Generalizing Biomedical Event Extraction

Jari Björne and **Tapio Salakoski**
Department of Information Technology, University of Turku
Turku Centre for Computer Science (TUCS)
Joukahaisenkatu 3-5, 20520 Turku, Finland
firstname.lastname@utu.fi

Abstract

We present a system for extracting biomedical events (detailed descriptions of biomolecular interactions) from research articles. This system was developed for the BioNLP'11 Shared Task and extends our BioNLP'09 Shared Task winning Turku Event Extraction System. It uses support vector machines to first detect event-defining words, followed by detection of their relationships. The theme of the BioNLP'11 Shared Task is generalization, extending event extraction to varied biomedical domains. Our current system successfully predicts events for every domain case introduced in the BioNLP'11 Shared Task, being the only system to participate in all eight tasks and all of their subtasks, with best performance in four tasks.

1 Introduction

Biomedical event extraction is the process of automatically detecting statements of molecular interactions in research articles. Using natural language processing techniques, an event extraction system predicts relations between proteins/genes and the processes they take part in. Manually annotated corpora are used to evaluate event extraction techniques and to train machine-learning based systems.

Event extraction was popularised by the BioNLP'09 Shared Task on Event Extraction (Kim et al., 2009), providing a more detailed alternative for the older approach of binary interaction detection, where each pair of protein names co-occurring in the text is classified as interacting or not. Events extend this formalism by adding to the relations *direction*, *type* and *nesting*. Events define the type of interaction, such as *phosphorylation*, and commonly mark in the text a *trigger word* (e.g. "phosphorylates") describing the interaction. Directed events can define the role of their protein or gene arguments as e.g. *cause* or *theme*, the agent or the target of the biological process. Finally, events can act as arguments of other events, creating complex nested structures that accurately describe the biological interactions stated in the text. For example, in the case of a sentence stating "Stat3 phosphorylation is regulated by Vav", a *phosphorylation*-event would itself be the argument of a *regulation*-event.

We developed for the BioNLP'09 Shared Task the Turku Event Extraction System, achieving the best performance at 51.95% F-score (Björne et al., 2009). This system separated event extraction into multiple classification tasks, detecting individually the trigger words defining events, and the arguments that describe which proteins or genes take part in these events. Other approaches used in the Shared Task included e.g. joint inference (Riedel et al., 2009). An overall notable trend was the use of full dependency parsing (Buyko et al., 2009; Van Landeghem et al., 2009; Kilicoglu and Bergler, 2009).

In the following years, event extraction has been the subject of continuous development. In 2009, after the BioNLP'09 Shared Task, we extended our system and improved its performance to 52.85% (Björne et al., 2011). In 2010, the system introduced by Miwa et. al. reached a new record performance of 56.00% (Miwa et al., 2010a).

Proceedings of BioNLP Shared Task 2011 Workshop, pages 183–191,
Portland, Oregon, USA, 24 June, 2011.

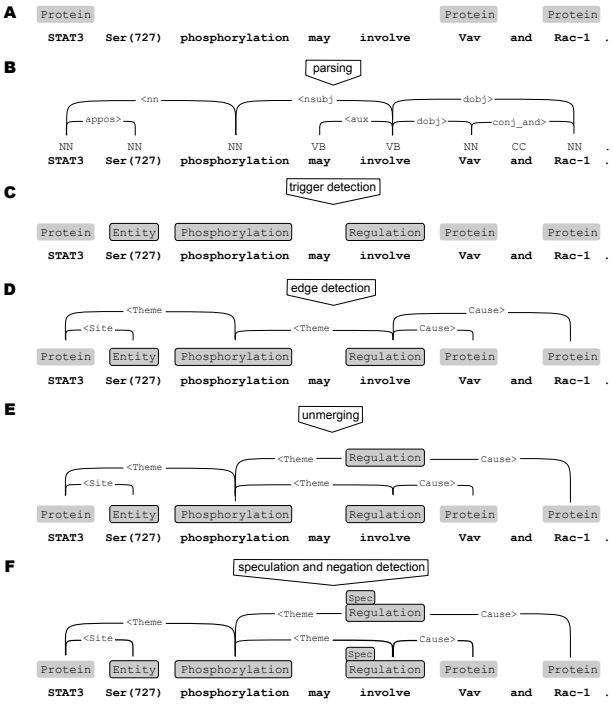

Figure 1: Event extraction. In most tasks named entities are given (A). Sentences are parsed (B) to produce a dependency parse. Entities not given are predicted through trigger detection (C). Edge detection predicts event arguments between entities (D) and unmerging creates events (E). Finally, event modality is predicted (F). When the graph is converted to the Shared Task format, site arguments are paired with core arguments that have the same target protein.

In 2010, we applied the Turku Event Extraction System to detecting events in all 18 million PubMed abstracts, showing its scalability and generalizability into real-world data beyond domain corpora (Björne et al., 2010). In the current BioNLP'11 Shared Task[1] (Kim et al., 2011), we demonstrate its generalizability to different event extraction tasks by applying what is, to a large extent, the same system to every single task and subtask.

2 System Overview

Our system divides event extraction into three main steps (Figure 1 C, D and E). First, entities are predicted for each word in a sentence. Then, arguments are predicted between entities. Finally, entity/argument sets are separated into individual events.

[1] http://sites.google.com/site/bionlpst/

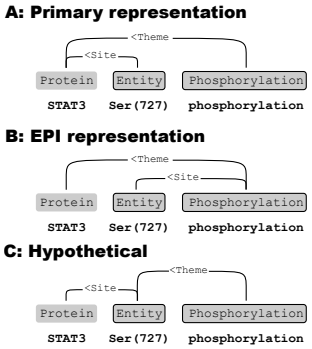

Figure 2: Site argument representation. Site arguments add detail to core arguments. (A) In most tasks we link both core and site arguments to given protein nodes. This minimizes the number of outgoing edges per trigger node, simplifying unmerging, but loses the connection between site and core arguments. (B) In the EPI task, all events with site-arguments have a single core argument, so linking sites to the trigger node preserves the site/core connection. (C) To both limit number of arguments in trigger nodes and preserve site information, event arguments using sites could be linked to protein nodes through the site entity. However, in this approach the core argument would remain undetected if the site wasn't detected.

2.1 Graph Representation

The BioNLP'11 Shared Task consists of eight separate tasks. Most of these follow the BioNLP'09 Shared Task annotation scheme, which defines events as having a trigger entity and one or more arguments that link to other events or protein/gene entities. This annotation can be represented as a graph, with trigger and protein/gene entities as nodes, and arguments (e.g. *theme*) as edges. In our graph representation, an event is defined implicitly as a trigger node and its outgoing edges (see Figure 1 F).

Most of the BioNLP'11 Shared Task tasks define task-specific annotation terminology, but largely follow the BioNLP'09 definition of events. Some new annotation schemes, such as the bracket notation in the CO-task can be viewed simply as alternative representations of arguments. The major new feature is *relations* or *triggerless events*, used in the REL, REN, BB and BI tasks. In our graph representation, this type of event is a single, directed edge.

Some event arguments have a matching *site* argument that determines the part of the protein the argument refers to (Figure 2). To allow detection of core arguments independently of site arguments, in

most tasks we link site arguments directly to proteins (Figure 2 A). This maximises extraction performance on core events, but losing the connection between site and core arguments limits performance on site arguments.

To further simplify event extraction all sentences are processed in isolation, so events crossing sentence boundaries (intersentence events, Table 2) cannot be detected. This also limits the theoretical maximum performance of the system (see Figure 3).

In the provided data an event is annotated only once for a set of equivalent proteins. For example, in the sentence "Ubiquitination of caspase 8 (casp8)" a *ubiquitination* event would be annotated only for "caspase 8", "casp8" being marked as equivalent to "caspase 8". To improve training data consistency, our system fully resolves these equivalences into new events, also recursively when a duplicated event is nested in another event (Table 2). Resolved equivalences were used for event extraction in the BioNLP'11 GE, ID, EPI and BB tasks, although based on tests with the GE dataset their impact on performance was negligible.

2.2 Machine Learning

The machine learning based event detection components classify examples into one of the positive classes or as negatives, based on a feature vector representation of the data. To make these classifications, we use the SVM^{multiclass} support vector machine[2] (Tsochantaridis et al., 2005) with a linear kernel. An SVM must be optimized for each classification task by experimentally determining the regularization parameter C. This is done by training the system on a training dataset, and testing a number of C values on a development dataset. When producing predictions for the test set, the classifier is retrained with combined training and development sets, and the test data is classified with the previously determined optimal value of C.

Unlike in the BioNLP'09 Shared Task where the three main parameters (trigger-detector, recall-adjustment and edge-detector) were optimized in an exhaustive grid search against the final metric, in the new system only the recall-adjustment parameter (see Section 2.5) is optimized against the final metric, edge and trigger detector parameters being optimized in isolation to speed up experiments.

[2] `http://svmlight.joachims.org/svm_multiclass.html`

2.3 Syntactic Analyses

The machine learning features that are used in event detection are mostly derived from the syntactic parses of the sentences. Parsing links together related words that may be distant in their linear order, creating a parse tree (see Figure 1 B).

We used the Charniak-Johnson parser (Charniak and Johnson, 2005) with David McClosky's biomodel (McClosky, 2010) trained on the GENIA corpus and unlabeled PubMed articles. The parse trees produced by the Charniak-Johnson parser were further processed with the Stanford conversion tool (de Marneffe et al., 2006), creating a dependency parse (de Marneffe and Manning, 2008).

In the supporting tasks (REL, REN and CO) this parsing was done by us, but in the main tasks the organizers provided official parses which were used. All parses for tasks where named entities were given as gold data were further processed with a *protein name splitter* that divides at punctuation tokens which contain named entities, such as "p50/p65" or "GATA3-binding".

2.4 Feature Groups

To convert text into features understood by the classifier, a number of analyses are performed on the sentences, mostly resulting in binary features stating the presence or absence of some feature. Applicable combinations of these features are then used by the trigger detection, edge detection and unmerging steps of the event extraction system.

Token features can be generated for each word token, and they define the text of the token, its Porter-stem (Porter, 1980), its Penn treebank part-of-speech-tag, character bi- and trigrams, presence of punctuation or numeric characters etc.

Sentence features define the number of named entities in the sentence as well as bag-of-words counts for all words.

Dependency chains follow the syntactic dependencies up to a depth of three, starting from a token of interest. They are used to define the immediate context of these words.

Dependency path *N*-grams, are built from the shortest undirected path of tokens and dependencies linking together two entities, and are used in edge detection. *N*-grams join together a token with its two flanking dependencies as well as each dependency with its two flanking tokens. While these *N*-grams follow the direction of the entire path, the governor-dependent directions of individual dependencies are used to define token bigrams.

Trigger features can be built in cases where triggers are already present, such as edge detection and event construction. These features include the types and supertypes of the trigger nodes, and combinations thereof.

External features are additional features based on data external to the corpus being processed. Such features can include e.g. the presence of a word in a list of key terms, Wordnet hypernyms, or other resources that enhance performance on a particular task. These are described in detail in Section 3.

2.5 Trigger Detection

Trigger words are detected by classifying each token as negative or as one of the positive trigger classes. Sometimes several triggers overlap, in which case a merged class (e.g. *phosphorylation–regulation*) is used. After trigger prediction, triggers of merged classes are split into their component classes.

Most tasks evaluate trigger detection using approximate span, so detecting a single token is enough. However, this token must be chosen consistently for the classifier to be able to make accurate predictions. For multi-token triggers, we select as the trigger word the *syntactic head*, the root token of the dependency parse subtree covering the entity.

When optimizing the SVM C-parameter for trigger and edge detection, it is optimized in isolation, maximizing the F-score for that classification task. Edges can be predicted for an event only if its trigger has been detected, but often the C-parameter that maximizes trigger detection F-score has too low recall for optimal edge detection. A *recall adjustment* step is used to fit together the trigger and edge detectors. For each example, the classifier gives a confidence score for each potential class, and picks as the predicted class the one with the highest score. In recall adjustment, the confidence score of each negative example is multiplied with a multiplier, and if the result falls below the score of another class, that class becomes the new classification. This multiplier is determined experimentally by optimizing against overall system performance, using the official task metric for cases where a downloadable evaluator is available (GE and BB).

2.6 Edge Detection

Edge detection is used to predict event arguments or triggerless events and relations, all of which are defined as edges in the graph representation. The edge detector defines one example per direction for each pair of entities in the sentence, and uses the SVM classifier to classify the examples as negatives or as belonging to one of the positive classes. As with the trigger detector, overlapping positive classes are predicted through merged classes (e.g. *cause–theme*). Task-specific rules defining valid argument types for each entity type are used to considerably reduce the number of examples that can only be negatives.

2.7 Unmerging

In the graph representation, events are defined through their trigger word node, resulting in overlapping nodes for overlapping events. The trigger detector can however predict a maximum of one trigger node per type for each token. When edges are predicted between these nodes, the result is a *merged graph* where overlapping events are merged into a single node and its set of outgoing edges. Taking into account the limits of trigger prediction, the edge detector is also trained on a merged graph version of the gold data.

To produce the final events, these merged nodes need to be "pulled apart" into valid trigger and argument combinations. In the BioNLP'09 Shared Task, this was done with a rule-based system. Since then, further research has been done on machine learning approaches for this question (Miwa et al., 2010b; Heimonen et al., 2010). In our current system, unmerging is done as an SVM-classification step. An example is constructed for each argument edge combination of each predicted node, and classified as a true event or a false event to be removed. Tested on the BioNLP'09 Shared Task data, this system performs roughly on par with our earlier rule-based system, but has the advantage of being more general and thus applicable to all BioNLP'11 Shared Task

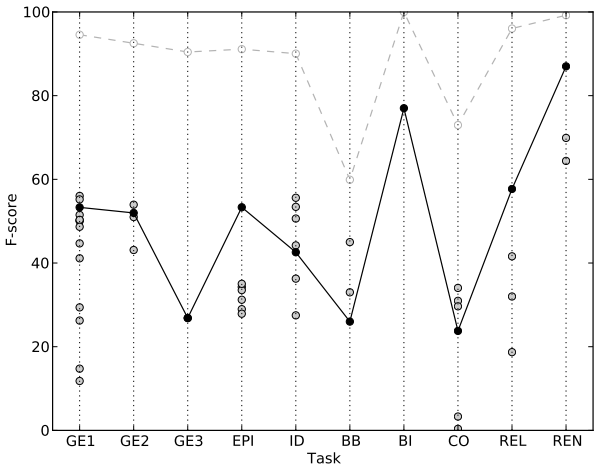

Figure 3: Ranking of the systems participating in the BioNLP'11 Shared Task. Our system is marked with black dots and the dotted line shows its theoretical maximum performance (see Section 2.1) with all correct classifications.

tasks. The unmerging step is not required for *triggerless events* which are defined by a single edge.

All of the tasks define varied, detailed limits on valid event type and argument combinations. A final validation step based on task-specific rules is used to remove structurally incorrect events left over from preceding machine learning steps.

2.8 Modality Detection

Speculation and negation are detected independently, with binary classification of trigger nodes. The features used are mostly the same as for trigger detection, with the addition of a list of speculation-related words based on the BioNLP'09 ST corpus.

3 Tasks and Results

The BioNLP'11 Shared Task consists of five main tasks and three supporting tasks. Additionally, many of these tasks specify separate subtasks. Except for the GE-task, which defines three main evaluation criteria, all tasks have a single primary evaluation criterion. All evaluations are based on F-score, the harmonic mean of precision and recall. Performance of all systems participating in the BioNLP'11 Shared Task is shown in Figure 3. Our system's performance on both development and test sets of all tasks is shown in Table 1.

Corpus	Devel F	Test F
GE'09 task 1	56.27	53.15
GE'09 task 2	54.25	50.68
GE task 1	55.78	53.30
GE task 2	53.39	51.97
GE task 3	38.34	26.86
EPI	56.41	53.33
ID	44.92	42.57
BB	27.01	26
BI	77.24	77
CO	36.22	23.77
REL	65.99	57.7
REN	84.62	87.0

Table 1: Devel and test results for all tasks. The performance of our new system on the BioNLP'09 ST GENIA dataset is shown for reference, with task 3 omitted due to a changed metric. For GE-tasks, the Approximate Span & Recursive matching criterion is used.

3.1 GENIA (GE)

The GENIA task is the direct continuation of the BioNLP'09 Shared Task. The BioNLP'09 ST corpus consisted only of abstracts. The new version extends this data by 30% with full text PubMed Central articles.

Our system applied to the GE task is the most similar to the one we developed for the BioNLP'09 Shared Task. The major difference is the replacement of the rule-based unmerging component with an SVM based one.

The GE task has three subtasks, task 1 is detection of events with their main arguments, task 2 extends this to detection of sites defining the exact molecular location of interactions, and task 3 adds the detection of whether events are stated in a negated or speculative context.

For task 3, speculation and negation detection, we considered the GE, EPI and ID task corpora similar enough to train a single model on. Compared to training on GE alone, example classification F-score decreased for negation by 8 pp and increased for speculation by 4 pp. Overall task 3 processing was considerably simplified.

Our system placed third in task 1, second in task 2 and first in task 3. Task 1 had the most participants, making it the most useful for evaluating overall performance. Our F-score of 53.30% was within three percentage points of the best performing system (by

Corpus	sentences	events	equiv events	nesting events	intersentence events	neg/spec events
GE'09	8906	11285	7.9%	38.8%	6.0%	12.1%
GE	11581	14496	6.6%	37.2%	6.0%	13.3%
EPI	7648	2684	9.1%	10.2%	9.3%	10.1%
ID	3193	2931	5.3%	21.3%	3.9%	4.9%
BB	1762	5843	79.4%	N/A	86.0%	0%
BI	120	458	0%	N/A	0%	0%
CO	8906	5284	0%	N/A	8.5%	N/A
REL	8906	2440	4.2%	N/A	0%	0%
REN	13235	373	0%	N/A	2.4%	0%

Table 2: Corpus statistics. Numbers are for all available annotated data, i.e. the merged training and development sets.

team FAUST), indicating that our chosen event detection approach still remains competitive. For reference, we ran our system also on the BioNLP'09 data, reaching an F-score of 53.15%, a slight increase over the 52.85% we previously reported in Björne et al. (2011).

3.2 Epigenetics and Post-translational Modifications (EPI)

All events in the EPI task that have additional arguments (comparable to the site-arguments in the GE-task) have a single core argument. We therefore use for this task a slightly modified graph representation, where all additional arguments are treated as core arguments, linking directly to the event node (Figure 2 B). The number of argument combinations per predicted event node remains manageable for the unmerging system and full recovery of additional arguments is possible.

Eight of the EPI event types have corresponding reverse events, such as *phosphorylation* and *dephosphorylation*. Many of these reverse events are quite rare, resulting in too little training data for the trigger detector to find them. Therefore we merge each reverse event type into its corresponding forward event type. After trigger detection, an additional rule-based step separates them again. Most of the reverse classes are characterized by a "de"-prefix in their trigger word. On the EPI training dataset, the rule-based step determined correctly whether an event was reversed in 99.6% of cases (1698 out of 1704 events). Using this approach, primary criterion F-score on the development set increased 1.33 percentage points from 55.08% to 56.41%. Several previously undetectable small reverse classes became detectable, with e.g. *deubiquitination* (8 instances in the development set) detected at 77.78% F-score.

Our system ranked first on the EPI task, outperforming the next-best system (team FAUST) by over 18 percentage points. On the alternative core metric our system was also the first, but the FAUST system was very close with only a 0.27 percentage point difference. Since the core metric disregards additional arguments, it may be that our alternative approach for representing these arguments (Figure 2 B) was important for the primary criterion difference.

3.3 Infectious Diseases (ID)

The annotation scheme for the ID task closely follows the GE task, except for an additional *process* event type that may have no arguments, and for five different entity types in place of the *protein* type. Our approach for the ID task was identical to the GE task, but performance relative to the other teams was considerably lower. Primary evaluation metric F-score was 42.57% vs. 43.44% for the core metric which disregards additional arguments, indicating that these are not the reason for low performance.

3.4 Bacteria Biotopes (BB)

The BB task considers detection of events describing bacteria and their habitats. The task defines only two event types but a large number of entity types which fall into five supertypes. All entities must be predicted and all events are triggerless.

Unlike in the other main tasks, in the BB task exact spans are required for *Bacterium*-type entities, which usually consist of more than one token (e.g. *B. subtilis*). After trigger detection, a rule-based step attempts to extend predicted trigger spans forwards and backwards to cover the correct span. When extending the spans of BB training set gold entity head

tokens, this step produced the correct span for 91% (399 out of 440) of *Bacterium*-type entities.

To aid in detecting *Bacterium*-entities a list of bacteria names from the List of Prokaryotic names with Standing in Nomenclature[3] was used (Euzéby, 1997) as external features. To help in detecting the heterogeneous habitat-entities, synonyms and hypernyms from Wordnet were used (Fellbaum, 1998). The development set lacked some event classes, so we moved some documents from the training set to the development set to include these.

Our F-score was the lowest of the three participating systems, and detailed results show a consistently lower performance in detecting the entities. The large number of intersentence events (Table 2) also considerably limited performance (Figure 3).

3.5 Bacteria Gene Interactions (BI)

The BI-task considers events related to genetic processes of the bacterium *Bacillus subtilis*. This task defines a large number of both entity and event types, but all entities are given as gold-standard data, therefore we start from edge detection (Figure 1 D). All BI events are triggerless.

In this task manually curated syntactic parses are provided. As also automated parses were available, we tested them as an alternative. With the Charniak-Johnson/McClosky parses overall performance was only 0.65 percentage points lower (76.59% vs. 77.24%). As with the BB task, we moved some documents from the training set to the development set to include missing classes.

Despite this task being very straightforward compared to the other tasks we were the only participant. Therefore, too many conclusions shouldn't be drawn from the performance, except to note that a rather high F-score is to be expected with all the entities being given as gold data.

3.6 Protein/Gene Coreference (CO)

In the CO supporting task the goal is to extract anaphoric expressions. Even though our event extraction system was not developed with coreference resolution in mind, the graph representation can be used for the coreference annotation, making coreference detection possible. *Anaphoras* and *Antecedents* are both represented as *Exp*-type entities, with *Coref*-type edges linking *Anaphora*-entities to *Antecedent*-entities and *Target*-type edges linking *Protein*-type entities to *Antecedent*-entities.

In the CO-task, character spans for detected entities must be in the range of a full span and minimum span. Therefore in this task we used an alternative trigger detector. Instead of predicting one trigger per token, this component predicted one trigger per each syntactic phrase created by the Charniak-Johnson parser. Since these phrases don't cover most of the CO-task triggers, they were further subdivided into additional phrases, e.g. by cutting away determiners and creating an extra phrase for each noun-token, with the aim of maximizing the number of included triggers and minimizing the number of candidates.

Our system placed fourth out of six, reaching an F-score of 23.77%. Coreference resolution being a new subject for us and our system not being developed for this domain, we consider this an encouraging result, but conclude that in general dedicated systems should be used for coreference resolution.

3.7 Entity Relations (REL)

The REL supporting task concerns the detection of static relationships, *Subunit-Complex* relations between individual proteins and protein complexes and *Protein-Component* relations between a gene or protein and its component, such as a protein domain or gene promoter. In the graph representation these relations are defined as edges that link together given protein/gene names and *Entity*-type entities that are detected by the trigger detector.

To improve entity detection, additional features are used. Derived from the REL annotation, these features highlight structures typical for biomolecular components, such as aminoacids and their shorthand forms, domains, motifs, loci, termini and promoters. Many of the REL entities span multiple tokens. Since the trigger detector predicts one entity per token, additional features are defined to mark whether a token is part of a known multi-token name.

Our system had the best performance out of four participating systems with an F-score of 57.7%, over 16 percentage points higher than the next. Development set results show that performance for the two event classes was very close, 66.40% for Protein-Component and 65.23% for Subunit-Complex.

[3] http://www.bacterio.cict.fr/

3.8 Bacteria Gene Renaming (REN)

The REN supporting task is aimed at detecting statements of *B. Subtilis* gene renaming where a synonym is introduced for a gene. The REL task defines a single relation type, *Renaming*, and a single entity type, *Gene*. All entities are given, so only edge detection is required. Unlike the other tasks, the main evaluation criterion ignores the direction of the relations, so they are processed as *undirected edges* in the graph representation.

Edge detection performance was improved with external features based on two sources defining known *B. Subtilis* synonym pairs: The Uniprot *B. Subtilis* gene list "bacsu"[4] and *Subti*Wiki[5], the *B. Subtilis* research community annotation wiki.

For the 300 renaming relations in the REN training data, the synonym pair was found from the Uniprot list in 66% (199 cases), from *Subti*Wiki in 79% (237 cases) and from either resource in 81.3% (244 cases). For the corresponding negative edge examples, Uniprot or *Subti*Wiki synonym pairs appeared in only 2.1% (351 out of 16640 examples).

At 87.0% F-score our system had the highest performance out of the three participants, exceeding the next highest system by 17.1 percentage points. If Uniprot and *Subti*Wiki features are not used, performance on the development set is still 67.85%, close to the second highest performing system on the task.

4 Conclusions

We have developed a system that addresses all tasks and subtasks in the BioNLP'11 Shared Task, with top performance in several tasks. With the modular design of the system, all tasks could be implemented with relatively small modifications to the processing pipeline. The graph representation which covered naturally all different task annotations was a key feature in enabling fast system development and testing. As with the Turku Event Extraction System developed for the BioNLP'09 Shared Task, we release this improved system for the BioNLP community under an open source license at `bionlp.utu.fi`.

Of all the tasks, the GE-task, which extends the BioNLP'09 corpus, is best suited for evaluating advances in event extraction in the past two years. Comparing our system's performance on the GE'09 corpus with the current one, we can assume that the two corpora are of roughly equal difficulty. Therefore we can reason that overall event extraction performance has increased about three percentage points, the highest performance on the current GE-task being 56.04% by team FAUST. It appears that event extraction is a hard problem, and that the immediate easy performance increases have already been found. We hope the BioNLP'11 Shared Task has focused more interest in the field, hopefully eventually leading to breakthroughs in event extraction and bringing performance closer to established fields of BioNLP such as syntactic parsing or named entity recognition.

That our system could be generalized to work on all tasks and subtasks, indicates that the event extraction approach can offer working solutions for several biomedical domains. A potential limiting factor currently is that most task-specific corpora annotate a non-overlapping set of sentences, necessitating the development of task-specific machine learning models. Training on multiple datasets could mean that positives of one task would be unannotated on text from the other task, confusing the classifier. On the other hand, multiple overlapping task annotations on the same text would permit the system to learn from the interactions and delineations of different annotations. System generalization has been successfully shown in the BioNLP'11 Shared Task, but has resulted in a number of separate extraction systems. It could well be that the future of event extraction requires also the generalization of corpus annotations.

As future directions, we intend to further improve the scope and usability of our event extraction system. We will also continue our work on PubMed-scale event extraction, possibly applying some of the new extraction targets introduced by the BioNLP'11 Shared Task.

Acknowledgments

We thank the Academy of Finland for funding, CSC — IT Center for Science Ltd for computational resources and Filip Ginter and Sofie Van Landeghem for help with the manuscript.

[4] http://www.uniprot.org/docs/bacsu

[5] http://subtiwiki.uni-goettingen.de/

References

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 BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 10–18, Boulder, Colorado. Association for Computational Linguistics.

Jari Björne, Filip Ginter, Sampo Pyysalo, Jun'ichi Tsujii, and Tapio Salakoski. 2010. Scaling up biomedical event extraction to the entire PubMed. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*, pages 28–36, Uppsala, Sweden, July. Association for Computational Linguistics.

Jari Björne, Juho Heimonen, Filip Ginter, Antti Airola, Tapio Pahikkala, and Tapio Salakoski. 2011. Extracting contextualized complex biological events with rich graph-based feature sets. *Computational Intelligence, Special issue on Extracting Bio-molecular Events from Literature*. To appear, accepted in 2009.

Ekaterina Buyko, Erik Faessler, Joachim Wermter, and Udo Hahn. 2009. Event extraction from trimmed dependency graphs. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 19–27. ACL.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180. Association for Computational Linguistics.

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

J. P. Euzéby. 1997. List of bacterial names with standing in nomenclature: a folder available on the internet. *Int J Syst Bacteriol*, 47(2):590–592.

Christiane Fellbaum, editor. 1998. *WordNet: an electronic lexical database*. MIT Press.

Juho Heimonen, Jari Björne, and Tapio Salakoski. 2010. Reconstruction of semantic relationships from their projections in biomolecular domain. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*, pages 108–116, Uppsala, Sweden, July. Association for Computational Linguistics.

Halil Kilicoglu and Sabine Bergler. 2009. Syntactic dependency based heuristics for biological event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 119–127. ACL.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. 2009. Overview of BioNLP'09 Shared Task on Event Extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 1–9, Boulder, Colorado, June. Association for Computational Linguistics.

Jin-Dong Kim, Sampo Pyysalo, Tomoko Ohta, Robert Bossy, and Jun'ichi Tsujii. 2011. Overview of BioNLP Shared Task 2011. In *Proceedings of the BioNLP 2011 Workshop Companion Volume for Shared Task*, Portland, Oregon, June. Association for Computational Linguistics.

Marie-Catherine de Marneffe and Christopher Manning. 2008. The Stanford typed dependencies representation. In *COLING Workshop on Cross-framework and Cross-domain Parser Evaluation*.

David McClosky. 2010. *Any Domain Parsing: Automatic Domain Adaptation for Natural Language Parsing*. Ph.D. thesis, Department of Computer Science, Brown University.

Makoto Miwa, Sampo Pyysalo, Tadayoshi Hara, and Jun'ichi Tsujii. 2010a. A comparative study of syntactic parsers for event extraction. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*, BioNLP '10, pages 37–45, Stroudsburg, PA, USA. Association for Computational Linguistics.

Makoto Miwa, Rune Sætre, Jin-Dong Kim, and Jun'ichi Tsujii. 2010b. Event extraction with complex event classification using rich features. *J Bioinform Comput Biol*, 8:131–146.

Martin F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

Sebastian Riedel, Hong-Woo Chun, Toshihisa Takagi, and Jun'ichi Tsujii. 2009. A markov logic approach to bio-molecular event extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*, BioNLP '09, pages 41–49, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research (JMLR)*, 6(Sep):1453–1484.

Sofie Van Landeghem, Yvan Saeys, Bernard De Baets, and Yves Van de Peer. 2009. Analyzing text in search of bio-molecular events: a high-precision machine learning framework. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 128–136. ACL.