EPE 2017: The Trento–Gothenburg Opinion Extraction System

Richard Johansson
Department of Computer Science and Engineering
University of Gothenburg and Chalmers University of Technology
Gothenburg, Sweden
richard.johansson@gu.se

Abstract

We give an overview of one of the three downstream systems in the Extrinsic Parser Evaluation shared task of 2017: the Trento–Gothenburg system for opinion extraction. We describe the modifications required to make the system agnostic to its input dependency representation, and discuss how the input affects the various submodules of the system. The results of the EPE shared task are presented and discussed, and to get a more detailed understanding of the effects of the dependencies we run two of the submodules separately. The results suggest that the module where the effects are strongest is the opinion holder extraction module, which can be explained by the fact that this module uses several dependency-based features. For the other modules, the effects are hard to measure.

1 Introduction

Applications that use dependency representations of sentences are affected by a number of interacting factors that can be hard to tease apart (Miyao et al., 2008; Johansson and Nugues, 2008b; Elming et al., 2013). We expect the quality of the parser to have an impact: in general, a good parser should also lead to the downstream application being more successful. But this is not the end of the story, because some types of dependency representations may be more or less suitable for a given application, or may be harder or easier for automatic parsers to produce. In the Extrinsic Parser Evaluation (EPE) shared task of 2017 (Oepen et al., 2017),[1] we aim to investigate these questions more systematically by considering several parsers and representations, and measuring their effect on three different downstream applications: opinion extraction, biomedical event extraction, and negation resolution.

In this paper, we describe how the Trento–Gothenburg opinion extraction system (Johansson and Moschitti, 2013) was adapted to the EPE shared task. This system extracts opinion expressions according to the annotation model in the MPQA corpus (Wiebe et al., 2005). The system consists of a number of submodules operating as a pipeline, with a reranker that selects the final output, and some of these modules use features derived from a dependency-parsed representation of the input sentence. For this reason, this application is a useful testbed for measuring the effect of representational design choices and the efficacy of parsers.

We discuss how the opinion extraction system is affected by the dependencies produced by the parsers participating in the EPE shared task. In particular, we are interested in the following questions:

- Can variations in the output of the opinion extraction system be attributed to differences in the dependency inputs, or to other aspects of the input such as tokenization, lemmatization, and tagging?
- In case the dependency structures do have an effect, what parts of the analysis are affected the most?
- Does the type of representation matter, or is the choice of parser more important? For instance, are the semantically oriented dependency representations investigated in the SDP Shared Task (Oepen et al., 2015) useful, or are more traditional syntactic dependencies more suitable?

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

We first give an introduction to the opinion extraction task as defined by Wiebe et al. (2005), after which we give an overview of the system by Johansson and Moschitti (2013) and how it was adapted to the EPE shared task. Next, we present the opinion extraction results in the shared task, and we carry out some additional experiments to try to understand to what extent and in what ways the quality of the system is affected by the parsers.

2 Task Description

The MPQA project (Wiebe et al., 2005) defined an annotation scheme and created a corpus of annotated expressions of opinions (or private states). The main building blocks in this scheme are three types of linguistic expressions:

- *direct-subjective expressions* (DSEs), which explitly mention emotions and opinions, such as *enjoy* or *disapproval*, or evaluative speech events, such as *criticize* or *label*;
- *expressive-subjective elements* (ESEs), which do not explicitly mention an emotion but in which the choice of words helps us understand an attitude – such as *great*, *heresy*, or *fifth column*;
- *objective statement expressions* (OSEs), which refer to speech events that do not express an opinion – such as *says* or *statement*.

Each instance of these types of expressions is connected to an *opinion holder* (or *source*, in the terminology of Wiebe et al., 2005). This is a lingustic expression that refers to the person expressing the opinion or experiencing the emotion. This person may not be explicitly mentioned in the text, for instance if this is the writer of the text. Furthermore, every DSE and ESE is associated with a *polarity*: positive, negative, or neutral.[2]

To exemplify, in the sentence

"The report is full of absurdities," Xirao-Nima said.

the expression *full of absurdities* is an ESE with a negative polarity, *said* a DSE, also with a negative polarity, and *Xirao-Nima* the opinion holder of the DSE as well as of the ESE.

[2]Following Choi and Cardie (2010) and Johansson and Moschitti (2013), we mapped the polarity value *both* to neutral, and e.g. *uncertain-positive* to positive, etc.

3 System Description

In this section, we give an overview of the opinion analysis system described by Johansson and Moschitti (2013). In particular, we focus on how the system was adapted for the EPE shared task, and how the parts of the pipeline are affected by the linguistic analysis provided by the participating parsing systems.

As a running example, Figure 1 shows how the sentence above could be analyzed by a hypothetical participant in the EPE shared task. In this case, the representation follows the CoNLL-2008 format (Surdeanu et al., 2008) and consists of a combination of syntactic edges, drawn above the sentence, and semantic edges, drawn below. Each word is tagged using a Penn Treebank-style (Marcus et al., 1993) part-of-speech tag.

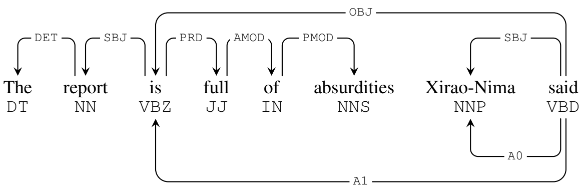

Figure 1: A hypothetical analysis of the example sentence.

3.1 Description of the Pipeline

The system by Johansson and Moschitti (2013) is implemented as a combination of three different submodules – opinion expression extraction, holder extraction, and polarity classification – followed by a reranker that picks the final output based on the results of the previous three steps. Figure 2 shows an overview.

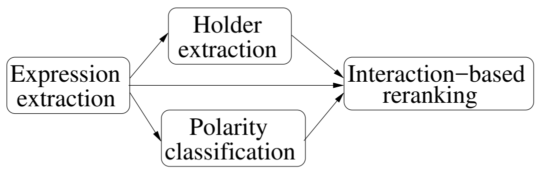

Figure 2: Parts of the opinion extraction system.

3.1.1 Extracting Opinion Expressions

The first step of the pipeline extracts opinion expressions (DSEs, ESEs, OSEs) using a standard sequence labeler (Collins, 2002) operating on the sequence of tokens. This sequence labeler extracts basic grammatical and lexical features (word, lemma, and PoS tag), as well as prior polarity and intensity features derived from the lexicon created by Wilson et al. (2005). Features based on words (and bigrams), lemmas, and PoS tags, as well as polarity and intensity values, were extracted in a window of size 3 around the word in focus. Expression brackets are encoded using IOB2 tags. The tagging model is trained using the online Passive–Aggressive algorithm (Crammer et al., 2006).

3.1.2 Polarity Classification

Each subjective expression (ESE or DSE) is assigned a polarity value by a separate classifier (a linear SVM). It uses a feature representation consisting of bags of words, word bigrams, and PoS tags inside the expression and in a small window around it, as well as prior polarity and intensity values from the MPQA lexicon.

3.1.3 Opinion Holder Extraction

The module that extracts opinion holders is also implemented as a linear SVM classifier. Given an opinion expression, for instance the DSE *said* in the example, the model assigns a score to each non-punctuation token in the sentence that is not contained in any opinion expression. In addition, it considers two special cases: the *writer* entity, representing the author of the text, and the *implicit* entity, for cases where the holder is not explicitly mentioned. The dependency node (or one of the two special cases) that maximizes this score is selected as the opinion holder.

The feature set used by the classifier in the holder extraction module makes heavy use of the dependency graph. This is because the holder extraction task consists of determining a *relation* between parts of the sentence; it is fairly similar to event participant or semantic role filler extraction; this is particularly true of DSEs, such as *Xirao-Nima* being the holder and filling the speaker role of *said*. For this reason, the feature set used by this module is fairly similar to a typical set of features used in semantic role labeling, relying heavily on paths and other syntactic patterns. Systems for such tasks tend to be sensitive to variations in the input representation (Johansson and Nugues, 2008b; Miyao et al., 2008).

3.1.4 Interaction-based Reranking

Johansson and Moschitti (2013) found considerable improvements in all subtasks by designing *interaction-based* features that describe the relations between different opinion expressions. For instance, these features can describe that

- a DSE (*said*) is connected to an ESE (*full of absurdities*) via an `OBJ` edge;
- a DSE and an ESE have the same opinion holder (*Xirao-Nima*);
- a DSE is connected to an ESE via an `OBJ` edge, and both of them are negative.

Three different groups of interaction features were investigated, based respectively on expressions holders, and polarity. In this work, we used a the combination of all three groups. Several of these features make use of the dependency representation of the sentence.

Considering pairs of opinion expressions makes inference harder, so Johansson and Moschitti (2013) used a reranking approach: first generate the top k solutions from the expression tagger, polarity classifier, and holder classifier, and then rescore the k candidates using a linear scoring model that uses the interaction-based features.

3.2 Changes for the EPE Task

Out of the four modules described above, the holder extraction module and the reranker required significant modifications for the EPE shared task in order to make them agnostic to the structure of the input dependency representation. The opinion extraction tagger and polarity classifier were unchanged from the implementation described by Johansson and Moschitti (2013), but we still expect these modules to see some effect from the expressivity of the PoS tagset and the quality of the PoS and lemma predictions.

The holder extraction and reranking modules in the system by Johansson and Moschitti (2013) depend on the output of the parser by Johansson and Nugues (2008a), which consists of a syntactic dependency tree and a separate semantic dependency graph representing PropBank and NomBank relations (Surdeanu et al., 2008), more or less corresponding to Figure 1. Several aspects of the system, including the design of features and a number of heuristics, require that the input conforms to this format. In particular, the system relies on the fact that the syntactic side of the representation is tree-structured.

In the reengineered system for the EPE shared task, all assumptions about the structure of the input were removed. Most importantly, this applies to several features based on *paths* through the dependency structure; as described in §3.1.3 and §3.1.4, such features are used by the holder extraction and reranking steps. The previous implementation assumed a tree-structured syntactic graph, which means that the path between two nodes in the graph is unique and easy to compute. For instance, when determining that *Xirao-Nima* is the opinion holder of the DSE *said*, we would extract one feature from the graph in Figure 1 that describes that *Xirao-Nima* is the syntactic subject (`SBJ`). The semantic dependency edge (`A0`, for the semantic role of speaker) would be represented as a separate feature. When relaxing the assumptions about the structural properties of the dependency graph, we instead use features based on *shortest paths*. In case there is no unique shortest path, i.e. if there is more than one with the minimal length, we create separate features for up to 8 paths with the minimal length. For example, the minimal path length between the DSE *said* and its opinion holder *Xirao-Nima* is 1, and there are two paths with this length: one via a syntactic edge (`SBJ`), and one via a semantic edge (`A0`).

Furthermore, in the reengineered system we removed the grammatical voice feature used by Johansson and Moschitti (2013) for the holder extraction module. The reason is that this feature was computed using hard-coded syntactic heuristics that are not applicable for general dependency representations. Hypothetically, we could imagine participating systems representing the voice of a verb as a morphological feature or via dependency edges (e.g. the `nsubj`/`nsubjpass` distinction in the Stanford representation).

Apart from these changes, we made no further adaptation of the system. In particular, we would like to point out that we did not have time to redesign and optimize features for each individual parser, which in principle could lead to a bias towards parsers or representations resembling those used by Johansson and Moschitti (2013). The only feature selection we did for individual parsers was that we investigated whether coarse-grained or fine-grained part-of-speech tags were more effective, and we found in all cases that the latter option was to be preferred.

4 Experimenal Setup

The systems participating in the EPE shared task were evaluated in three different subtasks, which correspond to the evaluations by Johansson and Moschitti (2013):

- marking up opinion expressions in the text, and determining their type (DSE, OSE, or ESE), for instance that the example contains an ESE (*full of absurdities*) and a DSE (*said*);
- determining the opinion holder for every extracted opinion expression, for instance that *Xirao-Nima* is the holder of the two expressions in the example;
- determining the polarity of each extracted subjective expression (that is, DSEs and ESEs), for instance that the two expressions in the examples are both negative.

4.1 Intersection-based Scoring

In all three evaluation scenarios mentioned above, the system marks up spans in the text (opinion expressions and holders), which are then compared to the gold-standard spans. However, the boundaries of opinion expressions in the annotation model of Wiebe et al. (2005) are not rigorously defined and the inter-annotator agreement at the boundary level tends to be low, particularly for ESEs. This makes it natural to apply evaluation metrics that allow for some leniency in evaluating the boundaries.

Johansson and Moschitti (2013) evaluated their system using *intersection-based precision and recall* metrics, which are based on the notion of *span coverage* to measure how well a span s covers another span s':

$$c(s, s') = \frac{|s \cap s'|}{|s'|}$$

The intersection $s \cap s'$ was defined as the set of shared tokens between the two spans, and the set cardinality $|\cdot|$ as the number of tokens. In label-aware evaluation scenarios, $c(s, s')$ was set to 0 if the labels of s and s' differ. The notion of span coverage was then used to define precision and recall measures. The precision measures how well the gold-standard spans cover the predicted spans, and vice versa for the recall:

$$P = \frac{\sum\limits_{\substack{s_i \in S \\ g_j \in G}} c(g_j, s_i)}{|S|} \qquad R = \frac{\sum\limits_{\substack{s_i \in S \\ g_j \in G}} c(s_i, g_j)}{|G|}$$

where S is the set of spans predicted by the system, and G the set of gold-standard spans.

In the EPE shared task, tokenization was provided by the different participating systems, which required us to change the evaluation procedure to take differences in tokenization into account. To achieve this, we redefined the span coverage to count the number of shared *characters* instead of tokens.

4.2 Dataset Details

We trained and evaluated the system on version 2.0 of the MPQA corpus.[3] This release contains 692 documents, out of which we discarded four documents where the annotation was difficult to align with the text, or which consisted mostly of non-English text. We split the remaining 688 into a training set (450 documents), a development set (90 documents) and a test set (148 documents). The split corresponds to the setup described by Johansson and Moschitti (2013), except that three additional documents were removed. This is a multi-domain corpus but the split was done randomly, so we do not expect that there are significant domain differences between the training and test sets.

5 Results

We first evaluate the system as a whole as described in §4, and then consider individual modules in the pipeline to try to tease out what effects are in play.

5.1 Overall Results

Table 1 shows the results of all the 44 participating parsers evaluated for the three subtasks, as well as the macro-average of the three scores. We observe that the scores for the holder extraction task shows much more variation than for the other tasks. As discussed in §3.1.3, the holder extraction module uses several features derived from the dependencies in the input, so it is logical that this task is the one where we see the largest effects of representational design choices and quality of the parsers.

5.2 Opinion Expression Extraction

We carried out an evaluation of the sequence labeler that marks up and labels opinion expressions (§3.1.1) by running it in isolation, without the interaction-based reranker. Table 2 shows the results. This module uses token-level information such as word forms, lemmas, and PoS tags, but no dependencies. We can thus see this experiment as an extrinsic evaluation of the non-dependency part of the input.

As the results show, the variation among systems is fairly small: if we remove the two outliers (UPF runs 1 and 2), the F-measure standard deviation is just 0.54 and 0.33 in the development and test set, respectively. This is likely because most systems use similar tokenization procedures and Penn Treebank-style tags. The two outliers by the UPF team used an unconventional tokenization scheme that excludes many function words, and this caused difficulties for the opinion expression tagger.

Furthermore, we considered the differences between the expression extraction results in Tables 1 and 2: we would expect that since the interaction-based reranker (§3.1.4) uses several features based on the dependency representation, the parser should have some impact. However, we see no systematic effect: the reranker consistently gives a relative improvement of around 5–7%, which suggests that the choice of representation or parser does not have a strong impact here. This result seems surprising; it is imaginable that the dependency features that have an impact on the reranker are relatively easy for parsers to extract, but we would need to carry out more thorough investigations of features to answer this question in a reliable manner.

5.3 Holder Extraction

We ran the holder extraction module separately, giving the gold-standard opinion expressions as input to the system. We refer to this experiment as *in vitro holder extraction*. Table 3 shows the results of this evaluation. As already mentioned, there are clear differences in performance between the different systems: the F-measure standard deviation on the development and test set is 3.28 and 2.53, respectively.

In an evaluation of this kind, the variation in performance can be explained by several interacting factors, including the design of the sentence representation and the quality of the parser. To exemplify the effects of the choice of parser, we can consider the variation among parsers based on Universal Dependencies (McDonald et al., 2013), of which there are several: the F-measure in this group ranges from about 59 points up to 65 points.

[3] http://www.cs.pitt.edu/mpqa/databaserelease/

One clearly discernible effect of the representation is that the small group of parsers producing semantic dependencies (Paris-Stanford runs 0–1, Peking, UPF 1–2, UW) give considerably lower holder extraction scores than those based on more traditional syntactic dependencies (using Universal Dependencies, Stanford Dependencies, or CoNLL dependencies). The mean F-score on the test set for the group of semantic parsers is 58.76, while the score for the syntactic parsers is 62.94. While this suggests that the more shallow syntactic parsers give more reliable features for this task, it should be noted that these parsers are probably more similar to that originally used by Johansson and Moschitti (2013), and that no effort has been spent on feature design or tuning for the semantically oriented parsers.

Among the systems producing purely syntactic dependencies, it seems that the parser implementation has a stronger impact than the choice of representation: among this group, the best-performing and worst-performing are UD-based, and the few CoNLL-based parsers achieve moderate to high performance (Szeged, UPF run 0).

6 Conclusions

We presented the Trento–Gothenburg opinion extraction system and how it was adapted for the EPE shared task of 2017. The previous implementation by Johansson and Moschitti (2013) made a number of assumptions about the structure of the input – that it consisted of a CoNLL-style syntactic tree and a separate set of semantic dependencies – that had to be relaxed. The modified system does not require a tree-structured input or that semantic edges are stored separately from the syntactic edges. Furthermore, a few hand-crafted features (mainly the voice feature) assuming a CoNLL-style input have been removed.

The outputs of the 44 participating parsers were used to train the opinion extraction system, and we investigated the general performance as well as the performance of individual submodules. It turned out that holder extraction seems to be the part of the analysis that is most affected by the dependencies, and for this subtask we saw much variation among systems. For the other subtasks, the differences attributable to the choice of dependencies are negligible, and token-level linguistic information such as tagging and lemmatization seems to cause much of the variation.

Since the holder extraction task was the one most affected by the dependencies, we ran this module in isolation to highlight the differences. We could see an effect of the choice of representation type, since it seems that semantically oriented parsers, e.g. those coming from the SDP shared task (Oepen et al., 2015), give weaker results. However, we should be careful to draw conclusions from this result, since it could possibly be attributed to the semantic dependency parsers being more different from those used by Johansson and Moschitti (2013), and they may require additional feature engineering and optimization. Apart from this result, there seems to be more variation among parsers using the same representation (e.g. Universal Dependencies) than between different types of syntactic dependencies.

Acknowledgments

The author was supported by the Swedish Research Council under grant 2013–4944.

References

Yejin Choi and Claire Cardie. 2010. Hierarchical sequential learning for extracting opinions and their attributes. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Uppsala, Sweden, pages 269–274.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*. University of Pennsylvania, United States, pages 1–8.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Schwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research* 2006(7):551–585.

Jakob Elming, Anders Johannsen, Sigrid Klerke, Emanuele Lapponi, Hector Martinez Alonso, and Anders Søgaard. 2013. Down-stream effects of tree-to-dependency conversions. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, United States, pages 617–626.

Richard Johansson and Alessandro Moschitti. 2013. Relational features in fine-grained opinion analysis. *Computational Linguistics* 39(3):473–509.

Richard Johansson and Pierre Nugues. 2008a. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *CoNLL 2008: Proceedings of the Twelfth Conference on Natural*

Language Learning. Manchester, United Kingdom, pages 183–187.

Richard Johansson and Pierre Nugues. 2008b. The effect of syntactic representation on semantic role labeling. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*. Manchester, United Kingdom, pages 393–400.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics* 19(2):313–330.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Sofia, Bulgaria, pages 92–97.

Yusuke Miyao, Rune Sætre, Kenji Sagae, Takuya Matsuzaki, and Jun'ichi Tsujii. 2008. Task-oriented evaluation of syntactic parsers and their representations. In *Proceedings of ACL-08: HLT*. Columbus, United States, pages 46–54.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, and Zdeňka Urešová. 2015. SemEval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. Denver, United States, pages 915–926.

Stephan Oepen, Lilja Øvrelid, Jari Björne, Richard Johansson, Emanuele Lapponi, Filip Ginter, and Erik Velldal. 2017. The 2017 Shared Task on Extrinsic Parser Evaluation. Towards a reusable community infrastructure. In *Proceedings of the 2017 Shared Task on Extrinsic Parser Evaluation at the Fourth International Conference on Dependency Linguistics and the 15th International Conference on Parsing Technologies*. Pisa, Italy, pages 1–12.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of CoNLL 2008*. Manchester, United Kingdom, pages 159–177.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation* 39(2-3):165–210.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*. Vancouver, Canada, pages 347–354.

A Evaluation Scores

		Development				Test			
System	Run	Expr	Holder	Polarity	Macro	Expr	Holder	Polarity	Macro
ECNU	0	58.99	46.36	51.18	52.18	59.10	44.18	49.74	51.01
	1	59.39	48.53	51.55	53.16	58.95	46.52	49.61	51.69
	2	59.58	49.18	**51.74**	53.50	58.85	46.04	49.47	51.45
	3	59.66	47.93	51.51	53.03	58.84	45.31	49.49	51.21
	4	**60.04**	**49.85**	51.47	**53.79**	**59.46**	**46.82**	**49.97**	**52.08**
Paris-Stanford	0	60.07	43.90	51.66	51.88	59.30	43.34	49.23	50.62
	1	60.28	45.60	**51.69**	52.52	59.00	44.04	49.31	50.78
	2	59.88	50.66	51.43	53.99	59.49	48.06	49.60	52.38
	3	59.87	50.40	51.31	53.86	58.91	**49.05**	49.52	**52.49**
	4	59.83	50.45	51.23	53.84	59.28	47.76	49.40	52.15
	5	60.01	**50.93**	51.41	54.12	59.31	48.82	49.69	52.61
	6	59.74	49.87	51.11	53.57	59.52	46.98	49.44	51.98
	7	60.12	50.12	51.23	53.82	**59.55**	47.66	**50.00**	52.40
	8	60.12	50.50	51.21	53.94	59.07	48.16	49.47	52.23
	9	**60.45**	50.41	51.51	**54.12**	59.27	48.54	49.41	52.41
	10	59.98	50.59	51.31	53.96	58.81	48.09	49.14	52.01
	11	60.19	49.92	51.46	53.86	59.16	47.82	49.20	52.06
Peking	0	**59.06**	**45.41**	50.18	**51.55**	58.14	43.33	48.67	50.05
	1	58.65	45.32	**50.52**	51.50	**58.15**	**43.78**	**48.90**	**50.28**
Prague	0	59.51	46.78	50.72	52.34	**59.33**	45.08	49.49	51.30
	1	**59.89**	**47.93**	**51.54**	**53.12**	59.06	**46.46**	49.50	**51.67**
	2	59.48	46.63	50.26	52.12	59.15	44.32	**49.60**	51.02
	3	59.75	45.88	51.23	52.29	58.89	44.46	49.11	50.82
	4	59.42	46.27	50.76	52.15	58.84	44.38	48.79	50.67
Stanford-Paris	0	60.27	51.28	51.40	54.32	59.61	49.52	49.75	52.96
	1	60.76	51.56	52.21	54.84	59.83	49.15	49.80	52.93
	2	60.76	52.06	51.69	54.84	59.89	**50.30**	**49.93**	**53.37**
	3	60.73	51.92	51.84	54.83	59.98	50.21	49.78	53.32
	4	**60.89**	52.84	52.11	55.28	**60.04**	49.91	49.79	53.25
	5	60.81	52.11	51.94	54.95	59.75	49.58	49.69	53.01
	6	60.67	52.93	52.26	55.29	59.73	49.98	49.62	53.11
	7	60.82	52.79	**52.49**	**55.37**	59.92	49.96	49.85	53.24
	8	60.60	**53.00**	51.85	55.15	59.53	49.91	49.76	53.07
	9	60.87	52.02	52.29	55.06	59.89	49.31	49.84	53.01
	10	60.63	52.57	52.04	55.08	59.53	49.50	49.64	52.89
Szeged	0	**59.82**	49.21	**52.47**	53.83	59.33	**50.61**	49.87	53.27
	1	59.76	**49.68**	52.43	**53.96**	59.32	50.52	50.02	**53.29**
	2	59.33	48.99	51.93	53.42	59.05	48.62	49.70	52.46
	3	59.29	48.86	52.12	53.42	**59.53**	48.49	**50.26**	52.76
	4	59.68	48.81	51.89	53.46	58.90	47.91	49.75	52.19
UPF	0	**59.60**	**50.01**	**51.19**	**53.60**	**59.26**	**48.81**	**49.37**	**52.48**
	1	56.04	45.68	47.57	49.76	56.02	45.51	46.81	49.45
	2	54.87	41.08	47.19	47.71	55.12	42.60	46.19	47.97
UW	0	**59.48**	**45.85**	**51.91**	**52.41**	**59.80**	**45.67**	**50.15**	**51.87**

Table 1: F-scores on the development and test sets. For each subtask, the best result for each team is in boldface and the best result overall is underlined.

		Development			Test		
System	Run	P	R	F	P	R	F
ECNU	0–3	65.62	49.27	56.28	66.69	48.84	56.38
	4	**66.06**	**49.68**	**56.71**	**66.83**	**49.38**	**56.80**
Paris-Stanford	0	**66.51**	50.43	57.37	**66.58**	49.21	56.59
	1	66.05	50.69	57.36	65.95	49.35	56.46
	2–11	66.26	**50.73**	**57.46**	66.27	**49.52**	**56.68**
Peking	0–1	**65.36**	**48.16**	**55.45**	**66.33**	**47.90**	**55.63**
Prague	0	65.26	49.61	56.37	66.20	**48.85**	56.22
	1	**65.71**	**50.00**	**56.79**	**66.48**	48.83	**56.31**
	2	65.06	49.30	56.10	66.47	48.70	56.21
	3	65.27	49.65	56.40	65.99	48.53	55.93
	4	65.22	49.35	56.19	65.99	48.67	56.02
Stanford-Paris	0–10	**65.87**	**50.46**	**57.15**	**66.45**	**49.76**	**56.90**
Szeged	0–4	**65.76**	**50.18**	**56.92**	**66.24**	**49.25**	**56.50**
UPF	0	**64.99**	**49.81**	**56.40**	**66.33**	**49.32**	**56.57**
	1	60.89	46.80	52.92	62.31	45.73	52.75
	2	62.04	45.12	52.24	62.09	43.81	51.37
UW	0	**65.76**	**50.18**	**56.92**	**66.24**	**49.25**	**56.50**

Table 2: Expression extraction scores without reranking.

		Development			Test		
System	Run	P	R	F	P	R	F
ECNU	0	62.28	60.46	61.36	60.27	57.42	58.81
	1	64.72	**63.71**	64.21	62.86	60.04	61.42
	2	64.94	63.31	64.12	62.15	59.75	60.92
	3	63.92	62.14	63.02	62.11	58.17	60.08
	4	**65.33**	63.41	**64.35**	**63.32**	**61.07**	**62.17**
Paris-Stanford	0	66.13	51.96	58.19	65.04	51.32	57.37
	1	66.80	51.48	58.15	65.80	52.73	58.55
	2	67.65	63.62	65.57	65.87	61.30	63.50
	3	67.61	62.91	65.18	66.22	**62.43**	**64.27**
	4	68.15	63.71	65.86	65.10	61.75	63.38
	5	67.83	**64.13**	**65.93**	66.62	62.03	64.24
	6	66.34	63.63	64.96	64.21	60.27	62.18
	7	67.57	62.35	64.85	65.78	60.96	63.28
	8	67.10	63.81	65.41	65.59	62.42	63.97
	9	68.19	61.71	64.79	**66.77**	61.04	63.78
	10	**68.24**	63.48	65.78	65.86	60.92	63.30
	11	66.33	62.77	64.50	64.90	60.56	62.66
Peking	0	66.02	**54.07**	59.45	65.63	53.64	59.04
	1	**66.21**	54.05	**59.52**	**66.57**	**54.55**	**59.96**
Prague	0	65.41	59.79	62.47	62.61	57.21	59.79
	1	64.21	**61.65**	**62.91**	62.31	**59.74**	**61.00**
	2	**65.59**	57.05	61.02	**63.45**	54.63	58.71
	3	63.50	58.89	61.11	61.26	56.72	58.90
	4	63.93	59.96	61.88	61.00	56.25	58.53
Stanford-Paris	0	69.48	64.01	66.63	67.26	60.54	63.72
	1	69.02	64.18	66.52	67.47	61.30	64.23
	2	70.59	64.94	67.65	67.69	61.02	64.18
	3	70.31	65.16	67.64	67.43	61.58	64.37
	4	70.56	66.42	68.43	66.68	61.95	64.23
	5	70.12	65.31	67.63	68.18	61.56	64.70
	6	**71.49**	65.80	68.53	**68.86**	61.81	65.14
	7	71.16	65.87	68.41	68.44	62.25	**65.20**
	8	71.05	**66.73**	**68.82**	67.64	**62.57**	65.01
	9	69.42	65.00	67.14	66.68	61.42	63.94
	10	70.21	65.85	67.96	67.30	62.01	64.55
Szeged	0	63.98	62.03	62.99	66.73	65.04	65.88
	1	64.53	**62.69**	**63.60**	**67.04**	**65.63**	**66.33**
	2	**66.44**	60.77	63.48	66.05	60.45	63.13
	3	64.93	61.31	63.06	65.35	61.28	63.25
	4	62.68	61.81	62.24	63.37	61.66	62.51
UPF	0	**65.70**	**60.41**	**62.94**	**66.25**	**61.19**	**63.62**
	1	63.87	56.29	59.84	64.65	56.71	60.42
	2	58.98	49.63	53.90	61.03	51.50	55.86
UW	0	**67.96**	**53.94**	**60.14**	**67.31**	**54.41**	**60.17**

Table 3: In vitro holder extraction scores.