
<tr>
<td>Transcription</td>
<td>50.0</td>
<td>39.1</td>
<td>43.9</td>
</tr>
<tr>
<td>Catabolism</td>
<td>52.4</td>
<td>100.0</td>
<td>68.9</td>
</tr>
<tr>
<td>Phosphorylation</td>
<td>61.7</td>
<td>82.9</td>
<td>70.7</td>
</tr>
<tr>
<td>Localization</td>
<td>52.8</td>
<td>100.0</td>
<td>69.1</td>
</tr>
<tr>
<td>Binding</td>
<td>20.2</td>
<td>92.7</td>
<td>33.2</td>
</tr>
<tr>
<td>Regulation</td>
<td>24.1</td>
<td>64.0</td>
<td>35.0</td>
</tr>
<tr>
<td>Positive regulation</td>
<td>17.4</td>
<td>63.8</td>
<td>27.4</td>
</tr>
<tr>
<td>Negative regulation</td>
<td>8.4</td>
<td>52.8</td>
<td>14.5</td>
</tr>
<tr>
<td>Total Event F1</td>
<td>27.9</td>
<td>72.2</td>
<td>40.2</td>
</tr>
</tbody>
</table>

Table 3: GENIA event extraction results of GUSPEE with five prototype words per event type

Figure 3: GENIA Event F1 of GUSPEE with prototypes, using incomplete database or text.

events such as Catabolism benefited the most from prototypes, as they have fewer variations in triggers. While the F1 score 40.2 still lags behind the supervised state of the art, it would have been competitive compared to the 24 teams participating in the original shared task, outperforming 19 of them (the top 5th system scored an F1 of 40.5, see www.nactem.ac.uk/tsujii/GENIA/SharedTask/results/results-master.html, Task 1).

4.3 Database-Text Mismatch

In our simulation of grounded learning, every event in the database is mentioned in some text and vice versa. In practice, however, there is usually a mismatch between database and text: the unannotated text generally contains more facts than are already populated in the database; conversely, a database fact may not be explicitly mentioned in the text.

The GENIA dataset offers an excellent opportunity to study the robustness of grounded learning in light of such mismatch. Specifically, we simulated a grounded learning scenario with an incomplete database by populating events from the annotations of a random fraction of training text, and then learning GUSPEE with this database and all training text. Likewise, we simulated a scenario with incomplete text using the training event database in full, but only a fraction of unannotated text.

Figure 3 shows the results of GUSPEE with prototypes as the fraction varies between 0.1 and 1, by averaging five random runs. In both scenarios, the F1 score degrades smoothly as the fraction gets smaller. Precision stays roughly the same while recall gradually degrades (curves not shown). This shows that GUSPEE is reasonably robust. Not surprisingly, the degradation is steeper with incomplete database than with incomplete text.

To further investigate the effect of unannotated text, we also randomly sampled a fraction of database events for grounded supervision, and evaluated GUSPEE with increasing amounts of unannotated text. Figure 4 shows the results by averaging nine random runs. The F1 increases steadily with additional unannotated text, mainly due to rising recall (curves not shown). This suggests that GUSPEE could potentially benefit from more unannotated text and is reasonably robust even when some text is not relevant to the available events. As expected, more grounded supervision (50% vs. 10% database) led to substantially better F1 and lower variation.

4.4 Error Analysis

Upon manual inspection, we found that syntactic errors considerably affect performance. Poon (2013)
introduced complex states such as Sinking and Implicit to combat syntax-semantics mismatch, which could also be incorporated into GUSPEE. Improving syntactic parsing, either separately by adapting to the biomedical domain, or jointly along with semantic parsing, is another important future direction. GUSPEE achieved better precision than recall, especially when learning with prototypes, and might benefit from augmenting prototypes by distributional similarity (Haghighi and Klein, 2006).

4.5 Comparison with Existing Distant Supervision Approaches

Existing distant supervision approaches are not directly applicable to extracting nested events. However, we can convert the extraction task into classifying minimum sub-events between proteins, for which existing methods can be applied. Specifically, we used binary sub-events in $S(K)$ (Section 3.3) for distant supervision, and evaluated on classifying test sentences. This would enable an interesting comparison with GUSPEE, as the latter also derived indirect supervision from $S(K)$ alone. Textual annotations of triggers and nested event structures in GUSPEE output were ignored, and prototypes were not used to enable a fair comparison. For distant supervision, we used the state-of-the-art MultiR system (Hoffmann et al., 2011) with standard lexical and syntactic features (Mintz et al., 2009). MultiR can be used for supervised learning by fixing relations according to the sentence-level annotations, which provides a supervised upper bound.

Table 4 shows the results. GUSPEE outperformed MultiR by a wide margin, improving F1 by 24%. Surprisingly, GUSPEE even surpassed the supervised upper bound of MultiR. This suggests that our semantic parsing formulation not only is superior in representation power, but also facilitates better learning. We also experimented with sharing parameters among related sub-events in a MultiR-like model, but it did not improve the performance. Upon close inspection, we found that MultiR mainly scored on Binding events and failed almost entirely on the more difficult Regulation events. GUSPEE was able to extract Regulation events, but incurred some precision errors.

<table>
<thead>
<tr>
<th>Method</th>
<th>Rec.</th>
<th>Prec.</th>
<th>F1 (Class.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MultiR</td>
<td>11.2</td>
<td>21.7</td>
<td>14.8</td>
</tr>
<tr>
<td>MultiR (Super.)</td>
<td>12.1</td>
<td>24.4</td>
<td>16.2</td>
</tr>
<tr>
<td>GUSPEE</td>
<td>22.9</td>
<td>15.3</td>
<td>18.4</td>
</tr>
</tbody>
</table>

Table 4: Classification results on GENIA when events are simplified to binary relations for distant supervision.

5 Summary

We generalize distant supervision to complex knowledge extraction and propose the first approach to learn a semantic parser from a database of nested events and unannotated text. Experiments on GENIA event extraction showed that our GUSPEE system could learn from and extract such complex events, and was competitive even among supervised systems after incorporating a few easily-obtainable prototype event trigger words.

Future directions include: PubMed-scale pathway extraction; application to other domains; incorporating additional complex states to address syntax-semantics mismatch; learning vector-space representations for complex states; joint syntactic-semantic parsing; incorporating reasoning and other sources of indirect supervision.

References


