Feature Derivation for Exploitation of Distant Annotation via Pattern Induction against Dependency Parses

Dayne Freitag and John Niekrasz
SRI International
9988 Hibert Street, Suite 203
San Diego, CA 92130
{freitag,niekrasz}@ai.sri.com

Abstract

We consider the use of distant supervision for biological information extraction, and introduce two understudied corpora of this form, the Biological Expression Language (BEL) Large Corpus and the Pathway Logic (PL) Datum Corpus. Each resource eschews annotation at the sentence constituent level, and the PL corpus requires synthesis of information across multiple sentences to construct composite knowledge frames. Decomposing this problem into feature induction for slot-level attributes, followed by event assembly over this space of features, we introduce a novel, general-purpose pattern induction procedure, evaluating it against these two corpora, demonstrating its ability to induce effective detection against dependency parses.

1 Introduction

Biological event and relation extraction have been the focus of considerable study in recent years, resulting in the availability of annotated corpora (Kim et al., 2003; Pyysalo et al., 2007; Kim et al., 2008; Thompson et al., 2009). In the interest of replicability and progress on critical challenges, such resources typically decompose the hard problem of factual understanding into several simpler problems, such as entity recognition, binary relation detection, and co-reference resolution.

This methodology is subject to several criticisms. The reliance on thorough annotation imposes overheads that prevent rapid progress. The targeting of a fixed set of simplified, typically binary relations does justice neither to the complexity of information expressed in a typical sentence, nor to the biological processes under discussion. And the methodology places a emphasis on pieces of information amenable to expression in individual sentences, leaving untouched information that can be assembled only through traversal of paragraphs or complete documents.

Some of these limitations can be mitigated through *distant supervision*, a technique deriving noisy annotation through the heuristic alignment of structured knowledge resources to texts (Craven et al., 1999). The biological domain affords a number of high-quality knowledge resources with good coverage, making possible strongly competitive distantly supervised solutions (Poon et al., 2015). However, the distance between resource and text is often not great in such work, which focuses on relations for which entity co-occurrence in a sentence is strong evidence that the sentence expresses the target relation.

In this paper we attempt to exploit two knowledge resources, neither of which has received much attention from the BioNLP community, that increase this distance in interesting and distinct ways. The Biological Expression Language (BEL) is a knowledge interchange format intended to encode qualitative causal and correlative relations that supports nested knowledge frames. One product of the OpenBEL initiative[1] is the "Large BEL Corpus," which explicitly pairs a large number of literature excerpts with the BEL assertions that each supports. The relation between sentence and BEL statement is many-to-many, with no provisions for aligning specific statement components with specific sentence constituents.

The Pathway Logic (PL) project pursues high-fidelity signaling pathway models centering on Ras (Eker et al., 2004). Part of the effort involves a manual curation of experimental results, which has resulted in approximately 40K records, each

[1]http://www.openbel.org/

containing a detailed formal representation of an experiment and its outcomes. Such records, called *datums*, retain pointers to the papers and figures from which they were derived. In general, assembling the rich information contained in a datum requires traversing multiple sentences, both in figure captions and paper bodies.

We view the problem of extracting composite knowledge frames based on these attenuated supervisory signals as having two parts. First, we seek to generate a set of features highly indicative of various aspects of the target frame (its type, various attributes, etc.). Second, we view the problem of assembling frames from the resulting enriched feature space as one of structured classification. Recent work on structured classification lends confidence that such empirical assembly models are possible in principle (Daum III et al., 2009) and applicable to discourse-level event extraction (Reschke et al., 2014).

In this paper we address the first problem, the derivation of features for downstream extraction. We treat this problem as one of sentence classification via pattern (or rule) set induction against dependency parses. Compared with related work involving rules in the BioNLP literature (Bunescu et al., 2005; Bui et al., 2013; Huang et al., 2004; Liu et al., 2011; Hunter et al., 2008; Valenzuela-Escarcega et al., 2015; Peng et al., 2014), our approach exhibits some interesting features, particularly the eschewal of domain heuristics and the nonreliance on constituent-level annotations. Our work can be viewed as complementary to manual rule writing, and we present evidence that our learned patterns outperform rules written by hand.

Our contributions in this paper are twofold:

- We present and evaluate a novel, general-purpose approach to the induction of classification and extraction patterns from dependency parses.
- We evaluate this approach against two BioNLP corpora that have received little attention in the literature. Each corpus presents an extraction problem of greater complexity than can be addressed by current methods, providing avenues toward models of greater scope and biological fidelity.

The remainder of the paper is organized as follows. In the next section, we describe these two data sources and the problems they pose. In Section 3 we present our approach to pattern induction. Then, we describe and discuss our experiments in Section 4. Finally, we compare our approach in Section 5 to related work.

2 Data

In this section we describe the BEL Large Corpus and the PL Datum Corpus, against which we evaluate our approach.

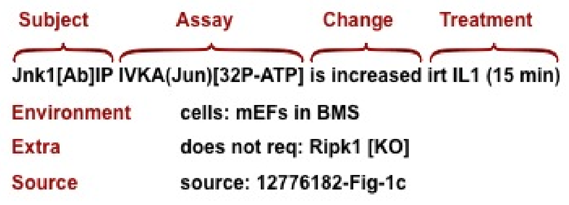
2.1 OpenBEL

The Biological Expression Language (BEL) is designed to capture rich qualitative biological relationships in context. For example, the BEL statement $[p(\text{HGNC:CCND1}) \Longrightarrow kin(p(\text{HGNC:CDK4}))]$ expresses that “increased abundance of the protein HGNC:CCND1 directly increases the kinase activity of the abundance of the protein HGNC:CDK4.” Here, the $\Longrightarrow$ symbol expresses a causal *directly increases* relationship between two BEL functions. Relationship types include causal (e.g., increases) and correlative (e.g., association) relationships. BEL functions are defined for abundances (e.g., protein or rna abundances), modifications (e.g., phosphorylation), activities (e.g., kinase or catalytic activity), processes (e.g., angiogenesis), and transformations (e.g., translocation and cell secretion). Depending on their definition, functions can be nested, accepting entity or other functions as arguments.

The OpenBEL initiative distributes the ‘large corpus.”[2], a collection of $\sim$80k statements, $\sim$74k of which are associated with natural language evidence passages and a PubMed article ID. The relationship between BEL statements and supporting sentences is many-to-many — some sentences are used to support multiple BEL statements, and whole paragraphs can be mustered in support of a given statement. After sentence segmentation and minor cleanup (e.g., removing inline comments from curators), we obtained a total of $\sim$40k unique supporting evidence sentences. In terms of biological content, the corpus contains independent observations (in human, mouse, and rat) not selected to represent any specific biological process. Given its size, this implies a lack of comprehensive coverage for any specific biological domain.

[2] `https://github.com/OpenBEL/openbel-framework-resources/blob/latest/knowledge/large_corpus.xbel.gz`

Figure 1: An example of a Pathway Logic datum.



Subject Assay Change Treatment

Jnk1[Ab]IP IVKA(Jun)[32P-ATP] is increased irt IL1 (15 min)

Environment	cells: mEFs in BMS
Extra	does not req: Ripk1 [KO]
Source	source: 12776182-Fig-1c

2.2 Pathway Logic

Pathway Logic (PL) is an approach to modeling biological entities and processes based on rewriting logic.[3] PL models can include specific facts and general principles relating entities and processes. PL is currently being used for the analysis of signal transduction and metabolic networks, including the STM model, a network of protein interactions and modifications used by the cell to transmit signals from its environment to the nucleus. Using STM, PL is able to predict and explain the effect of interventions (removal/inhibition/mutation of proteins) on downstream events.

In PL, reactions are curated by expert biologists from published experimental evidence (normally a Pubmed article). This experimental evidence is captured in formal expressions called "datums," encoded in a structured syntax over a controlled vocabulary, each representing one assay. An example datum is shown in Figure 1. Each datum captures, among other things, the protein(s) that are observed (Subject), the assay (Assay), the stimulus (Treatment), a result (Change), and the cells and culture conditions that were used in the assay (Environment).

Importantly, each datum also includes a reference to the figure in its source article containing the experimental result. These references allow us to link each datum to a small set of natural language sentences, namely those in the caption of the referenced figure or citing it in the paper body. This alignment, along with any mentions of entities listed in key datum roles, provides our supervisory signal. Note that the datum corpus consists primarily of PDF documents, necessitating a somewhat noisy conversion and alignment. We use a version of the PL knowledge base that contains ∼39k unique datums sourced from figures in ∼2,000 Pubmed articles.

In Section 4 we benchmark our pattern induction procedure against rule sets written by hand using the ODIN framework (Valenzuela-Escarcega et al., 2015). We created these rules over a period of several weeks while implementing a datum extraction system evaluated under DARPA Big Mechanism. This activity took place before the work described here had begun. Rule authors had full access to the datum corpus and possessed tools that exploited the same sentence-to-datum alignment heuristics used in this paper's experiments. Thus, although we cannot claim to have produced optimal manual rule sets, these sets can be viewed as characteristic of what can be achieved with reasonable effort. Of course, the two rule sources are not mutually exclusive. We are currently extending the datum extraction system to use *both* manual and automatically induced rules, and expect to see improvements in both precision and recall.

[3]`http://pl.csl.sri.com/`

3 Approach

3.1 The Setting

We frame our approach as a problem of Boolean classification over dependency parses (more generally, graphs with multiply labeled nodes and edges), where the positive class typically reflects that a sentence communicates some information we seek to detect. For conciseness, in the remainder of the paper we will refer simply to "parses", leaving the "dependency" modifier implied.

Formally, we are given *data* as a set of examples from (X, Y), with $Y = \{0, 1\}$ reflecting class membership. Each member of X is a parse taking the form (V_d, E_d), with V_d a set of vertices and E_d a set of directed edges (v_i, v_j). In addition, we are given two feature spaces, $F_V : V_d \mapsto \{0, 1\}$ and $F_E : E_d \mapsto \{0, 1\}$, that range over vertices and edges, respectively, which represent things such as a vertex word or the dependency label of an edge.

We seek to classify such parses using *patterns*, which take the form $(r, V_p, E_p, T_V, T_E, D_E)$. As with parses, the components V_p and E_p define a tree, which in this case is rooted in the distinguished vertex r. D_E denotes the *direction* of an edge, taking the form $D_E : E_p \mapsto \{\uparrow, \downarrow\}$. As this implies, we allow pattern edges to traverse up or down a parse.

T_V and T_E represent the *types* of vertices and edges, respectively. The vertex type, $T_V : V_p \mapsto \{\lambda\} \cup F_V$, is intended to constrain compatibility (call it C_V) with parse vertices:

$$C_V(v_d, v_p, T_V) = \begin{cases} 1 & : \quad T_V(v_p) = \lambda \vee \\ & \quad\;\; (T_V(v_p))(v_d) = 1 \\ 0 & : \quad \text{otherwise} \end{cases}$$

In other words, a *null* pattern vertex is compatible with any parse vertex, while a *feature* pattern vertex is compatible only with parse vertices for which the feature tests true, i.e., that *have* that feature. Thus, a feature pattern vertex with type f_{the} will match only parse vertices that correspond to the word "the", f_{NN} only to nouns, etc.

Edge types, $T_E : E_p \mapsto \{\lambda, *\} \cup F_E$, are analogous to vertex types. A pattern edge with type f_{amod} will match only parse edges having the "amod" dependency label. Edges with type "*", which we call *Kleene* edges, are explained in the next section.

3.2 Matching

We say that a pattern *matches* a parse if we can find a one-to-one alignment between pattern vertices and a subset of parse vertices proceeding recursively from the root node. This matching procedure is most easily described by means of a hypothetical Boolean function MATCH that returns true if a specified pattern vertex matches a specified parse vertex.

Algorithm 1 Procedure for matching patterns to parses.

```
 1: function MATCH($X, v_d, P, v_p$)
 2:     $(V_d, E_d) \leftarrow X$
 3:     $(r, V_p, E_p, T_V, T_E, D_E) \leftarrow P$
 4:     if not $C_V(v_d, v_p, T_V)$ then return false
 5:     for $e_p$ in $E_p$ s.t. $e_p = (v_p, v'_p)$ do
 6:         Found = false
 7:         for $e_d, v'_d$ in CandEdges$(e_p, E_d, v_d)$ do
 8:             if $T_E(e_p) = *$ then
 9:                 if KleeneMatch$(X, e_d, v'_d, P, v'_p)$ then
10:                     Found = true
11:             else if $T_E(e_p) = \lambda \vee T_E(e_p)(e_d) = 1$ then
12:                 if Match$(X, v'_d, P, v'_p)$ then
13:                     Found = true
14:         if not Found then return false
15:     return true
```

MATCH, shown in Algorithm 1, can be broken into three parts: a check for vertex compatibility (Line 4); a check for edge (or edge-to-path) compatibility (Line 5–13); and a recursive call to align unmatched pattern vertices (Lines 9 and 12). For brevity, we assume the existence of two helper functions. CANDEDGES (Line 7) merely selects and returns all edges (with destination vertex) attached to v_d that are compatible with the directional restriction of e_p. KLEENEMATCH (Line 9) enumerates all nodes on any path in the direction selected by e_p, internally calling MATCH on each until a match is found.

3.3 Induction

Linguistic variation usually ensures that no single pattern can adequately account for the ways in which target information is expressed. Therefore, our objective is to learn a set of patterns covering the forms observed in the training data. In pursuit of this objective we follow a top-down set covering procedure. At each step in this procedure, a single pattern is learned from the training data, all positive parses matching the new pattern are removed from the training set, and the process repeats. If no positive parses remain, or if the algorithm fails to induce a pattern, the process terminates.

Algorithm 2 Pattern induction procedure.

```
 1: function INDUCE($T, V, \alpha$)
 2:     $P \leftarrow \{\}$
 3:     $p \leftarrow$ null vertex
 4:     while $p' \leftarrow$ Specialize$(T, p, \alpha)$ do
 5:         $p \leftarrow p'$
 6:         $s \leftarrow$ Score$(p, V)$
 7:         $P \leftarrow P \cup \{(p, s)\}$
 8:         $T \leftarrow T - \{(x, y) | y = 1 \wedge \text{Match}(p, x)\}$
 9:     return $(p, s) \in P$ with max $s$
10:
11: function SPECIALIZE($T, p, \alpha$)
12:     $E \leftarrow \{\}$
13:     for $(x, y) \in T$ s.t. $y = 1$ do
14:         $M \leftarrow$ Matches$(p, x)$
15:         for $o_1 \cdots o_k \in$ Extensions$(M, x, \alpha)$ do
16:             $E \leftarrow E \cup \{o_1 \cdots o_k \mapsto (0, 0)\}$
17:     for $(x, y) \in T$ do
18:         $M \leftarrow$ Matches$(p, x)$
19:         for $o_1 \cdots o_k \in$ Extensions$(M, x, \alpha)$ do
20:             if $o_1 \cdots o_k \in E$ then
21:                 $E[o_1 \cdots o_k][y]$ += 1
22:     $o_1 \cdots o_k \leftarrow$ BestExtention$(E)$
23:     if no best extension found then
24:         return false
25:     return $p$ extended with $o_1$
```

The procedure for inducing a single pattern is presented as Algorithm 2. The top-level function INDUCE (Line 1) subjects an initial pattern containing a single null vertex (i.e., a pattern matching any non-empty parse—Line 3) to a series of specializations selected by the function SPECIALIZE (Line 4), and scores them against hold-out training data V (Line 6). The score of a rule is its precision, or $(p + m)/(p + n + 2m)$, where p is the number of matching positive parses, n the number of negative, and $m > 0$ a smoothing parameter.

SPECIALIZE (Line 11) takes the training set T, the pattern p in its current form, and an integer "look-ahead" parameter α. The procedure involves two passes over the training data, one collecting a set of candidate extensions to p (Lines 13–16), the other accumulating statistics for those extensions (Lines 17–21), which, as implied by Lines 16 and 21, are simple counts of the number of positive and negative parses matching each extension. These two steps assume the existence of two procedures: MATCHES (a straightforward variant of MATCH, Lines 14 and 18) returns all alignments between p and x; and EXTENSIONS (Lines 15 and 19), the behavior of which will be described in the next section.

Extensions are sequences of specialization operations $o_1 \cdots o_k$, $1 \leq k \leq \alpha$. Once statistics have been collected, the extension that best favors positive examples at the expense of negative ones is selected (Line 22). We use the "FOIL gain" in making this determination (Quinlan, 1990):

$$y' \cdot (\log(\frac{y'}{y'+n'}) - \log(\frac{y}{y+n})) \qquad (1)$$

where y and n (respectively, y' and n') are the number of positive and negative parses matched by p (respectively, p' formed by extending p). [4]. Importantly, once the best extension is identified, only the first specialization operation in the sequence is applied (Line 25) In this way, some of the greediness of the extension search is mitigated.

3.4 Specialization Operations

When considering specializations of a pattern, we have as reference its alignment to some parse, each vertex to some parse vertex, and each non-Kleene edge to some parse edge. We generate extensions by iterating over this alignment and collecting all possible specialization operations supported by it. Let us use v_p (pattern) and v_d (data) to represent two vertices in an alignment (similarly e_p and e_d for edges). We consider the following specialization operations (the o_i in Algorithm 2):

- **Specialize a null vertex**. If v_p is null, change it to require a feature of v_d. Because parse nodes may have multiple features, this operation in general generates multiple specializations.
- **Specialize a null edge**. Analogous to the previous item, but defined on edges. We currently only consider features based on dependency labels, but others are possible in principle.
- **Add a null edge**. If v_d has edges to unaligned parse vertices, add a null edge in the appropriate direction from v_p to a new null vertex.
- **Add a Kleene edge**. Except for the type, the conditions and effects of this operation are identical to the previous item.

Because these operations are typically considered in the context of a multi-step extension search, we can align any newly introduced vertices and edges to the parse, and consider further specializations either to the current element or to any newly introduced elements, up to the limit specified by α. Being a hyperparameter that inversely affects accuracy and running time, we set α manually to maximize accuracy given practical constraints on compute resources ($\alpha = 3$ for experiments in this paper).

Feature	**Description**
Word	Word associated with a vertex
POS	Part-of-speech tag
NER	The named entity type, if any
Cluster	Cluster of a word derived through distributional clustering

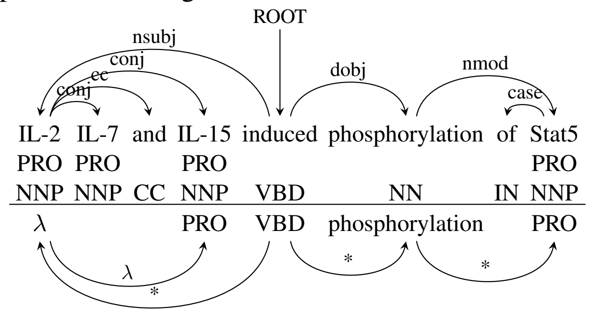
Table 1: Vertex feature types.

The specialization operations are defined in part by the feature spaces available, particularly by the vertex (word) feature space F_V. Conceptually, these features belong to an extensible set of types at the lexical level. The types used in our experiments are presented in Table 1.

Figure 2 shows an example pattern aligned to a matching sentence. In addition to the words in the sentence, the diagram also shows the associated POS and NER tags (clusters are not shown). An induced pattern matching the sentence is shown below the horizontal line. The pattern has three Kleene edges and one null edge. It has five nodes: two matching the NER=PRO (protein) feature, one matching the word "phosphorylation",

[4]We experimented with several comparable objective functions, including mutual information and kappa, and found results to be largely insensitive to this choice

Figure 2: An example of an automatically induced pattern matching a sentence.


ROOT
nsubj
conj
cc
conj
dobj
nmod
case
IL-2 IL-7 and IL-15 induced phosphorylation of Stat5
PRO PRO PRO PRO
NNP NNP CC NNP VBD NN IN NNP
λ PRO VBD phosphorylation PRO
λ
*
*
*

another matching POS=VBD, and another matching any word.

3.5 Pattern Application

The output of the induction process is a set of patterns scored for precision against hold-out data. Our primary interest in these patterns is as feature detectors used by some downstream process that assembles composite events, but the set of patterns can be used and evaluated as a stand-alone classification model. In this mode, the scores attached to the rules enable us to estimate the precision of matches against novel parses, and therefore to control precision and recall.

Multiple patterns may and often do match an individual parse. In such cases we estimate the precision of the ensemble match M as:

$$1 - \prod_{p \in M} (1 - s_p) \qquad (2)$$

where p is a pattern and s_p is its estimated precision. This estimate essentially treats the individual estimates as mutually independent.

4 Experiments

We evaluated our pattern induction procedure on the BEL and PL corpora for its effectiveness in detecting sentences expressing information needed for composite knowledge frames.

4.1 BEL evaluation

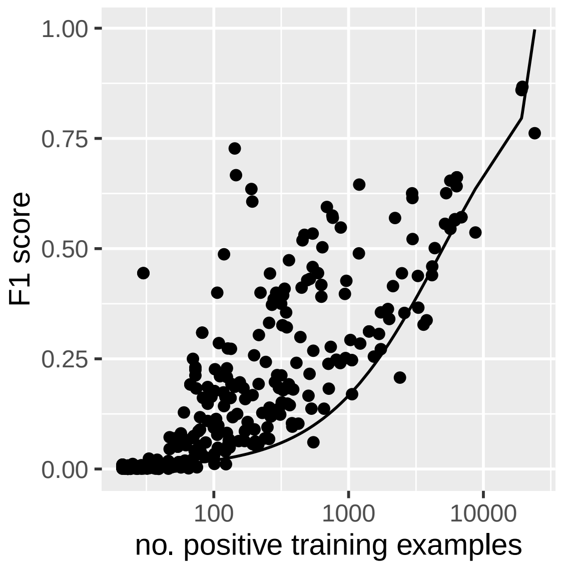
We converted BEL statements into a set of overlapping binary distinctions, called *fragments*, each a possible abstraction of the statement. Our objective is to convert each BEL extraction into a large set of redundant simpler problems, from which the original statement might be reconstituted. For example, the BEL statement "p(HGNC:CCND1) $\Longrightarrow kin(p$(HGNC:CDK4))" yields (among other fragments) "kin" (describes kinase activity), "$kin(p)$" (kinase activity of a protein), and "$\Longrightarrow kin(p)$" (kinase activity of a protein resulting from unspecified cause). Generating BEL fragments from fully specified BEL statements proceeds by first abstracting away any entity or numeric function arguments. Then, fragments are generated for every subtree in the abstract syntax tree of the statement. (We distinguish between functions occurring in the statement's subject and object position, in other words treating "subject" and "object" as a named element of the syntax tree.) Additionally, a fragment is generated for the relationship type and all functions occurring anywhere in the statement. Table 2 lists examples of BEL fragments with the number of positive training set sentences associated with each of them in the corpus.

BEL fragment	**no. pos. sentences**
$p \Longrightarrow ?$	8728
r	6355
$p \Longrightarrow bp$	1737
$c(p, p, p)$	127
$trans(p) \Longrightarrow r$	22

Table 2: Example BEL statement fragments (p = $proteinAbundance$; r = $rnaAbundance$; bp = $biologicalProcess$; $trans$ = $translocation$; c = $complexAbundance$).

From the ∼74,000 BEL statements associated with validated evidence, this process generates ∼3k unique fragments, of which we retained those associated with at least 20 sentences (resulting in ∼400 unique fragments). For each unique fragment, all sentences associated with the fragment were labeled as positive, and all other sentences were labeled as negative. Patterns were then induced and evaluated on a 60/20/20% train/validation/test split defined on each of the sets. Note that the corpus sometimes associates BEL statements with *multiple* sentences, e.g., and entire paragraph, which means our labeling procedure sometimes treats proximal sentences as positive, even though they may not directly instantiate a target statement. The inductive procedure, which assumes that the target class is a disjunction of mutually exclusive cases, handles such "noise" well, essentially failing to derive patterns from (i.e., ignoring) the superfluous "positive" examples.

Figure 3: Scatterplot showing results of the ∼400 sentence classification experiments run on the BEL corpus.



We evaluate the induced rule ensembles by calculating an idealized F1 score, identifying a threshold for classification based upon the validation estimated precision of the ensemble match (Eq. 2) that maximizes F1 on the test set. We acknowledge that finding this threshold using test data produces an overly-optimistic result, but doing so provides us with an informative upper bound. In practice, the threshld would be tuned using an alternative validation set to prevent overfitting. The mean cardinality of the induced rule ensembles was 7, with a total of 2991 rules (2402 *unique*) induced across the ∼400 fragments.

Results are plotted in Figure 3, with one dot per classification experiment, each being an application of the rule induction approach against a single unique BEL statement fragment. For each experiment, the F1 classification result is plotted against the size of the positive training set. The plot also contains a line showing the F1 results of a random chance baseline. To calculate the baseline, we assume a classifier that randomly labels sentences as positive or negative with the same marginal probabilities as observed in the training set.

4.2 Pathway Logic evaluation

We next conducted a set of experiments targeting classification of sentences associated with various PL datum fragments. In this case, we employed named entity resolution for proteins, labeling parse nodes as to whether they refer to the value of associated Subject or Treatment fields, and restricting rules to necessarily include nodes that match the protein mention. Results are shown in Table 3. These experiments targeted the two datum fields (*subject* and *treatment*) that correspond to extractible entities, and focus on the three important assay types (*phos*, *ubiq*, and *GTP-assoc*) for which we had written ODIN rules while implementing our heuristic datum extractor.

PL datum fragment	Learned	Written
phos & subject	0.54	0.37
ubiq & subject	0.53	0.41
GTP-assoc & subject	0.60	0.20
phos & treatment	0.48	0.35
ubiq & treatment	0.29	0.17
GTP-assoc & treatment	0.32	0.05

Table 3: F1 performance in the extraction of datum fields by learned and hand-written rules.

4.3 Discussion

The results presented in Figure 3 clearly establish the effectiveness of the induced rule ensembles at detecting the information conveyed in the BEL statements. Not surprisingly, this effectiveness increases with increased training data, though there is considerable variation. We attribute this variation to the indiscriminate way in which fragments were generated. Presumably, some kinds of information correspond more strongly to detectible linguistic regularities than others. A brief investigation informally confirmed our intuitions. For example, a common but difficult fragment is "$complex(p, p) \implies ?$" (about 0.17 F1), or, roughly, "a complex between two proteins increases some effect." A key insight is that "$\implies ?$" corresponds to a large range of effects, and these effects govern the form a sentence takes. In contrast, the fact that a complex is the agent is often expressed subtly, often requiring inference over multiple sentences. Note that the set of classes we target deliberately *overdetermine* the typical BEL statement, making it possible in principle to reassemble many statements.

The results presented in Table 3 promise immediate practical value. As described in Section 2.2, the hand-written rules were used in a system extracting simplified datums. Although this system produced very noisy outputs, we were able to show in an official evaluation that the extractions could be used to corroborate mechanistic assertions extracted by other systems, increasing baseline precisions of 50% to 80% at the strictest corroboration levels. Based on these results, we intend to supplement or replace the hand-written rules with learned ones.

In the slightly longer term, it remains to validate the second part of our hypothesis: that *assembly* of information captured by these induced patterns into composite frames (BEL statements or PL datums) similarly can be realized. To this end, we plan to explore the selection and use of learned patterns as features, as well as alternative approaches to induction, such as boosting. Experience has shown that contextualizing symbolic learning methods in this way yields performance as strong as other leading learning paradigm (Caruana and Niculescu-Mizil, 2006).

5 Related Work

Progress in biological information extraction (BioIE) is measured against shared annotated corpora that decompose the problem into entity extraction and sentence-level relation detection (Kim et al., 2003; Pyysalo et al., 2007; Kim et al., 2008; Thompson et al., 2009). The BEL corpus has recently joined the ranks of these shared corpora as part of BioCreative, where early F1 scores on the task of assembling complete BEL statements average about 0.2, reflecting the difficulty of the task (Fluck et al., 2015).

Corpora annotated for entities and pairwise relations enable the application of machine learning, which has been shown to be as effective for such problems (Bunescu et al., 2005). Perhaps because of a relative wealth of structured knowledge resources, there are several competitive rule-based approaches to BioIE, involving both hand-written rules (Hunter et al., 2008; Valenzuela-Escarcega et al., 2015; Peng et al., 2014) and rules induced or tuned from data. For the most part, these rule learning approaches introduce domain-specific heuristics that limit their generality. Bui et al (2013) begin with a pre-specified library of syntactic patterns, which are instantiated from training data through a domain-specific procedure. Huang et al (2004) heuristically simplify training sentences, align them in pairs using a specialized edit distance, derive a pattern from the alignment, applying a series of heuristic checks to discard problematic patterns. Liu et al (2011) presents what is in some ways a generalization of this procedure, deriving a graph structure that is the union of individual simplified parses. It is unclear how the resulting extractor controls for overgeneration.

In contrast to these rather specific solutions, Bunescu et al (2005) evaluate a number of machine learning approaches to BioIE problems, finding them generally viable and noting that the rule-based learners yield high precision. These rule learners, which were drawn from a tradition of general-purpose rule induction for IE (Ciravegna and others, 2001; Soderland, 1997; Califf and Mooney, 1999; Freitag, 2000; Freitag and Kushmerick, 2000), notably make no or relatively modest assumptions about syntax. We are aware of no prior work applying such techniques to full dependency or constituent parses.

The relaxations of the annotation requirement that we explore in this paper (absence of phrase-level annotations, distant supervision) have been thoroughly studied in other contexts. An early instance of the IE-as-classification idea was the AutoSlog system (Riloff, 1996), which gave birth to bootstrapping techniques commonly used for many NLP problems (Riloff et al., 1999). Similarly, distant supervision, pioneered in the biological domain (Craven et al., 1999), has matured toward yielding performance comparable to complete supervision on certain problems (Poon et al., 2015). In contrast with such work, which focuses on sentence-local targets, relatively little work has been done on *discourse-level* distant supervision. A counter-example is Reschke et al (2014), which addresses event extraction at the document level, showing promising results but leaving many unanswered questions.

6 Conclusion

We have presented two understudied corpora providing distant annotation for the extraction of composite frames constituted from multiple sentences, none of which are annotated at the constituent level. We have argued for a two-phase approach to the exploitation of these resources—feature derivation and frame assembly—and presented a novel pattern induction procedure applicable to the first phase. Experiments with the two corpora demonstrate the procedure is effective, yielding patterns superior to those authored by humans in a comparable pattern language.

References

Quoc-Chinh Bui, David Campos, Erik van Mulligen, and Jan Kors. 2013. A fast rule-based approach for biomedical event extraction. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 104–108. Association for Computational Linguistics.

Razvan Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun K. Ramani, and Yuk Wah Wong. 2005. Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. *Artificial Intelligence in Medicine: Special Issue on Summarization and Information Extraction from Medical Documents*.

M.E. Califf and R.J. Mooney. 1999. Relational learning of pattern-match rules for information extraction. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, volume 334.

Rich Caruana and Alexandru Niculescu-Mizil. 2006. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, pages 161–168. ACM.

Fabio Ciravegna and others. 2001. Adaptive information extraction from text by rule induction and generalisation. In *International joint conference on artificial intelligence*, volume 17, pages 1251–1256. LAWRENCE ERLBAUM ASSOCIATES LTD.

Mark Craven, Johan Kumlien, and others. 1999. Constructing biological knowledge bases by extracting information from text sources. In *ISMB*, volume 1999, pages 77–86.

Hal Daum III, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine learning*, 75(3):297–325.

Steven Eker, Merrill Knapp, Keith Laderoute, Patrick Lincoln, and Carolyn Talcott. 2004. Pathway logic: Executable models of biological networks. *Electronic Notes in Theoretical Computer Science*, 71:144–161.

Juliane Fluck, Sumit Madan, Tilia Renate Ellendorff, Theo Mevissen, Simon Clematide, Adrian van der Lek, and Fabio Rinaldi. 2015. Track 4 Overview: Extraction of causal network information in biological expression language (bel). In *Fifth BioCreative Challenge Evaluation Workshop*, pages 333–346.

Dayne Freitag and Nicholas Kushmerick. 2000. Boosted wrapper induction. In *AAAI/IAAI*, pages 577–583.

Dayne Freitag. 2000. Machine learning for information extraction in informal domains. *Machine learning*, 39(2-3):169–202.

Minlie Huang, Xiaoyan Zhu, Yu Hao, Donald G. Payan, Kunbin Qu, and Ming Li. 2004. Discovering patterns to extract proteinprotein interactions from full texts. *Bioinformatics*, 20(18):3604–3612.

Lawrence Hunter, Zhiyong Lu, James Firby, William A. Baumgartner, Helen L. Johnson, Philip V. Ogren, and K. Bretonnel Cohen. 2008. OpenDMAP: an open source, ontology-driven concept analysis engine, with applications to capturing knowledge regarding protein transport, protein interactions and cell-type-specific gene expression. *BMC bioinformatics*, 9(1):78.

J.-D. Kim, Tomoko Ohta, Yuka Tateisi, and Junichi Tsujii. 2003. GENIA corpusa semantically annotated corpus for bio-textmining. *Bioinformatics*, 19(suppl 1):i180–i182.

Jin-Dong Kim, Tomoko Ohta, and Jun'ichi Tsujii. 2008. Corpus annotation for mining biomedical events from literature. *BMC bioinformatics*, 9(1):1.

Haibin Liu, Ravikumar Komandur, and Karin Verspoor. 2011. From graphs to events: A subgraph matching approach for information extraction from biomedical text. In *Proceedings of the BioNLP Shared Task 2011 Workshop*, pages 164–172. Association for Computational Linguistics.

Yifan Peng, Manabu Torii, Cathy H. Wu, and K. Vijay-Shanker. 2014. A generalizable NLP framework for fast development of pattern-based biomedical relation extraction systems. *BMC bioinformatics*, 15(1):1.

Hoifung Poon, Kristina Toutanova, and Chris Quirk. 2015. Distant supervision for cancer pathway extraction from text. In *Pac. Symp. Biocomput.*, pages 120–131.

Sampo Pyysalo, Filip Ginter, Juho Heimonen, Jari Bjrne, Jorma Boberg, Jouni Jrvinen, and Tapio Salakoski. 2007. BioInfer: a corpus for information extraction in the biomedical domain. *BMC bioinformatics*, 8(1):50.

J. Ross Quinlan. 1990. Learning logical definitions from relations. *Machine learning*, 5(3):239–266.

Kevin Reschke, Martin Jankowiak, Mihai Surdeanu, Christopher D. Manning, and Daniel Jurafsky. 2014. Event Extraction Using Distant Supervision. In *LREC*, pages 4527–4531.

Ellen Riloff, Rosie Jones, and others. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI/IAAI*, pages 474–479.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proceedings of the national conference on artificial intelligence*, pages 1044–1049.

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

Paul Thompson, Syed A. Iqbal, John McNaught, and Sophia Ananiadou. 2009. Construction of an annotated corpus to support biomedical information extraction. *BMC bioinformatics*, 10(1):1.

Marco A. Valenzuela-Escarcega, Gus Hahn-Powell, Thomas Hicks, and Mihai Surdeanu. 2015. A Domain-independent Rule-based Framework for Event Extraction. In *ACL-IJCNLP 2015 System Demonstrations*.