First Shared Task on Extrinsic Parser Evaluation (EPE 2017): Biomedical Event Extraction Downstream Application

Jari Björne, Filip Ginter and Tapio Salakoski
Department of Future Technologies, University of Turku
Turku Centre for Computer Science (TUCS)
Faculty of Mathematics and Natural Sciences, FI-20014, Turku, Finland
firstname.lastname@utu.fi

Abstract

The First Shared Task on Extrinsic Parser Evaluation (EPE 2017) compares different dependency representations by evaluating their impact on downstream applications that utilize these parses for other text mining tasks. In the Biomedical Event Extraction downstream task parses are evaluated by using the Turku Event Extraction System (TEES) with the BioNLP'09 Shared Task as the model challenge. The participants parse the BioNLP'09 dataset, after which the TEES system is run, using the parses as features for predicting events on which the parses are compared. Eight teams submitted a total of 44 runs generated with various parsers, and an additional 13 runs were produced with parsers available via the TEES preprocessing system. Although the TEES system has been developed and optimized using the Stanford Dependencies parsing scheme, among the EPE submissions good performance on the TEES system was achieved also with the Universal Dependencies version 1 scheme.

1 Introduction

The goal of the The First Shared Task on Extrinsic Parser Evaluation (EPE 2017)[1] is to evaluate different dependency representations by comparing their performance on different downstream systems (EPE17Overview, 2017). Three downstream applications are used, 1) Biomedical Event Extraction, 2) Fine-Grained Opinion Analysis and 3) Negation Scope Resolution. In this paper we present the results for the Biomedical Event Extraction downstream application and describe the work on upgrading the TEES system to process the varying parse schemes submitted for the task.

Biomedical Event Extraction refers to the process of automatically detecting specific statements of interest from biomedical scientific publications. The biomedical literature is expanding at a rapid pace, with the central PubMed publication database containing as of 2017 over 27 million citations[2]. Text mining is required to search this mass of literature and to extract and summarize the common themes across the millions of publications. Common tasks in biomedical text mining include named entity recognition and normalization (detection of mentions of e.g. genes and mapping them to standardized database ids) as well as interaction extraction (detection of statements of e.g. molecular interactions), where the resulting information can be applied for tasks such as biochemical pathway curation.

In earlier work, biomedical interaction extraction has usually been approached through relation extraction, where all pairs of named entities detected within a span of text (usually a sentence) can either have or not have a stated (sometimes typed and directed) interaction linking them together. On the other hand, events consist of a trigger word (often a verb) and 0–n related arguments, some of which can be other events, allowing complex nested structures. For example, the sentence "Protein A regulates the binding of proteins B and C" can be annotated with a two-event nested structure REGULATION(A, BINDING(B, C)).

2 TEES Overview

The Turku Event Extraction System was originally developed for participation in the BioNLP'09 Shared Task on Biomedical Event Extraction. This task utilized the GENIA corpus, which was the

[1] http://epe.nlpl.eu/

[2] https://www.ncbi.nlm.nih.gov/pubmed/

first large-scale, annotated resource (+10,000 sentences) for biomedical events (Kim et al., 2008). In total, 24 teams participated in this task, with TEES achieving the first place with 51.95 F-score (Kim et al., 2009). The TEES system has later reached several first places in further shared tasks and has been used as the engine behind several PubMed-scale text mining resources (Björne, 2014).

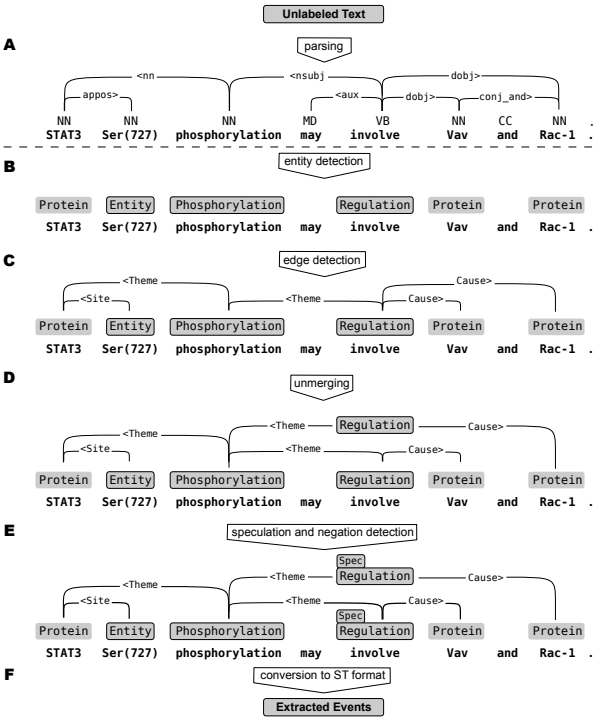

Figure 1: Event extraction. Before event extraction, (A) the text is split into sentences and parsed to produce dependency parse graphs. Relying on this information (B) keywords of interest (entities) are detected in the parsed sentences, after which (C) interaction edges can be detected between the entities. The graph is then (D) unmerged into individual events for which (E) various modifiers can be detected. Finally, the generated events can be (F) exported into the BioNLP Shared Task format. Figure adapted from (Björne et al., 2012).

The TEES system was built around the approach of modelling events as a graph. With relation extraction, graph-based approaches, such as the graph-kernel, have demonstrated good performance (Airola et al., 2008). If the word tokens of the sentence are thought of as the nodes of a graph, the relations can then be seen as edges. This formalism can be extended for events by considering the trigger word as the root node of the entire event, and all arguments as the outgoing edges of this node. Arising from this model of events as graphs, the TEES system is defined as a pipeline of multiclass-classification steps (using SVM^{multiclass}, Tsochantaridis et al. (2005)), the first of which detects word entities of interest (nodes), the second the arguments between these nodes (edges) and finally an unmerging step duplicates certain nodes to separate overlapping events. Optionally, a modifier detection step can be used to detect event labels such as negation and speculation (See Figure 1).

Dependency parses can similarly be modelled as graphs, with tokens as nodes and the dependencies as the edges. By starting from an automated dependency analysis, the TEES event extraction process can be seen as converting the syntactic dependency parse graph into the semantic event graph, the two graphs being linked via the shared token nodes. However, while the tokens form directly the nodes of the dependency graph, the entity and trigger nodes of the event graph may not follow the same tokenization, and can often cover multiple tokens (such as "human protein actin"). To align the graphs, TEES uses a heuristic to map each entity to the head token of the span of text covered by that entity.

In the context of the EPE task, there are three main attributes that determine the performance of a parse when used with TEES. First, the tokenization will determine at how fine-grained a level entities get mapped to tokens. Second, the dependencies determine the *shortest path* between two tokens, which is the central source of information used by TEES to detect the event argument edges. Finally, the labeling of the dependency graph affects the features generated: TEES uses the dependency types for edge detection and the token POS tags for entity detection. For the token POS tags, EPE participants could either use a generic POS attribute, or alternatively define both XPOS and UPOS attributes. If both XPOS and UPOS were used, the TEES system was run separately for both tag types and the higher performing result was used as the final result for that submission.

3 Adapting TEES for EPE

A number of improvements were developed for the TEES system in order to more easily apply it for the EPE task. In order to utilize the participants' submissions, the system was updated to im-

port the JSON EPE file format.

3.1 Importing the EPE parses

The EPE JSON format is the common interchange file format for the EPE task, allowing the participating parses to be used with the different downstream applications. As with the Interaction XML format used in TEES, dependency parses are modelled as graphs in the EPE format, with word tokens becoming the nodes and the dependency relations the edges of the graph. Each node must also be bound unambiguously to the source text with character offsets. While importing the EPE format is quite straightforward, challenges arise from the variation in the valid ways in which parses can be stored in this format.

For example, several of the participating systems produce parses where only tokens of interest have defined nodes. Previously, all parses used by TEES had annotated each word token in the sentence, leading to several mechanisms that relied on this implicit assumption to be true. In order to handle the parses which provide only partial tokenization, the TEES parse importer was updated to whitespace-tokenize and generate dummy tokens for the spans of text not part of partially tokenized parses.

TEES detects events by aligning the dependency parse graph with the event graph, using the parse tokens as shared nodes. However, the original nodes of the event graph are the annotated entities, which may consist of longer spans of text than single tokens. In order to align the graphs, each entity is mapped to a single *head token* using a heuristic for detecting the syntactic head of the parse subgraph contained within the entity. This heuristic assigns each token initial scores (0 for tokens with no dependencies, 1 for tokens connected by dependencies and -1 for special character tokens that should never be the head). After this, the score of each governor token of a dependency is increased to be higher than the score of the dependent token, until scoring no longer changes or a loop cutoff count is reached.

The variation in the parse schemes used by the EPE task participants means that the downstream applications can no longer rely on any parse specific information, such as conventions in POS tag or dependency type naming. In earlier TEES versions the likelihood of loops happening in head token detection was reduced by only considering a subset of primary dependency types for the iterative scoring. Since these dependency types were specific for the Stanford collapsed dependencies scheme they could no longer be used with the various dependencies submitted for the EPE task, so the limitation on dependency types was removed, with only the loop count cutoff now terminating loops if they happen. In practice, the impact on performance was minimal and the new system no longer depends on the specific naming of Stanford collapsed dependency types.

3.2 Parse Format Conversion

Originally, the TEES preprocessor was to be updated in order to provide another tool for converting a number of parser output formats into the EPE interchange format. This work was not completely successful, but for most texts the updated TEES parse converter can now reasonably well convert back and forth between the EPE, Penn TreeBank, Stanford Dependencies, CoNLL, CoNLL-X, CoNLL-U and CoreNLP formats using Interaction XML as the interchange format. The primary upgrade related to handling these formats was the improved alignment of parsed tokens with the input text, which is required in order to convert common parser output formats into textbound formats such as EPE and Interaction XML.

Most parsers do not provide the character offsets in the original text for the generated tokens, and this becomes a problem when such parsers also modify the input text. Common modifications include file format related escapings such as converting left and right parentheses into *-LRB-* and *-RRB-* tags. Such modifications can be detected and reversed relatively easily, but unfortunately many parsers perform also more unpredictable modifications, such as replacing British spellings with American ones (such as ”labour” becoming ”labor”).

Detecting the full list of such modifications would be an open ended problem, so in order to have a decent chance of aligning most modified tokens with the input text the TEES preprocessor now uses the Needleman-Wunsch global alignment algorithm (Needleman and Wunsch, 1970). A fast shortcut for aligning unmodified tokenizations that differ from the original text only in whitespace is tried at first, with reversals of various common parser modifications then being tried consecutively, with the alignment with the least

mismatches being the chosen one. In this manner, the updated TEES preprocessor can convert most parse formats into text-bound ones, although parses with extensive modifications of the original text can still be difficult to align fully.

3.3 Updating the Preprocessor

In the TEES system, importing parses, parsing and other preliminary tasks such as named entity recognition are performed with the TEES preprocessor tool. In previous versions, the preprocessor was implemented as a fixed pipeline of steps. These steps were, in order, conversion of plain text or the BioNLP Shared Task format to the Interaction XML format (the file format used internally by TEES), sentence splitting (with the GENIA sentence splitter) (Sætre et al., 2007), named entity recognition (with BANNER) (Leaman and Gonzalez, 2008), constituency parsing (with the BLLIP parser) (Charniak and Johnson, 2005), dependency conversion (with the Stanford Tools) (de Marneffe and Manning, 2008), named entity token splitting, entity syntactic head detection and division into train, devel and test sets. Individual steps in the pipeline could be turned off or modified with parameters, but the pipeline itself could not be redefined.

For the EPE task, the TEES preprocessor was updated into a fully configurable pipeline. The user can now define with the command line interface any list of consecutive steps, with the only limitations imposed by the input and output formats of these steps. Most steps take as both input and output an Interaction XML structure, but e.g. beginning steps can convert an input in the form of a directory of txt files into an Interaction XML structure, and final export steps can likewise convert an Interaction XML structure into a number of output formats.

In earlier TEES versions, parameters could be passed for the individual steps with a separate parameter option, using the step name (e.g. *--steps A,B,C --parameters A.parameter=value*). In the fully customizable pipeline the same step may appear multiple times, so the command line interface was updated to use regular Python syntax, evaluated at run-time, to configure both the steps and their parameters (e.g. *--steps A(parameter=value),B,C*). This approach allows not only using the same step multiple times in the pipeline, but also a consistent way of passing arbitrarily complex Python data structures as parameters for any preprocessing step.

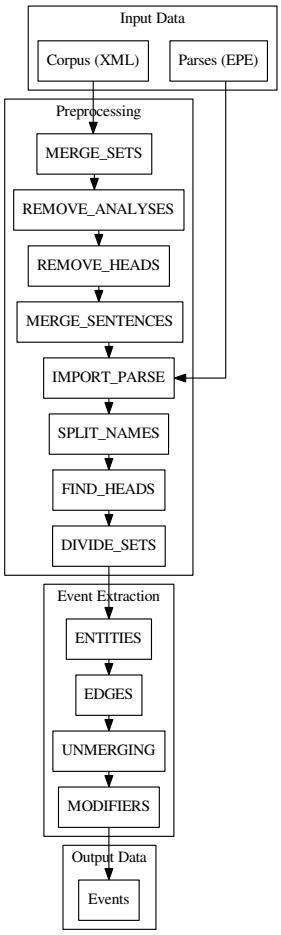

Figure 2: The TEES Preprocessor. The TEES Preprocessor is a fully configurable pipeline, which in the example shown in this figure is used to import an EPE task parse submission from the EPE interchange format and insert it into the BioNLP'09 Shared Task corpus from which existing parse information is removed. Once the parse is inserted, other preprocessing steps such as entity head token detection and entity token splitting are performed to prepare the parsed corpus for event extraction. The Interaction XML output from the Preprocessor is used as the input for Event Extraction, which finally generates the predicted events on which the different parses are evaluated in this EPE downstream task.

The updated TEES Preprocessor can now be used to perform a larger number of supporting tasks useful for event extraction. For example, the default parses for the BioNLP'09 Shared Task GENIA corpus (installed with TEES) can be exported to the EPE JSON format with the

pipeline *[LOAD(corpusName='GE09'), EXPORT(formats='epe')]*. The original TEES preprocessing pipeline was designed to prepare an unparsed corpus for event extraction by parsing it, and this same preprocessing can now be performed with the customized pipeline *[LOAD, GENIA_SPLITTER, BLLIP_BIO, STANFORD_CONVERT, SPLIT_NAMES, FIND_HEADS, SAVE]*. These steps correspond to the fixed preprocessing pipeline of earlier TEES versions.

In evaluating the EPE task, pregenerated parses provided by the participants need to be inserted instead of running a parser (See Figure 2). This can be achieved by changing a few of the preprocessing steps, resulting in the pipeline *[LOAD, REMOVE_ANALYSES, REMOVE_HEADS, MERGE_SENTENCES, IMPORT_PARSE(parseDir='x'), SPLIT_NAMES, FIND_HEADS, SAVE]*. Here, existing parse information and the sentence division based on the parse are removed from the loaded corpus using the *REMOVE_ANALYSES*, *REMOVE_HEADS* and *MERGE_SENTENCES* steps. A new parse is loaded (for example from a directory containing files in the EPE JSON format) using the *IMPORT_PARSE* step, and the rest of the pipeline performs the named entity token splitting, syntactic head detection and saving steps used in the normal parsing pipeline. In this manner, preprocessing steps can be freely combined to quickly define new experimental setups.

4 EPE 2017 Results and Discussion

In total, 44 runs were submitted by eight teams for the event downstream task. In addition, 13 "baseline" runs were generated by the organizers using the various parsers available via the TEES preprocessing pipeline, leading to a total of 57 individual parses for the biomedical event downstream task (See Table 1, Appendix A).

4.1 Baselines

The baseline runs were generated by parsing the BioNLP'09 corpus with parsers available via the TEES preprocessing pipeline, then converting the output to the EPE format, and finally running it through the same EPE evaluation pipeline as the participants' submissions. Therefore, these baselines are not to be seen necessarily as baselines in terms of performance, but as additional points of comparison that could easily be generated by using publicly available parsing tools.

The first baseline in Table 1, the *BioNLP'09 Analyses*, uses the official, BioNLP'09 organizer provided syntactic analyses from 2009. The analyses used were the PTB trees produced with the BLLIP parser using David McClosky's biomedical parsing model and the dependency parses generated with the Stanford Converter (Kim et al., 2009). The TEES preprocessor was used to insert these supporting resources into the BioNLP'09 corpus, followed by exporting them to the EPE format for evaluation as with the other baselines. Running the event extraction system using these official parses resulted in an F-score of 47.87, somewhat lower than the official UTurku result of 51.95 from 2009. The difference is most likely explained by the fact that the 2009 system included e.g. two parallel entity detection systems combined into an ensemble system and used a more corpus-specific, rule-based unmerging system. Therefore, the official BioNLP'09 UTurku result is not directly comparable with the baselines and the EPE submissions, all of which are evaluated using the current version of TEES.

The other baselines use various combinations of the BLLIP parser (commit 558adf6, Jan 9, 2016) (Charniak and Johnson, 2005), the Stanford Converter (version 2012-03-09) (de Marneffe and Manning, 2008) and the SyntaxNet parser (using the Parsey McParseface and Parsey Universal models) (Andor et al., 2016). The BLLIP parser is used with either the standard English model, or David McClosky's biomedical parsing model (McClosky, 2009). The highest performance of 50.48 is achieved with the TEES default parsing settings, using BLLIP with the McClosky biomodel, followed by Stanford conversion using the CCProcessed output format. The BLLIP biomodel parses outperform every other baseline parse regardless of the type of Stanford conversion used, showing a consistent gain from domain adapted parsing. At 47.52 F-score, the SyntaxNet Parsey McParseface model has the best performance out of all the non-biomodel baseline parses, but the Parsey Universal (Universal Dependencies) model is behind all other baselines by several percentage points at 42.79 F-score.

4.2 Submissions

A wide variety of submissions were provided by the task participants, using several different parsers and parsing approaches (See Table 1, Appendix A). The event extraction performance when using the various parses ranged from 42.70 to 50.26 F-score. The highest performance of 50.26 F-score was reached by the stanford-paris run 0, using the Stanford Basic scheme with the *XPOS* POS tag type. This parse, like nine out of the ten best submissions (48.99–50.26) was adapted for the biomedical domain by training also on the GENIA corpus.

However, paris-stanford run 2, which was trained only on the WSJ corpus, reached a performance of 49.55, less than a percentage point below the best performing domain adapted results. Although too many conclusions cannot be drawn from this single result, it is encouraging to see improved performance for general English parsers on this biomedical text mining task, perhaps indicating less need in the future for time-consuming and resource-dependent domain adaptation.

For the 14 submissions that did not mention using a Universal Dependencies (Nivre et al., 2016, 2017) model, performance was in the range 42.84–50.29. The 21 UD v1.x parses resulted in performances in the range 43.22–50.23, and the nine UD v2.0 submissions were in the range 42.70–45.54. At least on the basis of these results, the UD v1.x scheme can achieve very good performance with an event extraction system such as TEES which was originally developed on the Stanford collapsed dependencies scheme. The UD v2.0 demonstrates overall lower performance, perhaps partially explained by this newer scheme diverging further from the underlying dependency parsing paradigms on which TEES still relies.

5 Conclusions

The First Shared Task on Extrinsic Parser Evaluation (EPE 2017) explored the feasibility of evaluating different parsers in light of the performance of downstream applications that use these parses as supporting analyses for some other text mining task. For the TEES downstream task, 44 participant submissions and 13 internally generated parses were compared.

While the best performance was achieved with the stanford-paris run 0, using the Stanford Basic dependencies scheme, also the UD v1.x scheme proved to work effectively with the existing event extraction framework. However, more work is still needed to make use of the UD v2.0 parses for event extraction. In evaluating the results for the EPE biomedical event extraction downstream challenge, it is important to remember that the TEES system has been developed and optimized since 2009 using Stanford collapsed dependencies parses.

Even if TEES is not bound to any single parse scheme, the iterative development and optimization using Stanford collapsed dependencies has no doubt biased the system to some degree towards parsing schemes similar to those collapsed dependencies. Such "overfitting" is of course an issue for any downstream task developed originally using a specific parser, but in any case means that these results must not be assumed to be completely objective evaluations of parser performance or suitability for biomedical event extraction.

The work on adapting the TEES system to not only use the EPE file format, but to work efficiently with the varying schemes of the different parsers, is published as part of the TEES open source project[3]. The improvements to the preprocessing system and the increased robustness in handling different parse schemes should make the system increasingly suitable for more varied text mining tasks. The organizers' work in adapting the downstream applications to use the EPE format, as well as the participants' efforts to export their parses in this common interchange format, build a strong foundation for continued shared evaluation of parsers using downstream text mining applications.

Acknowledgments

We thank Dr. Matthew Shardlow and Professor Sophia Ananiadou of the National Centre for Text Mining (Manchester Institute of Biotechnology, University of Manchester)[4], OpenMinTeD project[5], for assistance with using the BioNLP'09 test set for the purpose of evaluating the EPE task submissions. We also thank CSC – IT Center for Science Ltd for computational resources.

[3] http://jbjorne.github.io/TEES/
[4] http://www.nactem.ac.uk/
[5] http://openminted.eu

References

Antti Airola, Sampo Pyysalo, Jari Björne, Tapio Pahikkala, Filip Ginter, and Tapio Salakoski. 2008. All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. *BMC bioinformatics* 9(11):S2.

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. *CoRR* abs/1603.06042. http://arxiv.org/abs/1603.06042.

Jari Björne. 2014. *Biomedical Event Extraction with Machine Learning*. Ph.D. thesis, University of Turku.

Jari Björne, Filip Ginter, and Tapio Salakoski. 2012. University of turku in the bionlp'11 shared task. *BMC bioinformatics* 13(11):S4.

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

EPE17Overview. 2017. Placeholder. In *Placeholder*.

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 Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*. Association for Computational Linguistics, Stroudsburg, PA, USA, BioNLP '09, pages 1–9. http://dl.acm.org/citation.cfm?id=1572340.1572342.

Jin-Dong Kim, Tomoko Ohta, and Jun'ichi Tsujii. 2008. Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics* 9(1):10. https://doi.org/10.1186/1471-2105-9-10.

R. Leaman and G. Gonzalez. 2008. BANNER: an executable survey of advances in biomedical named entity recognition. *Pacific Symposium on Biocomputing* pages 652–663.

David McClosky. 2009. *Any Domain Parsing: Automatic Domain Adaptation for Natural Language Parsing*. Ph.D. thesis, Department of Computer Science, Brown University.

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

Saul B Needleman and Christian D Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology* 48(3):443–453.

Joakim Nivre, Željko Agić, Lars Ahrenberg, Maria Jesus Aranzabe, Masayuki Asahara, Aitziber Atutxa, Miguel Ballesteros, John Bauer, Kepa Bengoetxea, Riyaz Ahmad Bhat, Eckhard Bick, et al. 2017. Universal dependencies 2.0. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University. http://hdl.handle.net/11234/1-1983.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D Manning, Ryan T McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *LREC*.

R. Sætre, K. Yoshida, A. Yakushiji, Y. Miyao, Y. Matsubyashi, and T. Ohta. 2007. AKANE system: protein-protein interaction pairs in BioCreAtIvE2 challenge, PPI-IPS subtask. In *Proceedings of the Second BioCreative Challenge Workshop*. Citeseer, pages 209–212.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. 2005. Large Margin Methods for Structured and Interdependent Output Variables. *Journal of Machine Learning Research (JMLR)* 6(Sep):1453–1484.

A Biomedical Event Extraction Results

					Development Set			Evaluation Set		
team	run	representation	training	POS	Approx. Span & Recursive			Approx. Span & Recursive		
					P	R	F	P	R	F
UTurku in BioNLP'09					55.62	51.54	53.50	58.48	46.73	51.95
BioNLP'09 Analyses	BL				61.83	45.33	52.31	60.06	39.79	47.87
BLLIP-Bio, SF-Conv	BL	CCprocessed			60.70	52.04	56.04	58.80	44.22	50.48
BLLIP-Bio, SF-Conv	BL	basic			59.69	49.30	54.00	57.18	44.78	50.23
BLLIP-Bio, SF-Conv	BL	collapsed			59.84	51.70	55.47	58.10	43.75	49.91
BLLIP-Bio, SF-Conv	BL	collapsedTree			61.33	50.59	55.44	59.06	43.21	49.91
BLLIP-Bio, SF-Conv	BL	nonCollapsed			58.73	49.13	53.50	56.87	43.81	49.49
BLLIP, SF-Conv	BL	CCprocessed			56.20	45.16	50.08	55.18	41.04	47.07
BLLIP, SF-Conv	BL	basic			54.52	47.96	51.03	52.68	42.93	47.31
BLLIP, SF-Conv	BL	collapsed			64.51	42.93	51.55	58.23	37.15	45.36
BLLIP, SF-Conv	BL	collapsedTree			58.60	45.00	50.91	55.46	39.75	46.31
BLLIP, SF-Conv	BL	nonCollapsed			55.15	48.52	51.62	53.36	41.99	47.00
SyntaxNet	BL				57.41	46.62	51.46	55.73	41.42	47.52
SyntaxNet	BL	UD			49.33	47.07	48.17	46.99	39.28	42.79
ecnu	0	UD v2.0		upos	54.12	45.61	49.50	49.48	39.00	43.62
ecnu	1	UD v2.0		upos	54.43	43.66	48.45	50.72	38.97	44.08
ecnu	2	UD v2.0		upos	52.91	45.28	48.80	52.24	40.23	45.46
ecnu	3	UD v2.0		upos	57.85	41.87	48.58	54.53	35.58	43.06
ecnu	4	UD v2.0		upos	62.90	43.66	51.54	60.69	35.76	45.00
paris-stanford	0	DM	WSJ 00-20 (SDP Sub-Set)		58.26	43.43	49.76	59.11	37.71	46.04
paris-stanford	1	PAS	WSJ 00-20 (SDP Sub-Set)		51.29	46.73	48.90	52.39	40.98	45.99
paris-stanford	2	UD v1 basic	WSJ 00-20 (SDP Sub-Set)		54.71	48.13	51.21	55.79	44.56	49.55
paris-stanford	3	UD v1 enh	WSJ 00-20 (SDP Sub-Set)		56.45	47.23	51.43	57.48	41.64	48.29
paris-stanford	4	UD v1 enh++	WSJ 00-20 (SDP Sub-Set)		61.53	46.00	52.64	58.55	39.50	47.17
paris-stanford	5	UD v1 enh++ dt	WSJ 00-20 (SDP Sub-Set)		54.84	49.13	51.83	55.58	43.37	48.72
paris-stanford	6	UD v1 enh++ dt--	WSJ 00-20 (SDP Sub-Set)		57.98	43.94	49.99	58.11	39.19	46.81
paris-stanford	7	UD v1 basic	WSJ, Brown, GENIA		61.05	49.02	54.38	57.69	42.80	49.14
paris-stanford	8	UD v1 enh	WSJ, Brown, GENIA		55.62	49.86	52.58	54.90	44.75	49.31
paris-stanford	9	UD v1 enh++	WSJ, Brown, GENIA		58.68	49.58	53.75	58.03	43.02	49.41
paris-stanford	10	UD v1 enh++ dt	WSJ, Brown, GENIA		60.04	46.84	52.62	59.88	40.19	48.10
paris-stanford	11	UD v1 enh++ dt--	WSJ, Brown, GENIA		61.63	44.77	51.86	58.92	40.07	47.70
peking	0	DM	SDP 2015		54.46	41.31	46.98	59.28	34.22	43.39
peking	1	CCD			56.15	45.72	50.40	58.26	40.07	47.48
prague	0	UD v2.0	English 2.0	xpos	52.86	38.85	44.78	53.84	36.61	43.58
prague	1	UD v2.0	English 2.0	xpos	55.70	42.09	47.95	56.35	38.21	45.54
prague	2	UD v2.0	English, LinES, ParTUT 2.0	xpos	52.69	42.15	46.83	53.22	37.87	44.25
prague	3	UD v2.0	English 2.0	xpos	53.44	39.58	45.48	51.91	36.27	42.70
prague	4	UD v1.2	English 2.0	xpos	52.96	41.53	46.55	51.71	37.12	43.22
stanford-paris	0	Stanford Basic	WSJ, Brown, GENIA	xpos	55.75	49.92	52.67	56.93	45.03	50.29
stanford-paris	1	UD v1 basic	WSJ 00-20 (SDP Sub-Set)	xpos	60.73	46.73	52.82	57.59	40.76	47.73
stanford-paris	2	UD v1 enh	WSJ 00-20 (SDP Sub-Set)	xpos	61.40	47.46	53.54	57.24	40.98	47.76
stanford-paris	3	UD v1 enh++	WSJ 00-20 (SDP Sub-Set)	xpos	58.88	48.97	53.47	56.76	42.74	48.76
stanford-paris	4	UD v1 enh++ dt	WSJ 00-20 (SDP Sub-Set)	xpos	63.62	46.00	53.39	58.86	40.51	47.99
stanford-paris	5	UD v1 basic	WSJ, Brown, GENIA	xpos	58.10	49.19	53.28	58.75	42.21	49.13
stanford-paris	6	UD v1 enh	WSJ, Brown, GENIA	xpos	59.51	50.42	54.59	58.36	44.09	50.23
stanford-paris	7	UD v1 enh++	WSJ, Brown, GENIA	xpos	63.45	46.73	53.82	62.30	41.55	49.85
stanford-paris	8	UD v1 enh++ dt	WSJ, Brown, GENIA	xpos	59.19	51.37	55.00	57.47	44.47	50.14
stanford-paris	9	UD v1 enh++ dt--	WSJ 00-20 (SDP Sub-Set)	xpos	58.15	49.92	53.72	55.29	43.21	48.51
stanford-paris	10	UD v1 enh++ dt--	WSJ, Brown, GENIA	xpos	56.87	50.25	53.36	57.22	42.83	48.99
szeged	0				59.33	45.89	51.75	60.20	39.69	47.84
szeged	1				58.11	45.11	50.79	59.09	39.53	47.37
szeged	2				57.28	46.23	51.17	57.93	39.13	46.71
szeged	3				54.85	45.89	49.97	55.14	40.48	46.69
szeged	4				56.14	44.77	49.81	55.12	39.41	45.96
upf	0	SSyntS			53.91	46.28	49.80	53.21	41.36	46.54
upf	1	DSyntS			53.56	44.61	48.68	54.06	39.94	45.94
upf	2	PredArg			55.38	44.38	49.27	56.37	39.63	46.54
uw	0	DM	SDP 2015		58.34	45.22	50.95	54.86	35.14	42.84

Table 1: EPE Biomedical Event Extraction downstream task results. *SF-Conv* refers to the Stanford Dependencies Converter and *BL* to the baseline parses generated via the TEES preprocessor. In *representation*, *enh* refers to enhanced and *dt* to diathesis. If no POS tag is defined, the generic EPE format POS attribute was used. The primary metric of evaluation is the evaluation (test) set F-score, evaluated using the official Task 1 Approximate Span & Recursive Mode of the BioNLP'09 Shared Task evaluation program.