A Graph Kernel for Protein-Protein Interaction Extraction

Antti Airola, Sampo Pyysalo, Jari Björne, Tapio Pahikkala, Filip Ginter and Tapio Salakoski
Turku Centre for Computer Science
and Department of IT, University of Turku
Joukahaisenkatu 3-5
20520 Turku, Finland
firstname.lastname@utu.fi

Abstract

In this paper, we propose a graph kernel based approach for the automated extraction of protein-protein interactions (PPI) from scientific literature. In contrast to earlier approaches to PPI extraction, the introduced all-dependency-paths kernel has the capability to consider full, general dependency graphs. We evaluate the proposed method across five publicly available PPI corpora providing the most comprehensive evaluation done for a machine learning based PPI-extraction system. Our method is shown to achieve state-of-the-art performance with respect to comparable evaluations, achieving 56.4 F-score and 84.8 AUC on the AImed corpus. Further, we identify several pitfalls that can make evaluations of PPI-extraction systems incomparable, or even invalid. These include incorrect cross-validation strategies and problems related to comparing F-score results achieved on different evaluation resources.

1 Introduction

Automated protein-protein interaction (PPI) extraction from scientific literature is a task of significant interest in the BioNLP field. The most commonly addressed problem has been the extraction of binary interactions, where the system identifies which protein pairs in a sentence have a biologically relevant relationship between them. Proposed solutions include both hand-crafted rule-based systems and machine learning approaches (see e.g. (Bunescu et al., 2005)). A wide range of results have been reported for the systems, but as we will show, differences in evaluation resources, metrics and strategies make direct comparison of these numbers problematic. Further, the results gained from the BioCreative II evaluation, where the best performing system achieved a 29% F-score (Hunter et al., 2008), suggest that the problem of extracting binary protein protein interactions is far from solved.

The public availability of large annotated PPI-corpora such as AImed (Bunescu et al., 2005), BioInfer (Pyysalo et al., 2007a) and GENIA (Kim et al., 2008), provides an opportunity for building PPI extraction systems automatically using machine learning. A major challenge is how to supply the learner with the contextual and syntactic information needed to distinguish between interactions and non-interactions. To address the ambiguity and variability of the natural language expressions used to state PPI, several recent studies have focused on the development, adaptation and application of NLP tools for the biomedical domain. Many high-quality domain-specific tools are now freely available, including full parsers such as that introduced by Charniak and Lease (2005). Additionally, a number of conversions from phrase structure parses to dependency structures that make the relationships between words more directly accessible have been introduced. These include conversions into representations such as the Stanford dependency scheme (de Marneffe et al., 2006) that are explicitly designed for information extraction purposes. However, specialized feature representations and kernels are required to make learning from such structures possible.

Approaches such as subsequence kernels (Bunescu and Mooney, 2006), tree kernels (Zelenko

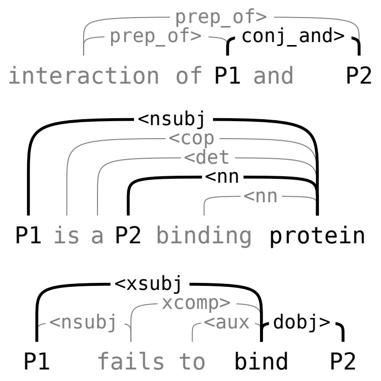

Figure 1: Stanford dependency parses ("collapsed" representation) where the shortest path, shown in bold, excludes important words.

et al., 2003) and shortest path kernels (Bunescu and Mooney, 2005) have been proposed and successfully used for relation extraction. However, these methods lack the expressive power to consider representations derived from general, possibly cyclic, dependency graph structures, such as those generated by the Stanford tools. The subsequence kernel approach does not consider parses at all, and the shortest path approach is limited to representing only a single path in the full dependency graph, which excludes relevant words even in many simple cases (Figure 1). Tree kernels can represent more complex structures, but are still restricted to tree representations.

Lately, in the framework of kernel-based machine learning methods there has been an increased interest in designing kernel functions for graph data. Building on the work of Gärtner et al. (2003), graph representations tailored for the task of dependency parse ranking were proposed by Pahikkala et al. (2006b). Though the proposed representations are not directly applicable to the task of PPI extraction, they offer insight in how to learn from dependency graphs. We develop a graph kernel approach for PPI extraction based on these ideas.

We next define a graph representation suitable for describing potential interactions and introduce a kernel which makes efficient learning from a general, unrestricted graph representation possible. Then we provide a short description of the sparse regularized least squares (sparse RLS) kernel-based machine learning method we use for PPI-extraction. Further, we rigorously assess our method on five publicly available PPI corpora, providing the first broad cross-corpus evaluation with a machine learning approach to PPI extraction. Finally, we discuss the effects that different evaluation strategies, choice of corpus and applied metrics have on measured performance, and conclude.

2 Method

We next present our graph representation, formalize the notion of graph kernels, and present our learning method of choice, the sparse RLS.

2.1 Graph encoding of sentence structure

As in most recent work on machine learning for PPI extraction, we cast the task as learning a decision function that determines for each unordered candidate pair of protein names occurring together in a sentence whether the two proteins interact. In the following, we first define the graph representation used to represent an interaction candidate pair. We then proceed to derive the kernel used to measure the similarities of these graphs.

We assume that the input of our learning method is a dependency parse of a sentence where a pair of protein names is marked as the candidate interaction for which an extraction decision must be made. Based on this, we form a weighted, directed graph that consists of two unconnected subgraphs. One represents the dependency structure of the sentence, and the other the linear order of the words (see Figure 2).

The first subgraph is built from the dependency analysis. One vertex and an associated set of labels is created in the graph for each token and for each dependency. The vertices that represent tokens have as labels the text and part-of-speech (POS) of the token. To ensure generalization of the learned extraction model, the labels of vertices that correspond to protein names are replaced with PROT1, PROT2 or PROT, where PROT1 and PROT2 are the pair of interest. The vertices that represent dependencies are labeled with the type of the dependency. The edges in the subgraph are defined so that each dependency vertex is connected by an incoming edge from the vertex representing its governor token, and by an outgoing edge to the vertex representing its de-

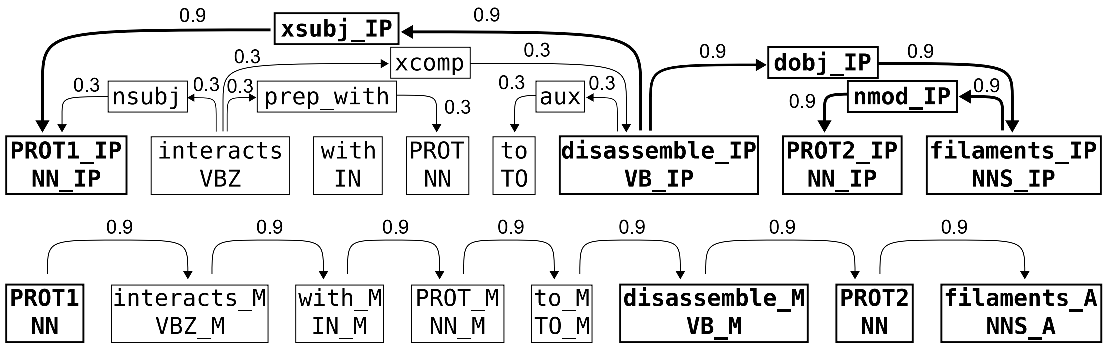

Figure 2: Graph representation generated from an example sentence. The candidate interaction pair is marked as PROT1 and PROT2, the third protein is marked as PROT. The shortest path between the proteins is shown in bold. In the dependency based subgraph all nodes in a shortest path are specialized using a post-tag (IP). In the linear order subgraph possible tags are (B)efore, (M)iddle, and (A)fter. For the other two candidate pairs in the sentence, graphs with the same structure but different weights and labels would be generated.

pendent token. The graph thus represents the entire sentence structure.

It is widely acknowledged that the words between the candidate entities or connecting them in a syntactic representation are particularly likely to carry information regarding their relationship; (Bunescu and Mooney, 2005) formalize this intuition for dependency graphs as the *shortest path hypothesis*. We apply this insight in two ways in the graph representation: the labels of the nodes on the shortest undirected paths connecting PROT1 and PROT2 are differentiated from the labels outside the paths using a special tag. Further, the edges are assigned weights; after limited preliminary experiments, we chose a simple weighting scheme where all edges on the shortest paths receive a weight of 0.9 and other edges receive a weight of 0.3. The representation thus allows us to emphasize the shortest path without completely disregarding potentially relevant words outside of the path.

The second subgraph is built from the linear structure of the sentence. For each token, a second vertex is created and the labels for the vertices are derived from the texts, POS-tags and named entity tagging as above. The labels of each word are specialized to denote whether the word appears before, in-between, or after the protein pair of interest. Each word node is connected by an edge to its succeeding word, as determined by sentence order the of the words. Each edge is given the weight 0.9.

2.2 The all-dependency-paths graph kernel

We next formalize the graph representation and present the all-dependency-paths kernel. This kernel can be considered as a practical instantiation of the theoretical graph kernel framework introduced by Gärtner et al. (2003). Let V be the set of vertices in the graph and $\mathcal{L}$ be the set of possible labels vertices can have. We represent the graph with an adjacency matrix $A \in \mathbb{R}^{|V|\times|V|}$, whose rows and columns are indexed by the vertices, and $[A]_{i,j}$ contains the weight of the edge connecting $v_i \in V$ and $v_j \in V$ if such an edge exists, and zero otherwise. Further, we represent the labels as a label allocation matrix $L \in \mathbb{R}^{|\mathcal{L}|\times|V|}$ so that $L_{i,j} = 1$ if the j-th vertex has the i-th label and $L_{i,j} = 0$ otherwise. Because only a very small fraction of all the possible labels are ever assigned to any single node, this matrix is extremely sparse.

It is well known that when an adjacency matrix is multiplied with itself, each element $[A^2]_{i,j}$ contains the summed weight of paths from vertex v_i to vertex v_j through one intervening vertex, that is, paths of length two. Similarly, for any length n, the summed weights from v_i to v_j can be determined by calculating $[A^n]_{i,j}$.

Since we are interested not only in paths of one specific length, it is natural to combine the effect of paths of different lengths by summing the powers of the adjacency matrices. We calculate the infinite sum of the weights of all possible paths connecting the vertices using the Neumann Series, defined as

$$(I - A)^{-1} = I + A + A^2 + ... = \sum_{k=0}^{\infty} A^k$$

if $|A| < 1$ where $|A|$ is the spectral radius of A (Meyer, 2000). From this sum we can form a new adjacency matrix

$$W = (I - A)^{-1} - I \ .$$

The final adjacency matrix contains the summed weights of all possible paths connecting the vertices. The identity matrix is subtracted to remove the paths of length zero, which would correspond to self-loops.

Next, we present the graph kernel that utilizes the graph representation defined previously. We define an instance G representing a candidate interaction as $G = LWL^{\mathrm{T}}$, where L and W are the label allocation matrix and the final adjacency matrix corresponding to the graph representation of the candidate interaction.

Following Gärtner et al. (2003) the graph kernel is defined as

$$k(G', G'') = \sum_{i=1}^{|\mathcal{L}|} \sum_{j=1}^{|\mathcal{L}|} G'_{i,j} G''_{i,j},$$

where G' and G'' are two instances formed as defined previously. The features can be thought as combinations of labels from connected pairs of vertices, with a value that represents the strength of their connection. In practical implementations, the full G matrices, which consist mostly of zeroes, are never explicitly formed. Rather, only the non-zero elements are stored in memory and used when calculating the kernels.

2.3 Scalable learning with Sparse RLS

RLS is a state-of-the-art kernel-based machine learning method which has been shown to have comparable performance to support vector machines (Rifkin et al., 2003). We choose the sparse version of the algorithm, also known as subset of regressors, as it allows us to scale up the method to very large training set sizes. Sparse RLS also has the property that it is possible to perform cross-validation and regularization parameter selection so that their time complexities are negligible compared to the training complexity. These efficient methods are analogous to the ones proposed by Pahikkala et al. (2006a) for the basic RLS regression.

We now briefly present the basic sparse RLS algorithm. Let m denote the training set size and $M = \{1, \ldots, m\}$ an index set in which the indices refer to the examples in the training set. Instead of allowing functions that can be expressed as a linear combination over the whole training set, as in the case of basic RLS regression, we only allow functions of the following restricted type:

$$f(\cdot) = \sum_{i \in B} a_i k(\cdot, x_i), \qquad (1)$$

where k is the kernel function, x_i are training data points, $a_i \in \mathbb{R}$ are weights, and the set indexing the basis vectors $B \subset M$ is selected in advance. The coefficients a_i that determine (1) are obtained by minimizing

$$\sum_{i=1}^{m} (y_i - \sum_{j \in B} a_j k(x_i, x_j))^2 + \lambda \sum_{i,j \in B} a_i a_j k(x_i, x_j),$$

where the first term is the squared loss function, the second term is the regularizer, and $\lambda \in \mathbb{R}_+$ is a regularization parameter. Note that all the training instances are used for determining the coefficient vector. The minimizer is obtained by solving the corresponding system of linear equations, which can be performed in $O(m|B|^2)$ time.

We set the maximum number of basis vectors to 4000 in all experiments in this study. The subset is selected randomly when the training set size exceeds this number. Other methods for the selection of the basis vectors were considered by Rifkin et al. (2003), who however reported that the random selection worked as well as the more sophisticated approaches.

3 Experimental evaluation

We next describe the evaluation resources and metrics used, provide a comprehensive evaluation of our method across five PPI corpora, and compare our results to earlier work. Further, we discuss the challenges inherent in providing a valid method evaluation and propose solutions.

	Statistics		Graph Kernel						Co-occ.	
Corpus	#POS.	#NEG.	P	R	F	σ_F	AUC	σ_{AUC}	P	F
AIMed	1000	4834	0.529	0.618	0.564	0.050	0.848	0.023	0.178	0.301
BioInfer	1370	8924	0.477	0.599	0.529	0.053	0.849	0.065	0.135	0.237
HPRD50	163	270	0.643	0.658	0.634	0.114	0.797	0.063	0.389	0.554
IEPA	335	482	0.696	0.827	0.751	0.070	0.851	0.051	0.408	0.576
LLL	164	166	0.725	0.872	0.768	0.178	0.834	0.122	0.559	0.703

Table 1: Counts of positive and negative examples in the corpora and (P)recision, (R)ecall (F)-score and AUC for the graph kernel, with standard deviations provided for F and AUC.

3.1 Corpora and evaluation criteria

We evaluate our method using five publicly available corpora that contain PPI interaction annotation: AImed (Bunescu et al., 2005), BioInfer (Pyysalo et al., 2007a), HPRD50 (Fundel et al., 2007), IEPA (Ding et al., 2002) and LLL (Nédellec, 2005). All the corpora were processed to a common format using transformations[1] that we have introduced earlier (Pyysalo et al., 2008). We parse these corpora with the Charniak-Lease parser (Charniak and Lease, 2005), which has been found to perform best among a number of parsers tested in recent domain evaluations (Clegg and Shepherd, 2007; Pyysalo et al., 2007b). The Charniak-Lease phrase structure parses are transformed into the collapsed Stanford dependency scheme using the Stanford tools (de Marneffe et al., 2006). We cast the PPI extraction task as binary classification, where protein pairs that are stated to interact are positive examples and other co-occuring pairs negative. Thus, from each sentence, $\binom{n}{2}$ examples are generated, where n is the number of occurrences of protein names in the sentence. Finally, we form the graph representation described earlier for each candidate interaction.

We evaluate the method with 10-fold document-level cross-validation on all of the corpora. This guarantees the maximal use of the available data, and also allows comparison to relevant earlier work. In particular, on the AImed corpus we apply the exact same 10-fold split that was used by Bunescu et al. (2006) and Giuliano et al. (2006). Performance is measured according to the following criteria: interactions are considered untyped, undirected pairwise relations between specific protein mentions, that is, if the same protein name occurs multiple times in a sentence, the correct interactions must be extracted for each occurrence. Further, we do not consider self-interactions as candidates and remove them from the corpora prior to evaluation.

[1]Available at `http://mars.cs.utu.fi/PPICorpora`.

The majority of PPI extraction system evaluations use the balanced F-score measure for quantifying the performance of the systems. This metric is defined as $F = \frac{2pr}{p+r}$, where p is precision and r recall. Likewise, we provide F-score, precision, and recall values in our evaluation. It should be noted that F-score is very sensitive to the underlying positive/negative pair distribution of the corpus — a property whose impact on evaluation is discussed in detail below. As an alternative to F-score, we also evaluate the performance of our system using the *area under the receiver operating characteristics curve* (AUC) measure (Hanley and McNeil, 1982). AUC has the important property that it is invariant to the class distribution of the used dataset. Due to this and other beneficial properties for comparative evaluation, the usage of AUC for performance evaluation has been recently advocated in the machine learning community (see e.g. (Bradley, 1997)). Formally, AUC can be defined as

$$AUC = \frac{\sum_{i=1}^{m_+} \sum_{j=1}^{m_-} H(x_i - y_i)}{m_+ m_-},$$

where m_+ and m_- are the numbers of positive and negative examples, respectively, and x_1,...,x_{m_+} are the outputs of the system for the positive, and y_1,...,y_{m_-} for the negative examples, and

$$H(r) = \begin{cases} 1, & \text{if } r > 0 \\ 0.5, & \text{if } r = 0 \\ 0, & \text{otherwise.} \end{cases}$$

The measure corresponds to the probability that given a randomly chosen positive and negative example, the system will be able to correctly disinguish which one is which.

3.2 Performance across corpora

The performance of our method on the five corpora for the various metrics is presented in Table 1. For reference, we show also the performance of the co-occurrence (or *all-true*) baseline, which simply assigns each candidate into the interaction class. The recall of the co-occurrence method is trivially 100%, and in terms of AUC it has a score of 0.5, the random baseline. All the numbers in Table 1 are averages taken over the ten folds. One should note that because of the non-linearity of the F-score measure, the average precision and recall will not produce exactly the average F.

The results hold several interesting findings. First, we briefly observe that on the AImed corpus, which has recently been applied in numerous evaluations (Sætre et al., 2008) and can be seen as an emerging *de facto* standard for PPI extraction method evaluation, the method achieves an F-score performance of 56.4%. As we argue in more detail below, this level of performance is comparable to the state-of-the-art in machine learning based PPI extraction. For the other large corpus, BioInfer, F-score performance is slightly lower.

Second, we observe that the F-score performance of the method varies strikingly between the different corpora, with results on IEPA and LLL approximately 20 percentage units higher than on AImed and BioInfer, despite the larger size of the latter two. In our previous work we have observed similar results with a rule-based extraction method (Pyysalo et al., 2008). As the first broad cross-corpus evaluation using a state-of-the-art machine learning method for PPI extraction, our results support and extend the key finding that F-score performance results measured on different corpora cannot, in general, be meaningfully compared.

The co-occurrence baseline numbers indicate one reason for the high F-score variance between the corpora. The F-score metric is not invariant to the distribution of positive and negative examples: for example, halving the number of negative test examples is expected to approximately halve the number of false positives at a given recall point. Thus, the greater the fraction of true interactions in a corpus is, the easier it is to reach high performance in terms of F-score. This is reflected in co-occurrence results, which range from 24% to 70% depending on the class distribution of the corpus.

This is a critical weakness of the F-score metric in cross-corpus comparisons as, for example, the fraction of true interactions out of all candidates is 50% on the LLL corpus but only 17% on AImed. By contrast to the large differences in performance measured using F-score, we find that for the distribution-invariant AUC measure the performance for all of the AImed, BioInfer, IEPA, and LLL corpora falls in the narrow range of 83-85%. In terms of AUC, performance on the HPRD50 corpus is an outlier, being approximately three percentage units lower than for any other corpus. Nevertheless, the results provide a strong argument in favor of applying the AUC metric instead of, or in addition to, F-score. AUC is also more stable in terms of variance.

Finally, we note that the similar performance in terms of AUC for corpora with as widely differing sizes as LLL and BioInfer indicates that past a relatively modest number of examples, increasing corpus size has a surprisingly small effect on the performance of the method. A similar finding can be seen, for example, in the relatively flat learning curve of Giuliano et al. (2006). While the issue requires further investigation, these results suggest that there may be more value in investing effort in developing better learning methods as opposed to larger corpora.

3.3 Performance compared to other methods

We next discuss the performance of our method compared to other methods introduced in the literature and the challenges of meaningful comparison, where we identify three major issues.

First, as indicated by the results above, differences in the makeup of different corpora render cross-corpus comparisons in terms of F-score essentially meaningless. As F-score is typically the only metric for which results are reported in the PPI extraction literature, we are limited to comparing against results on single corpora. We consider the AImed and BioInfer evaluations to be the most relevant ones, as these corpora are sufficiently large for training and reliably testing machine learning methods. As the present study is, to the best of our knowl-

	P	R	F
(Giuliano et al., 2006)	60.9%	57.2%	59.0%
All-dependency-paths graph kernel	52.9%	61.8%	56.4%
(Bunescu and Mooney, 2006)	65.0%	46.4%	54.2%
(Sætre et al., 2008)	64.3%	44.1%	52.0%
(Mitsumori et al., 2006)	54.2%	42.6%	47.7%
(Yakushiji et al., 2005)	33.7%	33.1%	33.4%

Table 2: (P)recision, (R)ecall and (F)-score results for methods evaluated on AImed with the correct cross-validation methodology.

edge, the first to report machine learning method performance on BioInfer, we will focus on AImed in the following comparison.

Second, the cross-validation strategy used in evaluation has a large impact on measured performance. In earlier system evaluations, two major strategies for defining the splits used in cross-validation can be observed. The approach used by Bunescu and Mooney (2006), which we consider the correct one, is to split the data into folds on level of documents. This guarantees that all pairs generated from the same document are always either in the training set or in the test set. Another approach is to pool all the generated pairs together, and then randomly split them to folds. To illustrate the significance of this choice, consider two interaction candidates extracted from the same sentence, e.g. from a statement of the form "P_1 and P_2 [...] P_3", where "[...]" is any statement of interaction or non-interaction. Due to the near-identity of contexts, a machine learning method will easily learn to predict that the label of the pair (P_1, P_2) should match that of (P_1, P_3). However, such "learning" will clearly not generalize. This approach must thus be considered invalid, because allowing pairs generated from same sentences to appear in different folds leads to an information leak between the training and test sets. Sætre et al. (2008) observed that adopting the latter cross-validation strategy on AImed could lead *up to 18 F-score percentage unit overestimation of performance*. For this reason, we will not consider results listed in the "False 10-fold cross-validation" table (2b) of Sætre et al. (2008).

With these restrictions in place, we now turn to comparison with relevant results reported in related research, summarized in Table 2. We note that Bunescu and Mooney (2006) only applied evaluation criteria where it is enough to extract only one occurrence of each mention of an interaction from each abstract, while the other results shown were evaluated using the same criteria as applied here. The former approach can produce higher performance: the evaluation of Giuliano et al. (2006) includes both alternatives, and their method achieves an F-score of 63.9% under the former criterion, which they term One Answer per Relation in a given Document (OARD). Our method outperforms most studies using similar evaluation methodology, with the exception being the approach of Giuliano et al. (2006). This result is somewhat surprising, as the method proposed by Giuliano does not apply any form of parsing but relies instead only on the sequential order of the words. This brings us to our third point regarding comparability of methods. As pointed out by Sætre et al. (2008), the AImed corpus allows remarkably different "interpretations" regarding the number of interacting and non-interacting pairs. For example, where we have identified 1000 interacting and 4834 non-interacting protein pairs in AImed, in the data used by Giuliano there are eight more interacting and 200 fewer non-interacting pairs. The corpus can also be preprocessed in a number of ways. In particular we noticed that whereas protein names are always blinded in our data, in the data used by Giuliano protein names are sometimes partly left visible. As Giuliano has generously made his method implementation available[2], we were able to test the performance of his system on the data we used in our experiments. This resulted in an F-score of 52.4%.

Finally, there remains an issue of parameter selection. For sparse RLS the values of the regular-

[2] Available at `http://tcc.itc.it/research/textec/tools-resources/jsre.html`.

ization parameter λ and the decision threshold separating the positive and negative classes must be chosen, which can be problematic when no separate data for choosing them is available. Choosing from several parameter values the ones that give best results in testing, or picking the best point from a precision/recall curve when evaluating in terms of F-score, will lead to an overoptimistic evaluation of performance. This issue has often not been addressed in earlier evaluations that do cross-validation on a whole corpus. We choose the parameters by doing further leave-one-document-out cross-validation within each round of 10-fold-cross-validation, on the nine folds that constitute the training set.

As a conclusion, we observe the results achieved with the all-dependency-paths kernel to be state-of-the-art level. However, differences in evaluation strategies and the large variance exhibited in the results make it impossible to state which of the systems considered can be expected in general to perform best. We encourage future PPI-system evaluations to report AUC and F-score results over multiple corpora, following clearly defined evaluation strategies, to bring further clarity to this issue.

4 Conclusions and future work

In this paper we have proposed a graph kernel approach to extracting protein-protein interactions, which captures the information in unrestricted dependency graphs to a format that kernel based learning algorithms can process. The method combines syntactic analysis with a representation of the linear order of the sentence, and considers all possible paths connecting any two vertices in the resulting graph. We demonstrate state-of-the art performance for the approach. All software developed in the course of this study is made publicly available at `http://mars.cs.utu.fi/PPICorpora`.

We identify a number of issues which make results achieved with different evaluation strategies and resources incomparable, or even incorrect. In our experimental design we consider the problems related to differences across corpora, the effects different cross-validation strategies have, and how parameter selection can be done. Our recommendation is to provide evaluations over different corpora, to use document-level cross-validation and to always selected parameters on the training set.

We draw attention to the behaviour of the F-score metric over corpora with differing pair distributions. The higher the relative frequency of interacting pairs is, the higher the performance can be expected to be. This is noticed both for the graph kernel method and for the naive co-occurrence baseline. Indeed, the strategy of just stating that all pairs interact leads to as high result as 70% F-score on one of the corpora. We consider AUC as an alternative measure that does not exhibit such behaviour, as it is invariant to the distribution of pairs. The AUC metric is much more stable across all the corpora, and never gives better results than random for approaches such as the naive co-occurrence.

Though we only consider binary interactions in this work, the graph representations have the property that they could be used to represent more complex structures than pairs. The availability of corpora that annotate complex interactions, such as the full BioInfer and GENIA, makes training a PPI extraction system for extracting complex interactions an important avenue of future research. However, how to avoid the combinatorial explosion following from considering triplets, quartets etc. remains an open question. Also, the performance of the current approaches may need to be yet improved before extending them to recognize complex interactions.

Acknowledgements

We would like to thank Razvan Bunescu, Claudio Giuliano and Rune Sætre for their generous assistance in providing us with data, software and information about their work on PPI extraction. Further, we thank CSC, the Finnish IT center for science, for providing us extensive computational resources. This work has been supported by the Academy of Finland and the Finnish Funding Agency for Technology and Innovation, Tekes.

References

Andrew P. Bradley. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159.

Razvan Bunescu and Raymond Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of HLT/EMNLP'05*, pages 724–731.

Razvan Bunescu and Raymond Mooney. 2006. Subsequence kernels for relation extraction. In *Proceedings of NIPS'05*, pages 171–178. MIT Press.

Razvan C. Bunescu, Ruifang Ge, Rohit J. Kate, Edward M. Marcotte, Raymond J. Mooney, Arun Kumar Ramani, and Yuk Wah Wong. 2005. Comparative experiments on learning information extractors for proteins and their interactions. *Artif Intell Med*, 33(2):139–155.

Eugene Charniak and Matthew Lease. 2005. Parsing biomedical literature. In *Proceedings of IJCNLP'05*, pages 58–69.

Andrew Brian Clegg and Adrian Shepherd. 2007. Benchmarking natural-language parsers for biological applications using dependency graphs. *BMC Bioinformatics*, 8(1):24.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC'06*, pages 449–454.

J. Ding, D. Berleant, D. Nettleton, and E. Wurtele. 2002. Mining MEDLINE: abstracts, sentences, or phrases? In *Proceedings of PSB'02*, pages 326–337.

Katrin Fundel, Robert Kuffner, and Ralf Zimmer. 2007. RelEx–Relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371.

Thomas Gärtner, Peter A. Flach, and Stefan Wrobel. 2003. On graph kernels: Hardness results and efficient alternatives. In *COLT'03*, pages 129–143. Springer.

Claudio Giuliano, Alberto Lavelli, and Lorenza Romano. 2006. Exploiting shallow linguistic information for relation extraction from biomedical literature. In *Proceedings of EACL'06*.

James A. Hanley and B. J. McNeil. 1982. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36.

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-specific gene expression. *BMC Bioinformatics*, 9(78).

Jin-Dong Kim, Tomoko Ohta, and Jun'ichi Tsujii. 2008. Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics*, 9(10).

Carl D. Meyer. 2000. *Matrix analysis and applied linear algebra*. Society for Industrial and Applied Mathematics.

Tomohiro Mitsumori, Masaki Murata, Yasushi Fukuda, Kouichi Doi, and Hirohumi Doi. 2006. Extracting protein-protein interaction information from biomedical text with svm. *IEICE - Trans. Inf. Syst.*, E89-D(8):2464–2466.

Claire Nédellec. 2005. Learning language in logic - genic interaction extraction challenge. In *Proceedings of LLL'05*.

Tapio Pahikkala, Jorma Boberg, and Tapio Salakoski. 2006a. Fast n-fold cross-validation for regularized least-squares. In *Proceedings of SCAI'06*, pages 83–90.

Tapio Pahikkala, Evgeni Tsivtsivadze, Jorma Boberg, and Tapio Salakoski. 2006b. Graph kernels versus graph representations: a case study in parse ranking. In *Proceedings of the ECML/PKDD'06 workshop on Mining and Learning with Graphs*.

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

Sampo Pyysalo, Filip Ginter, Veronika Laippala, Katri Haverinen, Juho Heimonen, and Tapio Salakoski. 2007b. On the unification of syntactic annotations under the stanford dependency scheme: A case study on BioInfer and GENIA. In *Proceedings of BioNLP'07*, pages 25–32.

Sampo Pyysalo, Antti Airola, Juho Heimonen, Jari Björne, Filip Ginter, and Tapio Salakoski. 2008. Comparative analysis of five protein-protein interaction corpora. *BMC Bioinformatics, special issue*, 9(Suppl 3):S6.

Ryan Rifkin, Gene Yeo, and Tomaso Poggio, 2003. *Regularized Least-squares Classification*, volume 190 of *NATO Science Series III: Computer and System Sciences*, chapter 7, pages 131–154. IOS Press.

Rune Sætre, Kenji Sagae, and Jun'ichi Tsujii. 2008. Syntactic features for protein-protein interaction extraction. In *Proceedings of LBM'07*, volume 319, pages 6.1–6.14.

Akane Yakushiji, Yusuke Miyao, Yuka Tateisi, and Jun'ichi Tsujii. 2005. Biomedical information extraction with predicate-argument structure patterns. In *Proceedings of SMBM'05*, pages 60–69.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *J. Mach. Learn. Res.*, 3:1083–1106.