Analyzing text in search of bio-molecular events: a high-precision machine learning framework

Sofie Van Landeghem^{1,2}**, Yvan Saeys**^{1,2}**, Bernard De Baets**^{3}**, Yves Van de Peer**^{1,2}
1. Dept. of Plant Systems Biology, VIB
2. Dept. of Plant Biotechnology and Genetics, Ghent University
3. Dept. of Applied Mathematics, Biometrics and Process Control, Ghent University
B-9000 Gent, Belgium
yves.vandepeer@psb.vib-ugent.be

Abstract

The BioNLP'09 Shared Task on Event Extraction is a challenge which concerns the detection of bio-molecular events from text. In this paper, we present a detailed account of the challenges encountered during the construction of a machine learning framework for participation in this task. We have focused our work mainly around the filtering of false positives, creating a high-precision extraction method. We have tested techniques such as SVMs, feature selection and various filters for data pre- and post-processing, and report on the influence on performance for each of them. To detect negation and speculation in text, we describe a custom-made rule-based system which is simple in design, but effective in performance.

1 Introduction

BioNLP recently emerged from the combined expertise of molecular biology and computational linguistics. At first, the community was mainly focused on named entity recognition (NER) and simple relation extraction, such as protein-protein interactions (Plake et al., 2005; Giuliano et al., 2006; Fundel et al., 2007; Saetre et al., 2008). However, the future of BioNLP lies in the ability to extract more complex events from text, in order to fully capture all available information (Altman et al., 2008).

Two recent community-wide challenges, Biocreative I (Hirschman et al., 2005) and II (Krallinger et al., 2008) have shown their merits by providing common benchmarking data and a meaningful comparison of various techniques. In contrast to the monolithic Biocreative tasks, the BioNLP'09 Shared Task has a more modular nature (Kim et al., 2009). It is not concerned with named entity recognition or normalization, but focuses on the task of event extraction itself.

This article is organized as follows: we first describe the Shared Task in a little more detail. Next, we present the methods used in our machine learning framework, carefully discussing our choices in design and their influence on performance. We then present the final results of our approach. Finally, we draw conclusions from our participation in this task, and suggest some future work for our own research as well as on a community-wide level.

2 BioNLP'09 Shared Task

2.1 Subtasks

The BioNLP'09 Shared Task was divided into three subtasks, of which only the first one was mandatory. We have participated in tasks 1 and 3, and will therefore only briefly discuss task 2. In accordance with the provided gold entity annotation, we will refer to all genes and gene products as *proteins*.

Task 1 represents the core of the challenge: detection and characterization of bio-molecular events from text. There are 9 distinct event types. Six events influence proteins directly, and we will refer to them as 'Protein events'. Five of them are unary: Localization, Gene expression, Transcription, Protein catabolism and Phosphorylation. The Binding event can be related to one protein (e.g. protein-DNA binding), two proteins (e.g. protein-protein interaction) or more (e.g. a complex). On top of these event types, there are three Regulation events: Regulation, Positive regulation and Negative regulation. Each of them can be unary or binary. In the latter case, an extra argument specifying the cause of the regulation is added. Each argument of a Regulation event can be either a protein or any other event.

Participants in task 2 had to recognise extra arguments for the events from task 1. For example, the cellular location should be added to a Localization event, and Site arguments had to be specified for Phosphorylation, Binding and Regulation.

Finally, task 3 was about detecting negation and speculation in text.

2.2 Examples

Suppose we are dealing with this sentence:

> "MAD-3 masks the nuclear localization signal of p65 and inhibits p65 DNA binding."

There are three proteins in this sentence:

- T1 : Protein : 'MAD-3'
- T2 : Protein : 'p65' (first occurrence)
- T3 : Protein : 'p65' (second occurrence)

There are also three triggers, which are defined by a contiguous stream of characters from the original text, and point to a specific event type:

- T27 : Negative regulation : 'masks'
- T29 : Negative regulation : 'inhibits'
- T30 : Binding : 'binding'

In this example, we see there is one binding event which involves trigger T30 and protein T3. Furthermore, this binding event is being influenced by protein T1, using trigger T29 which implies a Negative regulation event. Similarly, T1 has a negative effect on protein T2, which is expressed by trigger T27. When participating in subtask 2, one should also find the extra Site argument T28 for this last event:

- T28 : Entity : 'nuclear localization signal'

Now look at the following example:

> "NF-kappa B p50 is not directly regulated by I kappa B."

This sentence expresses a Regulation event involving the trigger 'regulated' and protein 'p50'. Participation in subtask 3 requires detecting the negation of this event.

2.3 Datasets

Both training and testing data consist of PubMed abstracts extracted from the GENIA corpus (Kim et al., 2008). All proteins are annotated and extra information is provided, such as analysis of sentence segmentation and tokenization, dependency graphs and phrase structure parses.

The training data consists of 800 articles. The development data contains an additional 150 articles with gold standard annotations. During development (6 weeks), the system's performance could be estimated with this dataset, using an online submission system. Participants had one week time to provide predictions for the final test dataset of 260 articles.

3 Methods

Our machine learning framework is tailored towards specific properties of different events, but is still kept sufficiently general to deal with new event types. The nature of the event extraction task leads to unbalanced datasets, with much more negative examples than positive ones. This is due to the fact that proteins could be involved in all possible event types, and each of the words in the text could be a trigger for an event. Finding the right events thus seems like looking for a needle in a haystack, which is why it is crucial to start with a good definition of candidate instances. This problem has motivated us to try and filter out as many irrelevant negative instances as possible by introducing specific preprocessing methods and filters. This reduces unbalancedness of the datasets and will lead to better precision as there will be less false positives (FPs). High-precision systems produce less noise and can be considered to be more useful when a researcher is trying to extract reliable interaction networks from text. There is a considerable degree of information redundancy in the original PubMed articles, which makes up for low recall when using the system in a real-world application. We have also tested a few post-processing techniques in order to remove FPs after classification.

Figure 1 shows a high-level overview of the different modules in our framework. More details are described in the next sections.

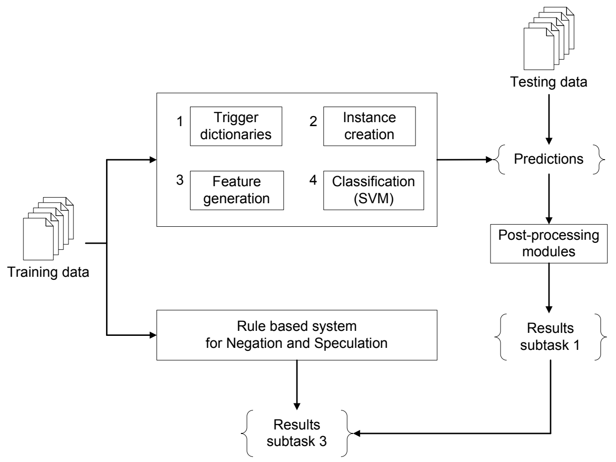

Figure 1: High-level overview of the modules used in our framework.

3.1 Parsing

For sentence segmentation, we made use of the provided tokenization files. Analysis of part-of-speech tags and dependency graphs was done using the Stanford parser (de Marneffe et al., 2006).

3.2 Dictionaries of triggers

From the training data, we automatically compiled dictionaries of triggers for each event type, applying the Porter stemming algorithm (Porter, 1980) to each trigger. This resulted in some entries in the dictionaries which were of limited use, such as 'through' for Binding, or 'are' for Localization. Such words are too general or too vague, and lead to many negative and irrelevant instances. For this reason, we *manually* cleaned the dictionaries, only keeping specific triggers for each event type (e.g. 'interaction' for Binding and 'secretion' for Localization).

During development, we noticed a significant difference between the triggers for unary Binding events (e.g. 'homodimer', 'binding site') and those for Binding events with multiple arguments (e.g. 'heterodimer', 'complex'). This motivated our choice to create two separate dictionaries and classifiers, thus discarding irrelevant candidate instances. Such an example would be a candidate binary Binding event with the trigger 'homodimer', while homodimerization is clearly a unary event. In the rest of this article, we will refer to these two event types as Single binding and Multiple binding events. The revision of the dictionaries resulted in a significant drop in the number of Binding instances in the training data, and improved the balancedness of the datasets: from a total of 34 612 instances (of which 2% positives) to 4708 Single binding instances (11% positives) and 3861 Multiple binding instances (5% positives).

Following the same reasoning, Regulation was also divided into unary and binary events. Furthermore, we have carefully analysed the nature of Binary regulation events, and noticed that a vast majority of these events had a protein in the 'cause' slot. We decided to split up the dictionaries of Binary regulations accordingly, differentiating between regulation events caused by proteins and those caused by other events. This keeps the more general words (e.g. 'causes') out of the dictionaries of events regulated by proteins (e.g. 'response'), again resulting in better balance of the datasets.

3.3 Instance creation

In a machine learning framework, a classifier tries to distinguish between positive instances (true biomolecular events) and negative instances (candidates which should be discarded). To run such a framework, one has to define candidate instances automatically by scanning the text. The first step towards instance creation consists of looking up triggers in text, using the constructed dictionaries for each event type. To this end, we have implemented a fast algorithm using Radix trees[1]. Next, candidate arguments have to be found. Initially, we have selected all (combinations of) proteins that were mentioned in the same sentence. However, this may result in a lot of negative and irrelevant instances, mainly in long sentences. This is why we have implemented a Negative-instances (NI) filter, which checks whether the length of the sub-sentence spanned by a candidate event does not exceed a certain value. Figure 2 shows the distribution of positive and negative Multiple binding events, according to the length of the relevant sub-sentence. It seems reasonable to only keep instances with a sub-sentence of less than 175 characters, as this includes almost all positive examples, while at the same time removing a significant amount of irrelevant negatives.

[1]Java implementation by Tahseen Ur Rehman, `http://code.google.com/p/radixtree/`

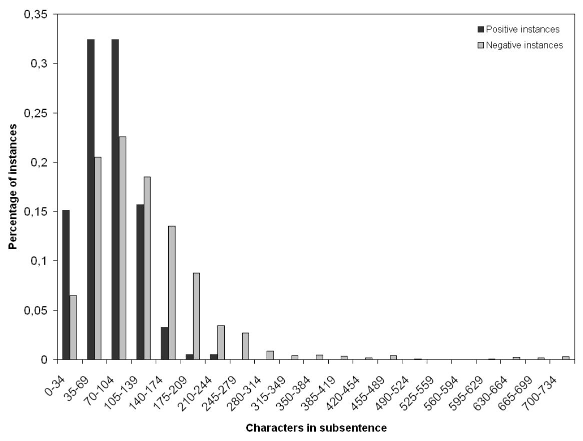

Figure 2: Distribution of Multiple binding instances, according to the length of the sub-sentence (training data).

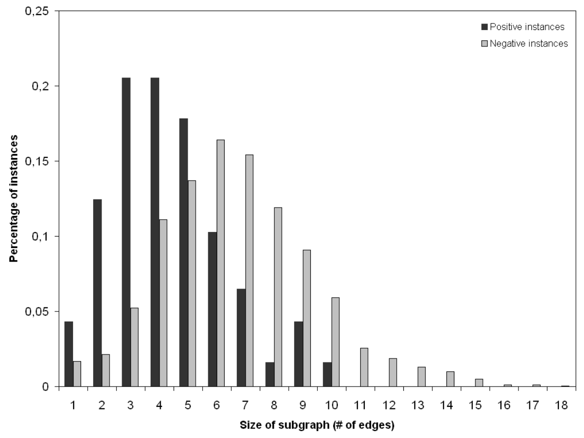

Figure 3: Distribution of Multiple binding instances, according to the size of the subgraph (training data).

Furthermore, for each instance, a minimal subgraph of the dependency graph was extracted, containing the full trigger and all arguments. The size of this subgraph was also used as a parameter for the NI filter, as positive instances are usually expressed in a smaller subtree than negative examples. In Figure 3 we see how the subgraphs of positive Multiple binding instances are never larger than 10 edges, while negative instances can contain up to 18 edges. In this case, only keeping instances with subgraphs smaller than 8 edges will discard many irrelevant negatives, while keeping most of the positive instances.

The NI filter further reduces noise in the data and unbalancedness. We now end up with 4070 Single binding instances (of which 13% positives) and 2365 Multiple binding instances (8% positives). Table 1 shows the final distribution of instances for all event types. Transcription, Localization and Multiple binding have the lowest percentage of positive instances, ranging between 7% and 8%, while Phosphorylation has up to 48% positive instances. It should be noted that the number of positive instances in Table 1 is lower than the actual number of positive examples in the training set, due to limitations of our instance definition method. However, a study regarding maximal recall shows that we do not remove too many true positives (TPs) (more details in Section 4.1).

3.4 Feature generation

For feature generation, we base our method on the rich feature set we previously used in our work on protein-protein interactions (Van Landeghem et al., 2008). The goal of that study was to extract binary relations and only one path in the dependency graph was analyzed for each instance. In the present work however, we are processing larger and more complex subgraphs. This is why we have excluded 'edge walks', i.e. patterns of two consecutive edges and their common vertex (e.g. 'nsubj VBZ prep'). To compensate for the loss of information, we have added trigrams to the feature set. These are three stemmed consecutive words from the sub-sentence spanning the event, e.g. 'by induc transcript', which is the stemmed variant of 'by inducing transcription'. Other features include

- A BOW-approach by looking at all the words which appear at a vertex of the subgraph. This automatically excludes uninformative words such as prepositions.
- Lexical and syntactic information of triggers.
- Size of the subgraph.

Event type	**# neg. inst.**	**# pos. inst.**	**% pos. inst.**
Localization	3415	249	7
Single binding	3548	522	13
Multiple binding	2180	185	8
Gene expression	5356	1542	22
Transcription	6930	489	7
Protein catabolism	175	96	35
Phosphorylation	163	153	48

Table 1: Distribution of instances

Event type	Features
Localization	18 121
Single binding	21 332
Multiple binding	11 228
Gene expression	31 332
Transcription	30 306
Protein catabolism	1 883
Phosphorylation	2 185

Table 2: Dimensionality of the datasets

- Length of the sub-sentence.
- Extra features for Regulation events, storing whether the arguments are proteins or events, specifying the exact event type.
- Vertex walks which consist of two vertices and their connecting edge. For these patterns, both lexical as well as syntactic information is kept. When using lexical information, protein names and triggers were blinded in order to extract more general patterns (e.g. 'trigger nsubj protx' which expresses that the given protein is the subject of a trigger). Blinding avoids overfitting of the classifier.

In the training phase, each instance generates different patterns, and each pattern is stored as a numeric feature in the feature vector. During testing, we count how many times each feature is found for each instance. This results in very sparse and high-dimensional datasets. Table 2 shows the dimensionality of the datasets for all event types. Protein catabolism has the lowest dimensionality with 1883 features, while Transcription and Gene expression produce over 30 000 features.

3.5 Classification

To process our dataset, we had to find a classifier able to deal with thousands of instances, thousands of features, and an unbalancedness of up to 93% negative instances. We have used the LibSVM implementation as provided by WEKA[2], as a few preliminary tests using different classifiers (such as Random Forests) gave worse results. We integrated an internal 5-fold cross-validation loop on the training portion of the data to determine a useful C-parameter. All other parameters were left unchanged, including the type of kernel which is a radial basis function by default.

In combination with the LibSVM, we have tried applying feature selection (FS). At first sight, FS did not seem to lead to gain in performance, although we were not able to test this hypothesis more thoroughly due to time limitations of the task. Finally, we have also tested the influence of assigning higher weights to positive training instances, in order to make up for the unbalanced nature of the data, but this had almost no effect on overall performance.

3.6 Post-processing

We have implemented a few custom-made post-processing modules, designed to further reduce FPs and improve precision of our method. We report here on their influence on performance.

Overlapping triggers of different event types

Predictions for different event types were processed in parallel and merged afterwards. This means that two triggers of different event types might overlap, based on the same words in the text. However, a word in natural language can only have one meaning at a time. When two such triggers lead to events with different event types, this means that some of these events should be FPs. When testing on the development data, we found a few predictions where this problem occurred. For example, the trigger 'expression' can lead to both a Transcription and a Gene expression event, but not at the same time. In such a case, we only select the prediction with the highest SVM score. However, thanks to careful construction of the dictionaries (Section 3.2), their mutual overlap is rather small, and thus this post-processing module has almost no influence on performance.

Events based on the same trigger

One trigger might be involved in different events from the same event type. For example, the sentence 'it induces expression of STAT5-regulated genes in CTLL-2, i.e. beta-casein, and oncostatin M (OSM)' mentions two Gene expression events based on the trigger 'expression', one involving beta-casein, and one involving OSM. For these two events, the subgraphs will be very similar, resulting in similar features and SVM scores. However, often a trigger only leads to one true event, while all other candi-

[2]Available at `http://www.cs.waikato.ac.nz/ml/weka/`

dates from the same event type are false positives. We have carefully benchmarked this hypothesis, and found that for Protein catabolism and Phosphorylation, we could achieve better performance by only keeping the top-ranked prediction. Up to 5% in F-score could be gained for these events. This is due to the fact that for these two event types, usually only one true event is linked to each trigger.

3.7 Negation

We found that there are three major categories of event negation:

1. A negation construct is found in the close vicinity of the trigger (e.g. 'no', 'failure to').
2. A trigger already expresses negation by itself (e.g. 'non-expressing', 'immobilization').
3. A trigger in a certain sentence expresses both positive as negative events. In this case, the pattern 'but not' is often used (e.g. 'overexpression of Vav, but not SLP-76, augments CD28-induced IL-2 promoter activity').

We have created a custom-made rule-based system to process these three categories. The rules make use of small dictionaries collected from the training data. For rule 1, we checked whether a negation word appears right in front of the trigger. To apply rule 2, we used a list of inherent negative triggers deduced from the training set. For rule 3, we checked whether we could find patterns such as 'but not' or 'whereas', negating only the event involving the protein mentioned right after that pattern.

3.8 Speculation

We identified two major reasons why the description of an event could be regarded as speculation instead of a mere fact. These categories are:

1. Uncertainty: the authors state the interactions or events they are investigating, without knowing the true results (yet). This is often indicated with expressions such as 'we have examined whether (...)'.
2. Hypothesis: authors formulate a hypothesis to try and explain the results of an experiment. Specific speculation words such as 'might' or 'appear to' often occur right before the trigger.

Event type	Maximal recall
Localization	84.91 %
Binding	78.23 %
Gene expression	91.57 %
Transcription	90.24 %
Protein catabolism	100 %
Phosphorylation	95.74 %
Regulation	46.15 %
Positive regulation	39.71 %
Negative regulation	43.88 %
Negation	28.97 %
Speculation	25.26 %

Table 3: Maximal recall for the development data

Similar to detecting negation, we compiled a list of relevant expressions from the training data and have used this to implement a simple rule-based system. For rule 1, we checked the appearance of such an expression in a range of 60 characters before the trigger and up to 60 characters after the trigger. Rule 2 was applied on a smaller range: only 20 characters right before the trigger were scanned.

4 Results

Our final machine learning framework consists of all the modules described in the previous section. To summarize, these design choices were made: automatically compiled dictionaries which were cleaned manually, usage of the NI filter, no weights on positive instances, a LibSVM classifier and no feature selection. We used both post-processing modules, but the second one only for Protein catabolism and Phosphorylation events. The best SVM cut-offs were chosen by determining the best F-score on the development data for each classifier.

4.1 Benchmarking on the development data

Protein events

To evaluate maximal recall of our instance extraction method, we executed an evaluation using an all-true classifier. As can be seen in Table 3, maximal recall is quite high for almost all Protein events, meaning that dictionary coverage is good, our NI filter does not remove too many TPs, and not too many events are expressed across sentences and thus not picked up by our method. Binding and Localization are the only events with less than 90% recall. Due to

Event type	**Recall**	**Precision**	**F-score**
Localization	77.36	91.11	83.67
Binding	45.16	37.21	40.80
Gene expression	70.79	79.94	75.08
Transcription	60.98	75.76	67.57
Protein catabolism	80.95	89.47	85.00
Phosphorylation	68.09	88.89	77.11
Total	62.45	64.40	63.41
Regulation	23.67	41.67	30.19
Positive regulation	21.56	38.00	27.51
Negative regulation	30.10	41.26	34.81
Total	23.63	39.39	29.54
Task 1	41.03	53.50	**46.44**
Negation	15.89	45.95	23.61
Speculation	20.00	26.87	22.93
Total	17.82	33.65	23.30
Task 3	38.77	52.24	**44.51**

Table 4: Final performance of all events for the development data

time constraints, we were not able to test which of our modules leads to false negative (FN) instances.

For each event, we have determined the best classifier cut-offs to achieve maximal F-score. Results of the final performance for the predictions of Protein events on the development data, can be seen in Table 4. For most events, we achieve very high precision, thanks to our careful definition of instances in combination with the NI-filter.

Looking at the F-measures, Transcription, Gene expression and Phosphorylation all perform between 67 and 77%, while Localization and Protein catabolism have an F-score of more than 83%. It becomes clear that Binding is the most difficult event type, with a performance of 41% F. Unfortunately, this group of events contains 44% of all Protein events, greatly influencing total performance. Average performance of predicting Protein events results in 63.41% F.

Regulation

When evaluating the predictions of Regulation events, one has to take into account that the performance greatly depends on the ability of our system to predict Protein events. Indeed, one FN Protein event can lead to multiple FN Regulation events, and the same holds for FPs. Furthermore, we do not try to extract events across sentences, which may lead to more FNs. To study maximal recall of the Regulation events, we have again applied an all-true classifier. Table 3 shows that the highest possible recall of the Regulation events is never above 50%, greatly limiting the performance of our method.

As regulation events can participate in new regulation events, one should run the regulation pipeline repeatedly until no more new events are found. In our experiments, we have found that even the first recursive run did not lead to much better performance, and only a few more Regulation events were found.

Final results are shown in Table 4. With recall being rather low, between 21% and 30%, at least we achieve relatively good precision: around 40% for each of the three regulation types. On average, the F-score is almost 30% for the regulation events, which is significantly lower than the performance of Protein events. On average, we obtain an F-score of 46.44% on the development data for task 1.

Negation and speculation

The performance of this subtask depends heavily on the performance of subtask 1. Again we have applied an all-true classifier to determine maximal recall (Table 3). Less than 30% of the events necessary for task 3 can be found with our setup; all of these FNs are due to FNs in task 1.

Final results are shown in Table 4. Performance of around 23% F-score is achieved on the development data. We take into consideration that according to the maximal recall study, only 29% of the necessary events for Negation were extracted by task 1. In the final results, 16% of all the negation events were found. This means that our rule-based method by itself achieves about 55% recall for Negation. Similarly, the system has a recall of 80% for Speculation when only considering events found in task 1. We conclude that our simple rule-based system performs reasonably well.

4.2 Scoring and ranking on final test set

Finally, our system was applied to the test data. Achieving a global F-score of 40.54% for subtask 1, we obtain a 5th place out of 24 participating teams. For subtask 3 of finding negation and speculation, we obtain a second place with a 37.80% F-score.

Final results for each of the event types are shown in Table 5. As on the development data, we see

Event type	Recall	Precision	F-score
Localization	43.68	78.35	56.09
Binding	38.04	38.60	38.32
Gene expression	59.42	81.56	68.75
Transcription	39.42	60.67	47.79
Protein catabolism	64.29	60.00	62.07
Phosphorylation	56.30	89.41	69.09
Total	50.75	67.24	57.85
Regulation	10.65	22.79	14.52
Positive regulation	17.19	32.19	22.41
Negative regulation	22.96	35.22	27.80
Total	17.36	31.61	22.41
Task 1	33.41	51.55	**40.54**
Negation	10.57	45.10	17.13
Speculation	8.65	15.79	11.18
Total	9.66	24.85	13.91
Task 3	30.55	49.57	**37.80**

Table 5: Performance of all events for the final test set

that the Binding event performs worst, and the same trend is found when analyzing results of other teams. In general however, we achieve a high precision: 67% for Protein events, 52% on average on subtask 1, and 50% on average on subtask 3. Another trend which is confirmed by other teams, is the fact that predicting Protein events achieves much higher performance than the prediction of Regulation events.

Compared to our results on the development data (Table 4), we notice a drop of performance for the Protein events of about 0.06F. This loss is propagated to the Regulation events and to Negation and Speculation, each also performing about 0.06F worse than on the development data. We believe this drop in performance might be due to overfitting of the system during training. It is difficult to find the best SVM cut-offs to achieve maximal performance. We have tuned these cut-offs on the development data, but they might not be ideal for the final test set. For this reason, we believe that it might be more representative to use evaluation schemes such as the area under the receiver operating characteristics curve (AUC) measure (Hanley and McNeil, 1982; Airola et al., 2008).

5 Conclusions and future work

We have participated in the BioNLP'09 Shared Task, joining the rest of the community in the progression of relation-based extraction towards the extraction of events from bio-molecular texts. Out of the 24 participants, we see quite some teams with a very good performance, with the highest result achieving an F-score of nearly 52%. We believe the community is off to a good start in this task, and we hope work in this field will continue afterwards.

In our own study, we notice that the task of extracting bio-molecular events leads to high-dimensional and unbalanced datasets. We carefully designed our system in order to improve balance of the datasets and to avoid false positives. For feature generation, we have made use of a modified bag-of-words approach, included trigrams extracted from the sentence, and derived patterns from dependency graphs. Our high-precision framework achieves a fifth position out of 24 participating teams in subtask 1, and second position out of six for subtask 3.

In the future, we would like to investigate the use of feature selection to produce better models for the classification task. Another interesting topic would be how to combine coreference resolution with dependency graphs in order to process events which span multiple sentences in text.

For the community as a whole, we think the next step would be to work on full articles instead of mere abstracts. Also, it might be interesting to investigate the use of text-bound annotation which is not necessarily contiguous, such as is the case in the Bioinfer corpus (Pyysalo et al., 2007), to be able to fully capture the semantics of a certain event.

Acknowledgments

SVL and YS would like to thank the Research Foundation Flanders (FWO) for funding their research. Furthermore, the authors would like to thank the organizers of the BioNLP'09 Shared Task for offering to the community a very valuable and well organized task about event extraction. We believe careful evaluation and discussion of the results will lead to a significant step forward in this domain.

References

A. Airola, S. Pyysalo, J. Björne, T. Pahikkala, F. Ginter and T. Salakoski. 2008. All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. *BMC Bioinformatics*, 9(Suppl 11):S2

R.B. Altman, C.M. Bergman, J. Blake, C. Blaschke, A. Cohen, F. Gannon, L. Grivell, U. Hahn, W. Hersh, L. Hirschman, L.J. Jensen, M. Krallinger, B. Mons, S.I. O'Donoghue, M.C. Peitsch, D. Rebholz-Schuhmann, H. Shatkay and A. Valencia. 2008. Text mining for biology - the way forward: opinions from leading scientists. *Genome Biology*, 9(Suppl 2):S7

B. Boser, I. Guyon and V.N. Vapnik. 1992. A training algorithm for optimal margin classifiers. *Proceedings of the 5th annual workshop on Computational learning theory (COLT)*, 144-152

K. Fundel, R. Küffner and R. Zimmer. 2007. RelEx—Relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365-371

C. Giuliano, A. Lavelli and L. Romano 2006. Exploiting shallow linguistic information for relation extraction from biomedical literature. *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 401-408

J. 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

L. Hirschman, A. Yeh, C. Blaschke and A. Valencia. 2005. Overview of BioCreAtIvE: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6(Suppl 1):S1

J.-D. Kim, T. Ohta and J. Tsujii. 2008. Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics*, 19(Suppl 1):i180-i182

J.-D. Kim, T. Ohta, S. Pyssalo, Y. Kano and J. Tsujii. 2009. Overview of BioNLP'09 Shared Task on Event Extraction, *Proceedings of Natural Language Processing in Biomedicine (BioNLP) NAACL 2009 Workshop*, to appear

M. Krallinger, A. Morgan, L. Smith, F. Leitner, L. Tanabe, J. Wilbur, L. Hirschman and A. Valencia. 2008. Evaluation of text-mining systems for biology: overview of the Second BioCreative community challenge. *Genome Biology*, 9(Suppl 2):S1

MC. de Marneffe, B. MacCartney and C.D. Manning. 2006. Generating typed dependency parses from phrase structure parses. *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC)*, 449-454

C. Plake, J. Hakenberg and U. Leser. 2005. Optimizing syntax patterns for discovering protein-protein interactions. *Proceedings of the 2005 ACM symposium on Applied computing (SAC)*, 195-201

M.F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3), 130-137

S. Pyysalo, F. Ginter, J. Heimonen, J. Björne, J. Boberg, J. Järvinen and T. Salakoski. 2007. BioInfer: A corpus for information extraction in the biomedical domain. *BMC Bioinformatics*, 8(50)

R. Saetre, K. Sagae and J. Tsujii. 2008. Syntactic features for protein-protein interaction extraction. *Proceedings of the 2nd International Symposium on Languages in Biology and Medicine (LBM)*, 6.1-6.14

S. Van Landeghem, Y. Saeys, B. De Baets and Y. Van de Peer. 2008. Extracting protein-protein interactions from text using rich feature vectors and feature selection. *Proceedings of the Third International Symposium on Semantic Mining in Biomedicine (SMBM)*, 77-84.