SnapToGrid: From Statistical to Interpretable Models for Biomedical Information Extraction

Marco A. Valenzuela-Escárcega, Gus Hahn-Powell, Dane Bell, Mihai Surdeanu
University of Arizona
Tucson, AZ 85721, USA
{marcov, hahnpowell, dane, msurdeanu}@email.arizona.edu

Abstract

We propose an approach for biomedical information extraction that marries the advantages of machine learning models, e.g., learning directly from data, with the benefits of rule-based approaches, e.g., interpretability. Our approach starts by training a feature-based statistical model, then converts this model to a rule-based variant by converting its features to rules, and "snapping to grid" the feature weights to discrete votes. In doing so, our proposal takes advantage of the large body of work in machine learning, but it produces an interpretable model, which can be directly edited by experts. We evaluate our approach on the BioNLP 2009 event extraction task. Our results show that there is a small performance penalty when converting the statistical model to rules, but the gain in interpretability compensates for that: with minimal effort, human experts improve this model to have similar performance to the statistical model that served as starting point.

1 Introduction

Due to the deluge of unstructured data, information extraction (IE) systems, which aim to translate this data to structured information, have become ubiquitous. For example, applications of IE range from parsing literature (Iyyer et al., 2016) to converting thousands of cancer research publications into complex proteins signaling pathways (Cohen, 2015).

By and large, in academia most of these approaches are implemented using machine learning (ML). This choice is warranted: generally, ML approaches, where the machine learns directly from the data, perform better than approaches where human domain experts encode the structure to be extracted manually. For example, the top systems in the BioNLP event extraction shared tasks have consistently been ML-based approaches (Kim et al., 2009; Kim et al., 2013). However, this is only part of the story: most of these models cannot be easily understood by their users, and, by and large, cannot be modified without retraining. This "technical debt" of ML (Sculley et al., 2014) is better understood in industry: Chiticariu et al. (2013) report that 67% of large commercial vendors of natural language processing (NLP) software focus on rule-based IE, and an additional 17% on hybrid systems that combine rule-based and ML approaches.

In this paper we focus on *interpretable models* for information extraction, i.e., models that: (a) can be understood by human users, and (b) can be directly edited and improved by these users. In particular, we focus on deterministic, rule-based models. Here, we introduce a novel approach to generate such models, which maintains both the advantages of ML such as learning from data, and the benefits of interpretability such as allowing human domain experts to directly edit and improve these models. Specifically, our contributions are:

(1) We introduce a simple strategy that converts statistical models for IE to rule-based models. We call the proposed algorithm SnapToGrid. Our approach works in three steps. First, we train a statistical model for the task at hand. Here we experiment with logistic regression, but the proposed method is, in principle, independent of the underlying statistical model. Further, our strategy can operate over multiple classifiers that are part of the same IE system (e.g., one classifier to identify event triggers, and another to identify event arguments). Second, we convert features

Proceedings of the 15th Workshop on Biomedical Natural Language Processing, pages 56–65,
Berlin, Germany, August 12, 2016.

to rules implemented in Odin, a modern declarative rule language (Valenzuela-Escarcega et al., 2016; Valenzuela-Escarcega et al., 2015). We also discard most of the statistical information acquired previously, by converting feature weights to discrete votes, which guarantees interpretability (hence the SnapToGrid name). Third, human domain experts inspect and manually improve the generated model, under certain time constraints.

(1) We evaluate our approach on the BioNLP 2009 core event extraction task, and demonstrate that the resulting interpretable model has similar performance to the statistical model that served as starting point.

2 Approach

Our motivation for this work is to keep the human domain expert in the loop when building IE systems. We show in Section 3 that this is beneficial, even when the domain experts have limited time to work on the task and no access to data other than the model itself. To achieve this "human in the loop" goal we propose the following three-step algorithm:

1. Train a statistical model for the IE task at hand (Section 2.1). The model may consist of several statistical classifiers. For example, for the BioNLP event extraction task, the most common approach involves two classifiers: one to identify event triggers, and a following classifier to identify event participants. One restriction is that these classifiers be feature-based classifiers, e.g., logistic regression, rather than the classifiers based on latent representations, e.g., neural networks.

2. Convert the statistical model into an interpretable, rule-based model (Section 2.2):

   (a) First, we convert the features to rules in the Odin language.

   (b) Then, we assign to each rules "votes" for a given class, by "snapping to grid", i.e., converting to discrete values, the weights computed by the above statistical model.

3. Domain experts edit the produced rule-based model directly, aiming to improve its quality with respect to both coverage and precision (Section 2.3).

We detail this process in the rest of this section, focusing on the BioNLP core event extraction task as the domain of interest.

2.1 Step 1: Build Statistical Model

Our statistical model is inspired by the top performing approach at the 2009 evaluation (Björne et al., 2009). The approach is summarized in Figure 1. Similar to (Björne et al., 2009), our approach consists of two classifiers: the first classifier detects and labels event trigger words in the input text; the second classifiers extracts and labels relations between event triggers and potential event participants, which can be either Protein entities or other event triggers. Both classifiers are implemented using multi-class logistic regression (LR), but our conversion process (Steps 2 and 3) is independent of the underlying statistical model, so, in principle, other feature-based classifiers that assign explicit weights to features could be used, e.g., perceptron, or linear support vector machines.

The Trigger Classifier

The first classifier sequentially labels each word in the input text as a trigger for a specific BioNLP event class, or as `Nil` otherwise. We implemented the following features:

Surface features: These features include the original and lemmatized words, and the presence of the word in a gazetteer of known event triggers (constructed automatically from the training data). These features are generated for the word being classified, as well as the words surrounding it inside a window of n tokens. We used two windows in our experiments, with $n = 1$ and $n = 4$. Further, bag-of-words features are generated for the windows and for the sentence as a whole.

Syntactic features: These features capture the syntactic dependencies (both incoming and outgoing) directly connected to the token. All syntactic information was represented using Stanford dependencies (De Marneffe and Manning, 2008), and was generated using the CoreNLP toolkit (Manning et al., 2014). For each of these paths, we generate two different versions: one containing just the label and direction of the syntactic dependencies, and another including also the destination words.

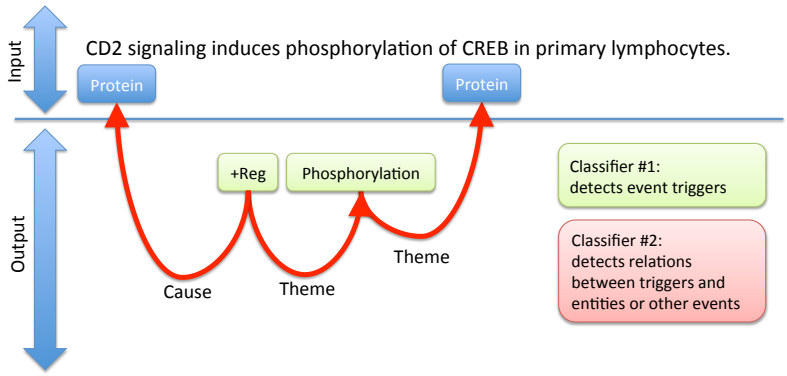

Figure 1: Architecture of the statistical model for the BioNLP core event extraction task.

Entity features: These features encode the number of other entities surrounding the token, both inside a window and in the sentence as a whole.

The Event Participant Classifier

This classifier pairs all the triggers detected by the previous classifier with other named entities (Proteins in this case) or event triggers that occur in the same sentence. These pairs are then classified into one of the possible participant relations, or `Nil` indicating that there is no relation between the pair. This classifier uses the following features:

Syntactic features: These features are based on the shortest path connecting the two mentions (trigger and candidate participant) in the Stanford syntactic dependency graph. Two versions of the shortest path are used: a lexicalized one (capturing the words along the path), and an unlexicalized one.

Surface features: These features include: the order of the two mentions in text, their distance in terms of tokens, the number of entities and triggers in the sentence, the parts of speech and words of the mentions, and the number of triggers and entities between the mentions.

Consistency features: These features encode the labels of the two mentions jointly, as well as the labels of their superclasses. For example, the features <Regulation, Phosphorylation> and <Regulation, Event> are generated for a relation between a Regulation event trigger and a Phosphorylation trigger as its theme. These feature capture selectional preferences for arguments, e.g., the Theme of a regulation event should be another event.

Graph features: The parent, children, and siblings of the mentions in the syntactic dependency graph.

Limitations

Not all of the above features can be represented as rules in the current implementation of the chosen rule language. Currently[1], Odin rules capture paths (over sequences or directed graphs) that are anchored at both ends (e.g., from an event trigger to an event argument) (Valenzuela-Escarcega et al., 2015; Valenzuela-Escarcega et al., 2016). Because of this, Odin cannot represent the following information: bag-of-word features, syntactic paths that are not anchored at both ends (such as dependencies connected only to event trigger candidates), and features that count occurrences of tokens or entities in text. In Section 3 we analyze the performance drop when such features are removed from the model.

2.2 Step 2: Convert the Statistical Model to a Rule-based Model

Once the statistical model is constructed, we employ the lossy process below to convert it to an interpretable one.

Converting Features to Rules

First, we convert the features encoded in the statistical model to rules in the Odin language (Valenzuela-Escarcega et al., 2015; Valenzuela-Escarcega et al., 2016). In general, the features previously introduced consist of conjunctions of information bits, each of which corresponds to a different rule fragment. For example, for the classification of event participants, one such conjunction captures the type of the expected trigger (e.g., Phosphorylation), combined

[1] As of June 2016

```
- name: phospho_event
  label: Phosphorylation
  pattern: |
    trigger:Phosphorylation
    theme:Protein = >nsubjpass
```

Figure 2: Example of a rule for event participant classification that is built from a single feature. The feature captures the passive nominal subject (`nsubjpass`) outgoing (>) from a Phosphorylation trigger and landing on a Protein. The bold font indicates the rule output, i.e., the nominal subject is the **theme** of a **Phosphorylation** event.

with the syntactic path that connects the trigger with the participant candidate (e.g., an outgoing passive nominal subject – `nsubjpass`), and a semantic constraint for the type of named entity of the participant (e.g., Protein). These are immediately translatable to Odin rules, as illustrated in Figure 2.

Importantly, the rules encode output information as well, e.g., the recognized event participant serves as a theme for a Phosphorylation event in Figure 2. At this stage, this information is exhaustively generated from all possible classifier labels (e.g., for the classification of event participants these labels are the cartesian product of {`theme`, `cause`} and possible event labels {`Phosphorylation`, `Binding`, ...}). Of course, some of these outputs do not apply. For example, it is highly unlikely that the rule shown in Figure 2 produces the cause of a Regulation event. We quantify the confidence in these outputs in the next stage of the algorithm.

Converting Weights to Votes

Feature weights are unbounded continuous values that are difficult to interpret and manually modify. For this reason, we would ideally prefer to exclude them completely from the interpretable model. Conceptually, this is simple: we could use the weights to choose the most likely output label for a rule (from the options generated previously), and discard them afterwards. However, our early experiments demonstrated that this performs poorly, because it forces the algorithm to ignore the inherent ambiguity of language, which is captured by the statistical model through weights. For example, the trigger classifier learns that "recruits" serves as trigger for two different events, Binding and Localization, and, consequently, assigns different weights to the two labels based on

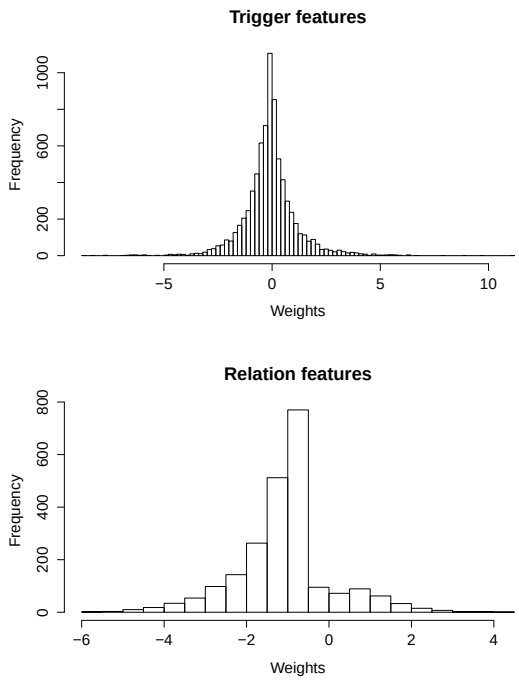

Figure 3: Weights of the two classifiers converted to votes (trigger classifier – top, participant classifier – bottom). Each histogram bin receives a number of votes (positive or negative) equal to its offset from 0.

the amount of evidence seen in training. During inference, the most likely class is chosen by aggregating the weights of all features that apply.

Given this observation, we chose to preserve the weights, but convert them from the original unbounded continuous values to discrete "votes" (positive or negative) that are then used during inference to resolve conflicts. This achieves two things. First, we increase the interpretability of the model: humans can now interpret these discrete votes, which mimic a Likert scale (Likert, 1932). Second, by keeping and using these discretized votes, we preserve some of the statistical power of the model. We show in Section 3 that some performance is indeed lost in this conversion, but the loss is small and the gain in interpretability compensates for that.

The conversion from continuous weights to discrete votes is a process similar to choosing the bins in a histogram. In our case, we first construct a histogram of all feature weights. Then, each histogram bin receives a number of votes equal to its offset (positive or negative) from 0. For example, all the weights in the second bin to the left of 0 receive two negative votes. Several methods have been proposed for selecting the number of bins in

a histogram, for example (Sturges, 1926; Doane, 1976; Freedman and Diaconis, 1981). Here, we use the formula proposed by (Scott, 1979):

$$h = 3.5\hat{\sigma}n^{-1/3} \quad (1)$$

where h is the estimated bin width, n is the sample size, and $\hat{\sigma}$ is the estimated standard deviation. We chose this formula because it gives a good compromise between retaining most of the information in the weights while minimizing the number of bins. The resulting binned weights for trigger and relation features (generated using the BioNLP 2009 training corpus) are shown in Figure 3.

2.3 Step 3: Edit the Rule-based Model

The output of the previous two steps is a model consisting of a set of rules. The association between rules and output classes is measured through votes that each matching rule gives to each output label. The last step in our proposed approach is to let human domain experts improve this model by directly editing it. The experts had complete freedom in the operations they were allowed to do. For example, they could improve the syntactic paths captured by the rules, or increase/decrease the number of votes assigned to a specific rule. The only constraints were: (a) they were not allowed to look only at the learned rules and not at the training data, and (b) they had to complete the process within one hour. This setting is of course artificial and unrealistic. We enforced it in this work to demonstrate the interpretability of the generated model.

3 Empirical Results

We analyze the performance of our approach on the core event extraction dataset from the BioNLP 2009 shared task (Kim et al., 2009). All the results reported in this section were measured on the *development* partition of the dataset, which was not used at all during training.[2] To minimize overfitting, we did not implement any feature selection or other hyper parameter tuning process.

Table 2 lists the results of the complete statistical model, i.e., using all features introduced in Section 2.1, trained using L_2-regularized LR. This configuration generated 1,190,029 features with non-zero weights. The table shows that this model achieved an overall F1 score of over 40 points, which likely puts it in the top 5 or 6 (out of 24) systems that participated in the actual challenge.[3] The performance of this system could be further improved by adding more features proposed in other event extraction approaches (Miwa et al., 2010), feature selection, hyper parameter tuning, etc.

Event Class	Recall	Precision	F1
Gene_expression	67.70	68.08	67.89
Transcription	57.32	50.00	53.41
Protein_catabolism	71.43	68.18	69.77
Phosphorylation	68.09	68.09	68.09
Localization	69.81	74.00	71.84
Binding	31.85	25.57	28.37
Event Total	55.89	51.48	53.59
Regulation	17.16	33.33	22.66
Positive_regulation	19.45	41.67	26.52
Negative_regulation	14.29	36.36	20.51
Regulation Total	18.02	39.16	24.69
All Total	35.10	47.29	40.30

Table 1: Performance of the statistical model using L_2-regularized LR, and all available features.

Event Class	Recall	Precision	F1
Gene_expression	57.58	74.28	64.87
Transcription	40.24	57.89	47.48
Protein_catabolism	61.90	86.67	72.22
Phosphorylation	51.06	82.76	63.16
Localization	47.17	92.59	62.50
Binding	18.15	34.62	23.81
Event Total	42.75	64.61	51.45
Regulation	8.28	40.00	13.73
Positive_regulation	17.18	42.74	24.51
Negative_regulation	7.14	40.00	12.12
Regulation Total	13.65	42.14	20.62
All Total	26.77	56.22	36.27

Table 2: Performance of the statistical model with L_2-regularized LR, using only features that can be converted to rules.

For a fair comparison, we next trained the same model but using only features that can be converted to rules. As discussed, the features that were removed include bag-of-word features and features that count occurrences of tokens or entities in text. These results, summarized in Table 2, show that the overall F1 score drops 4 points. This suggests that rule languages need to be extended if they are to have the same representational power as feature-based models. Given that the focus of

[2]The online scoring website, which would have allowed us to also obtain scores on the official test partition, was down due to updates during the development of this work.

[3](Kim et al., 2009) report results on the official test partition, which are not directly comparable with our results. However, in the authors' experience, the difference in scores between the development and test partitions in this dataset tend to be small. Since the 2009 evaluation, several works have improved upon these results, with performance reaching 58 F1 points, but using more complex methods, including joint inference, coreference resolution, and domain adaptation (Miwa et al., 2012; Bui and Sloot, 2012; Venugopal et al., 2014).

Event Class	Recall	Precision	F1
Gene_expression	58.71	78.28	67.09
Transcription	37.80	55.36	44.93
Protein_catabolism	61.90	86.67	72.22
Phosphorylation	46.81	84.62	60.27
Localization	56.60	88.24	68.97
Binding	16.13	33.33	21.74
Event Total	42.75	66.60	52.08
Regulation	8.88	65.22	15.62
Positive_regulation	13.13	40.50	19.83
Negative_regulation	8.16	55.17	14.22
Regulation Total	11.41	44.44	18.15
All Total	25.54	59.35	35.72

Table 3: Performance of the statistical model with L_1-regularized LR, using only features that can be converted to rules.

Event Class	Recall	Precision	F1
Gene_expression	55.34	76.95	64.38
Transcription	28.05	53.49	36.80
Protein_catabolism	57.14	85.71	68.57
Phosphorylation	40.43	90.48	55.88
Localization	45.28	88.89	60.00
Binding	12.90	33.33	18.60
Event Total	38.04	67.18	48.58
Regulation	5.33	75.00	9.94
Positive_regulation	10.70	48.89	17.55
Negative_regulation	5.61	55.00	10.19
Regulation Total	8.76	51.50	14.97
All Total	21.97	62.98	32.57

Table 4: Performance of the rule-based model before expert intervention.

this work is not on the design of rule-based languages for IE, we will use this latter model as the starting point of our approach, ignoring (for now) the performance penalty observed above.

Importantly, a system with more than 1 million features is not interpretable. To address this, we trained the same system using L_1 regularization as a form of feature selection. This reduced the number of features with non-zero weights by two orders of magnitude: from over 1 million to 10,926. The performance of this model is shown in Table 3. The results demonstrate that this drastic reduction in the number of useful features came with a small performance cost, of less than 1 F1 point.

Given this successful compression of the feature space, we next convert this L_1-regularized model to rules, using the approach discussed in Section 2.2. The performance of the rule-based model (before expert intervention!) is summarized in Table 4. The table shows that the overall cost of "snapping to grid" the statistical model is approximately 3 F1 points, which come from a drop in recall. This happens because many feature weights associated with specific labels (such as specific event triggers) have low values (due to sparsity), and, after the discretization process, the model can no longer prioritize these labels over the `Nil` class. Interestingly, the same process yielded a small increase in precision from 59% to 62%.

All in all, we consider a drop of 3 F1 points for the gain of interpretability an acceptable tradeoff. To empirically demonstrate the value of interpretability, we let two Linguistics PhD students edit the generated rule-based model for one hour, aiming to improve its generalization, robustness to syntactic errors, and readability. The students were familiar with the Odin language (Valenzuela-Escarcega et al., 2015) so they could "read" the model, and had a high-level understanding of the BioNLP shared task (although they did not participate in it). To guarantee that their recommendations came from understanding the model rather than other external factors, they were not given access to the BioNLP dataset. Given the large number of rules at this point, the students tended to randomly sample the rules in the model attempting to find repeated mistakes, rather than linearly inspect the list of rules. Table 5 summarizes the experts' recommendations. As shown, several of the experts' suggestions involved removing or collapsing rules, which reduced the number of rules from 10,926 to 8,868.

Table 6 lists the performance of the resulting model, after implementing the experts' recommendations. The table shows that most of the F1 loss has been recovered: the overall F1 score for this system approaches 35 F1 points, and is less than 1 F1 point behind the $L1$-regularized LR statistical model. In addition of reducing the number of rules in the model, the experts' recommendations increased recall by over 4%, which is more than what was lost during the conversion to rules. However, the precision of this configuration decreased by 11%, which we blame on the experts' limited familiarity with the BioNLP task, and the strict settings of the experiment (no access to data, limited time). However, all in all, this experiment demonstrates that the rule-based model produced by the proposed approach is interpretable: the experts understood the model, and were able to improve it, both with respect to its generalization power and its readability.

Lastly, Figure 4 shows a learning curve for the statistical model and the corresponding rule-based model (before expert intervention). The curve shows that the rule-based model follows closely

Suggested Change	Description
Generalization	
Add `/conj_(and|or|nor)|dep|cc|nn|prep_of/{,2}` to the end of Theme paths.	This transformation adds an optional modifier dependency to capture event participants when they appear either as nominal heads or modifiers. For example, because of this transformation, the model handles both these phrases similarly: "phosphorylation of MEK" and "phosphorylation of the MEK protein"
Ensure that all syntactic paths end in `appos?`.	This change handles optional apposition to increase rule coverage. For example, in the sentence "we found that A20 binds to a novel protein, ABIN", the word ABIN is an appositive for the word protein, so ABIN can serve as an argument in the binding event.
Replace all specific named entities with their label.	For example, in rules such as `[word=phosphorylates] (?=MEK)` that reference a specific protein, this replaces the specific protein (MEK) with the label `Protein`. This improves rule generalization and, at the same time, reduces the total number of rules.
Make the `>nn` dependency optional in `Theme:Protein = >nsubjpass >nn`.	The output of this transformation is similar to the first suggested change, i.e., the same rule captures event participants when they appear either as nominal heads or modifiers.
Robustness	
Replace `agent` with `/^(agent|prep_by)$/`.	This modification is designed to account for a common parsing error of passive sentences, where `agent` dependencies are incorrectly parsed as `prep_by`.
Change `ccomp` to `/(c|x)comp/` and `acomp` to `/(a|x)comp/`.	Parsers often confuse clausal and adjectival complements with open clausal complements. This transformation allows the rules to be robust to these errors.
Readability	
Merge rules when possible, e.g. `prep_of`, `prep_of nn`, `prep_of appos` become `prep_of (nn? appos | nn appos? nn?)?`.	This transformation collapses rules to improve readability.
Eliminate trigger rules that are not sufficiently discriminative (e.g., `(?<=[lemma="be"]) [tag=/^(V|N|J)/)`.	Some uninformative rules survived feature regularization but should be removed, as with the example rule which looks for any verb, noun, or adjective preceded by any conjugation of the verb "be". These rules inflate the grammar without adding discriminative power.
Do not use `word` constraints. Only use `lemma` and `tag` features in trigger rules for simple events (other than transcription and binding).	This modification prefers lexical constraints on lemmas, because they generalize better than constraints on actual words.
Remove redundant constraints.	For example, in patterns like `[incoming=nsubj & tag=/^N/]` the POS tag is redundant because it is implicitly defined through the incoming dependency (nominal subject).

Table 5: Representative examples of the rule changes suggested by linguistic experts.

the behavior of its statistical counterpart, with a small penalty of 1-2 F1 points throughout. As discussed before, this performance loss can be mitigated through interventions by domain experts.

4 Related Work

Most of the biomedical IE systems in academia rely on supervised machine learning. This includes the top performing system at the BioNLP 2009 shared task (Björne et al., 2009), as well as several following approaches that improve upon its performance (Miwa et al., 2010; McClosky et al., 2012; Miwa et al., 2012; Bui and Sloot, 2012; Venugopal et al., 2014).

However, rule-based approaches (Appelt et al., 1993; Cunningham et al., 2002; Piskorski et al., 2004; Li et al., 2011; Chang and Manning, 2014) are preferable when the corresponding systems have to be deployed for long periods of time, during which they have to be maintained and improved. This has been recognized in industry (Chiticariu et al., 2013).

We bring together these two diverging directions by combining the advantages of ML with the interpretability of rule-based approaches. By representing the model as a collection of declarative rules, experts can directly edit the model, thus guaranteeing that the desired changes are actually applied. This is in contrast with methods such as active learning, in which the learning algorithm

Event Class	Recall	Precision	F1
Gene_expression	60.39	70.49	65.05
Transcription	31.71	57.78	40.94
Protein_catabolism	61.90	81.25	70.27
Phosphorylation	42.55	86.96	57.14
Localization	45.28	88.89	60.00
Binding	22.18	23.50	22.82
Event Total	43.74	54.31	48.46
Regulation	10.06	40.48	16.11
Positive_regulation	12.80	44.89	19.92
Negative_regulation	10.71	51.22	17.72
Regulation Total	11.91	45.17	18.86
All Total	26.27	51.71	34.84

Table 6: Performance of the rule-based model, after expert intervention.

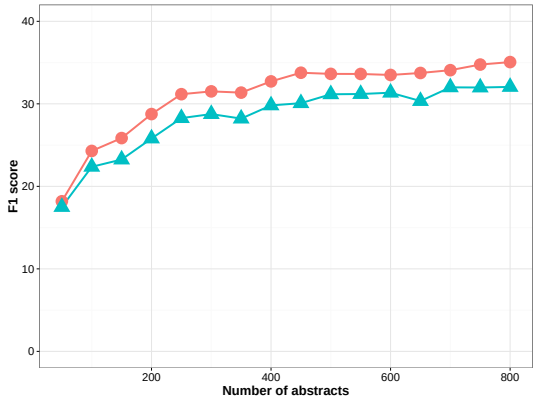

Figure 4: Learning curve showing the change in F1 performance as a function of the amount of training data. We compare the performance of the L_1-regularized logistic regression (shown using circles) with the rule-based model prior to the expert intervention (shown using triangles).

presents the "human in the loop" with new examples to annotate (Thompson et al., 1999). Although active learning may require less domain expertise than our proposal, it generally does not guarantee that the examples provided are actually propagated in the model (the learning algorithm may choose to override them with other data).

5 Conclusion and Future Work

We have proposed a simple approach that marries the advantages of machine learning models for information extraction (such as learning directly from data) with the benefits of rule-based approaches (interpretability, easier maintainability). Our approach starts by training a feature-based statistical model, then converts this model to a rule-based variant by converting its features to rules and its feature weights to discrete votes. In doing so, our proposal learns from data similar to other machine learning approaches, but produces an interpretable rule-based model that can be directly edited by experts. Using the BioNLP 2009 event extraction task as a test bed, we show that while there is a small performance penalty when converting the statistical model to rules, the gain in interpretability compensates for that.

In this work, we focused on building upon feature-based classifiers, in particular logistic regression, due to their potential extensions to distant supervision (DS), where training data is generated automatically by aligning a knowledge base (KB) of known examples (e.g., known drug-gene interactions) with text (e.g., scientific publications). Distant supervision has obvious applications to bioinformatics (Craven et al., 1999), but it generally suffers from noise in the automatically-generated annotations (Riedel et al., 2010). In future work, we plan to combine our work with distant supervision by adapting our proposal to logistic regression variants that are robust to the noise introduced in DS (Surdeanu et al., 2012). This extension would make it possible to generate rules even when no annotated examples are available, as long as a suitable KB of known examples exists.

Another planned extension of this work focuses on reducing the number of generated rules by merging/collapsing similar paths into a single pattern. This can be achieved by constructing a minimal deterministic acyclic finite-state automaton (DAFSA) (Daciuk et al., 2000) with the paths that are similar, and then converting the DAFSA into a single pattern (Neumann, 2005). For example, such approaches would collapse the two patterns: `dobj` and `dobj nn`, into a single one: `dobj nn?`. This is fundamental for the long-term maintainability of the rule-based model, because the human experts would have to maintain considerably fewer rules.

Lastly, we plan to improve the "snap to grid" algorithm. Currently, the conversion of weights to votes is implemented using Scott's rule (Scott, 1979), which is one method among several available to choose a histogram's bin size. Scott's method assumes that all bins have the same size, which may not be the best solution if interpretability is the goal. A potentially better approach is to select the bin divisions in a way that retains as much of the information contained in the weights as possible, while minimizing the number of bins.

Acknowledgments

This work was funded by the Defense Advanced Research Projects Agency (DARPA) Big Mechanism program under ARO contract W911NF-14-1-0395.

References

Douglas E. Appelt, Jerry R. Hobbs, John Bear, David Israel, and Mabry Tyson. 1993. Fastus: A finite-state processor for information extraction from real-world text. In *Proceedings of the International Conferences on Artificial Intelligence (IJCAI)*.

Jari Björne, Juho Heimonen, Filip Ginter, Antti Airola, Tapio Pahikkala, and Tapio Salakoski. 2009. Extracting complex biological events with rich graph-based feature sets. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*, pages 10–18. Association for Computational Linguistics.

Quoc-Chinh Bui and Peter MA Sloot. 2012. A robust approach to extract biomedical events from literature. *Bioinformatics*, 28(20):2654–2661.

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

Laura Chiticariu, Yunyao Li, and R. Reiss, Frederick. 2013. Rule-based Information Extraction is Dead! Long Live Rule-based Information Extraction Systems! In *Proceedings of 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Paul R. Cohen. 2015. DARPA's Big Mechanism program. *Physical Biology*, 12(4):045008.

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

Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. 2002. A framework and graphical development environment for robust nlp tools and applications. In *ACL*, pages 168–175.

Jan Daciuk, Stoyan Mihov, Bruce W Watson, and Richard E Watson. 2000. Incremental construction of minimal acyclic finite-state automata. *Computational linguistics*, 26(1):3–16.

Marie-Catherine De Marneffe and Christopher D Manning. 2008. Stanford typed dependencies manual. Technical report, Stanford University.

David P Doane. 1976. Aesthetic frequency classifications. *The American Statistician*, 30(4):181–183.

David Freedman and Persi Diaconis. 1981. On the histogram as a density estimator: l_2 theory. *Probability theory and related fields*, 57(4):453–476.

Mohit Iyyer, Anupam Guha, and Jordan Boyd-Graber. 2016. Feuding Families and Former Friends: Unsupervised Learning for Dynamic Fictional Relationships. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Junichi Tsujii. 2009. Overview of BioNLP09 Shared Task on Event Extraction. In *Proceedings of the Workshop on BioNLP: Shared Task*.

Jin-Dong Kim, Yue Wang, and Yamamoto Yasunori. 2013. The Genia Event Extraction Shared Task, 2013 Edition - Overview. In *Proceedings of the BioNLP Shared Task 2013 Workshop*.

Yunyao Li, Frederick R Reiss, and Laura Chiticariu. 2011. Systemt: A declarative information extraction system. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Systems Demonstrations*, pages 109–114. Association for Computational Linguistics.

Rensis Likert. 1932. A technique for the measurement of attitudes. *Archives of psychology*.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60.

David McClosky, Sebastian Riedel, Mihai Surdeanu, Andrew McCallum, and Christopher D. Manning. 2012. Combining joint models for biomedical event extraction. *BMC Bioinformatics*, 13(Suppl 11)(S9).

Makoto Miwa, Rune Sætre, Jin-Dong Kim, and Jun'ichi Tsujii. 2010. Event extraction with complex event classification using rich features. *Journal of bioinformatics and computational biology*, 8(01):131–146.

Makoto Miwa, Paul Thompson, and Sophia Ananiadou. 2012. Boosting automatic event extraction from the literature using domain adaptation and coreference resolution. *Bioinformatics*, 28(13):1759–1765.

Christoph Neumann. 2005. Converting deterministic finite automata to regular expressions. http://neumannhaus.com/christoph/papers/2005-03-16.DFA_to_RegEx.pdf.

Jakub Piskorski, Ulrich Schäfer, and Feiyu Xu. 2004. Shallow processing with unification and typed feature structures–foundations and applications. *Knstliche Intelligenz*, 1(1).

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

David W Scott. 1979. On optimal and data-based histograms. *Biometrika*, 66(3):605–610.

D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, and Dan Dennison. 2014. Hidden Technical Debt in Machine Learning Systems. In *Proceedings of SE4ML: Software Engineering for Machine Learning (NIPS 2014 Workshop)*.

Herbert A Sturges. 1926. The choice of a class interval. *Journal of the american statistical association*, 21(153):65–66.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465. Association for Computational Linguistics.

Cynthia A Thompson, Mary Elaine Califf, and Raymond J Mooney. 1999. Active learning for natural language parsing and information extraction. In *ICML*, pages 406–414. Citeseer.

Marco A. Valenzuela-Escarcega, Gustave Hahn-Powell, and Mihai Surdeanu. 2015. Description of the Odin Event Extraction Framework and Rule Language. In *arXiv:1509.07513*.

Marco A. Valenzuela-Escarcega, Gustave Hahn-Powell, and Mihai Surdeanu. 2016. Odin's Runes: A Rule Language for Information Extraction. In *Proceedings of the 10th edition of the Language Resources and Evaluation Conference (LREC)*.

Deepak Venugopal, Chen Chen, Vibhav Gogate, and Vincent Ng. 2014. Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features. In *EMNLP*, pages 831–843.