A Semi-Supervised Approach To Learning Relevant Protein-Protein Interaction Articles

Mark A. Greenwood `m.greenwood@dcs.shef.ac.uk`
Mark Stevenson `m.stevenson@dcs.shef.ac.uk`
Department of Computer Science, University of Sheffield, Regent Court, 211 Portobello Street, Sheffield, S1 4DP, UK.

Abstract

This paper describes an Information Extraction system that can be used to identify articles containing protein-protein interactions. The approach relies on the automatic acquisition of dependency tree based patterns which can be used to identify these interactions and consequently select relevant documents. Evaluation shows an F-Score performance of approximately 64%.

Keywords: semi-supervised learning, dependency trees, relation extraction, linked chains

1 Approach

Our approach to the Protein-Protein Interaction (PPI) article subtask (IAS) of the 2nd BioCreAtIvE workshop follows on from previous work on relation extraction which has been applied to several problems including the identification of gene-gene interactions [2]. This method, briefly outlined below, uses a semi-supervised algorithm to learn a relation extraction system given a few example seed patterns which illustrate protein-protein interactions.

Each abstract is pre-processed before extraction patterns are learned. Abstracts are split into sentences. Protein names are identified, using NLProt[1], and substituted with a generic token (`PROTEIN`). The text is then parsed, using the Stanford parser[2], to produce a dependency tree for each sentence.

The patterns we use to identify relations consist of chains and linked chains in dependency trees [2]. A chain is a path from a verb to any of its descendants in the dependency tree, passing through zero or more nodes. A linked chain is a pair of chains which share the same verb as their root but do not have any other nodes in common. For example, the linked chain $PROTEIN \xrightarrow{subj} interact \xleftarrow{with} PROTEIN$[3] would be found in a dependency parse for the sentence "PROTEIN frequently interacts with PROTEIN". It has been shown that chain and linked chain patterns are expressive enough to represent the majority of relations within a dependency analysis without generating an unwieldy number of patterns [4].

Space limitations prevent us from describing our learning algorithm in detail, a fuller description is available elsewhere [1]. Briefly, our algorithm for learning linked chain patterns begins with a small number of seed patterns used to provide examples of good patterns. Eight seeds, shown in Table 1, were used for the experiments described here. Our approach extracts all possible chain and linked chain patterns from the corpus and compares each against the seed patterns. Patterns whose similarity score is above a threshold, α, are assumed to be useful extraction patterns and the β of these with the highest score are added to the set of seeds.[4] This process is then repeated with the remaining patterns being compared against the expanded set of seed patterns. The algorithm continues until no more patterns can be learned.

[1] `http://cubic.bioc.columbia.edu/services/nlprot/`

[2] `http://www-nlp.stanford.edu/software/lex-parser.shtml`

[3] $X \xrightarrow{reln} Y$ indicates that nodes X and Y are connected by the dependency relation $reln$ and that X is Y's daughter.

[4] Based on previous experiments [1], α was set to 0.9 times the score of the best matching pattern and β to 4.

Table 1: Initial Seed Patterns

$PROTEIN \xrightarrow{of} reduce \xleftarrow{to} PROTEIN$	$PROTEIN \xrightarrow{pnmod} colocalized \xrightarrow{with} PROTEIN$
$PROTEIN \xrightarrow{subj} link \xleftarrow{obj} PROTEIN$	$PROTEIN \xrightarrow{subj} interact \xleftarrow{with} PROTEIN$
$PROTEIN \xrightarrow{obj} connect \xleftarrow{to} PROTEIN$	$PROTEIN \xrightarrow{obj} associate \xleftarrow{with} PROTEIN$
$PROTEIN \xrightarrow{subj} encode \xleftarrow{obj} PROTEIN$	$PROTEIN \xrightarrow{subj} express \xleftarrow{obj} PROTEIN$

A key choice in our approach is the method which is used to compare patterns against the seeds. We use a similarity function which is inspired by work on tree kernels [1], although the function used is not itself a kernel. This function compares pairs of patterns by starting at each of their root nodes and comparing their structure until they diverge too far to be considered similar.

Each node n in an extraction pattern has three features associated with it: the word, the relation to a parent, and the part-of-speech (POS) tag. These features are denoted by n_{word}, n_{reln} and n_{pos} respectively. Pairs of nodes can be compared by examining the values of these features and also by determining the semantic similarity of the words. A set of four functions, $F = \{word, relation, pos, semantic\}$, is used to compare nodes. The first three of these correspond to the node features with the same names and return 1 if the value of the feature is equal for the two nodes and 0 otherwise. The remaining function, *semantic*, returns a value between 0 and 1 to signify the semantic similarity of lexical items contained in the word feature of each node. This similarity is computed using Lin's lexical similarity function [3] which relies on an information-theoretic measure based on the WordNet hierarchy. The similarity of two nodes, $s(n_1, n_2)$ is 0 if their part of speech tags are different and, otherwise, is simply the sum of the scores provided by the four functions in F.

The similarity of a pair of linked chain patterns, l_1 and l_2, is determined by the function sim where r_1 and r_2 are the root nodes of patterns l_1 and l_2 and C_r is the set of children of node r. The final part of the similarity function, sim_c, calculates the similarity between the child nodes of n_1 and n_2. Using this similarity function a pair of identical nodes have a similarity score of four. Consequently, the similarity score for a pair of linked chain patterns can be normalised by dividing the similarity score by 4 times the size (in nodes) of the larger pattern. This results in a similarity function that is not biased towards either small or large patterns but will select the most similar pattern to those already accepted as representative of the domain.

$$s(n_1, n_2) = \begin{cases} 0 \text{ if } pos(n_1, n_2) = 0 \\ \sum_{f \in F} f(n_1, n_2) \text{ otherwise} \end{cases}$$

$$sim(l_1, l_2) = \begin{cases} 0 \text{ if } s(r_1, r_2) = 0 \\ s(r_1, r_2) + \\ sim_c(C_{r_1}, C_{r_2}) \text{ otherwise} \end{cases}$$

$$sim_c(C_{n_1}, C_{n_2}) = \sum_{c_1 \in C_{n_1}} \sum_{c_2 \in C_{n_2}} sim(c_1, c_2)$$

These acquired patterns can then be used to perform the IAS task. The abstracts in the test set are processed, as above, reducing each to a set of patterns. Each abstract is then scored based on the number of acquired patterns it contains. An abstract which does not contain any of the acquired patterns is deemed irrelevant. Relevance of the remaining abstracts is determined by ranking them based on the number of acquired patterns each contains.

2 Results and Analysis

The algorithm ran for 241 iterations before it was unable to acquire any more patterns. Patterns acquired up to iterations 241, 160, and 80 were submitted for the formal evaluation as runs #1, #2 and #3 respectively. After the 80th, 160th and 241st iteration the learning algorithm had acquired 320, 640, and 964 patterns respectively. These were combined with the eight seeds to perform the evaluation task. Results of this evaluation are shown in Table 2. The bottom portion of this table shows the mean and standard deviation, σ, of all 51 submitted runs. These results show that recall increases substantially as the algorithm learns without overly reducing precision, the net result of

Table 2: Official Evaluation Figures

Run	Precision	Recall	Accuracy	F-Score	FPR	TPR	Error Rate	AUC
#1	0.668	0.616	0.655	0.641	0.307	0.616	0.345	0.681
#2	0.735	0.547	0.675	0.627	0.197	0.547	0.325	0.692
#3	0.805	0.373	0.641	0.510	0.091	0.373	0.359	0.664
Mean	0.664	0.764	0.671	0.687	-	-	-	0.735
σ	0.081	0.193	0.064	0.104	-	-	-	0.074

which is an increase in F-Score.

Figure 1 shows the F-Score calculated at each iteration of the learning process. The eight seed patterns achieve an F-Score of 19.6%. The graph demonstrates that there is a steady increase in performance to a maximum of 64.3% (iteration 179), almost 45% more than the seed patterns. The F-Score at the final iteration (64.1%) is slightly lower than the maximum but the graph shows that the algorithm reaches a plateau so that the system submitted as run #1 was close to the best achievable by the system.

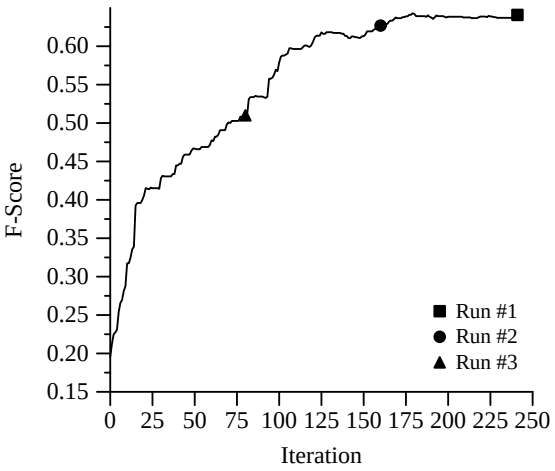

Figure 1: F-Score For All 241 Iterations

Our highest scoring official run and the results from the algorithm's best performing iteration are comparable with the mean scores of all submitted systems (within one standard deviation). However, our system has the advantage of employing a semi-supervised learning algorithm which requires a very small amount of annotated data (eight seed patterns).

3 Conclusion

This paper has described how an algorithm for learning relation extraction patterns can be used to identify articles containing interactions between proteins. The approach is semi-supervised and requires only a small number of example seed patterns. Analysis shows that the patterns learned by the system improve substantially on the performance of the seeds to produce a system which is comparable to the average score of the systems submitted for this task.

Acknowledgements The research described in this paper was funded by the UK Engineering and Physical Sciences Research Council via the RESuLT project (GR/T06391).

References

[1] M. A. Greenwood and M. Stevenson. Improving Semi-supervised Acquisition of Relation Extraction Patterns. In *Proceedings of the Information Extraction Beyond The Document Workshop (COLING/ACL 2006)*, Sydney, Australia, 2006.

[2] M. A. Greenwood, M. Stevenson, Y. Guo, H. Harkema and A. Roberts. Automatically Acquiring a Linguistically Motivated Genic Interaction Extraction System. In *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, Bonn, Germany, 2005.

[3] D. Lin. An Information-theoretic Definition of Similarity. In *Proceedings of the Fifteenth International Conference on Machine learning (ICML-98)*, Madison, Wisconsin, 1998.

[4] M. Stevenson and M. A. Greenwood. Comparing Information Extraction Pattern Models. In *Proceedings of the Information Extraction Beyond The Document Workshop (COLING/ACL 2006)*, Sydney, Australia, 2006.