CoNLL-2009: Shared Task

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task

Conference Chairs:
Suzanne Stevenson and Xavier Carreras

Shared Task Organizing Committee Chair:
Jan Hajič

June 4, 2009
Boulder, Colorado

Production and Manufacturing by
Omnipress Inc.
2600 Anderson Street
Madison, WI 53707
USA

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
209 N. Eighth Street
Stroudsburg, PA 18360
USA
Tel: +1-570-476-8006
Fax: +1-570-476-0860
acl@aclweb.org

ISBN 978-1-932432-29-9

Introduction

Welcome to the ACL Workshop on Unresolved Matters. We received 17 submissions, and due to a rigerous review process, we rejected 16.

Organizers:

Jan Hajič (chair), Charles University in Prague (Czech Republic)
Massimiliano Ciaramita, Google Inc., Zürich (Switzerland)
Richard Johansson, University of Trento (Italy)
Daisuke Kawahara, National Institute of Information and Communications Technology (Japan)
Maria Antònia Martí, University of Barcelona (Spain)
Lluís Màrquez, Technical University of Catalonia, Barcelona (Spain)
Adam Meyers, New York University (USA)
Joakim Nivre, Uppsala University and Växjö University (Sweden)
Sebastian Padó, Stuttgart University (Germany)
Jan Štěpánek, Charles University, Prague (Czech Republic)
Pavel Straňák, Charles University, Prague (Czech Republic)
Mihai Surdeanu, Stanford University (USA)
Nianwen (Bert) Xue, Brandeis University, Boston (USA)
Yi Zhang, DFKI GmbH, Saarbrücken (Germany)

Program Committee:

Massimiliano Ciaramita (chair), Google Inc., Zürich (Switzerland)

Reviewers:

Masayuki Asahara, Nara Institute of Science and Technology (Japan)
Bernd Bohnet, International Computer Science Institute, Berkeley (USA)
Wanxiang Che, Harbin Institute of Technology (China)
Wenliang Chen, National Institute of Information and Communications Technology (Japan)
Massimiliano Ciaramita (chair), Google Inc., Zürich (Switzerland)
Qifeng Dai, University of Science and Technology of China (China)
Jan Hajič, Charles University in Prague (Czech Republic)
James Henderson, University of Geneva (Switzerland)
Richard Johansson, University of Trento (Italy)
Daisuke Kawahara, National Institute of Information and Communications Technology (Japan)
Baoli Li, Trinity College Dublin (Ireland)
Lu Li, Harbin Institute of Technology (China)
Xavier Lluís, Technical University of Catalonia, Barcelona (Spain)
Lluís Màrquez, Technical University of Catalonia, Barcelona (Spain)
Maria Antònia Martí, University of Barcelona (Spain)
Adam Meyers, New York University (USA)
Roser Morante, University of Antwerp (Belgium)
Joakim Nivre, Uppsala University and Växjö University (Sweden)
Pierre Nugues, Lund University (Sweden)
Sebastian Padó, Stuttgart University (Germany)
Xipeng Qiu, Fudan University, Shanghai (China)

Han Ren, Wuhan University (China)
Antoine Rozenknop, Swiss Federal Institute of Technology in Lausanne (Switzerland)
Pavel Straňák, Charles University, Prague (Czech Republic)
Mihai Surdeanu, Stanford University (USA)
Oscar Täckström, Swedish Institute of Computer Science (Sweden)
Ivan Meza-Ruiz, University of Edinburgh (United Kingdom)
Yotaro Watanabe, Nara Institute of Science and Technology (Japan)
Nianwen (Bert) Xue, Brandeis University, Boston (USA)
Daniel Zeman, Charles University, Prague (Czech Republic)
Yi Zhang, DFKI GmbH, Saarbrücken (Germany)
Hai Zhao, City University of Hong Kong (China)

Table of Contents

The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages
Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue and Yi Zhang 1

An Iterative Approach for Joint Dependency Parsing and Semantic Role Labeling
Qifeng Dai, Enhong Chen and Liu Shi 19

Joint Memory-Based Learning of Syntactic and Semantic Dependencies in Multiple Languages
Roser Morante, Vincent Van Asch and Antal van den Bosch 25

Hybrid Multilingual Parsing with HPSG for SRL
Yi Zhang, Rui Wang and Stephan Oepen 31

A Latent Variable Model of Synchronous Syntactic-Semantic Parsing for Multiple Languages
Andrea Gesmundo, James Henderson, Paola Merlo and Ivan Titov 37

Multilingual Semantic Role Labeling
Anders Björkelund, Love Hafdell and Pierre Nugues 43

Multilingual Dependency-based Syntactic and Semantic Parsing
Wanxiang Che, Zhenghua Li, Yongqiang Li, Yuhang Guo, Bing Qin and Ting Liu 49

Multilingual Dependency Learning: A Huge Feature Engineering Method to Semantic Dependency Parsing
Hai Zhao, Wenliang Chen, Chunyu Kity and Guodong Zhou 55

Multilingual Dependency Learning: Exploiting Rich Features for Tagging Syntactic and Semantic Dependencies
Hai Zhao, Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto and Kentaro Torisawa 61

Efficient Parsing of Syntactic and Semantic Dependency Structures
Bernd Bohnet 67

Exploring Multilingual Semantic Role Labeling
Baoli Li, Martin Emms, Saturnino Luz and Carl Vogel 73

A Second-Order Joint Eisner Model for Syntactic and Semantic Dependency Parsing
Xavier Lluís, Stefan Bott and Lluís Màrquez 79

Multilingual Semantic Role Labelling with Markov Logic
Ivan Meza-Ruiz and Sebastian Riedel 85

The Crotal SRL System : a Generic Tool Based on Tree-structured CRF
Erwan Moreau and Isabelle Tellier 91

Parsing Syntactic and Semantic Dependencies for Multiple Languages with A Pipeline Approach
Han Ren, Donghong Ji, Jing Wan and Mingyao Zhang 97

Multilingual Semantic Parsing with a Pipeline of Linear Classifiers
Oscar Täckström 103

A Joint Syntactic and Semantic Dependency Parsing System based on Maximum Entropy Models
Buzhou Tang, Lu Li, Xinxin Li, Xuan Wang and Xiaolong Wang 109

Multilingual Syntactic-Semantic Dependency Parsing with Three-Stage Approximate Max-Margin Linear Models
Yotaro Watanabe, Masayuki Asahara and Yuji Matsumoto 114

A Simple Generative Pipeline Approach to Dependency Parsing and Semantic Role Labeling
Daniel Zeman 120

Conference Program

Thursday, June 4, 2009

Shared Task Session 1: Overview and Oral Presentations

14:00–14:20 *The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages*
Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue and Yi Zhang

14:20–14:30 *An Iterative Approach for Joint Dependency Parsing and Semantic Role Labeling*
Qifeng Dai, Enhong Chen and Liu Shi

14:30–14:40 *Joint Memory-Based Learning of Syntactic and Semantic Dependencies in Multiple Languages*
Roser Morante, Vincent Van Asch and Antal van den Bosch

14:40–14:50 *Hybrid Multilingual Parsing with HPSG for SRL*
Yi Zhang, Rui Wang and Stephan Oepen

14:50–15:00 *A Latent Variable Model of Synchronous Syntactic-Semantic Parsing for Multiple Languages*
Andrea Gesmundo, James Henderson, Paola Merlo and Ivan Titov

15:00–15:10 *Multilingual Semantic Role Labeling*
Anders Björkelund, Love Hafdell and Pierre Nugues

15:10–15:20 *Multilingual Dependency-based Syntactic and Semantic Parsing*
Wanxiang Che, Zhenghua Li, Yongqiang Li, Yuhang Guo, Bing Qin and Ting Liu

15:20–15:30 *Multilingual Dependency Learning: A Huge Feature Engineering Method to Semantic Dependency Parsing*
Hai Zhao, Wenliang Chen, Chunyu Kity and Guodong Zhou

15:30–15:40 *Multilingual Dependency Learning: Exploiting Rich Features for Tagging Syntactic and Semantic Dependencies*
Hai Zhao, Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto and Kentaro Torisawa

15:40–16:10 **Coffee Break**

Thursday, June 4, 2009 (continued)

Shared Task Session 2: Poster Session (16:10–18:00)

Multilingual Semantic Role Labeling
Anders Björkelund, Love Hafdell and Pierre Nugues

Efficient Parsing of Syntactic and Semantic Dependency Structures
Bernd Bohnet

Multilingual Dependency-based Syntactic and Semantic Parsing
Wanxiang Che, Zhenghua Li, Yongqiang Li, Yuhang Guo, Bing Qin and Ting Liu

An Iterative Approach for Joint Dependency Parsing and Semantic Role Labeling
Qifeng Dai, Enhong Chen and Liu Shi

A Latent Variable Model of Synchronous Syntactic-Semantic Parsing for Multiple Languages
Andrea Gesmundo, James Henderson, Paola Merlo and Ivan Titov

Exploring Multilingual Semantic Role Labeling
Baoli Li, Martin Emms, Saturnino Luz and Carl Vogel

A Second-Order Joint Eisner Model for Syntactic and Semantic Dependency Parsing
Xavier Lluís, Stefan Bott and Lluís Màrquez

Multilingual Semantic Role Labelling with Markov Logic
Ivan Meza-Ruiz and Sebastian Riedel

Joint Memory-Based Learning of Syntactic and Semantic Dependencies in Multiple Languages
Roser Morante, Vincent Van Asch and Antal van den Bosch

The Crotal SRL System : a Generic Tool Based on Tree-structured CRF
Erwan Moreau and Isabelle Tellier

Parsing Syntactic and Semantic Dependencies for Multiple Languages with A Pipeline Approach
Han Ren, Donghong Ji, Jing Wan and Mingyao Zhang

Multilingual Semantic Parsing with a Pipeline of Linear Classifiers
Oscar Täckström

Thursday, June 4, 2009 (continued)

A Joint Syntactic and Semantic Dependency Parsing System based on Maximum Entropy Models
Buzhou Tang, Lu Li, Xinxin Li, Xuan Wang and Xiaolong Wang

Multilingual Syntactic-Semantic Dependency Parsing with Three-Stage Approximate Max-Margin Linear Models
Yotaro Watanabe, Masayuki Asahara and Yuji Matsumoto

A Simple Generative Pipeline Approach to Dependency Parsing and Semantic Role Labeling
Daniel Zeman

Hybrid Multilingual Parsing with HPSG for SRL
Yi Zhang, Rui Wang and Stephan Oepen

Multilingual Dependency Learning: Exploiting Rich Features for Tagging Syntactic and Semantic Dependencies
Hai Zhao, Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto and Kentaro Torisawa

Multilingual Dependency Learning: A Huge Feature Engineering Method to Semantic Dependency Parsing
Hai Zhao, Wenliang Chen, Chunyu Kity and Guodong Zhou

The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages

Jan Hajič[⋆] **Massimiliano Ciaramita**[†] **Richard Johansson**[‡] **Daisuke Kawahara**[◇]
Maria Antònia Martí[⋆⋆] **Lluís Màrquez**[⋆†] **Adam Meyers**[⋆‡] **Joakim Nivre**[⋆◇] **Sebastian Padó**[◇⋆]
Jan Štěpánek[⋆] **Pavel Straňák**[⋆] **Mihai Surdeanu**[‡⋆] **Nianwen Xue**[‡‡] **Yi Zhang**[‡◇]

⋆: Charles University in Prague, {hajic,stepanek,stranak}@ufal.mff.cuni.cz
†: Google Inc., massi@google.com
‡: University of Trento, johansson@disi.unitn.it
◇: National Institute of Information and Communications Technology, dk@nict.go.jp
⋆⋆: University of Barcelona, amarti@ub.edu
⋆†: Technical University of Catalonia, Barcelona, lluism@lsi.upc.edu
⋆‡: New York University, meyers@cs.nyu.edu
⋆◇: Uppsala University and Växjö University, joakim.nivre@lingfil.uu.se
◇⋆: Stuttgart University, pado@ims.uni-stuttgart.de
‡⋆: Stanford University, mihais@stanford.edu
‡‡: Brandeis University, xuen@brandeis.edu
‡◇: Saarland University, yzhang@coli.uni-sb.de

Abstract

For the 11th straight year, the Conference on Computational Natural Language Learning has been accompanied by a shared task whose purpose is to promote natural language processing applications and evaluate them in a standard setting. In 2009, the shared task was dedicated to the joint parsing of syntactic and semantic dependencies in multiple languages. This shared task combines the shared tasks of the previous five years under a unique dependency-based formalism similar to the 2008 task. In this paper, we define the shared task, describe how the data sets were created and show their quantitative properties, report the results and summarize the approaches of the participating systems.

1 Introduction

Every year since 1999, the Conference on Computational Natural Language Learning (CoNLL) launches a competitive, open "Shared Task". A common ("shared") task is defined and datasets are provided for its participants. In 2004 and 2005, the shared tasks were dedicated to semantic role labeling (SRL) in a monolingual setting (English). In 2006 and 2007 the shared tasks were devoted to the parsing of syntactic dependencies, using corpora from up to 13 languages. In 2008, the shared task (Surdeanu et al., 2008) used a unified dependency-based formalism, which modeled both syntactic dependencies and semantic roles for English. The CoNLL-2009 Shared Task has built on the 2008 results by providing data for six more languages (Catalan, Chinese, Czech, German, Japanese and Spanish) in addition to the original English[1]. It has thus naturally extended the path taken by the five most recent CoNLL shared tasks.

As in 2008, the CoNLL-2009 shared task combined dependency parsing and the task of identifying and labeling semantic arguments of verbs (and other parts of speech whenever available). Participants had to choose from two tasks:

- Joint task (syntactic dependency parsing *and* semantic role labeling), or
- SRL-only task (syntactic dependency parses have been provided by the organizers, using state-of-the art parsers for the individual languages).

[1]There are some format changes and deviations from the 2008 task data specification; see Sect. 2.3

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 1–18,
Boulder, Colorado, June 2009.

In contrast to the previous year, the evaluation data indicated which words were to be dealt with (for the SRL task). In other words, (predicate) disambiguation was still part of the task, whereas the *identification* of argument-bearing words was not. This decision was made to compensate for the significant differences between languages and between the annotation schemes used.

The "closed" and "open" challenges have been kept from last year as well; participants could have chosen one or both. In the closed challenge, systems had to be trained strictly with information contained in the given training corpus; in the open challenge, systems could have been developed making use of any kind of external tools and resources.

This paper is organized as follows. Section 2 defines the task, including the format of the data, the evaluation metrics, and the two challenges. A substantial portion of the paper (Section 3) is devoted to the description of the conversion and development of the data sets in the additional languages. Section 4 shows the main results of the submitted systems in the Joint and SRL-only tasks. Section 5 summarizes the approaches implemented by participants. Section 6 concludes the paper. In all sections, we will mention some of the differences between last year's and this year's tasks while keeping the text self-contained whenever possible; for details and observations on the English data, please refer to the overview paper of the CoNLL-2008 Shared Task (Surdeanu et al., 2008) and to the references mentioned in the sections describing the other languages.

2 Task Definition

In this section we provide the definition of the shared task; after introducing the two challenges and the two tasks the participants were to choose, we continue with the format of the shared task data, followed by a description of the evaluation metrics used.

For three of the languages (Czech, English and German), out-of-domain data (OOD) have also been prepared for the final evaluation, following the same guidelines and formats.

2.1 Closed and Open Challenges

Similarly to the CoNLL-2005 and CoNLL-2008 shared tasks, this shared task evaluation is separated into two challenges:

Closed Challenge The aim of this challenge was to compare performance of the participating systems in a fair environment. Systems had to be built strictly with information contained in the given training corpus, and tuned with the development section. In addition, the lexical frame files (such as the PropBank and NomBank for English, the valency dictionary PDT-Vallex for Czech etc.) were provided and may have been used. These restrictions mean that outside parsers (not trained by the participants' systems) could not be used. However, we did provide the output of a single, state-of-the-art dependency parser for each language so that participants could build a SRL-only system (using the provided parses as inputs) within the closed challenge (as opposed to the 2008 shared task).

Open Challenge Systems could have been developed making use of any kind of external tools and resources. The only condition was that such tools or resources must not have been developed with the annotations of the test set, both for the input and output annotations of the data. In this challenge, we were interested in learning methods which make use of any tools or resources that might improve the performance. The comparison of different systems in this setting may not be fair, and thus ranking of systems is not necessarily important.

2.2 Joint and SRL-only tasks

In 2008, systems participating in the open challenge could have used state-of-the-art parsers for the syntactic dependency part of the task. This year, we have provided the output of these parsers for all the languages in an uniform way, thus allowing an orthogonal combination of the two tasks and the two challenges. For the SRL-only task, participants in the closed challenge simply had to use the provided parses only.

Despite the provisions for the SRL-only task, we are more interested in the approaches and results of the Joint task. Therefore, primary system ranking is provided for the Joint task while additional measures

are computed for various combinations of parsers and SRL methods across the tasks and challenges.

2.3 Data Format

The data format used in this shared task has been based on the CoNLL-2008 shared task, with some differences. The data follows these general rules:

- The files contain sentences separated by a blank line.
- A sentence consists of one or more tokens and the information for each token is represented on a separate line.
- A token consists of at least 14 fields. The fields are separated by one or more whitespace characters (spaces or tabs). Whitespace characters are not allowed within fields.

The data is thus a large table with whitespace-separated fields (columns). The fields provided in the data are described in Table 1. They are identical for all languages, but they may differ in contents; for example, some fields might not be filled for all the languages provided (such as the FEAT or PFEAT fields).

For the SRL-only task, participants have been provided will all the data but the PRED and APREDs, which they were supposed to fill in with their correct values. However, they did not have to determine which tokens are predicates (or more precisely, which are the argument-bearing tokens), since they were marked by 'Y' in the FILLPRED field.

For the Joint task, participants could not (in addition to the PRED and APREDs) see the gold-standard nor the predicted syntactic dependencies (HEAD, PHEAD) and their labels (DEPREL, PDEPREL). These syntactic dependencies were also to be filled by participants' systems.

In both tasks, participants have been free to use any other data (columns) provided, except the LEMMA, POS and FEAT columns (to get more 'realistic' results using only their automatically predicted variants PLEMMA, PPOS and PFEAT).

Besides the corpus proper, predicate dictionaries have been provided to participants in order to be able to properly match the predicates to the tokens in the corpus; their contents could have been used e.g. as features for the PRED/APREDs predictions (or even for the syntactic dependencies, i.e., for filling in the PHEAD and PDEPREL fields).

The system of filling-in the APREDs follows the 2008 pattern; for each argument-bearing token (predicate), a new APREDn column is created in the order in which the predicate token is encountered within the sentence (i.e., based on its ID seen as a numerical value). Then, for each token in the sentence, the value in the intersection of the APREDn column and the token row is either left unfilled (if the token is not an argument), or a predicate-argument label(s) is(are) filled in.

The differences between the English-only 2008 task and this year's multilingual task can be briefly summarized as follows:

- only "split"[2] lemmas and forms have been provided in the English datasets (for the other languages, original tokenization from the respective treebanks has been used);
- rich morphological features have been added wherever available;
- syntactic dependencies by state-of-the-art parsers have been provided (for the SRL-only task);
- multiple semantic labels for a single token have been allowed (and properly evaluated) in the APREDs columns;
- predicates have been pre-identified and marked in both the training and test data;
- some of the fields (e.g. the `APREDx`) and values (`ARG0` → `A0` etc.) have been renamed.

2.4 Evaluation Measures

It was required that participants submit results in all seven languages in the chosen task and in any of (or both) the challenges. Submission of out-of-domain data files has been optional.

The main evaluation measure, according to which systems are primarily compared, is the Joint task,

[2] Splitting of forms and lemmas in English has been introduced in the 2008 shared task to match the tokenization convention for the arguments in NomBank.

Field #	Name	Description
1	ID	Token counter, starting at 1 for each new sentence
2	FORM	Form or punctuation symbol (the token; "split" for English)
3	LEMMA	Gold-standard lemma of FORM
4	PLEMMA	Automatically predicted lemma of FORM
5	POS	Gold-standard POS (major POS only)
6	PPOS	Automatically predicted major POS by a language-specific tagger
7	FEAT	Gold-standard morphological features (if applicable)
8	PFEAT	Automatically predicted morphological features (if applicable)
9	HEAD	Gold-standard syntactic head of the current token (ID or 0 if root)
10	PHEAD	Automatically predicted syntactic head
11	DEPREL	Gold-standard syntactic dependency relation (to HEAD)
12	PDEPREL	Automatically predicted dependency relation to PHEAD
13	FILLPRED	Contains 'Y' for argument-bearing tokens
14	PRED	(sense) identifier of a semantic "predicate" coming from a current token
15...	APREDn	Columns with argument labels for each semantic predicate (in the ID order)

Table 1: Description of the fields (columns) in the data provided. The values of columns 9, 11 and 14 and above are not provided in the evaluation data; for the Joint task, columns 9–12 are also empty in the evaluation data.

closed challenge, Macro F_1 score. However, scores can also be computed for a number of other conditions:

- Task: Joint or SRL-only
- Challenge: open or closed
- Domain: in-domain data (IDD, separated from training corpus) or out-of-domain data (OOD)

Joint task participants are also evaluated separately on the syntactic dependency task (labeled attachment score, LAS). Finally, systems competing in both tasks are compared on semantic role labeling alone, to assess the impact of the the joint parsing/SRL task compared to an SRL-only task on preparsed data.

Finally, as an explanatory measure, precision and recall of the semantic labeling task have been computed and tabulated.

We have decided to omit several evaluation figures that were reported in previous years, such as the percentage of completely correct sentences ("Exact Match"), unlabeled scores, etc. With seven languages, two tasks (plus two challenges, and the IDD/OOD distinction), there are enough results to get lost even as it is.

2.4.1 Syntactic Dependency Measures

The LAS score is defined similarly as in the previous shared tasks, as the percentage of tokens for which a system has predicted the correct HEAD and DEPREL columns. The unlabeled attachment score (UAS), i.e., the percentage of tokens with correct HEAD regardless if the DEPREL is correct, has not been officially computed this year. No precision and recall measures are applicable, since all systems are supposed to output a single dependency with a single label (see also below the footnote to the description of the combined score).

2.4.2 Semantic Labeling Measures

The semantic propositions are evaluated by converting them to semantic dependencies, i.e., we create n semantic dependencies from every predicate to its n arguments. These dependencies are labeled with the labels of the corresponding arguments. Additionally, we create a semantic dependency from each predicate to a virtual ROOT node. The latter dependencies are labeled with the predicate senses. This approach guarantees that the semantic dependency structure conceptually forms a single-rooted, connected (but not necessarily acyclic) graph. More importantly, this scoring strategy implies that if a system assigns the incorrect predicate sense, it still receives some points for the arguments correctly assigned. For example, for the correct proposition:

```
verb.01: A0, A1, AM-TMP
```

the system that generates the following output for the same argument tokens:

```
verb.02: A0, A1, AM-LOC
```

receives a labeled precision score of 2/4 because two out of four semantic dependencies are incorrect: the dependency to ROOT is labeled `02` instead of `01` and the dependency to the `AM-TMP` is incorrectly labeled `AM-LOC`. Using this strategy we compute precision, recall, and F_1 scores for semantic dependencies (labeled only).

For some languages (Czech, Japanese) there may be more than one label in a given argument position; for example, this happens in Czech in special cases of reciprocity when the same token serves as two or more arguments to the same predicate. The scorer takes this into account and considers such cases to be (as if) multiple predicate-argument relations for the computation of the evaluation measures.

For example, for the correct proposition:

```
v1f1: ACT|EFF, ADDR
```

the system that generates the following output for the same argument tokens:

```
v1f1: ACT, ADDR|PAT
```

receives a labeled precision score of 3/4 because the PAT is incorrect and labeled recall 3/4 because the EFF is missing (should the ACT|EFF and ADDR|PAT be taken as atomic values, the scores would then be zero).

2.4.3 Combined Syntactic and Semantic Score

We combine the syntactic and semantic measures into one global measure using macro averaging. We compute macro precision and recall scores by averaging the labeled precision and recall for semantic dependencies with the LAS for syntactic dependencies:[3]

$$LMP = W_{sem} * LP_{sem} + (1 - W_{sem}) * LAS \quad (1)$$

$$LMR = W_{sem} * LR_{sem} + (1 - W_{sem}) * LAS \quad (2)$$

where LMP is the labeled macro precision and LP_{sem} is the labeled precision for semantic dependencies. Similarly, LMR is the labeled macro recall and LR_{sem} is the labeled recall for semantic dependencies. W_{sem} is the weight assigned to the semantic task.[4] The macro labeled F_1 score, which was used for the ranking of the participating systems, is computed as the harmonic mean of LMP and LMR.

[3] We can do this because the LAS for syntactic dependencies is a special case of precision and recall, where the predicted number of dependencies is equal to the number of gold dependencies.

3 Data

The unification of the data formats for the various languages appeared to be a challenge in itself. We will briefly describe the processes of the conversion of the existing treebanks in the seven languages of the CoNLL-2009 shared task. In many instances, the original treebanks had to be not only converted format-wise, but also merged with other resources in order to generate useful training and testing data that fit the task description.

3.1 The Input Corpora

The data used as the input for the transformations aimed at arriving at the data contents and format described in Sect. 2.3 are described in (Taulé et al., 2008), (Xue and Palmer, 2009), (Hajič et al., 2006), (Surdeanu et al., 2008), (Burchardt et al., 2006) and (Kawahara et al., 2002).

In the subsequent sections, the procedures for the data conversion for the individual languages are described. The data has been collected by the main organization site and checked for format errors, and repackaged for distribution.

There were three packages of the data distributed to the participants: Trial, Training plus Development, and Evaluation. The Trial data were rather small, just to give the feeling of the format and languages involved. A visual representation of the Trial data was also created to make understanding of the data easier. Any data in the same format can be transformed and displayed in the Tree Editor TrEd[5] (Pajas and Štěpánek, 2008) with the CoNLL 2009 Shared Task extension that can be installed from within the editor. A sample visualization of an English sentence after its conversion to the shared task format (Sect. 2.3) is in Fig. 1.

Due to licensing requirements, every package of the data had to be split into two portions. One portion (Catalan, German, Japanese, and Spanish data) was published on the task's webpage for down-

[4] We assign equal weight to the two tasks, i.e., $W_{sem} = 0.5$.

[5] `http://ufal.mff.cuni.cz/~pajas/tred`

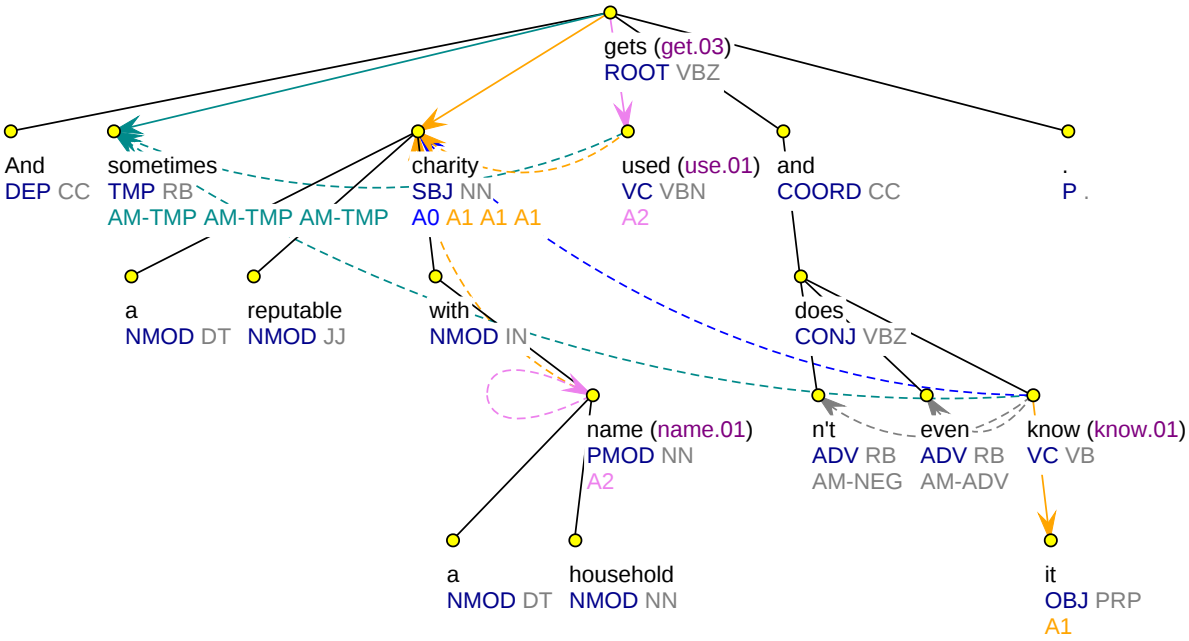

Figure 1: Visualisation of the English sentence *"And sometimes a reputable charity with a houshold name gets used and doesn't even know it."* (Penn Treebank, wsj_0559) showing jointly the labeled syntactic and semantic dependencies. The basic tree shape comes from the syntactic dependencies; syntactic labels and POS tags are on the 2^{nd} line at each node. Semantic dependencies which do not follow the syntactic ones use dotted lines. Predicate senses in parentheses (`use:01`, ...) follow the word label. SRLs (`A0`, `AM-TMP`, ...) are on the last line. Please note that multiple semantic dependencies (e.g., there are four for *charity*: A0 ← *know*, A1 ← *gets*, A1 ← *used*, A1 ← *name*) and self-dependencies (*name*) appear in this sentence.

load, the other portion (Czech, English, and Chinese data) was invoiced and distributed by the Linguistic Data Consortium under a special agreement free of charge.

Distribution of the Evaluation package was a bit more complicated, because there were two types of the packages - one for the Joint task and one for the SRL-only task. Every participant had to subscribe to one of the two tasks; subsequently, they obtained the appropriate data (again, from the webpage and LDC).

Prior to release, each data file was checked to eliminate errors. The following test were carried out:

- For every sentence, number of PREDs rows matches the number of APREDs columns.
- The first line of each file is never empty, while the last line always is.
- The first character on a non-empty line is always a digit, the last one is never a whitespace.
- The number of empty lines (i.e. the number of sentences) equals the number of lines beginning with "1".
- The data contain no spaces nor double tabs.

Some statistics on the data can be seen in Tables 2, 3 and 4. Whereas the training sizes of the data have not been that different as they were e.g. for the 2007 shared task on multilingual dependency parsing (Nivre et al., 2007)[6], substantial differences existed in the distribution of the predicates and arguments, the input features, the out-of-vocabulary rates, and other statistical characteristics of the data.

Data sizes have been relatively uniform in all the datasets, with Japanese having the smallest dataset

[6] `http://nextens.uvt.nl/depparse-wiki/DataOverview`

containing data for SRL annotation training. To compensate at least for the dependency parsing part, an additional, large Japanese corpus with syntactic dependency annotation has been provided.

The average sentence length, the vocabulary sizes for FORM and LEMMA fields and the OOV rates characterize quite naturally the properties of the respective languages (in the domain of the training and evaluation data). It is no surprise that the FORM OOV rate is the highest for Czech, a highly inflectional language, and that the LEMMA OOV rate is the highest for German (as a consequence of keeping compounds as a single lemma). The other statistics also reflect (to a large extent) the annotation specification and conventions used for the original treebanks and/or the result of the conversion process to the unified CoNLL-2009 Shared Task format.

Starting with the POS and FEAT fields, it can be seen that Catalan, Czech and Spanish use only the 12 major part-of-speech categories as values of the POS field (with richly populated FEAT field); English and Chinese are the opposite extreme, disregarding the use of the FEAT field completely and coding everything as a POS value. While for Chinese this is quite understandable, English follows the PTB tradition in this respect. German and Japanese use relatively rich set of values in both the POS and FEAT fields.

For the dependency relations (DEPREL), all the languages use a similarly-sized set except for Japanese, which only encodes the distinction between a root and a dependent node (and some infrequent special ones).

Evaluation data are over 10% of the size of the training data for Catalan, Chinese, Czech, Japanese and Spanish and roughly 5% for English and German.

Table 3 shows the distribution of the five most frequent dependency relations (determined as part of the subtask of syntactic parsing). With the exception of Japanese, which essentially does not label dependency relations at this level, all the other languages show little difference in this distribution. For example, the unconditioned probability of "subjects" is almost the same for all the six other languages (between 6 and 8 percent). The probability mass covered by the first five most frequent DEPRELs is also almost the same (again, except for Japanese), suggesting that the labeling task might have similar difficulty[7]. The most skewed one is for Czech (after Japanese).

Table 4 shows similar statistics for the argument labels (PRED/APREDs); it also adds the average number of arguments per "predicate" token, since this is part of the SRL task[8]. It is apparent from the comparison of the "Total" rows in this table and Table 3 that the first five argument labels cover more that their syntactic counterparts. For example, the arguments A0-A4 account for all but 3% of all arguments labels, whereas Spanish and Catalan have much more rich set of argument labels, with a high entropy of the most-frequent-label distribution.

3.2 Catalan and Spanish

The Catalan and Spanish datasets (Taulé et al., 2008) were generated from the AnCora corpora[9] through an automatic conversion process from a constituent-based formalism to dependencies (Civit et al., 2006).

AnCora corpora contain about half million words for Catalan and Spanish annotated with syntactic and semantic information. Text sources for the Catalan corpus are EFE news agency (∼75Kw), ACN Catalan news agency (∼225Kw), and 'El Periódico' newspaper (∼200Kw). The Spanish corpus comes from the Lexesp Spanish balanced corpus (∼75Kw), the EFE Spanish news agency (∼225Kw), and the Spanish version of 'El Periódico' (∼200Kw). The subset from 'El Periódico' corresponds to the same news in Catalan and Spanish, spanning from January to December 2000.

Linguistic annotation is the same in both languages and includes: PoS tags with morphological features (gender, number, person, etc.), lemmatization, syntactic dependencies (syntactic functions), semantic dependencies (arguments and thematic roles), named entities and predicate semantic classes (Lexical Semantic Structure, LSS). Tag sets are shared by the two languages.

If we take into account the complete PoS tags,

[7] Yes, this is overgeneralization since this distribution does not condition on the features, dependencies etc. But as a rough measure, it often correlates well with the results.

[8] A number below 1 means there are some argument-bearing words (often nouns) which have no arguments in the particular sentence in which they appear.

[9] `http://clic.ub.edu/ancora`

Characteristic	Catalan	Chinese	Czech	English	German	Japanese	Spanish
Training data size (sentences)	13200	22277	38727	39279	36020	4393[a]	14329
Training data size (tokens)	390302	609060	652544	958167	648677	112555[a]	427442
Avg. sentence length (tokens)	29.6	27.3	16.8	24.4	18.0	25.6	29.8
Tokens with arguments[b] (%)	9.6	16.9	63.5	18.7	2.7	22.8	10.3
DEPREL types	50	41	49	69	46	5	49
POS types	12	41	12	48	56	40	12
FEAT types	237	1	1811	1	267	302	264
FORM vocabulary size	33890	40878	86332	39782	72084	36043	40964
LEMMA vocabulary size	24143	40878	37580	28376	51993	30402	26926
Evaluation data size (sent.)	1862	2556	4213	2399	2000	500	1725
Evaluation data size (tokens)	53355	73153	70348	57676	31622	13615	50630
Evaluation FORM OOV[c]	5.40	3.92	7.98/8.62[d]	1.58/3.76[d]	7.93/7.57[d]	6.07	5.63
Evaluation LEMMA OOV[c]	4.14	3.92	3.03/4.29[d]	1.08/2.30[d]	5.83/7.36[d]	5.21	3.69

Table 2: Elementary data statistics for the CoNLL-2009 Shared Task languages. The data themselves, the original treebanks they were derived from and the conversion process are described in more detail in sections 3.2-3.7. All evaluation data statistics are derived from the in-domain evaluation data.

[a]There were additional 33257 sentences (839947 tokens) available for syntactic dependency parsing of Japanese; the type and vocabulary statistics are computed using this larger dataset.

[b]Percentage of tokens with FILLPRED='Y'.

[c]Percentage of FORM/LEMMA tokens not found in the respective vocabularies derived solely from the training data.

[d]OOV percentage for in-domain/out-of-domain data.

DEPREL	Catalan	Chinese	Czech	English	German	Japanese	Spanish
Labels	sn 0.16	COMP 0.21	Atr 0.26	NMOD 0.27	NK 0.31	D 0.93	sn 0.16
	spec 0.15	NMOD 0.14	AuxP 0.10	P 0.11	PUNC 0.14	ROOT 0.04	spec 0.15
	f 0.11	ADV 0.10	Adv 0.10	PMOD 0.10	MO 0.12	P 0.03	f 0.12
	sp 0.09	UNK 0.09	Obj 0.07	SBJ 0.07	SB 0.07	A 0.00	sp 0.08
	suj 0.07	SBJ 0.08	Sb 0.06	OBJ 0.06	ROOT 0.06	I 0.00	suj 0.08
Total	0.58	0.62	0.59	0.61	0.70	1.00	0.59

Table 3: Unigram probability for the five most frequent DEPREL labels in the training data of the CoNLL-2009 Shared Task is shown. Total is the probability mass covered by the five dependency labels shown.

APRED	Catalan	Chinese	Czech	English	German	Japanese	Spanish
Labels	arg1-pat 0.22	A1 0.30	RSTR 0.30	A1 0.37	A0 0.40	GA 0.33	arg1-pat 0.20
	arg0-agt 0.18	A0 0.27	PAT 0.18	A0 0.25	A1 0.39	WO 0.15	arg0-agt 0.19
	arg1-tem 0.15	ADV 0.20	ACT 0.17	A2 0.12	A2 0.12	NO 0.15	arg1-tem 0.15
	argM-tmp 0.08	TMP 0.07	APP 0.06	AM-TMP 0.06	A3 0.06	NI 0.09	arg2-atr 0.08
	arg2-atr 0.08	DIS 0.04	LOC 0.04	AM-MNR 0.03	A4 0.01	DE 0.06	argM-tmp 0.08
Total	0.71	0.91	0.75	0.83	0.97	0.78	0.70
Avg.	2.25	2.26	0.88	2.20	1.97	1.71	2.26

Table 4: Unigram probability for the five most frequent APRED labels in the training data of the CoNLL-2009 Shared Task is shown. Total is the probability mass covered by the five argument labels shown. The "Avg." line shows the average number of arguments per predicate or other argument-bearing token (i.e. for those marked by FILLPRED='Y').

AnCora has 280 different labels. Considering only the main syntactic categories, the tag set is reduced to 47 tags. The syntactic tag set consists of 50 different syntactic functions. Regarding semantic arguments, we distinguish Arg0, Arg1, Arg2, Arg3, Arg4, ArgM, and ArgL. The first five tags are numbered from less to more obliqueness with respect to the verb, ArgM corresponds to adjuncts. The list of thematic roles consists of 20 different labels: AGT (Agent), AGI (Induced Agent), CAU (Cause), EXP (Experiencer), SCR (Source), PAT (Patient), TEM (Theme), ATR (Attribute), BEN (Beneficiary), EXT (Extension), INS (Instrument), LOC (Locative), TMP (Time), MNR (Manner), ORI (Origin), DES (Goal), FIN (Purpose), EIN (Initial State), EFI (Final State), and ADV (Adverbial). Each argument position can map onto specific thematic roles. By way of example, Arg1 can be PAT, TEM or EXT. For Named Entities, we distinguish six types: Organization, Person, Location, Date, Number, and Others.

An incremental process guided the annotation of AnCora, since semantics depends on morphosyntax, and syntax relies on morphology. This procedure made it possible to check, correct, and complete the previous annotations, thus guaranteeing the final quality of the corpora and minimizing the error rate. The annotation process was carried out sequentially from lower to upper layers of linguistic description. All resulting layers are independent of each other, thus making easier the data management. The initial annotation was performed manually for syntax, semiautomatically in the case of arguments and thematic roles, and fully automatically for PoS (Martí et al., 2007; Màrquez et al., 2007).

The Catalan and Spanish AnCora corpora were straightforwardly translated into the CoNLL-2009 shared task formatting (information about named entities was skipped in this process). The resulting Catalan corpus (including training, development and test partitions) contains 16,786 sentences with an average length of 29.59 lexical tokens per sentence. Long sentences abound in this corpus. For instance, 10.73% of the sentences are longer than 50 tokens, and 4.42% are longer than 60. The corpus contains 47,537 annotated predicates (2.83 predicates per sentence, on average) with 107,171 arguments (2.25 arguments per predicate, on average). From the latter, 73.89% correspond to core arguments and 26.11% to adjuncts. Numbers for the Spanish corpus are comparable in all aspects: 17,709 sentences with 29.84 lexical tokens on average (11.58% of the sentences longer than 50 tokens, 4.07% longer than 60); 54,075 predicates (3.05 per sentence, on average) and 122,478 arguments (2.26 per predicate, on average); 73.34% core arguments and 26.66% adjuncts.

The following are important features of the Catalan and Spanish corpora in the CoNLL-2009 shared task setting: (1) all dependency trees are projective; (2) no word can be the argument of more than one predicate in a sentence; (3) semantic dependencies completely match syntactic dependency structures (i.e., no new edges are introduced by the semantic structure); (4) only verbal predicates are annotated (with exceptional cases referring to words that can be adjectives and past participles); (5) the corpus is segmented so multi-words, named entities, temporal expressions, compounds, etc. are grouped together; and (6) segmentation also accounts for elliptical pronouns (there are marked as empty lexical tokens '_' with a pronoun POS tag).

Finally, the predicted columns (PLEMMA, PPOS, and PFEAT) have been generated with the FreeLing Open source suite of Language Analyzers[10]. Accuracy in PLEMMA and PPOS columns is above 95% for the two languages. PHEAD and PDEPREL columns have been generated using MaltParser[11]. Parsing accuracy (LAS) is above 86% for the the two languages.

3.3 Chinese

The Chinese Corpus for the 2009 CoNLL Shared Task was generated by merging the Chinese Treebank (Xue et al., 2005) and the Chinese Proposition Bank (Xue and Palmer, 2009) and then converting the constituent structure to a dependency formalism as specified in the CoNLL Shared Task. The Chinese data used in the shared task is based on Chinese Treebank 6.0 and the Chinese Proposition Bank 2.0, both of which are publicly available via the Linguistic Data Consortium.

The Chinese Treebank Project originated at Penn and was later moved to University of Colorado at

[10] `http://www.lsi.upc.es/~nlp/freeling`
[11] `http://w3.msi.vxu.se/~jha/maltparser`

Boulder. Now it is the process of being to moved to Brandeis University. The data sources of the Chinese Treebank range from Xinhua newswire (mainland China), Hong Kong news, and Sinorama Magazine (Taiwan). More recently under DARPA GALE funding it has been expanded to include broadcast news, broadcast conversation, news groups and web log data. It currently has over one million words and is fully segmented, POS-tagged and annotated with phrase structure. The version of the Chinese Treebank used in this shared task, CTB 6.0, includes newswire, magazine articles, and transcribed broadcast news [12]. The training set has 609,060 tokens, the development set has 49,620 tokens, and the test set has 73,153 tokens.

The Chinese Proposition Bank adds a layer of semantic annotation to the syntactic parses in the Chinese Treebank. This layer of semantic annotation mainly deals with the predicate-argument structure of Chinese verbs and their nominalizations. Each major sense (called *frameset*) of a predicate takes a number of *core* arguments annotated with numerical labels *Arg0* through *Arg5* which are defined in a predicate-specific manner. The Chinese Proposition Bank also annotates adjunctive arguments such as locative, temporal and manner modifiers of the predicate. The version of the Chinese Propbank used in this CoNLL Shared Task is CPB 2.0, but nominal predicates are excluded because the annotation is incomplete.

Since the Chinese Treebank is annotated with constituent structures, the conversion and merging procedure converts the constituent structures to dependencies by identifying the head for each constituent in a parse tree and making its sisters its dependents. The Chinese Propbank pointers are then shifted from the entire constituent to the head of that constituent. The conversion procedure identifies the head by first exploiting the structural information in the syntactic parse and detecting six broad categories of syntactic relations that hold between the head and its dependents (*predication*, *modification*, *complementation*, *coordination*, *auxiliary*, and *flat*) and then designating the head based on these relations. In particular, the first conjunct of a coordination structure is designated as the head and the heads of the other conjuncts are the conjunctions preceding them. The conjunctions all "modify" the first conjunct.

[12] A small number of files were taken out of the CoNLL shared task data due to conversion problems and time constraints to fix them.

3.4 Czech

For the training, development and evaluation data, Prague Dependency Treebank 2.0 was used (Hajič et al., 2006). For the out-of-domain evaluation data, part of the Czech side of the Prague Czech-English Dependency Treebank (version 2, under construction) was used[13], see also (Čmejrek et al., 2004). For the OOD data, no manual annotation of LEMMA, POS, and FEAT existed, so the predicted values were used. The same conversion procedure has been applied to both sources.

The FORM column was created from the `form` element of the morphological layer, not from the "token" from the word-form layer. Therefore, most typos, errors in word segmentation and tokenization are corrected and numerals are normalized.

The LEMMA column was created from the `lemma` element of the morphological layer. Only the initial string of the element was used, so there is no distinction between homonyms. However, some components of the detailed lemma explanation were incorporated into the FEAT column (see below).

The POS column was created form the morphological `tag` element, its first character more precisely.

The FEAT column was created from the remaining characters of the `tag` element. In addition, the special feature "Sem" corresponds to a semantic feature of the lemma.

For the HEAD and DEPREL columns, the PDT analytical layer was used. The DEPREL was taken from the analytic function (the `afun` node attribtue). There are 27 possible values for `afun` element: `Pred`, `Pnom`, `AuxV`, `Sb`, `Obj`, `Atr`, `Adv`, `Atv`, `AtvV`, `Coord`, `Apos`, `ExD`, and a number of auxiliary and "double-function" labels. The first nine of these are the "most interesting" from the point of view of the shared task, since they relate to semantics more closely than the rest (at least from the linguistic point of view). The HEAD is a pointer to its parent, which means the PDT's `ord` attribute

[13] `http://ufal.mff.cuni.cz/pedt`

(within-sentence ID / word position number) of the parent. If a node is a member of a coordination or apposition (`is_member` element), its DEPREL obtains the `_M` suffix. The parenthesis annotation (`is_parenthesis_root` element) was ignored.

The PRED and APREDs columns were created from the tectogrammatical layer of PDT 2.0 and the valency lexicon PDT-Vallex according to the following rules:

- Every line corresponding to an analytical node referenced by a lexical reference (`a/lex.rf`) from the tectogrammatical layer has a PRED value filled. If the referring non-generated tectogrammatical node (`is_generated` not equal to 1) has a valency frame assigned (`val_frame.rf`), the value of PRED is the identifier of the frame. Otherwise, it is set to the same value as the LEMMA column.
- For every tectogrammatical node, a corresponding analytical node is searched for:
  1. If the tectogrammatical node is not generated and has a lexical reference (`a/lex.rf`), the referenced node is taken.
  2. Otherwise, if the tectogrammatical node has a coreference (`coref_text.rf` or `coref_gram.rf`) or complement reference (`compl.rf`) to a node that has an analytical node assigned (by 1. or 2.), the assigned node is taken.

APRED columns are filled with respect to the following correspondence: for a tectogrammatical node P and its effective child C with functor F, the column for P's corresponding analytical node at the row for C's corresponding analytical node is filled with F. Some nodes can thus have several functors in one APRED column, separated by a vertical bar (see Sect. 2.4.2).

PLEMMA, PPOS and PFEAT were generated by the (cross-trained) morphological tagger MORCE (Spoustová et al., 2009), which gives full combined accuracy (PLEMMA+PPOS+PFEAT) slightly under 96%.

PHEAD and PDEPREL were generated by the (cross-trained) MST parser for Czech (Chu–Liu/Edmonds algorithm, (McDonald et al., 2005)), which has typical dependency accuracy around 85%.

The valency lexicon, converted from (Hajič et al., 2003), has four columns:

1. lemma (can occur several times in the lexicon, with different frames)
2. frame identifier (as found in the PRED column)
3. list of space-separated actants and obligatory members of the frame
4. example(s)

The source of the out-of-domain data uses an extended valency lexicon (because of out-of-vocabulary entries). For simplicity, the extended lexicon was not provided; instead, such words were not marked as predicates in the OOD data (their FILLPRED was set to '_') and thus not evaluated.

3.5 English

The English corpus is almost identical to the corpus used in the closed challenge in the CoNLL-2008 shared task evaluation (Surdeanu et al., 2008). This corpus was generated through a process that merges several input corpora and converts them from the constituent-based formalism to dependencies. The following corpora were used as input to the merging procedure:

- **Penn Treebank 3** – The Penn Treebank 3 corpus (Marcus et al., 1994) consists of hand-coded parses of the Wall Street Journal (test, development and training) and a small subset of the Brown corpus (W. N. Francis and H. Kucera, 1964) (test only).
- **BBN Pronoun Coreference and Entity Type Corpus** – BBN's NE annotation of the Wall Street Journal corpus (Weischedel and Brunstein, 2005) takes the form of SGML inline markup of text, tokenized to be completely compatible with the Penn Treebank annotation. For the CoNLL-2008 shared task evaluation, this corpus was extended by the task organizers to cover the subset of the Brown corpus used as a secondary testing dataset. From this corpus we only used NE boundaries to derive NAME

dependencies between NE tokens, e.g., we create a NAME dependency from *Mary* to *Smith* given the NE mention *Mary Smith*.

- **Proposition Bank I (PropBank)** – The PropBank annotation (Palmer et al., 2005) classifies the arguments of all the main verbs in the Penn Treebank corpus, other than *be*. Arguments are numbered (Arg0, Arg1, ...) based on lexical entries or frame files. Different sets of arguments are assumed for different rolesets. Dependent constituents that fall into categories independent of the lexical entries are classified as various types of adjuncts (ArgM-TMP, -ADV, etc.).

- **NomBank** – NomBank annotation (Meyers et al., 2004) uses essentially the same framework as PropBank to annotate arguments of nouns. Differences between PropBank and NomBank stem from differences between noun and verb argument structure; differences in treatment of nouns and verbs in the Penn Treebank; and differences in the sophistication of previous research about noun and verb argument structure. Only the subset of nouns that take arguments are annotated in NomBank and only a subset of the non-argument siblings of nouns are marked as ArgM.

The complete merging process and the conversion from the constituent representation to dependencies is detailed in (Surdeanu et al., 2008).

The main difference between the 2008 and 2009 version of the corpora is the generation of word lemmas. In the 2008 version the only lemmas provided were predicted using the built-in lemmatizer in WordNet (Fellbaum, 1998) based on the most frequent sense for the form and the predicted part-of-speech tag. These lemmas are listed in the 2009 corpus under the PLEMMA column. The LEMMA column in the 2009 version of the corpus contains lemmas generated using the same algorithm but using the correct Treebank part-of-speech tags. Additionally, the PHEAD and PDEPREL columns were generated using MaltParser[14], similarly to the open challenge corpus in the CoNLL 2008 shared task.

[14] `http://w3.msi.vxu.se/~nivre/research/MaltParser.html`

3.6 German

The German in-domain dataset is based on the annotated verb instances of the SALSA corpus (Burchardt et al., 2006), a total of around 40k sentences[15]. SALSA provides manual semantic role annotation on top of the syntactically annotated TIGER newspaper corpus, one of the standard German treebanks. The original SALSA corpus uses semantic roles in the FrameNet paradigm. We constructed mappings between FrameNet frame elements and PropBank argument positions at the level of frame-predicate pairs semi-automatically. For the frame elements of each frame-predicate pair, we first identified the semantically defined PropBank Arg-0 and Arg-1 positions. To do so, we annotated a small number of very abstract frame elements with these labels (Agent, Actor, Communicator as Arg-0, and Theme, Effect, Message as Arg-1) and percolated these labels through the FrameNet hierarchy, adding further manual labels where necessary. Then, we used frequency and grammatical realization information to map the remaining roles onto higher-numbered Arg roles. We considerably simplified the annotations provided by SALSA, which use a rather complex annotation scheme. In particular, we removed annotation for multi-word expressions (which may be non-contiguous), annotations involving multiple frames for the same predicate (metaphors, underspecification), and inter-sentence roles.

The out-of-domain dataset was taken from a study on the multi-lingual projection of FrameNet annotation (Pado and Lapata, 2005). It is sampled from the EUROPARL corpus and was chosen to maximize the lexical coverage, i.e., it contains of a large number of infrequent predicates. Both syntactic and semantic structure were annotated manually, in the TIGER and SALSA format, respectively. Since it uses a simplified annotation schemes, we did not have to discard any annotation.

For both datasets, we converted the syntactic TIGER (Brants et al., 2002) representations into dependencies with a similar set of head-finding rules used for the preparation of the CoNLL-X shared task German dataset. Minor modifications (for the con-

[15] Note, however, that typically not all predicates in each sentence are annotated (cf. Table 2).

version of person names and coordinations) were made to achieve better consistency with datasets of other languages. Since the TIGER annotation allows non-contiguous constituents, the resulting dependencies can be non-projective. Secondary edges were discarded in the conversion. As for the automatically constructed features, we used TreeTagger (Schmid, 1994) to produce the PLEMMA and PPOS columns, and the Morphisto morphology (Zielinski and Simon, 2008) for PFEAT.

3.7 Japanese

For Japanese, we used the Kyoto University Text Corpus (Kawahara et al., 2002), which consists of approximately 40k sentences taken from *Mainichi* Newspapers. Out of them, approximately 5k sentences are annotated with syntactic and semantic dependencies, and are used the training, development and test data of this year's shared task. The remaining sentences, which are annotated with only syntactic dependencies, are provided for the training corpus of syntactic dependency parsers.

This corpus adopts a dependency structure representation, and thus the conversion to the CoNLL-2009 format was relatively straightforward. However, since the original dependencies are annotated on the basis of phrases (Japanese *bunsetsu*), we needed to automatically convert the original annotations to word-based ones using several criteria. We used the following basic criteria: the words except the last word in a phrase depend on the next (right) word, and the last word in a phrase basically depends on the head word of the governing phrase.

Semantic dependencies are annotated for both verbal predicates and nominal predicates. The semantic roles (APRED columns) consist of 41 surface cases, many of which are case-marking postpositions such as *ga* (nominative), *wo* (accusative) and *ni* (dative). Semantic frame discrimination is not annotated, and so the PRED column is the same as the LEMMA column. The original corpus contains coreference annotations and inter-sentential semantic dependencies, such as inter-sentential zero pronouns and bridging references, but we did not use these annotations, which are not the target of this year's shared task.

To produce the PLEMMA, PPOS and PFEAT columns, we used the morphological analyzer JUMAN [16] and the dependency and case structure analyzer KNP [17]. To produce the PHEAD and PDEPREL columns, we used the MSTParser [18].

4 Submissions and Results

Participants uploaded the results through the shared task website, and the official evaluation was performed centrally. Feedback was provided if any formal problems were encountered (for a list of checks, see the previous section). One submission had to be rejected because only English results were provided. After the evaluation period had passed, the results were anonymized and published on the web.

A total of 20 systems participated in the closed challenge; 13 of them in the Joint task and seven in the SRL-only task. Two systems participated in the open challenge (Joint task). Moreover, 17 systems provided output in the out-of-domain part of the task (11 in the OOD Joint task and six in the OOD SRL-only task).

The main results for the core task - the Joint task (dependency syntax *and* semantic relations) in the context of the closed challenge - are summarized and ranked in Table 5.

The largest number of systems can be compared in the SRL results table (Table 6), where all the systems have been evaluated solely on the SRL performance regardless whether they participated in the Joint or SRL-only task. However, since the results might have been influenced by the supplied parser, separate ranking is provided for both types of the systems.

Additional breakdown of the results (open challenge, precision and recall tables for the semantic labeling task, etc.) are available from the CoNLL-2009 Shared Task website[19].

5 Approaches

Table 7 summarizes the properties of the systems that participated in the closed the open challenges.

[16] http://nlp.kuee.kyoto-u.ac.jp/nl-resource/juman-e.html

[17] http://nlp.kuee.kyoto-u.ac.jp/nl-resource/knp-e.html

[18] http://sourceforge.net/projects/mstparser

[19] http://ufal.mff.cuni.cz/conll2009-st

Rank	System	Average	Catalan	Chinese	Czech	English	German	Japanese	Spanish
1	Che	82.64	81.84	76.38	**83.27**	87.00	**82.44**	**85.65**	81.90
2	Chen	82.52	**83.01**	76.23	80.87	**87.69**	81.22	85.28	**83.31**
3	Merlo	82.14	82.66	76.15	83.21	86.03	79.59	84.91	82.43
4	Bohnet	80.85	80.44	75.91	79.57	85.14	81.60	82.51	80.75
5	Asahara	78.43	75.91	73.43	81.43	86.40	69.84	84.86	77.12
6	Brown	77.27	77.40	72.12	75.66	83.98	77.86	76.65	77.21
7	Zhang	76.49	75.00	73.42	76.93	82.88	73.76	78.17	75.25
8	Dai	73.98	72.09	72.72	67.14	81.89	75.00	80.89	68.14
9	Lu Li	73.97	71.32	65.53	75.85	81.92	70.93	80.49	71.72
10	Lluís	71.49	56.64	66.18	75.95	81.69	72.31	81.76	65.91
11	Vallejo	70.81	73.75	67.16	60.50	78.19	67.51	77.75	70.78
12	Ren	67.81	59.42	75.90	60.18	77.83	65.77	77.63	57.96
13	Zeman	51.07	49.61	43.50	57.95	50.27	49.57	57.69	48.90

Table 5: Official results of the Joint task, closed challenge. Teams are denoted by the last name (first name added only where needed) of the author who registered for the evaluation data. Results are sorted in descending order of the language-averaged macro F_1 score on the closed challenge Joint task. Bold numbers denote the best result for a given language.

Rank	Rank in task	System	Average	Catalan	Chinese	Czech	English	German	Japanese	Spanish
1	1 (SRLonly)	Zhao	80.47	**80.32**	77.72	85.19	85.44	75.99	78.15	**80.46**
2	2 (SRLonly)	Nugues	80.31	80.01	**78.60**	85.41	85.63	**79.71**	76.30	76.52
3	1 (Joint)	Chen	79.96	80.10	76.77	82.04	**86.15**	76.19	78.17	80.29
4	2 (Joint)	Che	79.94	77.10	77.15	**86.51**	85.51	78.61	**78.26**	76.47
5	3 (Joint)	Merlo	78.42	77.44	76.05	86.02	83.24	71.78	77.23	77.19
6	3 (SRLonly)	Meza-Ruiz	77.46	78.00	77.73	75.75	83.34	73.52	76.00	77.91
7	4 (Joint)	Bohnet	76.00	74.53	75.29	79.02	80.39	75.72	72.76	74.31
8	5 (Joint)	Asahara	75.65	72.35	74.17	84.69	84.26	63.66	77.93	72.50
9	6 (Joint)	Brown	72.85	72.18	72.43	78.02	80.43	73.40	61.57	71.95
10	7 (Joint)	Dai	70.78	66.34	71.57	75.50	78.93	67.43	71.02	64.64
11	8 (Joint)	Zhang	70.31	67.34	73.20	78.28	77.85	62.95	64.71	67.81
12	9 (Joint)	Lu Li	69.72	66.95	67.06	79.08	77.17	61.98	69.58	66.23
13	4 (SRLonly)	Baoli Li	69.26	74.06	70.37	57.46	69.63	67.76	72.03	73.54
14	10 (Joint)	Vallejo	68.95	70.14	66.71	71.49	75.97	61.01	68.82	68.48
15	5 (SRLonly)	Moreau	66.49	65.60	67.37	71.74	72.14	66.50	57.75	64.33
16	11 (Joint)	Lluís	63.06	46.79	59.72	76.90	75.86	62.66	71.60	47.88
17	6 (SRLonly)	Täckström	61.27	57.11	63.41	71.05	67.64	53.42	54.74	61.51
18	7 (SRLonly)	Lin	57.18	61.70	70.33	60.43	65.66	59.51	23.78	58.87
19	12 (Joint)	Ren	56.69	41.00	72.58	62.82	67.56	54.31	58.73	39.80
20	13 (Joint)	Zeman	32.14	24.19	34.71	58.13	36.05	16.44	30.13	25.36

Table 6: Official results of the semantic labeling, closed challenge, all systems. Teams are denoted by the last name (first name added only where needed) of the author who registered for the evaluation data. Results are sorted in descending order of the semantic labeled F_1 score (closed challenge). Bold numbers denote the best result for a given language. Separate ranking is provided for SRL-only systems.

The second column of the table highlights the overall architectures. We used + to indicate that the components are sequentially connected. The lack of a + sign indicates that the corresponding tasks are performed jointly.

It is perhaps not surprising that most of the observations from the 2008 shared task still hold; namely, the best systems overall do not use joint learning or optimization (the best such system was placed third in the Joint task, and there were only four systems where the learning methodology can be considered "joint").

Therefore, most of the observations and conclusions from 2008 shared task hold as well for the current results. For details, we will leave it to the reader to interpret the architectures and methods

System[a]	Overall Arch.[b]	D Arch.	D Comb.	D Inference[c]	PA Arch.	PA Comb.	PA Inference	Joint Learning/Opt.	ML Methods
Zhao	PAIC	(SRL-only)	(SRL-only)	(SRL-only)	class	no	greedy/global search	(SRL-only)	ME
Nugues	(PC+AI+AC) + AIC	(SRL-only)	(SRL-only)	(SRL-only)	class	no	beam search + reranking	(SRL-only)	L2-regularized lin. regression
Chen	P + PC + AI + AC	graph	partially	$MST^{CL/E}$	class	no	greedy (?)	no	ME
Che	D+PC+AIC	graph	no	MST^{HOE}	class	no	ILP	no	SVM, ME
Merlo	DPAIC+D	generative, trans	no	beam search	trans	no	beam search	synchronized derivation	ISBN
Meza-Ruiz	PAIC	(SRL-only)	(SRL-only)	(SRL-only)	Markov LN	no	Cutting Plane	(SRL-only)	MIRA
Bohnet	D + AI + AC + PC	graph	no	MST^{C}+rearrange	class	no	greedy	no	SVM (MIRA)
Asahara	D + PIC + AIC	graph	no	MST^{C}	class	no	n-best relax.	no	perceptron
Dai	D + PC + AC	graph	no	MST^{C}	class	no	prob	iterative	ME
Zhang	D + AI + AC + PC	graph	no	MST^{E}	class	no	classification	no	MIRA, ME
Lu Li	D + (PC || AIC)	graph	for each lang.	$MST^{CL/E}$,MST^{E}	class	no	greedy	no	ME
Baoli Li	PC + AIC	(SRL-only)	(SRL-only)	(SRL-only)	class	no	greedy	(SRL-only)	SVM, kNN, ME
Vallejo[d]	[D+P+A]C + DI	class	no	reranking	class	no	reranking	unified labels	MBL
Moreau	D + PI + Clustering + AI + AC	(SRL-only)	(SRL-only)	(SRL-only)	class	no	CRF	(SRL-only)	CRF
Lluís	D+DAIC+PC	graph	no	MST^{E}	graph	no	MST^{E}	yes, MST^{E}	Avg. Perceptron
Täckström	D + PI + AI + AC + Constraint Satisfaction	(SRL-only)	(SRL-only)	(SRL-only)	class	no	greedy	(SRL-only)	SVM
Ren	D + PC + AIC	trans	no	greedy	class	no	greedy	no	SVM (Malt), ME
Zeman	DI+DC+PC+AI+AC	trans	no	greedy with heuristics	class	no	greedy	no	cooccurrence

Table 7: Summary of system architectures for the CoNLL-2009 shared task; all systems are included. SRL-only systems do not have the D columns and the Joint Learing/Opt. columns filled in. The systems are sorted by the semantic labeled F_1 score averaged over all the languages (same as in Table 6). Only the systems that have a corresponding paper in the proceedings are included. Acronyms used: **D** - syntactic dependencies, **P** - predicate, **A** - argument, I - identification, **C** - classification. **Overall arch.** stands for the complete system architecture; **D Arch.** stands for the architecture of the syntactic parser; **D Comb.** indicates if the final parser output was generated using parser combination; **D Inference** stands for the type of inference used for syntactic parsing; **PA Arch.** stands the type of architecture used for PAIC; **PA Comb.** indicates if the PA output was generated through system combination; **PA Inference** stands for the the type of inference used for PAIC; **Joint Learning/Opt.** indicates if some form of joint learning or optimization was implemented for the syntactic + semantic global task; **ML Methods** lists the ML methods used throughout the complete system.

[a] Authors of two systems: "Brown" and "Lin" didn't submit a paper, so their systems' architectures are unknown.

[b] The symbol + indicates sequential processing (otherwise, parallel/joint). The || means that several different architectures spanning multiple subtasks ran in parallel.

[c] $MST^{CL/E}$ as used by McDonald (2005), MST^{C} by Carreras (2007), MST^{E} by Eisner (2000), MST^{HOE} = MST^{E} with higher-order features (siblings + all grandchildren).

[d] The system unifies the syntactic and semantic labels into one label, and trains classifiers over them. It is thus difficult to split the system characteristic into a "D"/"PA" part.

when comparing Table 7 with the Tables 5 and 6).

6 Conclusion

This year's task has been demanding in several respects, but certainly the most difficulty came from the fact that participants had to tackle all seven languages. It is encouraging that despite this added affort the number of participating systems has been almost the same as last year (20 vs. 22 in 2008).

There are several positive outcomes from this year's enterprise:

- we have prepared a unified format and data for several very different lanaguages, as a basis for possible extensions towards other languages and unified treatment of syntactic depenndecies and semantic role labeling across natural languages;
- 20 participants have produced SRL results for all seven languages, using several different methods, giving hope for a combined system with even substantially better performance;
- initial results have been provided for three languages on out-of-domain data (being in fact quite close to the in-domain results).

Only four systems tried to apply what can be described as joint learning for the syntactic and semantic parts of the task. (Morante et al., 2009) use a true joint learning formulation that phrases syntactico-semantic parsing as a series of classification where the class labels are concatenations of syntactic and semantic edge labels. They predict (a), the set of syntactico-semantic edge labels for each pair of tokens; (b), the set of incoming syntactico-semantic edge labels for each individual token; and (c), the existence of an edge between each pair of tokens. Subsequently, they combine the (possibly conflicting) output of the three classifiers by a ranking approach to determine the most likely structure that meets all well-formedness constraints. (Lluís et al., 2009) present a joint approach based on an extension of Eisner's parser to accommodate also semantic dependency labels. This architecture is similar to the one presented by the same authors in the past edition, with the extension to a second-order syntactic parsing and a particular setting for Catalan and Spanish. (Gesmundo et al., 2009) use an incremental parsing model with synchronous syntactic and semantic derivations and a joint probability model for syntactic and semantic dependency structures. The system uses a single input queue but two separate stacks and synchronizes syntactic and semantic derivations at every word. The synchronous derivations are modeled with an Incremental Sigmoid Belief Network that has latent variables for both syntactic and semantic states and connections from syntax to semantics and vice versa. (Dai et al., 2009) designed an iterative system to exploit the inter-connections between the different subtasks of the CoNLL shared task. The idea is to decompose the joint learning problem into four subtasks – syntactic dependency identification, syntactic dependency labeling, semantic dependency identification and semantic dependency labeling. The initial step is to use a pipeline approach to use the input of one subtask as input to the next, in the order specified. The iterative steps then use additional features that are not available in the initial step to improve the accuracy of the overall system. For example, in the iterative steps, semantic information becomes available as features to syntactic parsing, so on and so forth.

Despite these results, it is still not clear whether joint learning has a significant advantage over other approaches (and if yes, then for what languages). It is thus necessary to carefully plan the next shared tasks; it might be advantageous to bring up a similar task in the future once again, and/or couple it with selected application(s). There, (we hope) the benefits of the dependency representation combined with semantic roles the way we have formulated it in 2008 and 2009 will really show up.

Acknowledgments

We would like to thank the Linguistic Data Consortium, mainly to Denise DiPersio, Tony Casteletto and Christopher Cieri for their help and handling of invoicing and distribution of the data for which LDC has a license. For all of the trial, training and evaluation data they had to act a very short notice. All the data has been at the participants' disposal (again) free of charge. We are grateful to all of them for LDC's continuing support of the CoNLL Shared

Tasks.

We would also like to thank organizers of the previous four shared tasks: Sabine Buchholz, Xavier Carreras, Ryan McDonald, Amit Dubey, Johan Hall, Yuval Krymolowski, Sandra Kübler, Erwin Marsi, Jens Nilsson, Sebastian Riedel and Deniz Yuret. This shared task would not have been possible without their previous effort.

We also acknowledge the support of the MŠMT of the Czech Republic, projects MSM0021620838 and LC536; the Grant Agency of the Academy of sciences of the Czech Republic 1ET201120505 (for Jan Hajič, Jan Štěpánek and Pavel Straňák).

Lluís Màrquez and M. Antònia Martí participation was supported by the Spanish Ministry of Education and Science, through the OpenMT and TextMess research projects (TIN2006-15307-C03-02, TIN2006-15265-C06-06).

The following individuals directly contributed to the Chinese Treebank (in alphabetic order): Meiyu Chang, Fu-Dong Chiou, Shizhe Huang, Zixin Jiang, Tony Kroch, Martha Palmer, Mitch Marcus, Fei Xia, Nianwen Xue. The contributors to the Chinese Proposition Bank include (in alphabetic order): Meiyu Chang, Gang Chen, Helen Chen, Zixin Jiang, Martha Palmer, Zhiyi Song, Nianwen Xue, Ping Yu, Hua Zhong. The Chinese Treebank and the Chinese Proposition Bank were funded by DOD, NSF and DARPA.

Adam Meyers' work on the shared task has been supported by the NSF Grant IIS-0534700 "Structure Alignment-based MT."

We thank the Mainichi Newspapers for the permission of distributing the sentences of the Kyoto University Text Corpus for this shared task.

References

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The TIGER treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol.

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of EMNLP-CoNLL 2007*, pages 957–961, June. Prague, Czech Republic.

Montserrat Civit, M. Antònia Martí, and Núria Bufí. 2006. Cat3LB and Cast3LB: from constituents to dependencies. In *Proceedings of the 5th International Conference on Natural Language Processing, FinTAL*, pages 141–153, Turku, Finland. Springer Verlag, LNAI 4139.

Qifeng Dai, Enhong Chen, and Liu Shi. 2009. An iterative approach for joint dependency parsing and semantic role labeling. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA. June 4-5.

Jason Eisner. 2000. Bilexical grammars and their cubic-time parsing algorithms. In Harry Bunt and Anton Nijholt, editors, *Advances in Probabilistic and Other Parsing Tehcnologies*, pages 29–62. Kluwer Academic Publishers.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge.

Andrea Gesmundo, James Henderson, Paola Merlo, and Ivan Titov. 2009. A latent variable model of synchronous syntactic-semantic parsing for multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA. June 4-5.

Jan Hajič, Jarmila Panevová, Zdeňka Urešová, Alevtina Bémová, Veronika Kolářová-Řezníčková‘, and Petr Pajas. 2003. PDT-VALLEX: Creating a Large-coverage Valency Lexicon for Treebank Annotation. In J. Nivre and E. Hinrichs, editors, *Proceedings of The Second Workshop on Treebanks and Linguistic Theories*, pages 57–68, Vaxjo, Sweden. Vaxjo University Press.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

Xavier Lluís, Stefan Bott, and Lluís Màrquez. 2009. A second-order joint eisner model for syntactic and semantic dependency parsing. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA. June 4-5.

M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1994. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, 19(2):313–330.

Lluís Màrquez, Luis Villarejo, M. Antònia Martí, and Mariona Taulé. 2007. SemEval-2007 Task 09: Multilevel semantic annotation of catalan and spanish. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-2007)*, pages 42–47, Prague, Czech Republic.

M. Antònia Martí, Mariona Taulé, Lluís Màrquez, and Manu Bertran. 2007. Anotación semiautomática con papeles temáticos de los corpus CESS-ECE. *Procesamiento del Lenguaje Natural, SEPLN Journal*, 38:67–76.

Ryan McDonald, Fernando Pereira, Jan Hajič, and Kiril Ribarov. 2005. Non-projective dependency parsing using spanning tree algortihms. In *Proceedings of NAACL-HLT'05, Vancouver, Canada*, pages 523–530.

A. Meyers, R. Reeves, C. Macleod, R. Szekely, V. Zielinska, B. Young, and R. Grishman. 2004. The NomBank Project: An Interim Report. In *NAACL/HLT 2004 Workshop Frontiers in Corpus Annotation*, Boston.

Roser Morante, Vincent Van Asch, and Antal van den Bosch. 2009. A simple generative pipeline approach to dependency parsing and semantic role labeling. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009)*, Boulder, Colorado, USA. June 4-5.

Joakim Nivre, Johann Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The conll 2007 shared task on dependency parsing. In *Proceedings of the EMNLP-CoNLL 2007 Conference*, pages 915–932, Prague, Czech Republic.

Sebastian Pado and Mirella Lapata. 2005. Cross-lingual projection of role-semantic information. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP-2005)*, pages 859–866, Vancouver, BC.

Petr Pajas and Jan Štěpánek. 2008. Recent advances in a feature-rich framework for treebank annotation. In *The 22nd International Conference on Computational Linguistics - Proceedings of the Conference (COLING'08)*, pages 673–680, Manchester.

M. Palmer, D. Gildea, and P. Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*.

Drahomíra "Johanka" Spoustová, Jan Hajič, Jan Raab, and Miroslav Spousta. 2009. Semi-supervised training for the averaged perceptron POS tagger. In *Proceedings of the European ACL Cenference EACL'09*, Athens, Greece.

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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*, pages 159–177.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco.

Martin Čmejrek, Jan Cuřín, Jan Hajič, Jiří Havelka, and Vladislav Kuboň. 2004. Prague Czech-English Dependency Treebank: Syntactically Anntoated Resources for Machine Translation. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC-2004)*, pages 1597–1600, Lisbon, Portugal.

W. N. Francis and H. Kucera. 1964. Brown Corpus Manual of Information to accompany A Standard Corpus of Present-Day Edited American English, for use with Digital Computers. Revised 1971, Revised and Amplified 1979, available at www.clarinet/brown.

R. Weischedel and A. Brunstein. 2005. BBN pronoun coreference and entity type corpus. Technical report, Lin- guistic Data Consortium.

Nianwen Xue and Martha Palmer. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

Nianwen Xue, Fei Xia, Fu Dong Chiou, and Martha Palmer. 2005. The Penn Chinese TreeBank: Phrase Structure Annotation of a Large Corpus. *Natural Language Engineering*, 11(2):207–238.

Andrea Zielinski and Christian Simon. 2008. Morphisto: An open-source morphological analyzer for german. In *Proceedings of the Conference on Finite State Methods in Natural Language Processing*.

An Iterative Approach for Joint Dependency Parsing and Semantic Role Labeling

Qifeng Dai
Department of Computer Science, University of Science and Technology of China, Hefei, China
daiqifeng001@126.com

Enhong Chen
Department of Computer Science, University of Science and Technology of China, Hefei, China
cheneh@ustc.edu.cn

Liu Shi
Department of Computer Science, University of Science and Technology of China, Hefei, China
shiliu@ustc.edu

Abstract

We propose a system to carry out the joint parsing of syntactic and semantic dependencies in multiple languages for our participation in the shared task of CoNLL-2009. We present an iterative approach for dependency parsing and semantic role labeling. We have participated in the closed challenge, and our system achieves 73.98% on labeled macro F1 for the complete problem, 77.11% on labeled attachment score for syntactic dependencies, and 70.78% on labeled F1 for semantic dependencies. The current experimental results show that our method effectively improves system performance.

1 Introduction

In this paper we describe the system submitted to the closed challenge of the CoNLL-2009 shared task on joint parsing of syntactic and semantic dependencies in multiple languages.

Give a sentence, the task of dependency parsing is to identify the syntactic head of each word in the sentence and classify the relation between the dependent and its head. The task of semantic role labeling is to label the senses of predicates in the sentence and labeling the semantic role of each word in the sentence relative to each predicate.

The difficulty of this shared task is to perform joint task on dependency parsing and semantic role labeling. We split the shared task into four subproblems: syntactic dependency parsing, syntactic dependency label classification, word sense disambiguation, and semantic role labeling. And we propose a novel iterative approach to perform the joint task. In the first step, the system performs dependency parsing and semantic role labeling in a pipelined manner and the four sub-problems extract features based on the known information. In the iterative step, the system performs the four tasks in a pipelined manner but uses features extracted from the previous parsing result.

The remainder of the paper is structured as follows. Section 2 presents the technical details of our system. Section 3 presents experimental results and the performance analysis. Section 4 looks into a few issues concerning our forthcoming work for this shared task, and concludes the paper.

2 System description

This section briefly describes the main components of our system: a) system flow; b) syntactic parsing; c) semantic role labeling; d) an iterative approach to perform joint syntactic-semantic parsing.

2.1 System flow

As many systems did in CoNLL Shared Task 2008, the most direct way for such task is pipeline approach. First, Split the system into four subtasks: syntactic dependency parsing, syntactic dependency relation labeling, predicate sense labeling and semantic role labeling. Then, execute them one by one. In our system, we extend this pipeline system to an iterative system so that it can do a joint labeling to improve the performance.

Our iterative system is based on the pipeline system. For the first iteration (original step), we use the pipeline system to parse and label the

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 19–24,
Boulder, Colorado, June 2009.

whole sentence. For the rest iterations (iterative step), we use another pipeline system to parse and label it. The structure of this pipeline is the same as the original one, but each subtask can have much more features than the original subtask. Because the whole sentence has been labeled in the original step, all information is available for every subtask. For example, when doing syntactic dependency relation labeling, we can add some features about sense and semantic role. It seems like using syntactic results to do semantic labeling, then using semantic results to improve syntactic labeling. This is the core idea of our joint system. Figure 1 shows the main flow of our system.

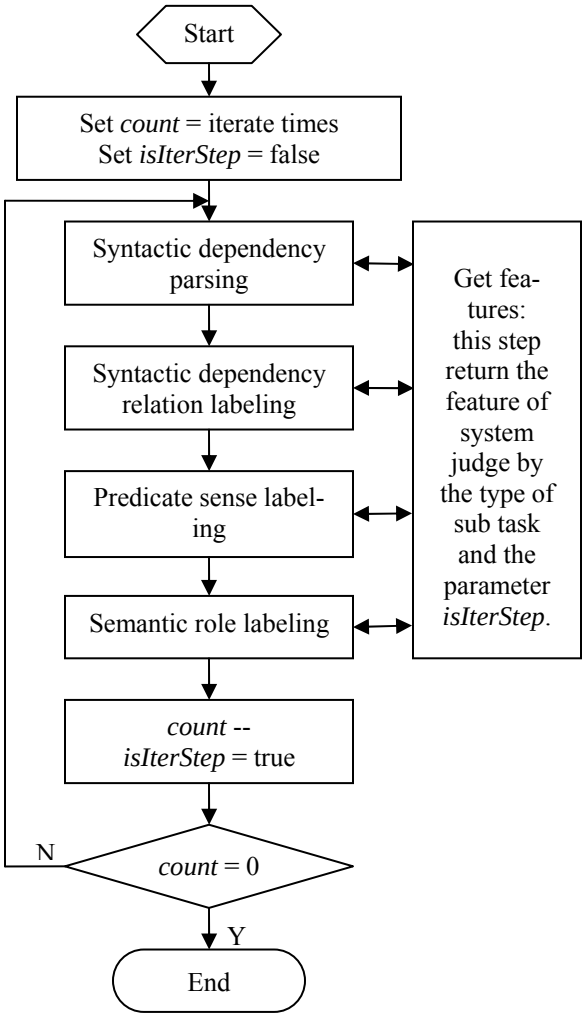

Figure 1. The main flow of iteration system

2.2 Dependency Parsing

In the dependency parsing step, we split the task into two sub-problems: syntactic dependency parsing and syntactic dependency relation labeling.

In the syntactic dependency parsing stage, MSTParser[1], a dependency parser that searches for maximum spanning trees over directed graphs, is applied. Due to the differences between the seven languages, we use different parameters to train a parsing model. Specifically, as Czech and German languages are none-projective and the others are projective, we train Czech and German languages with parameter "none-projective" and the others with "projective".

On the syntactic dependency label classification step, we used the max-entropy classification algorithm to train the model. This step contains two processes. In the first process the sub-problem trains the model with the following basic features:

- FORM1: FORM of the head.
- LEMMA1: LEMMA of the head.
- STEM1 (English only): STEM of the head.
- POS1: POS of the head.
- IS_PRED1: the value of FILLPRED of the head.
- FEAT1: FEAT of the head.
- LM_STEM1 (English only): the left-most modifier's STEM of head.
- LM_POS1: the left-most modifier's POS of head.
- L_NUM1: number of the head's left modifiers.
- RM_STEM1 (English only): the right-most modifier's STEM of head.
- RM_POS1: the right-most modifier's POS of head.
- M_NUM1: number of modifiers of the head.
- SUFFIX1 (English only): suffix of the head.
- FORM2: FORM of the dependent.
- LEMMA2: LEMMA of the dependent.
- STEM2 (English only): STEM of the dependent.
- POS2: POS of the dependent.
- IS_PRED2: the value of FILLPRED of the dependent.

[1] http://sourceforge.net/projects/mstparser

- FEAT2: FEAT of the dependent.
- LM_STEM2 (English only): the left-most modifier's STEM of dependent.
- LM_POS2: the left-most modifier's POS of dependent.
- L_NUM2: number of the dependent's left modifiers.
- RM_STEM2 (English only): the right-most modifier's STEM of dependent.
- RM_POS2: the right-most modifier's POS of dependent.
- M_NUM2: number of modifiers of the dependent.
- SUFFIX2 (English only): suffix of the dependent.
- DEP_PATH_ROOT_POS2: POS list from dependent to tree's root through the syntactic dependency path.
- DEP_PATH_ROOT_LEN2: length from dependent to tree's root through the syntactic dependency path.
- POSITION: The position of the word with respect to its predicate. It has three values, "before", "is" and "after", for the predicate.

In the iterative step, in addition to the features mentioned above, the sub-task trains the model with the following features:

- DEP_PATH_ROOT_POS1: POS list from head to tree's root through the syntactic dependency path.
- DEP_PATH_ROOT_REL1: length from dependent to tree's root through the syntactic dependency path.
- PRED_POS: POS list of all predicates in the sentence.
- FORM2 + DEP_PATH_REL: component of FORM2 and the POS list from head to the dependent through the syntactic dependency path.
- POSITION + FORM2
- STEM1 + FORM2 (English only)
- STEM1 + STEM2 (English only)
- POSITION + POS2
- ROLE_LIST2: list of APRED when the dependent is a predicate.
- ROLE: list of APRED and PRED when the head is predicate.
- L_ROLE: the nearest semantic role in its left side when head is a predicate.
- R_ROLE: the nearest semantic role in its right side when head is a predicate.
- IS_ROLE1: whether dependent is a semantic role of head when head is a predicate.

2.3 Semantic role labeling

Unlike CoNLL-2008 shared task, this shared task does not need to identify predicates. So the main task of this step is to label the sense of each predicate and label the semantic role for each predicate.

When labeling the sense of each predicate, we build a classification model for each predicate. As the senses of different predicates are usually unrelated even if they have the same sense label, this makes it difficult for us to use only one classifier to label them. But this approach leads to another issue. The set of predicates in the training set cannot cover all predicates. For new predicates in the test set, no classification model can be found for them, and we build a most common sense for them. The features we used are as follow:

- DEPREL1: DEPREL of the predicate.
- STEM1
- POS1
- RM_STEM1 (English only)
- RM_POS1
- FORM2
- POS2
- SUFFIX2
- VOICE (English only): VOICE of predicate.
- POSITION + POS2
- L_POS1 + POS1 + R_POS1: component of left word's POS and predicate POS and right word's POS.
- FORM2 + DEP_PATH_REL
- DEP_PATH_ROOT_POS1
- DEP_PATH_ROOT_REL1

When labeling the semantic role, we use a similar approach as we did in CoNLL Shared Task 2008. However, as the frames information is not supplied for all languages, we do not use it in this task. The features we use are as follows:

- DEPREL1
- STEM1 (English only)
- POS1
- RM_STEM1 (English only)
- RM_POS1

- FORM2
- POS2
- SUFFIX2
- VOICE2 (English only)
- POSITION
- DEP_PATH_REL
- DEP_PATH_POS
- SENSE2
- SENSE2 + VOICE2
- POSITION + VOICE2
- DEP_PATH_LEN
- DEP_PATH_ROOT_REL1

Moreover, we build an iterative model in this shared task. When doing an iterative labeling, the previous labeling results are known. So we can design some new features for checking the previous results in a global view. The features we add for the iterative model are as follows:

- SENSE1: SENSE of the predicate.
- SENSE1 + VOICE1: component of the SENSE + VOICE of predicate.
- VOICE1 + FORM1: component of VOICE and FORM.
- ROLE_LIST1: list of APRED of predicate.

2.4 Iterative Approach

As described above, some subtasks have two groups of features. One is for the pipeline model, and the other is for the iterative model. The usage of these two types of model is the same. The only difference is that they use different features. The iterative model can get more information, so they can use more features. These additional features can contain some joint and global (like frame and global structure) information. The performance may be improved because the viewer is extended. Some structural error and semantic conflict can be fixed.

Although the usage of the two types of model is the same, there are some differences when building the models.

In the iterative step, all information is available for doing parsing and labeling. For example, when doing syntactic dependency relation labeling in the iterative step, the fields "HEAD", "DEPREL", "PRED" and "APREDs" are filled by the pervious iteration. So all these information can be used in the iterative step. This will cause one issue: use "HEAD1" to label "HEAD2". When training the model, "HEAD1" is golden. The classifier will build a model directly and let "HEAD2" equal to "HEAD1". However, in the iterative step, "HEAD1" is not golden, but such model makes it impossible to change the results.. The iterative step will be useless.

We design a simple method to avoid this issue.

- Firstly, split the training set into N (N>1) subsets.
- Secondly, for each subset, use the left N-1 subsets to build an original sub-model (use features in the pipeline step).
- Thirdly, use each sub-model to label the corresponding subset.
- Lastly, use these labeled N subsets to extract samples (use features in the iterative step) for building the iterative model.

In this way, the "HEAD1" is not golden any more. And for each sub-task, we can use the similar method to build the original model and the iterative model.

Moreover, in our system, we only build the iterative models for syntactic dependency relation labeling and semantic role labeling. For syntactic dependency parsing, we use an approach with very high time and space complexity, so it is not added to the iterative step. Thus, its results will not be changed in the iterative step. For sense labeling, we build classification models for every predicate. There are too many models and each model contains only a few classes. We think they are not suitable for building the iterative model. But, as its previous sub-task (syntactic dependency relation labeling) is added to the iterative step, it is useful to add it to the iterative step. Though we do not build an iterative model for sense labeling, we can directly use its pipeline model. This is another advantage of our iterative model: if one subtask is not suitable for doing iterative labeling/parsing, we can use its pipeline model instead.

3 Experiments and Results

We have tested our system with the test set and obtained official results as shown in Table 1. We have tried to find how the iterative step influences syntactic dependency parsing and semantic role labeling. For syntactic dependency parsing and semantic role labeling, we do experiments on the test set.

	Macro F1 Score
Average	73.98
Catalan	72.09
Chinese	72.72
Czech	67.14
English	81.89
German	75.00
Japanese	80.89
Spanish	68.14

Table 1. The Macro F1 Score of every languages and the average value.

3.1 Syntactic Dependency Parsing

Dependency Parsing can be split into two subproblems: syntactic dependency parsing and syntactic dependency label classification. We use the iterative method on syntactic dependency label classification. We do experiments on the test set.

On the test set, we do two group experiments. In the first group, we build a subtest to test this subtask only. All other information is given, and we just label the dependency relation. The results are shown in Table 2. The row of "Initial step" shows the results of this sub task in the original step. The left two rows show the results in the iterative step with iterating once and twice. The table shows that the iterative approach improves the performance. Especially for Catalan, the performance increases by 2.89%.

Certainly, in the whole system, this subtask cannot get golden information about sense and semantic roles. So we test it in the whole system (joint test) on the test set in the second group of experiments. As shown in Table 3, the iterative step is not as good as previous test. But it is still useful for some languages. The reason that some languages have no improvements on the iterative step is that the result of the initial step is not so good.

3.2 Semantic Role Labeling

Like syntactic dependency parsing, we do two tests on Semantic Role Labeling. This result is not consistent with the official data because we have added some features of the subtask. The results of subtest can be found in Table 4. And Table 5 shows the results of the joint test. These two groups of results show that the advantage of the iterative step is not as good as that of syntactic dependency labeling in subtest. But it improves the performance for most languages. The iterative step improves the performance in both two tests.

3.3 Analysis of Results

From the experimental results, we can see that the effect of each part of the iterative step depends on the overall labeling result of the previous step. And the labeling effect varies with different languages. Iterative approach can improve the performance of the system but it strongly depends on the initial labeling result.

4 Conclusion and Future Work

This paper has presented a simple discriminative system submitted to the CoNLL-2009 shared task to address the learning task of syntactic and semantic dependencies. The paper first describes how to carry out syntactic dependency parsing and semantic role labeling, and then a new iterative approach is presented for joint parsing. The experimental results show that the iterative process can improve the labeling accuracy on syntactic and semantic analysis. However, this approach probably depends on the accuracy of the initial labeling results. The results of the initial labeling results will affect the effect of the iterative process.

Because of time constraints and inadequate experimental environment, our first results do not meet our expectation, and the effect of the iterative step is not so clear. Next, we will strive to refine our approach to produce good results for the syntactic dependency parsing, since it has a great impact on the final parsing results.

Acknowledgments

The authors would like to thank the reviewers for their helpful comments. This work was supported by National Natural Science Foundation of China (No.60573077, No.60775037) and the National High Technology Research and Development Program of China (863 Program) (grant no. 2009AA01Z123). We also thank the High-Performance Center of USTC for providing us with the experimental platform.

	Average	Catalan	Chinese	Czech	Czech-ood	English	English-ood	German	German-ood	Japanese	Spanish
Initial step	93.64	95.66	95.01	88.10	88.10	96.79	92.98	96.41	89.71*	98.17	95.48
Iteration 1	94.60	98.56*	96.08*	88.59	88.29	97.31*	94.57*	96.63*	89.31	98.34	98.30
Iteration 2	94.65	98.55	96.08*	88.68*	88.45*	97.29	94.56	96.63*	89.53	98.35*	98.33*

Table 2. The subtest result of Labeled Syntactic Accuracy of each language and the average performance value on test set. (* denotes the best score for the system)

	Average	Catalan	Chinese	Czech	Czech-ood	English	English-ood	German	German-ood	Japanese	Spanish
Initial step	74.02	77.75	73.81	58.69*	55.50*	84.75	78.85	82.45	66.27*	90.45*	71.64
Iteration 1	73.90	77.82	73.86*	58.17	54.95	84.81	78.95	82.51*	65.78	90.43	71.68
Iteration 2	73.94	77.85*	73.86*	58.31	55.13	84.82*	79.02*	82.46	65.85	90.45*	71.69*

Table 3. The joint test result of Labeled Syntactic Accuracy of each language and the average performance value on test set. (* denotes the best score for the system)

	Average	Catalan	Chinese	Czech	Czech-ood	English	English-ood	German	German-ood	Japanese	Spanish
Initial step	83.83	88.56	85.86	88.08	86.20*	86.23	82.09	80.98	78.82	74.32*	87.45
Iteration 1	84.34	89.02*	87.14*	87.88	86.09	86.66	82.07	83.66*	79.28*	74.06	87.59
Iteration 2	84.36	89.02*	87.01	88.10*	86.17	86.78*	82.34*	83.15	79.18	74.06	87.81*

Table 4. The sub test result of Semantic Labeled F1 of each language and the average performance value on test set. (* denotes the best score for the system)

	Average	Catalan	Chinese	Czech	Czech-ood	English	English-ood	German	German-ood	Japanese	Spanish
Initial step	70.01	66.87	71.63	75.50	75.71	78.97	69.87	67.50	58.47	70.91*	64.64
Iteration 1	70.15	67.12	71.98	75.54	75.68	79.40	70.17*	68.08*	58.55*	70.69	64.32
Iteration 2	70.20	67.33*	71.99*	75.65*	75.90*	79.47*	69.98	67.98	58.33	70.70	64.65*

Table 5. The joint test result of Semantic Labeled F1 of each language and the average performance value on test set. (* denotes the best score for the system)

References

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antonia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue and Yi Zhang. 2009. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009). Boulder, Colorado, USA. June 4-5. pp. 3-22.

Mariona Taulé, Maria Antònia Martí and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008). Marrakech, Morocco.

Nianwen Xue and Martha Palmer. 2009. Adding semantic roles to the Chinese Treebank. Natural Language Engineering, 15(1):143-172.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová and Zdeněk Žabokrtský. 2006. The Prague Dependency Treebank 2.0. CD-ROM. Linguistic Data Consortium, Philadelphia, Pennsylvania, USA. ISBN 1-58563-370-4. LDC Cat. No. LDC2006T01. URL: http://ldc.upenn.edu.

Surdeanu, Mihai, 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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008).

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó and Manfred Pinkal. 2006. The SALSA Corpus: a German Corpus Resource for Lexical Semantics. Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006). Genoa, Italy.

Daisuke Kawahara, Sadao Kurohashi and Koiti Hasida. 2002. Construction of a Japanese Relevance-tagged Corpus. *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002).* Las Palmas, Spain. pp. 2008-2013.

McDonald, Ryan. 2006. Discriminative learning and Spanning Tree Algorithms for Dependency parsing. Ph.D. thesis, University of Pennyslvania.

Stanley F. Chen and Ronald Rosenfeld. 1999. A gaussian prior for smoothing maximum entropy models. Technical. *Report CMU-CS-99-108.*

Joint memory-based learning of syntactic and semantic dependencies in multiple languages

Roser Morante, Vincent Van Asch
CNTS - Language Technology Group
University of Antwerp
Prinsstraat 13
B-2000 Antwerpen, Belgium
{Roser.Morante,Vincent.VanAsch}@ua.ac.be

Antal van den Bosch
Tilburg University
Tilburg centre for Creative Computing
P.O. Box 90153
NL-5000 LE Tilburg, The Netherlands
Antal.vdnBosch@uvt.nl

Abstract

In this paper we present a system submitted to the CoNLL Shared Task 2009 performing the identification and labeling of syntactic and semantic dependencies in multiple languages. Dependencies are truly jointly learned, i.e. as if they were a single task. The system works in two phases: a classification phase in which three classifiers predict different types of information, and a ranking phase in which the output of the classifiers is combined.

1 Introduction

In this paper we present the machine learning system submitted to the CoNLL Shared Task 2009 (Hajič et al., 2009). The task is an extension to multiple languages (Burchardt et al., 2006; Hajič et al., 2006; Kawahara et al., 2002; Palmer and Xue, 2009; Surdeanu et al., 2008; Taulé et al., 2008) of the CoNLL Shared Task 2008, combining the identification and labeling of syntactic dependencies and semantic roles. Our system is a joint-learning system tested in the "closed" challenge, i.e. without making use of external resources.

Our system operates in two phases: a classification phase in which three memory-based classifiers predict different types of information, and a ranking phase in which the output of the classifiers is combined by ranking the predictions. Semantic and syntactic dependencies are jointly learned and processed. In the task description no precise definition is given of joint learning. We consider that a joint-learning system is one in which semantic and syntactic dependencies are learned and processed jointly as a single task. In our system this is achieved by fully merging semantic and syntactic dependencies at the word level as the first step.

One direct consequence of merging the two tasks, is that the class space becomes more complex; the number of classes increases. Many machine-learning approaches do not scale well to larger class spaces in terms of efficiency and computer resource requirements. Memory-based learning is a noted exception, as it is largely insensitive to the number of classes in terms of efficiency. This is the primary reason for using memory-based learning. Memory-based language processing (Daelemans and van den Bosch, 2005) is based on the idea that NLP problems can be solved by storing solved examples of the problem in their literal form in memory, and applying similarity-based reasoning on these examples in order to solve new ones. Memory-based algorithms have been previously applied to semantic role labeling and parsing separately (Morante et al., 2008; Canisius and Tjong Kim Sang, 2007).

We briefly discuss the issue of true joint learning of two tasks in Section 2. The system is described in Section 3, Section 4 presents and discusses the results, and in Section 5 we put forward some conclusions and future research.

2 Joint learning

When two tasks share the same feature space, there is the natural option to merge them and consider the merge as a single task. The merging of two tasks will typically lead to an increase in the number of classes, and generally a more complex class space. In practice, if two combined tasks are to some ex-

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 25–30,
Boulder, Colorado, June 2009.

tent related, the increase will tend to be less than the product of the number of classes in the two original tasks, as classes from both tasks will tend to correlate. Yet, even a mild increase of the number of classes leads to a further fragmentation of the class space, and thus to less training examples per class label. Joint learning can therefore only lead to positive results if the data sparsity effect of the fragmentation of the class space is counter-balanced by an improved learnability.

Here, we treat the syntactic and semantic tasks as one and the same task. At the word level, we merge the class labels of the two tasks into single labels, and present the classifiers with these labels. Further on in our system, as we describe in the next section, we do make use of the compositionality of the labels, as the semantic and syntactic output spaces represented two different types of structure.

3 System description

The joint system that we submitted works in two phases: a classification phase in which three memory-based classifiers predict different aspects of joint syntactic and semantic labeling, and a ranking phase in which the output of the classifiers is combined. Additionally, a memory-based classifier is used for predicate sense disambiguation. As a first step, before generating the instances of the classifiers we merge the semantic and syntactic dependencies into single labels. The merged version of the dependencies from an example sentence is shown in Table 1, where column MERGED DEPs contains all the dependencies of a token separated by a blank space expressed in labels with the following format: PHEAD::PDEPREL:APRED.

3.1 Phase 1: Classification

In the classification phase, three classifiers predict different local aspects of the global output structure. The classifiers have been optimized for English, by training on the full training set and testing on the development set; these optimized settings were then used for the other six languages. We experimented with manually selected parameters and with parameters selected by a genetic algorithm, but the parameters found by the genetic algorithm did not yield better results than the manually selected parameters.

N	Token	Merged Dependencies
1	Housing	2::NMOD:A1
2	starts	2::_:A2 3::SBJ:_ 4::_:A1 6::_:A1 13::_:A0
3	are	0::ROOT:_
4	expected	3::VC:_
5	to	4::OPRD:C-A1
6	quicken	5::IM:_
7	a	8::NMOD:_
8	bit	6::OBJ:A2
9	from	6::ADV:A3
10	August	13::NMOD:AM-TMP
11	's	10::SUFFIX:_
12	annual	13::NMOD:AM-TMP
13	pace	9::PMOD:_
14	of	13::NMOD:A2
15	1,350,000	16::NMOD:_
16	units	14::PMOD:_
17	.	3::P:_

Table 1: Example sentence with merged dependency labels.

3.1.1 Classifier 1: Pairwise semantic and syntact dependencies

Classifier 1 predicts the merged semantic and syntactic dependencies that hold between two tokens. Instances represent combinations of pairs of tokens within a sentence. Each token is combined with all other tokens in the sentence. The class predicted is the PDEPREL:APRED label. The amount of classes per language is shown in Table 2 ("Classifier 1").

	Number of classes	
Lang.	Classifier 1	Classifier 2
Cat	111	111
Chi	309	1209
Cze	395	1221
Eng	351	1957
Ger	152	300
Jap	103	505
Spa	124	124

Table 2: Number of classes per language predicted by Classifiers 1 and 2.

We use an IB1 memory–based algorithm as implemented in TiMBL (version 6.1.2) [1], a memory-based classifier based on the k-nearest neighbor

[1] TiMBL: `http://ilk.uvt.nl/timbl`

rule. The IB1 algorithm was parameterised by using modified value difference as the similarity metric, gain ratio for feature weighting, using 11 k-nearest neighbors, and weighting the class vote of neighbors as a function of their inverse linear distance. Because of time limitations we used TRIBL for Czech and Chinese to produce the official results, although we also provide postevaluation results produced with IB1. TRIBL is a hybrid combination of IB1 and IGTREE, a fast decision-tree approximation of k-NN (Daelemans and van den Bosch, 2005), trading off fast decision-tree lookup on the most important features (in our experiments, five) with slower k-NN classification on the remaining features.

The features[2] used by this classifier are:

- The word, lemma, POS and FILLPRED[3] of the token, the combined token and of two tokens before and after token and combined token.
- POS and FILLPRED of the third token before and after token and combined token.
- Distance between token and combined token, location of token in relation to combined token.

Because data are skewed towards the NONE class, we downsampled the training instances so that there would be a negative instance for every positive instance. Instances with the NONE class to be kept were randomly selected.

3.1.2 Classifier 2: Per-token relations

Classifier 2 predicts the labels of the dependency relations of a token with its syntactic and/or semantic head(s). Instances represent a token. As an example, the instance that represents token 2 in Table 1 would have as class: _:A2-SBJ:_-_:A1-_:A1-_:A0. The amount of classes per language is shown in Table 2 under "Classifier 2". The number of classes exceeds 1,000 for Chinese, Czech, and English.

The features used by the classifier are the word, lemma, POS and FILLPRED of the token and two tokens before and after the token. We use the IB1 memory–based algorithm parameterised in the same way as Classifier 1.

[2] *POS* refers to predicted part-of-speech and *lemma* to predicted lemma in the description of features for all classifiers.

[3] The FILLPRED column has value Y if a token is a predicate.

3.1.3 Classifier 3: Pairwise detection of a relation

Classifier 3 is a binary classifier that predicts whether two tokens have a dependency relation. Instance representation follows the same scheme as with Classifier 1. We use the IGTREE algorithm as implemented in TiMBL. The data are also skewed towards the NONE class, so we downsampled the training instances so that there would be a negative instance for every four positive instances.

The features used by this classifier are:

- The word, lemma, POS and FILLPRED of the token, of the combined token, and of two tokens before and after the token.
- Word and lemma of two tokens before and after combined token.
- Distance between token and combined token.

3.1.4 Results

The results of the Classifiers are presented in Table 3. The performance of Classifiers 1 and 3 is similar across languages, whereas the scores for Classifier 2 are lower for Chinese, Czech and English. This can be explained by the fact that the number of classes that Classifier 2 predicts for these languages is significantly higher.

Lang.	C1	C2	C3
Cat	94.77	86.30	97.96
Chi	92.10	70.11	95.47
Cze	87.33	67.87	93.88
Eng	94.17	76.16	95.37
Ger	92.76	83.23	93.77
Jap	91.55	81.22	96.75
Spa	94.76	84.40	96.39

Table 3: Micro F1 scores per classifier (C) and per language.

Training times for the three classifiers were reasonably short, as is to be expected with memory-based classification. With English, C2 takes just over two minutes to train, and C3 half a minute. C1 takes 8 hours and 18 minutes, due to the much larger amount of examples and features.

3.2 Phase 2: Ranking

The classifier that is at the root of generating the desired output (dependency graphs and semantic

role assignments) is Classifier 1, which predicts the merged semantic and syntactic dependencies that hold between two tokens (PDEPREL:APRED labels). If this classifier would be able to predict the dependencies with 100% accuracy, no further processing would be necessary. Naturally, however, the classifier predicts incorrect dependencies to a certain degree, and does not provide a graph in wich all tokens have at least a syntactic head. It achieves 51.3% labeled macro F1. The ranking phase improves this performance. This is done in three steps: (i) ranking the predictions of Classifier 1; (ii) constructing an intermediate dependency tree, and (iii) adding extra semantic dependencies to the tree.

3.2.1 Ranking predictions of Classifier 1

In order to disambiguate between all possible dependencies predicted by this classifier, the system applies ranking rules. It analyses the dependency relations that have been predicted for a token with its potential parents in the sentence and ranks them. For example, for a sentence with 10 tokens, the system would make 10 predictions per token. The predictions are first ranked by entropy of the class distribution for that prediction, then using the output of Classifier 2, and next using the output of Classifier 3.

Ranking by entropy In order to compute entropy we use the (inverse-linear) distance-weighted class label distributions among the nearest neighbors that Classifier 1 was able to find. For example, the prediction for an instance can be: { NONE (2.74), NMOD:_ (0.48) }. We can compute the entropy for this instance using the formula in (1):

$$-\sum_{i=1}^{n} P(label_i) log_2(P(label_i)) \qquad (1)$$

with
- n: the total number of different labels in the distribution, and
- $P(label_i)$: $\frac{\text{the weight of label i}}{\text{the total sum of the weights in the distribution}}$

The system ranks the prediction with the lowest entropy in position 1, while the prediction with the highest entropy is ranked in the last position. The rationale behind this is that the lower the entropy, the more certain the classifier is about the predicted dependency. Table 4 lists the first six heads for the predicate word 'starts' ranked by entropy (cf. Table 1).

Head	Predicted label	Distribution	Entropy
Housing	NONE	{ NONE (8.51) }	0.0
expected	_:A1	{ _:A1 (5.64) }	0.0
to	NONE	{ NONE (4.74) }	0.0
quicken	_:A0	{ _:A0 (4.13), _:A1 (0.18), _:A2 (0.31) }	0.56
are	NONE	{ NONE (2.56), SBJ:_ (0.52) }	0.65
starts	_:A0	{ _:A0 (7.90), _:A1 (0.61), _:A2 (1.50) }	0.93

Table 4: Output of Classifier 1 for the first six heads of 'starts', ranked by entropy.

On the development data for English, applying this rule causes a marked error reduction of 26.5% on labeled macro F1: from 51.3% to 64.2%.

Ranking by Classifier 2 The next ranking step is performed by using the predictions of Classifier 2, i.e. the estimated labels of the dependency relations of a token with its syntactic and/or semantic head(s). The system ranks the predictions that are not in the set of possible dependencies predicted by Classifier 2 at the bottom of the ranked list.

Head	Predicted label	Distribution	Entropy
expected	_:A1	{ _:A1 (5.64) }	0.0
Housing	NONE	{ NONE (8.51) }	0.0
to	NONE	{ NONE (4.74) }	0.0
quicken	_:A0	{ _:A0 (4.13), _:A1 (0.18), _:A2 (0.31) }	0.56
are	NONE	{ NONE (2.56), SBJ:_ (0.52) }	0.65
starts	_:A0	{ _:A0 (7.90), _:A1 (0.61), _:A2 (1.50) }	0.93

Table 5: Output of Classifier 1 for the first six heads of 'starts'. Ranked by entropy and Classifier 2.

Because this is done after ranking by entropy, the instances with the lowest entropy are still at the top of the list. Table 5 displays the re-ranked six heads of 'starts', given that Classifier 2 has predicted that possible relations to heads are SBJ:A1 and _:A1, and given that only 'expected' is associated with one of these two relations.

On the development data for English, applying this rule induces a 9.0% error reduction on labeled macro F1: from 64.2% to 67.4%.

Ranking by Classifier 3 The final ranking step makes use of Classifier 3, which predicts the relation that holds between two tokens. The dependency relations predicted by Classifier 1 that are not confirmed by Classifier 3 predicting that a relation exists are moved to the end of the ranked list. Table 6 lists the resulting ranked list. On the development data for English, applying this rule yields another 5.2%

error reduction on labeled macro F1: from 67.4% to 69.1%.

Head	Predicted label	Distribution	Entropy
expected	_:A1	{ _:A1 (5.64) }	0.0
quicken	_:A0	{ _:A0 (4.13), _:A1 (0.18), _:A2 (0.31) }	0.56
starts	_:A0	{ _:A0 (7.90), _:A1 (0.61), _:A2 (1.50) }	0.93
Housing	NONE	{ NONE (8.51) }	0.0
to	NONE	{ NONE (4.74) }	0.0
are	NONE	{ NONE (2.56), SBJ:_ (0.52) }	0.65

Table 6: Output of Classifier 1 for the first six heads of 'starts'. Ranked by entropy, Classifier 2, and Classifier 3.

3.2.2 Construction of the intermediate dependency tree

After ranking the predictions of Classifier 1, the system selects a syntactic head for every token. This is motivated by the fact that every token has one and only one syntactic head. The system selects the prediction with the best ranking that has in the PDEPREL part a value different than "_".

The intermediate tree can have more than one root or no root at all. To make sure that every sentence has one and only one root we apply some extra rules. If the sentence does not have a token with a root label, the system checks the distributions of Classifier 1. The token with the rootlabel in its distribution that is the head of the biggest number of tokens is taken as root. If the intermediate tree has more than one root, the last root is taken as root. The other root tokens get the label with a syntax part (PDEPREL) that has the highest score in the distribution of Classifier 1.

The product of this step is a tree in which every token is uniquely linked to a syntactic head. Because syntactic and semantic dependencies have been linked, the tree contains also semantic dependencies. However, the tree is missing the purely semantic dependencies. The next step adds these relations to the dependency tree.

3.2.3 Adding extra semantic dependencies

In order to find the tokens that have only a semantic relation with a predicate, the system analyses for each predicate (i.e. tokens marked with Y in FILLPRED) the list of predictions made by Classifier 1 and selects the predictions in which the PDEPREL part of the label is "_" and the APRED part of the label is different than "_". On the development data for English, applying this rule produces a 6.7% error reduction on labeled macro F1: from 69.1% to 71.1%.

3.3 Predicate sense disambiguation

Predicate sense disambiguation is performed by a classifier per language that predicts the sense of the predicate, except for Japanese, as with that language the lemma is taken as the sense. We use the IGTREE algorithm. Instances represent predicates and the features used are the word, lemma and POS of the predicate, and the lemma and POS of two tokens before and after the predicate. The results per language are presented in Table 7.

Lang.	Cat	Chi	Cze	Eng	Ger	Spa
F1	82.40	94.85	87.84	93.64	73.57	81.13

Table 7: Micro F1 for the predicate sense disambiguation.

4 Overall results

The system was developed by training on the training set provided by the task organisers and testing on the development set. The final results were obtained by testing on the testing set. Table 8 shows the global results of the system for syntactic and semantic dependencies.

Lang.	F1	Precision	Recall
Cat	73.75	74.91	72.63
Chi	67.16	68.09	66.26
Chi*	67.79	68.70	66.89
Cze	60.50	62.55	58.58
Cze*	68.68	70.38	67.07
Eng	78.19	79.69	76.74
Ger	67.51	69.52	65.62
Jap	77.75	81.91	73.98
Spa	70.78	71.34	70.22
Av.	70.81	72.57	69.15

Table 8: Macro F1, precision and recall for all dependencies per language. Postevaluation results are marked with *.

Table 9 shows the scores of syntactic and semantic dependencies in isolation.

	Syntax	Semantics		
Lang.	LA	F1	Precision	Recall
Cat	77.33	70.14	72.49	67.94
Chi	67.58	66.71	68.59	64.93
Chi*	67.92	67.63	69.48	65.86
Cze	49.41	71.49	75.68	67.75
Cze*	60.03	77.28	80.73	74.11
Eng	80.35	75.97	79.04	73.13
Ger	73.88	61.01	65.15	57.36
Jap	86.17	68.82	77.66	61.80
Spa	73.07	68.48	69.62	67.38
Av.	72.54	68.95	72.60	65.76

Table 9: Labeled attachment (LA) score for syntactic dependencies and Macro F1, precision and recall of semantic dependencies per language. Postevaluation results are marked with *.

5 Conclusions

In this paper we presented the system that we submitted to the "closed" challenge of the CoNLL Shared Task 2009. We observe fairly low scores, which can be possibly improved for all languages by making use of the available morpho-syntactic features, which we did not use in the present system, by optimising the classifiers per language, and by improving the reranking algorithm. We also observe a relatively low recall on the semantic task as compared to overall recall, indicating that syntactic dependencies are identified with a better precision-recall balance. A logical continuation of this study is to compare joint learning to learning syntactic and semantic dependencies in isolation, using the same architecture. Only then will we be able to put forward conclusions about the performance of a joint learning system versus the performance of a system that learns syntax and semantics independently.

Acknowledgments

This study was made possible through financial support from the University of Antwerp (GOA project BIOGRAPH), and from the Netherlands Organisation for Scientific Research.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

S. Canisius and E. Tjong Kim Sang. 2007. A constraint satisfaction approach to dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 1124–1128.

W. Daelemans and A. van den Bosch. 2005. *Memory-based language processing.* Cambridge University Press, Cambridge, UK.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

R. Morante, W. Daelemans, and V. Van Asch. 2008. A combined memory-based semantic role labeler of english. In *Proc. of the CoNLL 2008*, pages 208–212, Manchester, UK.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco.

Hybrid Multilingual Parsing with HPSG for SRL

Yi Zhang
Language Technology
DFKI GmbH, Germany
yzhang@coli.uni-sb.de

Rui Wang
Computational Linguistics
Saarland University, Germany
rwang@coli.uni-sb.de

Stephan Oepen
Informatics
University of Oslo, Norway
oe@ifi.uio.no

Abstract

In this paper we present our syntactic and semantic dependency parsing system submitted to both the closed and open challenges of the CoNLL 2009 Shared Task. The system extends the system of Zhang, Wang, & Uszkoreit (2008) in the multilingual direction, and achieves 76.49 average macro F1 Score on the closed joint task. Substantial improvements to the open SRL task have been observed that are attributed to the HPSG parses with hand-crafted grammars. †

1 Introduction

The CoNLL 2009 shared task (Hajič et al., 2009) continues the exploration on learning syntactic and semantic structures based on dependency notations in previous year's shared task. The new addition to this year's shared task is the extension to multiple languages. Being one of the leading competitions in the field, the shared task received submissions from systems built on top of the state-of-the-art data-driven dependency parsing and semantic role labeling systems. Although it was originally designed as a task for machine learning approaches, CoNLL shared tasks also feature an 'open' track since 2008, which encourages the use of extra linguistic resources to further improve the performance. This makes the task a nice testbed for the cross-fertilization of various language processing techniques.

As an example of such work, Zhang et al. (2008) have shown in the past that deep linguistic parsing outputs can be integrated to help improve the performance of the English semantic role labeling task. But several questions remain unanswered. First, the integration only experimented with the semantic role labeling part of the task. It is not clear whether syntactic dependency parsing can also benefit from grammar-based parsing results. Second, the English grammar used to achieve the improvement is one of the largest and most mature hand-crafted linguistic grammars. It is not clear whether similar improvements can be achieved with less developed grammars. More specifically, the lack of coverage of hand-crafted linguistic grammars is a major concern. On the other hand, the CoNLL task is also a good opportunity for the deep processing community to (re-)evaluate their resources and software.

2 System Architecture

The overall system architecture is shown in Figure 1. It is similar to the architecture used by Zhang et al. (2008). Three major components were involved. The HPSG parsing component utilizes several hand-crafted grammars for deep linguistic parsing. The outputs of deep parsings are passed to the syntactic dependency parser and semantic role labeler. The syntactic parsing component is composed of a modified MST parser which accepts HPSG parsing results as extra features. The semantic role labeler is comprised of a pipeline of 4 sub-components (predicate identification is not necessary in this year's task). Comparing to Zhang et al. (2008), this architecture simplified the syntactic component, and puts more focus on the integration of deep parsing outputs. While Zhang et al. (2008) only used seman-

†We are indebted to our DELPH-IN colleagues, specifically Peter Adolphs, Francis Bond, Berthold Crysmann, and Montserrat Marimon for numerous hours of support in adapting their grammars and the PET software to parsing the CoNLL data sets. The first author thanks the German Excellence Cluster of Multimodal Computing and Interaction for the support of the work. The second author is funded by the PIRE PhD scholarship program. Participation of the third author in this work was supported by the University of Oslo, as part of its research partnership with the Center for the Study of Language and Information at Stanford University. Our deep parsing experimentation was executed on the TITAN HPC facilities at the University of Oslo.

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 31–36,
Boulder, Colorado, June 2009.

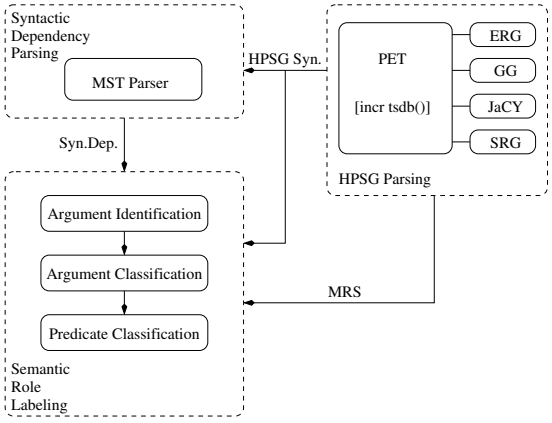

Figure 1: Joint system architecture.

tic features from HPSG parsing in the SRL task, we added extra syntactic features from deep parsing to help both tasks.

3 HPSG Parsing for the CoNLL Data

DELPH-IN (Deep Linguistic Processing with HPSG) is a repository of open-source software and linguistic resources for so-called 'deep' grammatical analysis.[1] The grammars are rooted in relatively detailed, hand-coded linguistic knowledge—including lexical argument structure and the linking of syntactic functions to thematic arguments—and are intended as general-purpose resources, applicable to both parsing and generation. Semantics in DELPH-IN is cast in the Minimal Recursion Semantics framework (MRS; Copestake, Flickinger, Pollard, & Sag, 2005), essentially predicate – argument structures with provision for underspecified scopal relations. For the 2009 'open' task, we used the DELPH-IN grammars for English (ERG; Flickinger, 2000), German (GG; Crysmann, 2005), Japanese (JaCY; Siegel & Bender, 2002), and Spanish (SRG; Marimon, Bel, & Seghezzi, 2007). The grammars vary in their stage of development: the ERG comprises some 15 years of continuous development, whereas work on the SRG only started about five years ago, with GG and JaCY ranging somewhere inbetween.

3.1 Overall Setup

We applied the DELPH-IN grammars to the CoNLL data using the PET parser (Callmeier, 2002) running it through the [incr tsdb()] environment (Oepen & Carroll, 2000), for parallelization and distribution. Also, [incr tsdb()] provides facilities for (re-)training the MaxEnt parse selection models that PET uses for disambiguation.

The two main challenges in applying DELPH-IN resources to parsing CoNLL data were (a) mismatches in basic assumptions, specifically tokenization and the inventory of PoS tags provided as part of the input, and (b) the need to adapt the resources for new domains and genres—in particular in terms of parse disambiguation—as the English and Spanish grammars at least had not been previously applied to the corpora used in the CoNLL shared task.

The importance of the first of these two aspects is often underestimated. A detailed computational grammar, inevitably, comes with its own assumptions about tokenization—the ERG, for example, rejects the conventional assumptions underlying the PTB (and derived tools). It opts for an analysis of punctuation akin to affixation (rather than as stand-alone tokens), does not break up contracted negated auxiliaries, and splits hyphenated words like *ill-advised* into two tokens (the hyphen being part of the first component). Thus, a string like *Don't you!* in the CoNLL data is tokenized as the four-element sequence $\langle do, n't, you, ! \rangle$,[2] whereas the ERG analysis has only two leaf nodes: $\langle don't, you! \rangle$.

Fortunately, the DELPH-IN toolchain recently incorporated a mechanism called *chart mapping* (Adolphs et al., 2008), which allows one to map flexibly from 'external' input to grammar-internal assumptions, while keeping track of external token identities and their contributions to the final analysis. The February 2009 release of the ERG already had this machinery in place (with the goal of supporting extant, PTB-trained PoS taggers in pre-processing input to the deep parser), and we found that only a tiny number of additional chart mapping rules was required to 'fix up' CoNLL-specific deviations from the PTB tradition. With the help of the original developers, we created new chart mapping configurations for the German and Japanese grammars (with 17 and 16 such accomodation rules, respectively) in a similar spirit. All four DELPH-IN grammars in-

[1] See `http://www.delph-in.net` for background.

[2] Note that the implied analogy to a non-contracted variant is linguistically mis-leading, as **Do not you!* is ungrammatical.

clude an account of unknown words, based on underspecified 'generic' lexical entries that are activated from PoS information.

The Japenese case was interesting, in that the grammar assumes a different pre-processor (ChaSen, rather than Juman), such that not only token boundaries but also PoS tags and morphological features had to be mapped. From our limited experience to date, we found the chart mapping approach adequate in accomodating such discrepancies, and the addition of this extra layer of input processing gave substantial gains in parser coverage (see below). For the Spanish data, on the other hand, we found it impossible to make effective use of the PoS and morphological information in the CoNLL data, due to more fundamental discrepancies (e.g. the treatment of enclitics and multi-word expressions).

3.2 Retraining Disambiguation Models

The ERG includes a domain-specific parse selection model (for tourism instructions); GG only a stub model trained on a handful of test sentences. For use on the CoNLL data, thus, we had to train new parse selections models, better adapted to the shared task corpora. Disambiguation in PET is realized by conditional MaxEnt models (Toutanova, Manning, Flickinger, & Oepen, 2005), usually trained on full HPSG treebanks. Lacking this kind of training material, we utilized the CoNLL dependency information instead, by defining an unlabeled *dependency accuracy* (DA) metric for HPSG analyses, essentially quantifying the degree of overlap in head–dependent relations against the CoNLL annotations.

Calculating DA for HPSG trees is similar to the procedure commonly used for extracting bi-lexical dependencies from phrase structure trees, in a sense even simpler as HPSG analyses fully determine headeness. Taking into account the technical complication of token-level mismatches, our DA metric loosely corresponds to the unlabeled attachment score. To train CoNLL-specific parse selection models, we parsed the development sections in 500-best mode (using the existing models) and then mechanically 'annotated' the HPSG analyses with maximum DA as preferred, all others as dis-preferred. In other words, this procedure constructs a 'binarized' empirical distribution where estimation of log-linear model parameters amounts to adjusting conditional probabilities towards higher DA values.[3]

Grammar	Coverage	Time
ERG	80.4%	10.06 s
GG	28.6%	3.41 s
JaCY	42.7%	2.13 s
SRG	7.5%	0.80 s

Table 1: Performance of the DELPH-IN grammars.

Using the [incr tsdb()] MaxEnt experimentation facilities, we trained new parse selection models for English and German, using the first 16,000 sentences of the English training data and the full German training corpus; seeing that only inputs that (a) parse successfully and (b) have multiple readings, with distinct DA values are relevant to this step, the final models reflect close to 13,000 sentences for English, and a little more than 4,000 items for German. Much like in the SRL component, these experiments are carried out with the TADM software, using ten-fold cross-validation and exact match ranking accuracy (against the binarized training distribution) to optimize estimation hyper-parameters

3.3 Deep Parsing Features

HPSG parsing coverage and average cpu time per input for the four languages with DELPH-IN grammars are summarized in Table 1. The PoS-based unknown word mechanism was active for all grammars but no other robustness measures (which tend to lower the quality of results) were used, i.e. only complete spanning HPSG analyses were accepted. Parse times are for 1-best parsing, using selective unpacking (Zhang, Oepen, & Carroll, 2007).

HPSG parsing outputs are available in several different forms. We investigated two types of structures: syntactic derivations and MRS meaningrepresentations. Representative features were extracted from both structures and selectively used in the statistical syntactic dependency parsing and semantic role labeling modules for the 'open' challenge.

[3] We also experimented with using DA scores directly as empirical probabilities in the training distribution (or some function of DA, to make it fall off more sharply), but none of these methods seemed to further improve parse selection performance.

Deep Semantic Features Similar to Zhang et al. (2008), we extract a set of features from the semantic outputs (MRS) of the HPSG parses. These features represent the basic predicate-argument structure, and provides a simplified semantic view on the target sentence.

Deep Syntactic Dependency Features A HPSG derivation is a tree structure. The internal nodes are labeled with identifiers of grammar rules, and leaves with lexical entries. The derivation tree provides complete information about the actual HPSG analysis, and can be used together with the grammar to reproduce complete feature structure and/or MRS. Given that the shared task adopts dependency representation, we further map the derivation trees into token-token dependencies, labeled by corresponding HPSG rules, by defining a set of head-finding rules for each grammar. This dependency structure is different from the dependencies in CoNLL dataset, and provides an alternative HPSG view on the sentences. We refer to this structure as the dependency backbone (DB) of the HPSG anaylsis. A set of features were extracted from the deep syntactic dependency structures. This includes: i) the POS of the DB parent from the predicate and/or argument; ii) DB label of the argument to its parent (only for AI/AC); iii) labeled path from predicate to argument in DB (only for AI/AC); iv) POSes of the predicate's DB dependents

4 Syntactic Dependency Parsing

For the syntactic dependency parsing, we use the MST Parser (McDonald et al., 2005), which is a graph-based approach. The best parse tree is acquired by searching for a spanning tree which maximizes the score on either a partially or a fully connected graph with all words in the sentence as nodes (Eisner, 1996; McDonald et al., 2005). Based on our experience last year, we use the second order setting of the parser, which includes features over pairs of adjacent edges as well as features over single edges in the graph. For the projective or non-projective setting, we compare the results on the development datasets of different languages. According to the parser performance, we decide to use non-projective parsing for German, Japanese, and Czech, and use projective parsing for the rest.

For the Closed Challenge, we first consider whether to use the morphological features. We find that except for Czech, parser performs better without morphological features on other languages (English and Chinese have no morphological features). As for the other features (i.e. lemma and pos) given by the data sets, we also compare the gold standard features and P-columns. For all languages, the performance decreases in the following order: training with gold standard features and evaluating with the gold standard features, training with P-columns and evaluating with P-columns, training with gold standard features and testing with P-columns. Consequently, in the final submission, we take the second combination.

The goal of the Open Challenge is to see whether using external resources can be helpful for the parsing performance. As we mentioned before, our deep parser gives us both the syntactic analysis of the input sentences using the HPSG formalism and also the semantic analysis using MRS as the representation. However, for the syntactic dependency parsing, we only extract features from the syntactic HPSG analyses and feed them into the MST Parser. Although, when parsing with gold standard lemma and POS features, our open system outperforms the closed system on out-domain tests (for English), when parsing with P-columns there is no substantial improvement observed after using the HPSG features. Therefore, we did not include it in the final submission.

5 Semantic Role Labeling

The semantic role labeling component used in the submitted system is similar to the one described by Zhang et al. (2008). Since predicates are indicated in the data, the predicate identification module is removed from this year's system. Argument identification, argument classification and predicate classification are the three sub-components in the pipeline. All of them are MaxEnt-based classifiers. For parameter estimation, we use the open source TADM system (Malouf, 2002).

The active features used in various steps of SRL are fine tuned separately for different languages using development datasets. The significance of feature types varies across languages and datasets.

		ca	zh	cs	en	de	ja	es
SYN	Closed	82.67	73.63	75.58	87.90	84.57	91.47	82.69
	ood	-	-	71.29	81.50	75.06	-	-
SRL	Closed	67.34	73.20	78.28	77.85	62.95	64.71	67.81
	ood	-	-	77.78	67.07	54.87	-	-
	Open	-	-	-	78.13 (↑0.28)	64.31 (↑1.36)	65.95 (↑1.24)	68.24 (↑0.43)
	ood	-	-	-	68.11 (↑1.04)	58.42 (↑3.55)	-	-

Table 2: Summary of System Performance on Multiple Languages

In the *open* challenge, two groups of extra features from HPSG parsing outputs, as described in Section 3.3, were used on languages for which we have HPSG grammars, that is English, German, Japanese, and Spanish.

6 Result Analysis

The evaluation results of the submitted system are summarized in Table 2. The overall ranking of the system is #7 in the closed challenge, and #2 in the open challenge. While the system achieves mediocre performance, the clear performance difference between the closed and open challenges of the semantic role labeler indicates a substantial gain from the integration of HPSG parsing outputs. The most interesting observation is that even with grammars which only achieve very limited coverage, noticeable SRL improvements are obtained. Confirming the observation of Zhang et al. (2008), the gain with HPSG features is more significant on out-domain tests, this time on German as well.

The training of the syntactic parsing models for all seven languages with MST parser takes about 100 CPU hours with 10 iterations. The dependency parsing takes 6 – 7 CPU hours. The training and testing of the semantic role labeler is much more efficient, thanks to the use of MaxEnt models and the efficient parameter estimation software. The training of all SRL models for 7 languages takes about 3 CPU hours in total. The total time for semantic role labeling on test datasets is less than 1 hour.

Figure 2 shows the learning curve of the syntactic parser and semantic role labeler on the Czech and English datasets. While most of the systems continue to improve when trained on larger datasets, an exception was observed with the Czech dataset on the out-domain test for syntactic accuracy. In most of the cases, with the increase of training data, the out-domain test performance of the syntactic parser and semantic role labeler improves slowly relative to the in-domain test. For the English dataset, the SRL learning curve climbs more quickly than those of syntactic parsers. This is largely due to the fact that the semantic role annotation is sparser than the syntactic dependencies. On the Czech dataset which has dense semantic annotation, this effect is not observed.

7 Conclusion

In this paper, we described our syntactic parsing and semantic role labeling system participated in both closed and open challenge of the (Joint) CoNLL 2009 Shared Task. Four hand-written HPSG grammars of a variety of scale have been applied to parse the datasets, and the outcomes were integrated as features into the semantic role labeler of the system. The results clearly show that the integration of HPSG parsing results in the semantic role labeling task brings substantial performance improvement. The conclusion of Zhang et al. (2008) has been reconfirmed on multiple languages for which we hand-built HPSG grammars exist, even where grammatical coverage is low. Also, the gain is more significant on out-of-domain tests, indicating that the hybrid system is more robust to cross-domain variation.

References

Adolphs, P., Oepen, S., Callmeier, U., Crysmann, B., Flickinger, D., & Kiefer, B. (2008). Some fine points of hybrid natural language parsing. In *Proceedings of the 6th International Conference on Language Resources and Evaluation.* Marrakech, Morocco.

Burchardt, A., Erk, K., Frank, A., Kowalski, A., Padó, S., & Pinkal, M. (2006). The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 4th International Conference on Language Resources and Evaluation.* Genoa, Italy.

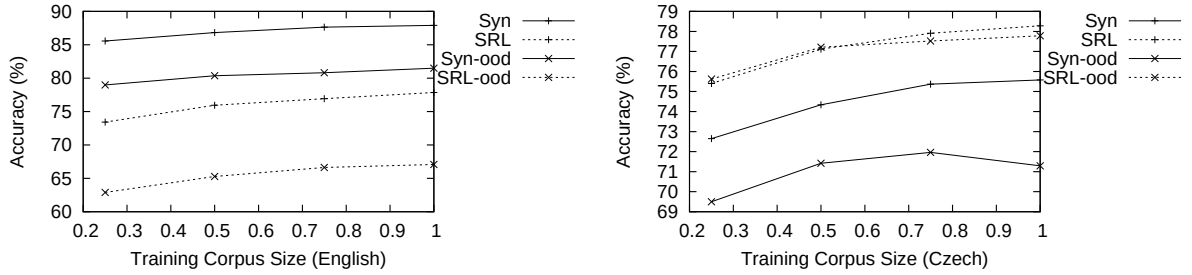

Figure 2: Learning curves of syntactic dependency parser and semantic role labeler on Czech and English datasets

Callmeier, U. (2002). Preprocessing and encoding techniques in PET. In S. Oepen, D. Flickinger, J. Tsujii, & H. Uszkoreit (Eds.), *Collaborative language engineering. A case study in efficient grammar-based processing.* Stanford, CA: CSLI Publications.

Copestake, A., Flickinger, D., Pollard, C., & Sag, I. A. (2005). Minimal Recursion Semantics. An introduction. *Journal of Research on Language and Computation*, *3*(4), 281 – 332.

Crysmann, B. (2005). Relative clause extraposition in German. An efficient and portable implementation. *Research on Language and Computation*, *3*(1), 61 – 82.

Flickinger, D. (2000). On building a more efficient grammar by exploiting types. *Natural Language Engineering*, *6 (1)*, 15 – 28.

Hajič, J., Ciaramita, M., Johansson, R., Kawahara, D., Martí, M. A., Màrquez, L., Meyers, A., Nivre, J., Padó, S., Štěpánek, J., Straňák, P., Surdeanu, M., Xue, N., & Zhang, Y. (2009). The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning.* Boulder, CO, USA.

Hajič, J., Panevová, J., Hajičová, E., Sgall, P., Pajas, P., Štěpánek, J., Havelka, J., Mikulová, M., & Žabokrtský, Z. (2006). *Prague Dependency Treebank 2.0* (Nos. Cat. No. LDC2006T01, ISBN 1-58563-370-4). Philadelphia, PA, USA: Linguistic Data Consortium.

Kawahara, D., Kurohashi, S., & Hasida, K. (2002). Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation* (pp. 2008–2013). Las Palmas, Canary Islands.

Malouf, R. (2002). A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the 6th conferencde on natural language learning (CoNLL 2002)* (pp. 49–55). Taipei, Taiwan.

Marimon, M., Bel, N., & Seghezzi, N. (2007). Test suite construction for a Spanish grammar. In T. H. King & E. M. Bender (Eds.), *Proceedings of the Grammar Engineering Across Frameworks workshop* (p. 250-264). Stanford, CA: CSLI Publications.

Oepen, S., & Carroll, J. (2000). Performance profiling for parser engineering. *Natural Language Engineering*, *6 (1)*, 81 – 97.

Palmer, M., Kingsbury, P., & Gildea, D. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, *31*(1), 71–106.

Palmer, M., & Xue, N. (2009). Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, *15*(1), 143–172.

Siegel, M., & Bender, E. M. (2002). Efficient deep processing of Japanese. In *Proceedings of the 3rd workshop on asian language resources and international standardization at the 19th international conference on computational linguistics.* Taipei, Taiwan.

Surdeanu, M., Johansson, R., Meyers, A., Màrquez, L., & Nivre, J. (2008). The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the 12th Conference on Computational Natural Language Learning.* Manchester, UK.

Taulé, M., Martí, M. A., & Recasens, M. (2008). AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation.* Marrakesh, Morroco.

Toutanova, K., Manning, C. D., Flickinger, D., & Oepen, S. (2005). Stochastic HPSG parse selection using the Redwoods corpus. *Journal of Research on Language and Computation*, *3*(1), 83 – 105.

Zhang, Y., Oepen, S., & Carroll, J. (2007). Efficiency in unification-based n-best parsing. In *Proceedings of the 10th International Conference on Parsing Technologies* (pp. 48 – 59). Prague, Czech Republic.

Zhang, Y., Wang, R., & Uszkoreit, H. (2008). Hybrid Learning of Dependency Structures from Heterogeneous Linguistic Resources. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL 2008)* (pp. 198–202). Manchester, UK.

A Latent Variable Model of Synchronous Syntactic-Semantic Parsing for Multiple Languages

Andrea Gesmundo
Univ Geneva
Dept Computer Sci
Andrea.Gesmundo@unige.ch

James Henderson
Univ Geneva
Dept Computer Sci
James.Henderson@unige.ch

Paola Merlo
Univ Geneva
Dept Linguistics
Paola.Merlo@unige.ch

Ivan Titov*
Univ Illinois at U-C
Dept Computer Sci
titov@uiuc.edu

Abstract

Motivated by the large number of languages (seven) and the short development time (two months) of the 2009 CoNLL shared task, we exploited latent variables to avoid the costly process of hand-crafted feature engineering, allowing the latent variables to induce features from the data. We took a pre-existing generative latent variable model of joint syntactic-semantic dependency parsing, developed for English, and applied it to six new languages with minimal adjustments. The parser's robustness across languages indicates that this parser has a very general feature set. The parser's high performance indicates that its latent variables succeeded in inducing effective features. This system was ranked third overall with a macro averaged F1 score of 82.14%, only 0.5% worse than the best system.

1 Introduction

Recent research in syntax-based statistical machine translation and the recent availability of syntactically annotated corpora for multiple languages (Nivre et al., 2007) has provided a new opportunity for evaluating the cross-linguistic validity of statistical models of syntactic structure. This opportunity has been significantly expanded with the 2009 CoNLL shared task on syntactic and semantic parsing of seven languages (Hajič et al., 2009) belonging to several different language families.

We participate in this task with a generative, history-based model proposed in the CoNLL 2008 shared task for English (Henderson et al., 2008) and further improved to tackle non-planar dependencies (Titov et al., 2009). This model maximises the joint probability of the syntactic and semantic dependencies and thereby enforces that the output structure be globally coherent, but the use of synchronous parsing allows it to maintain separate structures for the syntax and semantics. The probabilistic model is based on Incremental Sigmoid Belief Networks (ISBNs), a recently proposed latent variable model for syntactic structure prediction, which has shown very good performance for both constituency (Titov and Henderson, 2007a) and dependency parsing (Titov and Henderson, 2007b). The use of latent variables enables this architecture to be extended to learning a synchronous parse of syntax and semantics without overly restrictive assumptions about the linking between syntactic and semantic structures.

In this work, we evaluate the ability of this method to generalise across several languages. We take the model as it was developed for English, and apply it directly to all seven languages. The only fine-tuning was to evaluate whether to include one feature type which we had previously found did not help for English, but helped overall. No other feature engineering was done. The use of latent variables to induce features automatically from the data gives our method the adaptability necessary to perform well across all seven languages, and demonstrates the lack of language specificity in the models of Henderson et al. (2008) and Titov et al. (2009).

The main properties of this model, that differentiate it from other approaches, is the use of synchronous syntactic and semantic derivations and the

[0] Authors in alphabetical order.

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 37–42,
Boulder, Colorado, June 2009.

use of online planarisation of crossing semantic dependencies. This system was ranked third overall with a macro averaged F1 score of 82.14%, only 0.5% worse than the best system.

2 The Synchronous Model

The use of synchronous parsing allows separate structures for syntax and semantics, while still modeling their joint probability. We use the approach to synchronous parsing proposed in Henderson et al. (2008), where we start with two separate derivations specifying each of the two structures, then synchronise these derivations at each word. The individual derivations are based on Nivre's shift-reduce-style parsing algorithm (Nivre et al., 2006), as discussed further below. First we illustrate the high-level structure of the model, discussed in more detail in Henderson et al. (2008).

Let T_d be a syntactic dependency tree with derivation $D_d^1, ..., D_d^{m_d}$, and T_s be a semantic dependency graph with derivation $D_s^1, ..., D_s^{m_s}$. To define derivations for the joint structure T_d, T_s, we divide the two derivations into the chunks between shifting each word onto the stack, $c_d^t = D_d^{b_d^t}, ..., D_d^{e_d^t}$ and $c_s^t = D_s^{b_s^t}, ..., D_s^{e_s^t}$, where $D_d^{b_d^t-1} = D_s^{b_s^t-1} = \textit{Shift}_{t-1}$ and $D_d^{e_d^t+1} = D_s^{e_s^t+1} = \textit{Shift}_t$. Then the actions of the synchronous derivations consist of quadruples $C^t = (c_d^t, \textit{Switch}, c_s^t, \textit{Shift}_t)$, where *Switch* means switching from syntactic to semantic mode. This gives us the following joint probability model, where n is the number of words in the input.

$$P(T_d, T_s) = \prod_{t=1}^{n} P(C^t | C^1, \ldots, C^{t-1}) \quad (1)$$

These synchronous derivations $C^1, \ldots, C^n$ only require a single input queue, since the *Shift* actions are synchronised, but they require two separate stacks, one for the syntactic derivation and one for the semantic derivation.

The probability of each synchronous derivation chunk C^t is the product of four factors, related to the syntactic level, the semantic level and the two synchronising steps. The probability of c_d^t is decomposed into one probability for each derivation action D^i, conditioned on its history using the chain rule, and likewise for c_s^t. These probabilities are estimated using the method described in section 3.

	Syn cross	Sem cross	Sem tree	No parse
Cat	0%	0%	61.4%	0%
Chi	0%	28.0%	28.6%	9.5%
Cze	22.4%	16.3%	6.1%	1.8%
Eng	7.6%	43.9%	21.4%	3.9%
Ger	28.1%	1.3%	97.4%	0.0%
Jap	0.9%	38.3%	11.2%	14.4%
Spa	0%	0%	57.1%	0%

Table 1: For each language, percentage of training sentences with crossing arcs in syntax and semantics, with semantic arcs forming a tree, and which were not parsable using the *Swap* action.

One of the main characteristics of our synchronous representation, unlike other synchronous representations of syntax and semantics (Nesson et al., 2008), is that the synchronisation is done on words, rather than on structural components. We take advantage of this freedom and adopt different methods for handling crossing arcs for syntax and for semantics.

While both syntax and semantics are represented as dependency graphs, these graphs differ substantially in their properties. Some statistics which indicate these differences are shown in table 1. For example, English syntactic dependencies form trees, while semantic dependency structures are only trees 21.4% of the time, since in general each structure does not form a connected graph and some nodes may have more than one parent. The syntactic dependency structures for only 7.6% of English sentences contain crossing arcs, while 43.9% of the semantic dependency structures contain crossing arcs. Due to variations both in language characteristics and annotation decisions across corpora, these differences between syntax and semantics vary across the seven languages, but they are consistent enough to motivate the development of new techniques specifically for handling semantic dependency structures. In particular, we use a different method for parsing crossing arcs.

For parsing crossing semantic arcs (i.e. non-planar graphs), we use the approach proposed in Titov et al. (2009), which introduces an action *Swap* that swaps the top two elements on the parser's stack. The *Swap* action allows the parser to reorder words online during the parse. This allows words to be processed in different orders during different

portions of the parse, so some arcs can be specified using one ordering, then other arcs can be specified using another ordering. Titov et al. (2009) found that only using the *Swap* action as a last resort is the best strategy for English (compared to using it preemptively to address future crossing arcs) and we use the same strategy here for all languages.

Syntactic graphs do not use a *Swap* action. We adopt the HEAD method of Nivre and Nilsson (2005) to de-projectivise syntactic dependencies outside of parsing.[1]

3 Features and New Developments

The synchronous derivations described above are modelled with a type of Bayesian Network called an Incremental Sigmoid Belief Network (ISBN) (Titov and Henderson, 2007a). As in Henderson et al. (2008), the ISBN model distinguishes two types of latent states: syntactic states, when syntactic decisions are considered, and semantic states, when semantic decision are considered. Latent states are vectors of binary latent variables, which are conditioned on variables from previous states via a pattern of connecting edges determined by the previous decisions. These latent-to-latent connections are used to engineer soft biases which reflect the relevant domains of locality in the structure being built. For these we used the set of connections proposed in Titov et al. (2009), which includes latent-to-latent connections both from syntax states to semantics states and vice versa. The latent variable vectors are also conditioned on a set of observable features of the derivation history. For these features, we start with the feature set from Titov et al. (2009), which extends the semantic features proposed in Henderson et al. (2008) to allow better handling of the non-planar structures in semantics. Most importantly, all the features previously included for the top of the stack were also included for the word just under the top of the stack. To this set we added one more type of feature, discussed below.

We made some modifications to reflect differences in the task definition between the 2008 and 2009 shared tasks, and experimented with one type of features which had been previously implemented. For the former modifications, the system was adapted to allow the use of the PFEAT and FILLPRED fields in the data, which both resulted in improved accuracy for all the languages. The PFEAT data field (automatically predicted morphological features) was introduced in the system in two ways, as an atomic feature bundle that is predicted when predicting the word, and split into its elementary components when conditioning on a previous word, as was done in Titov and Henderson (2007b). Because the testing data included a specification of which words were annotated as predicates (the FILLPRED data field), we constrained the parser's output so as to be consistent with this specification. For rare predicates, if the predicate was not in the parser's lexicon (extracted from the training set), then a sense was taken from the list of senses reported in the Lexicon and Frame Set resources available for the closed challenge. If this information was not available, then a default sense was constructed based on the automatically predicted lemma (PLEMMA) of the predicate.

We also made use of a previously implemented type of feature that allows the prediction of a semantic link between two words to be conditioned on the syntactic dependency already predicted between the same two words. While this feature had previously not helped for English, it did result in an overall improvement across the languages.

Also, in comparison with previous experiments, the search beam used in the decoding phase was increased from 50 to 80, producing a small improvement in the overall development score.

All development effort took about two person-months, mostly by someone who had no previous experience with the system. Most of this time was spent on the above differences in the task definition between the 2008 and 2009 shared tasks.

4 Results and Discussion

We participated in the joint task of the closed challenge, as described in Hajič et al. (2009). The datasets used in this challenge are described in Taulé et al. (2008) (Catalan and Spanish), Palmer and Xue (2009) (Chinese), Hajič et al. (2006) (Czech), Surdeanu et al. (2008) (English), Burchardt et al. (2006) (German), and Kawahara et al. (2002) (Japanese).

[1] The statistics in Table 1 suggest that, for some languages, swapping might be beneficial for syntax as well.

	Rank	Average	Catalan	Chinese	Czech	English	German	Japanese	Spanish
macro F1	3	82.14	82.66	76.15	83.21	86.03	79.59	84.91	82.43
syntactic acc	1	@85.77	@87.86	76.11	@80.38	88.79	87.29	92.34	@87.64
semantic F1	3	78.42	77.44	76.05	86.02	83.24	71.78	77.23	77.19

Table 2: The three main scores for our system. Rank is within task.

	Rank	Ave	Cze-ood	Eng-ood	Ger-ood
macro F1	3	75.93	@80.70	75.76	71.32
syn Acc	2	78.01	@76.41	80.84	76.77
sem F1	3	73.63	84.99	70.65	65.25

Table 3: Results on out-of-domain for our system. Rank is within task.

The official results on the testing set are shown in tables 2, 3, and 4. The symbol "@" indicates the best result across systems. In table 5, we show our rankings across the different datasets, amongst systems submitted for the same task.

The overall score used to rank systems is the unweighted average of the syntactic labeled accuracy and the semantic labeled F1 measure, across all languages ("macro F1" in table 2). We were ranked third, out of 14 systems. There was only a 0.5% difference between our score and that of the best system, while there was a 1.29% difference between our score and the fourth ranked system. Only considering syntactic accuracy, we had the highest average score of all systems, with the highest individual score for Catalan, Czech, and Spanish. Only considering semantic F1, we were again ranked third. Our results for out-of-domain data (table 3) achieved a similar level of success, although here we were ranked second for average syntactic accuracy. Our precision on semantic arcs was generally much better than our recall (shown in table 4). However, other systems had a similar imbalance, resulting in no change in our third place ranking for semantic precision and for semantic recall. Only when the semantic precision is averaged with syntactic accuracy do we squeeze into second place ("macro Prec").

To get a more detailed picture of the strengths and weaknesses of our system, we computed its rank within each dataset, shown in table 5. Overall, our system is robust across languages, with little fluctuation in ranking for the overall score, including for out-of-domain data. The one noticeable exception to this consistency is the syntactic score for English out-of-domain data. The other ranks for English out-of-domain and English in-domain scores are also on the poor side. These results support our claim that our parser has not undergone much hand-tuning, since it was originally developed for English. It is not currently clear whether this relative difference reflects a English-specific weakness in our system, or that many of the other systems have been fine-tuned for English.

	data	time (min)	macro F1
Czech	25%	5007	73.84
	50%	3699	77.57
	75%	4201	79.10
	100%	6870	80.55
English	25%	1300	79.02
	50%	1899	81.61
	75%	3196	82.41
	100%	3191	83.27

Table 6: Training times and development set accuracies using different percentages of the training data, for Czech and English.

On the higher end of our dataset rankings, we do relatively well on Catalan, Czech, and Spanish. Catalan and Spanish are unique amongst these datasets in that they have no crossing arcs in their semantic structure. Czech seems to have semantic structures which are relatively well handled by our derivations with *Swap*. As indicated above in table 1, only 2% of sentences are unparsable, despite 16% requiring the *Swap* action. However, this argument does not explain why our parser did relatively poorly on German semantic dependencies. Regardless, these observations would suggest that our system is still having trouble with crossing dependencies, despite the introduction of the *Swap* operation, and that our learning method could achieve better performance with an improved treatment of crossing semantic dependencies.

Table 6 shows how accuracies and training times vary with the size of the training dataset, for Czech and English. Training times vary in part because

	Rank	Ave	Cat	Chi	Cze	Eng	Ger	Jap	Spa	Cze-ood	Eng-ood	Ger-ood
semantic Prec	3	81.60	79.08	80.93	87.45	84.92	75.60	83.75	79.44	85.90	72.89	75.19
semantic Rec	3	75.56	75.87	71.73	@84.64	81.63	68.33	71.65	75.05	@84.09	68.55	57.63
macro Prec	2	83.68	83.47	78.52	83.91	86.86	81.44	88.05	83.54	81.16	76.86	@75.98
macro Rec	3	80.66	@81.86	73.92	@82.51	85.21	77.81	81.99	81.35	@80.25	74.70	67.20

Table 4: Semantic precision and recall and macro precision and recall for our system. Rank is within task.

Rank by	Ave	Cat	Chi	Cze	Eng	Ger	Jap	Spa	Ave-ood	Cze-ood	Eng-ood	Ger-ood
macro F1	3	2	3	2	4	4	3	2	3	1	4	3
syntactic Acc	1	1	4	1	3	2	2	1	2	1	7	2
semantic F1	3	2	4	2	4	5	4	2	3	2	4	3

Table 5: Our system's rank within task according to the three main measures, for each dataset.

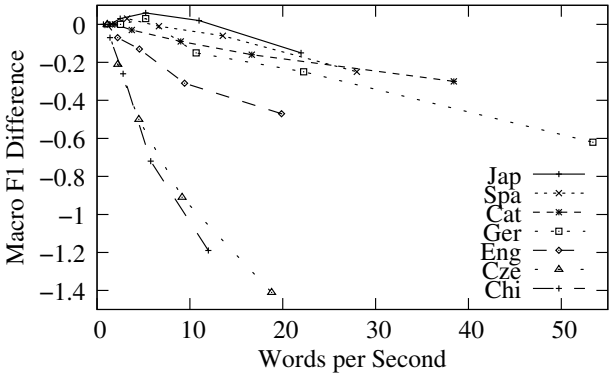

Figure 1: Difference in development set macro F1 as the search beam is decreased from the submitted beam (80) to 40, 20, 10, and 5, plotted against parser speed.

random variations can result in different numbers of training cycles before convergence. Accuracies appear to be roughly log-linear with data size.

Figure 1 shows how the accuracy of the parser degrades as we speed it up by decreasing the search beam used in decoding, for each language. For some languages, a slightly smaller search beam is actually more accurate,[2] but for smaller beams the trade-off of accuracy versus words-per-second is roughly linear. Parsing time per word is also linear in beam width, with a zero intercept.

5 Conclusion

In the joint task of the closed challenge of the CoNLL 2009 shared task (Hajič et al., 2009), we investigated how well a model of syntactic-semantic dependency parsing developed for English would generalise to the other six languages. This model provides a single generative probability of the joint syntactic and semantic dependency structures, but allows separate representations for these two structures by parsing the two structures synchronously. Finding the statistical correlations both between and within these structures is facilitated through the use of latent variables, which induce features automatically from the data, thereby greatly reducing the need for hand-coded feature engineering.

This latent variable model proved very robust across languages, achieving a ranking of between second and fourth on each language, including for out-of-domain data. The extent to which the parser does not rely on hand-crafting is underlined by the fact that its worst ranking is for English, the language for which it was developed (particularly for out-of-domain data). The parser was ranked third overall out of 14 systems, with a macro averaged F1 score of 82.14%, only 0.5% worse than the best system.

Both joint learning and conditioning decisions about semantic dependencies on latent representations of syntactic parsing states were crucial to the success of our model, as was previously demonstrated in Henderson et al. (2008). There, removing this conditioning led to a 3.5% drop in the SRL score. This result seems to contradict the general trend in the CoNLL-2008 shared task, where joint learning had only limited success. The latter fact may be explained by recent theoretical results demonstrating that pipelines can be preferable to joint learning (Roth et al., 2009) when no shared hidden representation is learnt. Our system (Henderson et al., 2008) was the only one which attempted to

[2]This fact suggests that we could have gotten improved results by tailoring the search beam to individual languages.

learn a common hidden representation for this multitask learning problem and also was the only one which achieved significant gain from joint parameter estimation. We believe that learning shared hidden representations for related NLP problems is a very promising direction for further research.

Acknowledgements

We thank Gabriele Musillo and Dan Roth for help and advice. This work was partly funded by Swiss NSF grants 100015-122643 and PBGE22-119276, European Community FP7 grant 216594 (CLASSiC, www.classic-project.org), US NSF grant SoD-HCER-0613885 and DARPA (Bootstrap Learning Program).

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

James Henderson, Paola Merlo, Gabriele Musillo, and Ivan Titov. 2008. A latent variable model of synchronous parsing for syntactic and semantic dependencies. In *Proceedings of CONLL 2008*, pages 178–182, Manchester, UK.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

Rebecca Nesson, Giorgio Satta, and Stuart M. Shieber. 2008. Optimal k-arization of synchronous tree-adjoining grammar. In *Proceedings of ACL-08: HLT*, pages 604–612, Columbus, Ohio, June.

Joakim Nivre and Jens Nilsson. 2005. Pseudo-projective dependency parsing. In *Proc. 43rd Meeting of Association for Computational Linguistics*, pages 99–106, Ann Arbor, MI.

Joakim Nivre, Johan Hall, Jens Nilsson, Gulsen Eryigit, and Svetoslav Marinov. 2006. Pseudo-projective dependency parsing with support vector machines. In *Proc. of the Tenth Conference on Computational Natural Language Learning*, pages 221–225, New York, USA.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 915–932, Prague, Czech Republic, June.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

Dan Roth, Kevin Small, and Ivan Titov. 2009. Sequential learning of classifiers for structured prediction problems. In *AISTATS*, Clearwater, Florida, USA.

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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco.

Ivan Titov and James Henderson. 2007a. Constituent parsing with Incremental Sigmoid Belief Networks. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 632–639, Prague, Czech Republic.

Ivan Titov and James Henderson. 2007b. Fast and robust multilingual dependency parsing with a generative latent variable model. In *Proc. Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, Prague, Czech Republic. (CoNLL Shared Task).

Ivan Titov, James Henderson, Paola Merlo, and Gabriele Musillo. 2009. Online graph planarisation for synchronous parsing of semantic and syntactic dependencies. In *Proc. Twenty-First International Joint Conference on Artificial Intelligence (IJCAI-09)*, Pasadena, California.

Multilingual Semantic Role Labeling

Anders Björkelund **Love Hafdell** **Pierre Nugues**
Department of Computer Science, Lund University
S-221 00 Lund, Sweden
fte04abj@student.lth.se
love_hafdell@hotmail.com
Pierre.Nugues@cs.lth.se

Abstract

This paper describes our contribution to the semantic role labeling task (SRL-only) of the CoNLL-2009 shared task in the closed challenge (Hajič et al., 2009). Our system consists of a pipeline of independent, local classifiers that identify the predicate sense, the arguments of the predicates, and the argument labels. Using these local models, we carried out a beam search to generate a pool of candidates. We then reranked the candidates using a joint learning approach that combines the local models and proposition features.

To address the multilingual nature of the data, we implemented a feature selection procedure that systematically explored the feature space, yielding significant gains over a standard set of features. Our system achieved the second best semantic score overall with an average labeled semantic F1 of 80.31. It obtained the best F1 score on the Chinese and German data and the second best one on English.

1 Introduction

In this paper, we describe a three-stage analysis approach that uses the output of a dependency parser and identifies the arguments of the predicates in a sentence. The first stage consists of a pipeline of independent classifiers. We carried out the predicate disambiguation with a set of greedy classifiers, where we applied one classifier per predicate lemma. We then used a beam search to identify the arguments of each predicate and to label them, yielding a pool of candidate propositions. The second stage consists of a reranker that we applied to the candidates using the local models and proposition features. We combined the score of the greedy classifiers and the reranker in a third stage to select the best candidate proposition. Figure 1 shows the system architecture.

We evaluated our semantic parser on a set of seven languages provided by the organizers of the CoNLL-2009 shared task: Catalan and Spanish (Taulé et al., 2008), Chinese (Palmer and Xue, 2009), Czech (Hajič et al., 2006), English (Surdeanu et al., 2008), German (Burchardt et al., 2006), and Japanese (Kawahara et al., 2002). Our system achieved an average labeled semantic F1 of 80.31, which corresponded to the second best semantic score overall. After the official evaluation was completed, we discovered a fault in the training procedure of the reranker for Spanish. The revised average labeled semantic F1 after correction was 80.80.

2 SRL Pipeline

The pipeline of classifiers consists of a predicate disambiguation (PD) module, an argument identification module (AI), and an argument classification (AC) module. Aside from the lack of a predicate identification module, which was not needed, as predicates were given, this architecture is identical to the one adopted by recent systems (Surdeanu et al., 2008), as well as the general approach within the field (Gildea and Jurafsky, 2002; Toutanova et al., 2005).

We build all the classifiers using the L2-regularized linear logistic regression from the LIBLINEAR package (Fan et al., 2008). The package implementation makes models very fast to train and

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 43–48,
Boulder, Colorado, June 2009.

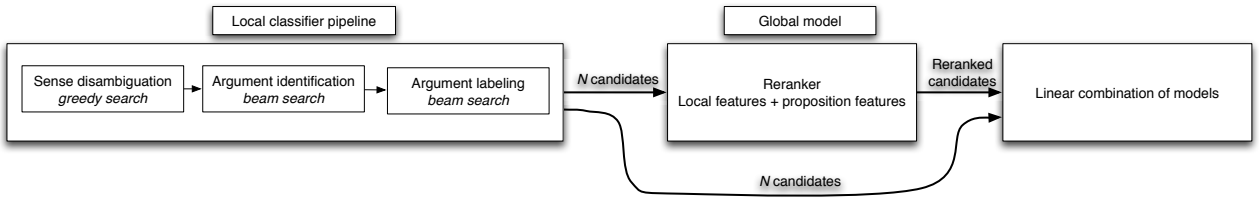

Figure 1: System architecture.

use for classification. Since models are logistic, they produce an output in the form of probabilities that we use later in the reranker (see Sect. 3).

2.1 Predicate Disambiguation

We carried out a disambiguation for all the lemmas that had multiple senses in the corpora and we trained one classifier per lemma. We did not use the predicate lexicons and we considered lemmas with a unique observed sense as unambiguous.

English required a special processing as the sense nomenclature overlapped between certain nominal and verbal predicates. For instance, the nominal predicate *plan.01* and the verbal predicate *plan.01* do not correspond to the same semantic frame. Hence, we trained two classifiers for each lemma *plan* that could be both a nominal and verbal predicate.

Table 1: Feature sets for predicate disambiguation.

	ca	**ch**	**cz**	**en**	**ge**	**sp**
PredWord	•		•			•
PredPOS	•			•		
PredDeprel		•	•	•		
PredFeats			•		•	•
PredParentWord	•	•	•	•	•	
PredParentPOS		•		•	•	
PredParentFeats			•		•	
DepSubCat	•	•	•	•	•	
ChildDepSet	•	•	•	•	•	•
ChildWordSet	•	•	•	•	•	•
ChildPOSSet		•	•	•	•	•

2.2 Argument Identification and Classification

We implemented the argument identification and classification as two separate stages, because it enabled us to apply and optimize different feature sets in each step. Arguments were identified by means of a binary classifier. No pruning was done, each word in the sentence was considered as a potential argument to all predicates of the same sentence.

Arguments were then labeled using a multiclass classifier; each class corresponding to a certain label. We did not apply any special processing with multiple dependencies in Czech and Japanese. Instead, we concatenated the composite labels (i.e. double edge) to form unique labels (i.e. single edge) having their own class.

2.3 Identification and Classification Features

For the English corpus, we used two sets of features for the nominal and the verbal predicates both in the AI and AC steps. This allowed us to create different classifiers for different kinds of predicates. We extended this approach with a default classifier catching predicates that were wrongly tagged by the POS tagger. For both steps, we used the union of the two feature sets for this catch-all class.

We wanted to employ this procedure with the two other languages, Czech and Japanese, where predicates had more than one POS type. As feature selection (See Sect. 2.4) took longer than expected, particularly in Czech due to the size of the corpus and the annotation, we had to abandon this idea and we trained a single classifier for all POS tags in the AI and AC steps.

For each data set, we extracted sets of features similar to the ones described by Johansson and Nugues (2008). We used a total of 32 features that we denote with the prefixes: Pred-, PredParent-, Arg-, Left-, Right-, LeftSibling-, and RightSibling- for, respectively, the predicate, the parent of the predicate, the argument, the leftmost and rightmost dependents of the argument, and the left and right

Table 2: Feature sets for argument identification and classification.

	Argument identification							Argument classification						
	ca	**ch**	**cz**	**en**	**ge**	**ja**	**sp**	**ca**	**ch**	**cz**	**en**	**ge**	**ja**	**sp**
PredWord									•		N		•	
PredPOS				N	•				•		V		•	
PredLemma				N	•			•	•	•	N,V	•		•
PredDeprel														
Sense		•	•	V	•			•	•	•	N,V	•	•	•
PredFeats			•							•		•		
PredParentWord				V					•		V		•	
PredParentPOS				V							V	•		
PredParentFeats			•											
DepSubCat	•	•												
ChildDepSet	•				•			•	•		V	•	•	•
ChildWordSet				N						•			•	
ChildPOSSet		•							•		N		•	
ArgWord	•	•		N,V	•	•	•	•	•	•	N,V	•	•	•
ArgPOS	•	•		N,V	•	•	•	•	•	•	N,V		•	
ArgFeats	•							•		•		•		•
ArgDeprel	•	•	•	V	•		•	•	•	•	V	•		•
DeprelPath	•	•	•	N,V	•	•	•	•	•		V	•		
POSPath	•	•	•	N,V	•	•	•			•	V	•	•	
Position			•	N,V		•		•	•	•	N,V		•	•
LeftWord	•				•			•	•		N		•	•
LeftPOS						•			•		V			
LeftFeats						•	•					•		
RightWord	•			N				•	•		N,V			•
RightPOS				N				•	•		N,V			•
RightFeats								•						•
LeftSiblingWord	•						•	•	•		N		•	
LeftSiblingPOS					•	•		•	•		N,V		•	
LeftSiblingFeats			•					•				•		
RightSiblingWord	•	•		V	•	•	•		•			•		•
RightSiblingPOS												•	•	
RightSiblingFeats													•	

sibling of the argument. The suffix of these names corresponds to the column name of the CoNLL format, except Word which corresponds to the Form column. Additional features are:

- Sense: the value of the Pred column, e.g. *plan.01*.
- Position: the position of the argument with respect to the predicate, i.e. before, on, or after.
- DepSubCat: the subcategorization frame of the predicate, e.g. OBJ+OPRD+SUB.
- DeprelPath: the path from predicate to argument concatenating dependency labels with the direction of the edge, e.g. OBJ↑OPRD↓SUB↓.
- POSPath: same as DeprelPath, but dependency labels are exchanged for POS tags, e.g. NN↑NNS↓NNP↓.
- ChildDepSet: the set of dependency labels of the children of the predicate, e.g. {OBJ, SUB}.
- ChildPOSSet: the set of POS tags of the children of the predicate, e.g. {NN, NNS}.
- ChildWordSet: the set of words (Form) of the children of the predicate, e.g. {*fish, me*}.

2.4 Feature Selection

We selected the feature sets using a greedy forward procedure. We first built a set of single features and, to improve the separability of our linear classifiers, we paired features to build bigrams. We searched the space of feature bigrams using the same procedure. See Johansson (2008, page 83), for a complete description. We intended to carry out a cross-validation search. Due to the lack of time, we resorted to using 80% of the training set for training and 20% for evaluating the features. Table 2 contains the complete list of single features we used. We omitted the feature bigrams.

Feature selection turned out to be a massive task. It took us three to four weeks searching the feature spaces, yet in most cases we were forced to interrupt the selection process after a few bigram features in order to have our system ready in time. This means that our feature sets can probably be further optimized.

When the training data was initially released, we used the exact feature set from Johansson and Nugues (2008) to compute baseline results on the development set for all the languages. After feature selection, we observed an increase in labeled semantic F1 close to 10% in most languages.

2.5 Applying Beam Search

The AI module proceeds left to right considering each word as an argument of the current predicate. The current partial propositions are scored by computing the product of the probabilities of all the words considered so far. After each word, the current pool of partial candidates is reduced to the beam size, k, and at the end of the sentence, the top k scoring propositions are passed on to the AC module.

Given k unlabeled propositions, the AC module applies a beam search on each of these propositions independently. This is done in a similar manner, proceeding from left to right among the identified arguments, keeping the l best labelings in its beam, and returning the top l propositions, when all identified arguments have been processed. This yields $n = k \times l$ complete propositions, unless one of the unlabeled propositions has zero arguments, in which case we have $n = (k - 1) \times l + 1$.

The probability of a labeled proposition according to the local pipeline is given by $P_{Local} = P_{AI} \times P_{AC}$, where P_{AI} and P_{AC} is the output probability from the AI and AC modules, respectively. In the case of empty propositions, P_{AC} was set to 1.

3 Global Reranker

We implemented a global reranker following Toutanova et al. (2005). To generate training examples for the reranker, we trained m AI and AC classifiers by partitioning the training set in m parts and using $m - 1$ of these parts for each AI and AC classifier, respectively.

We applied these AI and AC classifiers on the part of the corpus they were *not* trained on and we then generated the top n propositions for each predicate. We ran the CoNLL evaluation script on the propositions and we marked the top scoring one(s) as positive. We marked the others negative. If the correct proposition was not in the pool of candidates, we added it as an *extra* positive example. We used these positive and negative examples as training data for the global reranker.

3.1 Reranker Features

We used all the features from the local pipeline for all the languages. We built a vector where the AI features were prefixed with AI- and the AC features prefixed with $lab-$, where lab was any of the argument labels.

We added one proposition feature to the concatenation of local features, namely the sequence of core argument labels, e.g. *A0+plan.01+A1*. In Catalan and Spanish, we considered all the labels prefixed by arg0, arg1, arg2, or arg3 as core labels. In Chinese and English, we considered only the labels A0, A1, A2, A3, and A4. In Czech, German, and Japanese, we considered all the labels as core labels.

Hence, the total size of the reranker vector space is $|AI| + |L| \times |AC| + |G|$, where $|AI|$ and $|AC|$ denotes the size of the AI and AC vector spaces, respectively, $|L|$ corresponds to the number of labels, and $|G|$ is the size of additional global features.

We ran experiments with the grammatical voice that we included in the string representing the sequence of core argument labels, e.g. *A1+plan.01/Passive+A0*. The voice was derived by hand-crafted rules in Catalan, English, German, and

Spanish, and given in the Feat column in Czech. However, we did not notice any significant gain in performance. The hand-crafted rules use lexical forms and dependencies, which we believe classifiers are able to derive themselves using the local model features. This also applies to Czech, as PredFeats was a feature used in the local pipeline, both in the AI and AC steps.

3.2 Weighting the Models

In Sect. 2.5, we described how the pipeline was used to generate the top n propositions, each with its own local probability P_{Local}. Similar to softmax, we normalized these local probabilities by dividing each of them by their total sum. We denote this normalized probability by P'_{Local}. The reranker gives a probability on the complete proposition, $P_{Reranker}$. We weighted these probabilities and chose the proposition maximizing $P_{Final} = (P'_{Local})^{\alpha} \times P_{Reranker}$. This is equivalent to a linear combination of the log probabilities.

3.3 Parameters Used

For the submission to the CoNLL 2009 Shared Task, we set the beam widths to $k = l = 4$, yielding candidate pools of size $n = 13$ or $n = 16$ (See Section 2.5). We used $m = 5$ for training the reranker and $\alpha = 1$ for combining the local model with the reranker.

4 Results

Our system achieved the second best semantic score, all tasks, with an average labeled semantic F1 of 80.31. It obtained the best F1 score on the Chinese and German data and the second best on English. Our system also reached the third rank in the out-of-domain data, all tasks, with a labeled semantic F1 of 74.38. Post-evaluation, we discovered a bug in the Spanish reranker model causing the poor results in this language. After correcting this, we could reach a labeled semantic F1 of 79.91 in Spanish. Table 3 shows our official results in the shared task as well as the post-evaluation update.

We also compared the performance of a greedy strategy with that of a global model. Table 4 shows these figures with post-evaluation figures in Spanish. Table 5 shows the training time, parsing time, and the parsing speed in predicates per second. These figures correspond to complete execution time of parsing, including loading models into memory, i.e. a constant overhead, that explains the low parsing speed in German. We implemented our system to be flexible for easy debugging and testing various ideas. Optimizing the implementation would reduce execution times significantly.

Table 3: Summary of submitted results: closed challenge, semantic F1. * denotes the post-evaluation results obtained for Spanish after a bug fix.

	Unlabeled	**Labeled**
Catalan	93.60	80.01
Chinese	84.76	78.60
Czech	92.63	85.41
English	91.17	85.63
German	92.13	79.71
Japanese	83.45	76.30
Spanish	92.69	76.52
Spanish*	93.76	79.91
Average	90.06	80.31
Average*	90.21	80.80

Table 4: Improvement of reranker. * denotes the post-evaluation results obtained for Spanish after a bug fix.

	Greedy	**Reranker**	**Gain**
Catalan	79.54	80.01	0.47
Chinese	77.84	78.60	0.76
Czech	84.99	85.41	0.42
English	84.44	85.63	1.19
German	79.01	79.71	0.70
Japanese	75.61	76.30	0.69
Spanish	79.28	76.52	-2.76
Spanish*	79.28	79.91	0.63
Average	80.10	80.31	0.21
Average*	80.10	80.80	0.70

5 Conclusion

We have built and described a streamlined and effective semantic role labeler that did not use any lexicons or complex linguistic features. We used a generic feature selection procedure that keeps language adaptation minimal and delivers a relatively even performance across the data sets. The system is

Table 5: Summary of training and parsing times on an Apple Mac Pro, 3.2 GHz.

	Training (min)	**Parsing (Greedy)** (min:sec)	**Speed (Greedy)** (pred/sec)	**Parsing (Reranker)** (min:sec)	**Speed (Reranker)** (pred/sec)
Catalan	46	1:10	71	1:21	62
Chinese	139	2:35	79	3:45	55
Czech	299	18:47	40	33:49	22
English	421	6:25	27	8:51	20
German	15	0:21	26	0:22	25
Japanese	48	0:37	84	1:02	50
Spanish	51	1:15	69	1:47	48

robust and can handle incorrect syntactic parse trees with a good level of immunity. While input parse trees in Chinese and German had a labeled syntactic accuracy of 78.46 (Hajič et al., 2009), we could reach a labeled semantic F1 of 78.60 and 79.71 in these languages. We also implemented an efficient global reranker in all languages yielding a 0.7 average increase in labeled semantic F1. The reranker step, however, comes at the expense of parsing times increased by factors ranging from 1.04 to 1.82.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *Proceedings of the Shared Task Session of CoNLL-2008*.

Richard Johansson. 2008. *Dependency-based Semantic Analysis of Natural-language Text*. Ph.D. thesis, Lund University, December 5.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2005. Joint learning improves semantic role labeling. In *Proceedings of ACL-2005*.

Multilingual Dependency-based Syntactic and Semantic Parsing

Wanxiang Che, Zhenghua Li, Yongqiang Li, Yuhang Guo, Bing Qin, Ting Liu
Information Retrieval Lab
School of Computer Science and Technology
Harbin Institute of Technology, China, 150001
{car, lzh, yqli, yhguo, qinb, tliu}@ir.hit.edu.cn

Abstract

Our CoNLL 2009 Shared Task system includes three cascaded components: syntactic parsing, predicate classification, and semantic role labeling. A pseudo-projective high-order graph-based model is used in our syntactic dependency parser. A support vector machine (SVM) model is used to classify predicate senses. Semantic role labeling is achieved using maximum entropy (MaxEnt) model based semantic role classification and integer linear programming (ILP) based post inference. Finally, we win the first place in the joint task, including both the closed and open challenges.

1 System Architecture

Our CoNLL 2009 Shared Task (Hajič et al., 2009): multilingual syntactic and semantic dependencies system includes three cascaded components: syntactic parsing, predicate classification, and semantic role labeling.

2 Syntactic Dependency Parsing

We extend our CoNLL 2008 graph-based model (Che et al., 2008) in four ways:

1. We use bigram features to choose multiple possible syntactic labels for one arc, and decide the optimal label during decoding.

2. We extend the model with sibling features (McDonald, 2006).

3. We extend the model with grandchildren features. Rather than only using the left-most and right-most grandchildren as Carreras (2007) and Johansson and Nugues (2008) did, we use all left and right grandchildren in our model.

4. We adopt the pseudo-projective approach introduced in (Nivre and Nilsson, 2005) to handle the non-projective languages including Czech, German and English.

2.1 Syntactic Label Determining

The model of (Che et al., 2008) decided one label for each arc before decoding according to unigram features, which caused lower labeled attachment score (LAS). On the other hand, keeping all possible labels for each arc made the decoding inefficient. Therefore, in the system of this year, we adopt approximate techniques to compromise, as shown in the following formulas.

$$\mathbf{f}^{lbl}_{uni}(h,c,l) = \mathbf{f}^{lbl}_{1}(h,1,d,l) \cup \mathbf{f}^{lbl}_{1}(c,0,d,l)$$

$$L_1(h,c) = \arg\max^{K_1}_{l \in L}(\mathbf{w} \cdot \mathbf{f}^{lbl}_{uni}(h,c,l))$$

$$\mathbf{f}^{lbl}_{bi}(h,c,l) = \mathbf{f}^{lbl}_{2}(h,c,l)$$

$$L_2(h,c) = \arg\max^{K_2}_{l \in L_1(h,c)}(\mathbf{w} \cdot \{\mathbf{f}^{lbl}_{uni} \cup \mathbf{f}^{lbl}_{bi}\})$$

For each arc, we firstly use unigram features to choose the K_1-best labels. The second parameter of $\mathbf{f}^{lbl}_1(\cdot)$ indicates whether the node is the head of the arc, and the third parameter indicates the direction. L denotes the whole label set. Then we re-rank the labels by combining the bigram features, and choose K_2-best labels. During decoding, we only use the K_2 labels chosen for each arc ($K_2 \ll K_1 < |L|$).

2.2 High-order Model and Algorithm

Following the Eisner (2000) algorithm, we use spans as the basic unit. A span is defined as a substring of the input sentence whose sub-tree is already produced. Only the start or end words of a span can link with other spans. In this way, the algorithm parses the left and the right dependence of a word independently, and combines them in the later stage.

We follow McDonald (2006)'s implementation of first-order Eisner parsing algorithm by modifying its scoring method to incorporate high-order features. Our extended algorithm is shown in Algorithm 1.

There are four different span-combining operations. Here we explain two of them that correspond to right-arc ($s < t$), as shown in Figure 1 and 2. We

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 49–54,
Boulder, Colorado, June 2009.

Algorithm 1 High-order Eisner Parsing Algorithm

1: $C[s][s][c] = 0, 0 \leq s \leq N, c \in cp, icp$ # cp: complete; icp: incomplete
2: **for** $j = 1$ to N **do**
3: **for** $s = 0$ to N **do**
4: $t = s + j$L
5: **if** $t > N$ **then**
6: break
7: **end if**
 # Create incomplete spans
8: $C[s][t][icp] = \max_{s \leq r < t; l \in L_2(s,t)}(C[s][r][cp] + C[t][r+1][cp] + S_{icp}(s, r, t, l))$
9: $C[t][s][icp] = \max_{s \leq r < t; l \in L_2(t,s)}(C[s][r][cp] + C[t][r+1][cp] + S_{icp}(t, r, s, l))$
 # Create complete spans
10: $C[s][t][cp] = \max_{s < r \leq t; l = C[s][r][icp].label}(C[s][r][icp] + C[r][t][cp] + S_{cp}(s, r, t, l))$
11: $C[t][s][cp] = \max_{s \leq r < t; l = C[t][r][icp].label}(C[r][s][cp] + C[t][r][icp] + S_{cp}(t, r, s, l))$
12: **end for**
13: **end for**

follow the way of (McDonald, 2006) and (Carreras, 2007) to represent spans. The other two operations corresponding to left-arc are similar.

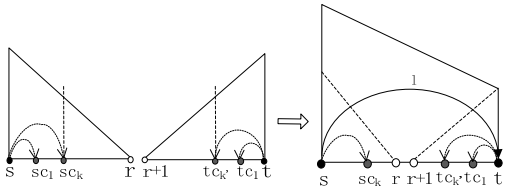

Figure 1: Combining two spans into an incomplete span

Figure 1 illustrates line 8 of the algorithm in Algorithm 1, which combines two complete spans into an incomplete span. A complete span means that only the head word can link with other words further, noted as "$\rightarrow$" or "$\leftarrow$". An incomplete span indicates that both the start and end words of the span will link with other spans in the future, noted as "$\dashrightarrow$" or "$\dashleftarrow$". In this operation, we combine two smaller spans, $sp_{s \rightarrow r}$ and $sp_{r+1 \leftarrow t}$, into $sp_{s \dashrightarrow t}$ with adding $arc_{s \rightarrow t}$. As shown in the following formulas, the score of $sp_{s \dashrightarrow t}$ is composed of three parts: the score of $sp_{s \rightarrow r}$, the score of $sp_{r+1 \leftarrow t}$, and the score of adding $arc_{s \rightarrow t}$. The score of $arc_{s \rightarrow t}$ is determined by four different feature sets: unigram features, bigram features, sibling features and left grandchildren features (or inside grandchildren features, meaning that the grandchildren lie between s and t). Note that the sibling features are only related to the nearest sibling node of t, which is denoted as sc_k here. And the inside grandchildren features are related to all the children of t. This is different from the models used by Carreras (2007) and Johansson and Nugues (2008). They only used the left-most child of t, which is $tc_{k'}$ here.

$$\mathbf{f}_{icp}(s, r, t, l) = \mathbf{f}_{uni}(s, t, l) \cup \mathbf{f}_{bi}(s, t, l) \cup \mathbf{f}_{sib}(s, sc_k, t) \cup \{\bigcup_{i=1}^{k'} \mathbf{f}_{grand}(s, t, tc_i, l)\}$$

$$S_{icp}(s, r, t, l) = \mathbf{w} \cdot \mathbf{f}_{icp}(s, r, t, l)$$

$$S(sp_{s \dashrightarrow t}) = S(sp_{s \rightarrow r}) + S(sp_{r+1 \leftarrow t}) + S_{icp}(s, r, t, l)$$

In Figure 2 we combine $sp_{s \dashrightarrow r}$ and $sp_{r \rightarrow t}$ into $sp_{s \rightarrow t}$, which explains line 10 in Algorithm 1. The score of $sp_{s \rightarrow t}$ also includes three parts, as shown in the following formulas. Although there is no new arc added in this operation, the third part is necessary because it reflects the right (or called outside) grandchildren information of $arc_{s \rightarrow r}$.

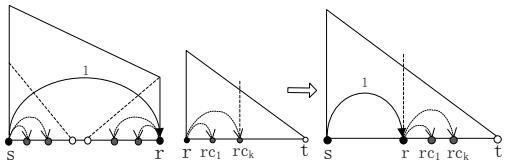

Figure 2: Combining two spans into a complete span

$$\mathbf{f}_{cp}(s, r, t, l) = \bigcup_{i=1}^{k} \mathbf{f}_{grand}(s, r, rc_i, l)$$

$$S_{cp}(s, r, t, l) = \mathbf{w} \cdot \mathbf{f}_{cp}(s, r, t, l)$$

$$S(sp_{s \rightarrow t}) = S(sp_{s \dashrightarrow r}) + S(sp_{r \rightarrow t}) + S_{cp}(s, r, t, l)$$

2.3 Features

As shown above, features used in our model can be decomposed into four parts: unigram features, bigram features, sibling features, and grandchildren features. Each part can be seen as two different sets: arc-related and label-related features, except sibling features, because we do not consider labels when using sibling features. Arc-related features can be understood as back-off of label-related features. Actually, label-related features are gained by simply attaching the label to the arc-features.

The unigram and bigram features used in our model are similar to those of (Che et al., 2008), except that we use bigram label-related features. The sibling features we use are similar to those of (McDonald, 2006), and the grandchildren features are similar to those of (Carreras, 2007).

3 Predicate Classification

The predicate classification is regarded as a supervised word sense disambiguation (WSD) task here. The task is divided into four steps:

1. Target words selection: predicates with multiple senses appearing in the training data are selected as target words.

2. Feature extraction: features in the context around these target words are extracted as shown in Table 4. The detailed explanation about these features can be found from (Che et al., 2008).

3. Classification: for each target word, a Support Vector Machine (SVM) classifier is used to classify its sense. As reported by Lee and Ng (2002) and Guo et al. (2007), SVM shows good performance on the WSD task. Here libsvm (Chang and Lin, 2001) is used. The linear kernel function is used and the trade off parameter C is 1.

4. Post processing: for each predicate in the test data which does not appear in the training data, its first sense in the frame files is used.

4 Semantic Role Labeling

The semantic role labeling (SRL) can be divided into two separate stages: semantic role classification (SRC) and post inference (PI).

During the SRC stage, a Maximum entropy (Berger et al., 1996) classifier is used to predict the probabilities of a word in the sentence to be each semantic role. We add a virtual role "NULL" (presenting none of roles is assigned) to the roles set, so we do not need semantic role identification stage anymore. For a predicate of each language, two classifiers (one for noun predicates, and the other for verb predicates) predict probabilities of each word in a sentence to be each semantic role (including virtual role "NULL"). The features used in this stage are listed in Table 4.

Language	No-duplicated-roles
Catalan	arg0-agt, arg0-cau, arg1-pat, arg2-atr, arg2-loc
Chinese	A0, A1, A2, A3, A4, A5,
Czech	ACT, ADDR, CRIT, LOC, PAT, DIR3, COND
English	A0, A1, A2, A3, A4, A5,
German	A0, A1, A2, A3, A4, A5,
Japanese	DE, GA, TMP, WO
Spanish	arg0-agt, arg0-cau, arg1-pat, arg1-tem, arg2-atr, arg2-loc, arg2-null, arg4-des, argL-null, argM-cau, argM-ext, argM-fin

Table 1: No-duplicated-roles for different languages

The probability of each word to be a semantic role for a predicate is given by the SRC stage. The results generated by selecting the roles with the largest probabilities, however, do not satisfy some constrains. As we did in the last year's system (Che et al., 2008), we use the ILP (Integer Linear Programming) (Punyakanok et al., 2004) to get the global optimization, which is satisfied with three constrains:

C1: Each word should be labeled with one and only one label (including the virtual label "NULL").

C2: Roles with a small probability should never be labeled (except for the virtual role "NULL"). The threshold we use in our system is 0.3.

C3: Statistics show that some roles (except for the virtual role "NULL") usually appear once for a predicate. We impose a no-duplicate-roles constraint with a no-duplicate-roles list, which is constructed according to the times of semantic roles' duplication for each single predicate. Table 1 shows the no-duplicate-roles for different languages.

Our maximum entropy classifier is implemented with Maximum Entropy Modeling Toolkit[1]. The classifier parameters are tuned with the development data for different languages respectively. lp_solve 5.5[2] is chosen as our ILP problem solver.

[1] http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html

[2] http://sourceforge.net/projects/lpsolve

5 Experiments

5.1 Experimental Setup

We participate in the CoNLL 2009 shared task with all 7 languages: Catalan (Taulé et al., 2008), Chinese (Palmer and Xue, 2009), Czech (Hajič et al., 2006), English (Surdeanu et al., 2008), German (Burchardt et al., 2006), Japanese (Kawahara et al., 2002), and Spanish (Taulé et al., 2008). Besides the closed challenge, we also submitted the open challenge results. Our open challenge strategy is very simple. We add the SRL development data of each language into their training data. The purpose is to examine the effect of the additional data, especially for out-of-domain (ood) data.

Three machines (with 2.5GHz Xeon CPU and 16G memory) were used to train our models. During the peak time, Amazon's EC2 (Elastic Compute Cloud)[3] was used, too. Our system requires 15G memory at most and the longest training time is about 36 hours.

During training the predicate classification (PC) and the semantic role labeling (SRL) models, golden syntactic dependency parsing results are used. Previous experiments show that the PC and SRL test results based on golden parse trees are slightly worse than that based on cross trained parse trees. It is, however, a pity that we have no enough time and machines to do cross training for so many languages.

5.2 Results and Discussion

In order to examine the performance of the ILP based post inference (PI) for different languages, we adopt a simple PI strategy as baseline, which selects the most likely label (including the virtual label "NULL") except for those duplicate non-virtual labels with lower probabilities (lower than 0.5). Table 2 shows their performance on development data.

We can see that the ILP based post inference can improve the precision but decrease the recall. Except for Czech, almost all languages are improved. Among them, English benefits most.

The final system results are shown in Table 3. Comparing with our CoNLL 2008 (Che et al., 2008) syntactic parsing results on English[4], we can see that our new high-order model improves about 1%.

[3] http://aws.amazon.com/ec2/

[4] devel: 85.94%, test: 87.51% and ood: 80.73%

	Precision	Recall	F1
Catalan simple	78.68	77.14	77.90
Catalan ILP	79.42	76.49	77.93
Chinese simple	80.74	74.36	77.42
Chinese ILP	81.97	73.92	77.74
Czech simple	88.54	84.68	86.57
Czech ILP	89.23	84.05	86.56
English simple	83.03	83.55	83.29
English ILP	85.63	83.03	84.31
German simple	78.88	75.87	77.34
German ILP	82.04	74.10	77.87
Japanese simple	88.04	70.68	78.41
Japanese ILP	89.23	70.16	78.56
Spanish simple	76.73	75.92	76.33
Spanish ILP	77.71	75.34	76.51

Table 2: Comparison between different PI strategies

For the open challenge, because we did not modify the syntactic training data, its results are the same as the closed ones. We can, therefore, examine the effect of the additional training data on SRL. We can see that along with the development data are added into the training data, the performance on the in-domain test data is increased. However, it is interesting that the additional data is harmful to the ood test.

6 Conclusion and Future Work

Our CoNLL 2009 Shared Task system is composed of three cascaded components. The pseudo-projective high-order syntactic dependency model outperforms our CoNLL 2008 model (in English). The additional in-domain (devel) SRL data can help the in-domain test. However, it is harmful to the ood test. Our final system achieves promising results. In the future, we will study how to solve the domain adaptive problem and how to do joint learning between syntactic and semantic parsing.

Acknowledgments

This work was supported by National Natural Science Foundation of China (NSFC) via grant 60803093, 60675034, and the "863" National High-Tech Research and Development of China via grant 2008AA01Z144.

		Syntactic Accuracy (LAS)			Semantic Labeled F1			Macro F1 Score		
		devel	test	ood	devel	test	ood	devel	test	ood
Catalan	closed	86.65	86.56	—	77.93	77.10	—	82.30	81.84	—
	open	—			—	77.36		—	81.97	
Chinese	closed	75.73	75.49	—	77.74	77.15	—	76.79	76.38	—
	open	—			—	77.23		—	76.42	
Czech	closed	80.07	80.01	76.03	86.56	86.51	85.26	83.33	83.27	80.66
	open	—			—	86.57	85.21	—	83.31	80.63
English	closed	87.09	88.48	81.57	84.30	85.51	73.82	85.70	87.00	77.71
	open	—			—	85.61	73.66	—	87.05	77.63
German	closed	85.69	86.19	76.11	77.87	78.61	70.07	81.83	82.44	73.19
	open	—			—	78.61	70.09	—	82.44	73.20
Japanese	closed	92.55	92.57	—	78.56	78.26	—	85.86	85.65	—
	open	—			—	78.35		—	85.70	
Spanish	closed	87.22	87.33	—	76.51	76.47	—	81.87	81.90	—
	open	—			—	76.66		—	82.00	
Average	closed	—	85.23	77.90	—	79.94	76.38	—	82.64	77.19
	open					80.06	76.32		82.70	77.15

Table 3: Final system results

References

Adam L. Berger, Stephen A. Della Pietra, and Vincent J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22.

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *LREC-2006*.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *EMNLP/CoNLL-2007*.

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*.

Wanxiang Che, Zhenghua Li, Yuxuan Hu, Yongqiang Li, Bing Qin, Ting Liu, and Sheng Li. 2008. A cascaded syntactic and semantic dependency parsing system. In *CoNLL-2008*.

Jason Eisner. 2000. Bilexical grammars and their cubic-time parsing algorithms. In *Advances in Probabilistic and Other Parsing Technologies*.

Yuhang Guo, Wanxiang Che, Yuxuan Hu, Wei Zhang, and Ting Liu. 2007. HIT-IR-WSD: A wsd system for english lexical sample task. In *SemEval-2007*.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *CoNLL-2009*.

Richard Johansson and Pierre Nugues. 2008. Dependency-based semantic role labeling of PropBank. In *EMNLP-2008*.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *LREC-2002*.

Yoong Keok Lee and Hwee Tou Ng. 2002. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In *EMNLP-2002*.

Ryan McDonald. 2006. *Discriminative Learning and Spanning Tree Algorithms for Dependency Parsing*. Ph.D. thesis, University of Pennsylvania.

Joakim Nivre and Jens Nilsson. 2005. Pseudo-projective dependency parsing. In *ACL-2005*.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1).

Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Coling-2004*.

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 *CoNLL-2008*.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *LREC-2008*.

	Catalan	Chinese	Czech	English	German	Japanese	Spanish
ChildrenPOS			♦ ◇				♦◇
ChildrenPOSNoDup				♦ ◇			♦ ◇
ConstituentPOSPattern	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇
ConstituentPOSPattern+DepRelation	♦ ◇		♦ ◇		♦ ◇		
ConstituentPOSPattern+DepwordLemma	♦ ◇		♦ ◇		♦ ◇		
ConstituentPOSPattern+HeadwordLemma		♦ ◇	♦ ◇	♦ ◇		♦ ◇	♦ ◇
DepRelation	▲ △ ♦ ◇	▲ △ ♦ ◇	▲ △ ♦ ◇	▲ △ ♦ ◇	▲ △	♦ ◇	▲ △ ♦ ◇
DepRelation+DepwordLemma	♦ ◇		♦ ◇				
DepRelation+Headword	▲ △	▲ △	▲	▲ △	▲ △		▲
DepRelation+HeadwordLemma	♦ ◇		♦ ◇		♦ ◇		♦ ◇
DepRelation+HeadwordLemma+DepwordLemma		♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇
DepRelation+HeadwordPOS	▲ △	▲ △	▲ △	▲ △	▲ △		▲
Depword	♦ ◇						♦ ◇
DepwordLemma	♦ ◇	♦ ◇	♦ ◇	♦ ◇		♦ ◇	♦ ◇
DepwordLemma+HeadwordLemma	♦ ◇				♦ ◇	♦ ◇	
DepwordLemma+RelationPath		♦ ◇	♦ ◇	♦ ◇		♦ ◇	♦ ◇
DepwordPOS	▲ △	▲ △	▲ △ ♦ ◇	▲ △	▲ △ ♦ ◇		▲ △
DepwordPOS+HeadwordPOS	♦ ◇		♦ ◇				
DownPathLength			♦ ◇				♦ ◇
FirstLemma	♦ ◇	♦ ◇	♦ ◇	♦ ◇		♦ ◇	♦ ◇
FirstPOS			♦ ◇		♦ ◇		
FirstPOS+DepwordPOS	♦ ◇		♦ ◇		♦ ◇		
FirstWord	♦ ◇				♦ ◇		
Headword	▲ △	▲ △	▲ △	▲ △	▲ △ ♦ ◇		▲
HeadwordLemma	▲ △ ♦ ◇	▲ △ ♦ ◇	▲ △ ♦ ◇	▲ △ ♦ ◇	▲ △ ♦ ◇	♦ ◇	▲
HeadwordLemma+RelationPath		♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇
HeadwordPOS	▲ △	▲ △	▲ △ ♦ ◇	▲ △ ♦ ◇	▲ △ ♦ ◇		▲ △
LastLemma	♦ ◇	♦ ◇	♦ ◇	♦ ◇		♦ ◇	
LastPOS	♦ ◇		♦ ◇				
LastWord	♦ ◇						
Path	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	
Path+RelationPath		♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	
PathLength		♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇
PFEAT	▲ △		▲ △				▲ △
PFEATSplit	▲ △ ♦ ◇		▲ △ ♦ ◇		▲ △ ♦ ◇		▲ △ ♦ ◇
PFEATSplitRemoveNULL	▲ △				▲ △		▲ △
PositionWithPredicate	♦ ◇	♦ ◇	♦ ◇	♦ ◇		♦ ◇	
Predicate	▲ △ ♦ ◇	▲ △	▲ △ ♦ ◇	▲ △	▲ △		▲ △ ♦ ◇
Predicate+PredicateFamilyship		♦ ◇	♦ ◇	♦ ◇		♦ ◇	♦ ◇
PredicateBagOfPOSNumbered	△	▲ △			▲ △		▲ △
PredicateBagOfPOSNumberedWindow5	▲ △	▲ △	▲	△	▲ △		▲ △
PredicateBagOfPOSOrdered	▲ △	▲ △	▲ △		▲ △		▲
PredicateBagOfPOSOrderedWindow5	▲ △	▲ △	▲ △	▲ △	▲ △		▲ △
PredicateBagOfPOSWindow5	▲	▲ △	▲ △	▲ △	▲ △		▲
PredicateBagOfWords		△	▲ △	▲	▲ △		▲ △
PredicateBagOfWordsAndIsDesOfPRED	▲ △	▲	△	△	▲ △		▲ △
PredicateBagOfWordsOrdered	△	▲ △	▲ △	△	▲ △		▲ △
PredicateChildrenPOS	▲ △ ♦ ◇	▲ △	▲ △	▲ △	▲ △		▲ △ ♦ ◇
PredicateChildrenPOSNoDup	▲ △	▲ △	▲ △	▲ △	▲ △		▲ △
PredicateChildrenREL	▲ △ ♦ ◇	▲ △	▲ △	▲ △	▲ △ ♦ ◇		▲ △
PredicateChildrenRELNoDup	▲ △ ♦ ◇	▲ △	▲ △	▲ △	▲ △ ♦ ◇		▲ △
PredicateFamilyship					♦ ◇		
PredicateLemma	▲ △ ♦ ◇	▲ △ ♦ ◇	▲ △ ♦ ◇	▲ △ ♦ ◇	▲ △ ♦ ◇	♦ ◇	▲ △ ♦ ◇
PredicateLemma+PredicateFamilyship	♦ ◇		♦ ◇		♦ ◇		
PredicateSense	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇
PredicateSense+DepRelation			♦ ◇		♦ ◇		
PredicateSense+DepwordLemma			♦ ◇		♦ ◇		
PredicateSense+DepwordPOS			♦ ◇		♦ ◇		
PredicateSiblingsPOS	▲ △	▲ △	▲	▲ △	▲ △		▲ △
PredicateSiblingsPOSNoDup	▲ △ ♦ ◇	▲ △	▲ △	▲ △	▲ △		▲ △ ♦ ◇
PredicateSiblingsREL	▲ △ ♦ ◇	▲ △	▲ △	▲ △	▲ △		▲ △
PredicateSiblingsRELNoDup	▲ △	▲ △ ♦ ◇	△	▲ △	▲ △ ♦ ◇		▲ △ ♦ ◇
PredicateVoiceEn				▲ △			
PredicateWindow5Bigram		▲ △	▲ △	▲ △			▲ △
PredicateWindow5BigramPOS	▲ △	▲ △	▲ △	▲ △	▲ △		▲ △
RelationPath	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇	♦ ◇
SiblingsPOS					♦ ◇		♦ ◇
SiblingsREL	♦						
SiblingsRELNoDup			♦ ◇		♦ ◇		
UpPath		♦ ◇	♦ ◇	♦ ◇		♦	
UpPathLength			♦ ◇				
UpRelationPath	♦ ◇				♦ ◇		♦ ◇
UpRelationPath+HeadwordLemma		♦ ◇	♦ ◇	♦ ◇		♦ ◇	

Table 4: Features that are used in predicate classification (PC) and semantic role labeling (SRL). ▲: noun predicate PC, △: verb predicate PC, ♦: noun predicate SRL, ◇: verb predicate SRL

Multilingual Dependency Learning: A Huge Feature Engineering Method to Semantic Dependency Parsing *

Hai Zhao(赵海)†*, Wenliang Chen(陈文亮)‡, Chunyu Kit†, Guodong Zhou*
†Department of Chinese, Translation and Linguistics
City University of Hong Kong
83 Tat Chee Avenue, Kowloon, Hong Kong, China
‡Language Infrastructure Group, MASTAR Project
National Institute of Information and Communications Technology
3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, Japan, 619-0289
*School of Computer Science and Technology
Soochow University, Suzhou, China 215006
haizhao@cityu.edu.hk, chenwl@nict.go.jp

Abstract

This paper describes our system about multilingual semantic dependency parsing (SRLonly) for our participation in the shared task of CoNLL-2009. We illustrate that semantic dependency parsing can be transformed into a word-pair classification problem and implemented as a single-stage machine learning system. For each input corpus, a large scale feature engineering is conducted to select the best fit feature template set incorporated with a proper argument pruning strategy. The system achieved the top average score in the closed challenge: 80.47% semantic labeled F1 for the average score.

1 Introduction

The syntactic and semantic dependency parsing in multiple languages introduced by the shared task of CoNLL-2009 is an extension of the CoNLL-2008 shared task (Hajič et al., 2009). Seven languages, English plus Catalan, Chinese, Czech, German, Japanese and Spanish, are involved (Taulé et al., 2008; Palmer and Xue, 2009; Hajič et al., 2006; Surdeanu et al., 2008; Burchardt et al., 2006; Kawahara et al., 2002). This paper presents our research for participation in the semantic-only (SRLonly) challenge of the CoNLL-2009 shared task, with a highlight on our strategy to select features from a large candidate set for maximum entropy learning.

This study is partially supported by CERG grant 9040861 (CityU 1318/03H), CityU Strategic Research Grant 7002037, Projects 60673041 and 60873041 under the National Natural Science Foundation of China and Project 2006AA01Z147 under the "863" National High-Tech Research and Development of China.

2 System Survey

We opt for the maximum entropy model with Gaussian prior as our learning model for all classification subtasks in the shared task. Our implementation of the model adopts L-BFGS algorithm for parameter optimization as usual. No additional feature selection techniques are applied.

Our system is basically improved from its early version for CoNLL-2008 (Zhao and Kit, 2008). By introducing a virtual root for every predicates, The job to determine both argument labels and predicate senses is formulated as a word-pair classification task in four languages, namely, Catalan, Spanish, Czech and Japanese. In other three languages, Chinese, English and German, a predicate sense classifier is individually trained before argument label classification. Note that traditionally (or you may say that most semantic parsing systems did so) argument identification and classification are handled in a two-stage pipeline, while ours always tackles them in one step, in addition, predicate sense classification are also included in this unique learning/test step for four of all languages.

3 Pruning Argument Candidates

We keep using a word-pair classification procedure to formulate semantic dependency parsing. Specifically, we specify the first word in a word pair as a predicate candidate (i.e., a semantic head, and noted as p in our feature representation) and the next as an argument candidate (i.e., a semantic dependent, and

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 55–60,
Boulder, Colorado, June 2009.

noted as a). We do not differentiate between verbal and non-verbal predicates and our system handles them in the exactly same way.

When no constraint available, however, all word pairs in the an input sequence must be considered, leading to very poor efficiency in computation for no gain in effectiveness. Thus, the training sample needs to be pruned properly. As predicates overtly known in the share task, we only consider how to effectively prune argument candidates.

We adopt five types of argument pruning strategies for seven languages. All of them assume that a syntactic dependency parsing tree is available.

As for Chinese and English, we continue to use a dependency version of the pruning algorithm of (Xue and Palmer, 2004) as described in (Zhao and Kit, 2008). The pruning algorithm is readdressed as the following.

Initialization: Set the given predicate candidate as the current node;

(1) The current node and all of its syntactic children are selected as argument candidates.

(2) Reset the current node to its syntactic head and repeat step (1) until the root is reached.

Note that the given predicate candidate itself is excluded from the argument candidate list for Chinese, that is slightly different from English.

The above pruning algorithm has been shown effective. However, it is still inefficient for a single-stage argument identification/classification classification task. Thus we introduce an assistant argument label '*_NoMoreArgument*' to alleviate this difficulty. If an argument candidate in the above algorithm is labeled as such a label, then the pruning algorithm will end immediately. In training, this assistant label means no more samples will be generated for the current predicate, while in test, the decoder will not search more argument candidates any more. This adaptive technique more effectively prunes the argument candidates. In fact, our experiments show 1/3 training memory and time may be saved from it.

As for Catalan and Spanish, only syntactic children of the predicate are considered as the argument candidates.

As for Czech, only syntactic children, grandchildren, great-grandchildren, parent and siblings of the predicate are taken as the argument candidates.

As for German, only syntactic children, grandchildren, parent, siblings, siblings of parent and siblings of grandparent of the predicate are taken as the argument candidates.

The case is somewhat sophisticated for Japanese. As we cannot identify a group of simple predicate-argument relations from the syntactic tree. Thus we consider top frequent 28 syntactic relations between the predicate and the argument. The parser will search all words before and after the predicate, and only those words that hold one of the 28 syntactic relations to the predicate are considered as the argument candidate. Similar to the pruning algorithm for Chinese/English/German, we also introduce two assistant labels '*_leftNoMoreArgument*' and '*_rightNoMoreArgument*' to adaptively prune words too far away from the predicate.

4 Feature Templates

As we don't think that we can benefit from knowing seven languages, an automatic feature template selection is conducted for each language.

About 1000 feature templates (hereafter this template set is referred to FT) are initially considered. These feature templates are from various combinations or integrations of the following basic elements.

Word Property. This type of elements include word *form*, *lemma*, part-of-speech tag (*PoS*), *FEAT* (additional morphological features), syntactic dependency label (*dprel*), semantic dependency label (*semdprel*) and characters (*char*) in the word form (only suitable for Chinese and Japanese)[1].

Syntactic Connection. This includes syntactic head (h), left(right) farthest(nearest) child (lm, ln, rm, and rn), and high(low) support verb or noun. We explain the last item, support verb(noun). From the predicate or the argument to the syntactic root along the syntactic tree, the first verb(noun) that is met is called as the low support verb(noun), and the nearest one to the root is called as the high support verb(noun).

Semantic Connection. This includes semantic

[1] All lemmas, PoS, and *FEAT* for either training or test are from automatically pre-analyzed columns of every input files.

FEATn	1	2	3	4	5	6	7	8	9	10	11
Catalan/Spanish	postype	gen	num	person	mood	tense	punct				
Czech	SubPOS	Gen	Num	Cas	Neg	Gra	Voi	Var	Sem	Per	Ten

Table 1: Notations of FEATs

head ($semhead$), left(right) farthest(nearest) semantic child ($semlm, semln, semrm, semrn$). We say a predicate is its argument's semantic head, and the latter is the former's child. Features related to this type may track the current semantic parsing status.

Path. There are two basic types of path between the predicate and the argument candidates. One is the linear path (*linePath*) in the sequence, the other is the path in the syntactic parsing tree (*dpPath*). For the latter, we further divide it into four sub-types by considering the syntactic root, *dpPath* is the full path in the syntactic tree. Leading two paths to the root from the predicate and the argument, respectively, the common part of these two paths will be *dpPathShare*. Assume that *dpPathShare* starts from a node r', then *dpPathPred* is from the predicate to r', and *dpPathArgu* is from the argument to r'.

Family. Two types of children sets for the predicate or argument candidate are considered, the first includes all syntactic children (*children*), the second also includes all but excludes the left most and the right most children (*noFarChildren*).

Concatenation of Elements. For all collected elements according to *linePath*, *children* and so on, we use three strategies to concatenate all those strings to produce the feature value. The first is *seq*, which concatenates all collected strings without doing anything. The second is *bag*, which removes all duplicated strings and sort the rest. The third is *noDup*, which removes all duplicated neighbored strings.

In the following, we show some feature template examples derived from the above mentioned items.

a.lm.lemma The lemma of the left most child of the argument candidate.

p.h.dprel The dependant label of the syntactic head of the predicate candidate.

a.pos+p.pos The concatenation of *PoS* of the argument and the predicate candidates.

p_{-1}*.pos+p.pos PoS* of the previous word of the predicate and *PoS* of the predicate itself.

a:p|dpPath.lemma.bag Collect all lemmas along the syntactic tree path from the argument to the predicate, then removed all duplicated ones and sort the rest, finally concatenate all as a feature string.

a:p.highSupportNoun|linePath.dprel.seq Collect all dependant labels along the line path from the argument to the high support noun of the predicate, then concatenate all as a feature string.

(a:p|dpPath.dprel.seq)+p.FEAT1 Collect all dependant labels along the line path from the argument to the predicate and concatenate them plus the first FEAT of the predicate.

An important feature for the task is *dpTreeRelation*, which returns the relationship of a and p in a syntactic parse tree and cannot be derived from combining the above basic elements. The possible values for this feature include `parent`, `sibling` etc.

5 Automatically Discovered Feature Template Sets

For each language, starting from a basic feature template set (a small subset of FT) according to our previous result in English dependency parsing, each feature template outside the basic set is added and each feature template inside the basic set is removed one by one to check the effectiveness of each feature template following the performance change in the development set. This procedure will be continuously repeated until no feature template is added or removed or the performance is not improved.

There are some obvious heuristic rules that help us avoid trivial feature template checking, for example, *FEAT* features are only suitable for Catalan, Czech and Spanish. Though *FEAT* features are also available for Japanese, we don't adopt them for this language due to the hight training cost. To simplify feature representation, we use *FEAT1*, *FEAT2*, and so on to represent different *FEAT* for every languages. A lookup list can be found in Table 1. According to the list, *FEAT4* represents $person$ for Catalan or Spanish, but Cas for Czech.

As we don't manually interfere the selection procedure for feature templates, ten quite different fea-

	Ca	Ch	Cz	En	Gr	Jp	Sp
Ca	53						
Ch	5	75					
Cz	11	10	76				
En	11	11	12	73			
Gr	7	7	7	14	45		
Jp	6	22	13	15	10	96	
Sp	22	9	18	15	9	12	66

Table 2: Feature template set: argument classifier

	Ch	En	Gr
Ch	46		
En	5	9	
Gr	17	2	40

Table 3: Feature template set: sense classifier

ture template sets are obtained at last. Statistical information of seven sets for argument classifiers is in Table 2, and those for sense classifiers are in Table 3. Numbers in the diagonals of these two tables mean the numbers of feature templates, and others mean how many feature templates are identical for every language pairs. The most matched feature template sets are for Catalan/Spanish and Chinese/Japanese. As for the former, it is not so surprised because these two corpora are from the same provider.

Besides the above statistics, these seven feature template sets actually share little in common. For example, the intersection set from six languages, as Chinese is excluded, only includes one feature template, $p.lemma$ (the lemma of the predicate candidate). If all seven sets are involved, then such an intersection set will be empty. Does this mean human languages share little in semantic representation? :)

It is unlikely to completely demonstrate full feature template sets for all languages in this short report, we thus only demonstrate two sets, one for English sense classification in Table 4 and the other for Catalan argument classification in Table 5[2].

6 Word Sense Determination

The shared task of CoNLL-2009 still asks for the predicate sense. In our work for CoNLL-2008 (Zhao and Kit, 2008), this was done by searching for a right example in the given dictionary. Unfortunately, we late found this caused a poor performance in sense determination. This time, an individual classifier is used to determine the sense for Chinese, English or German, and this is done by the argument classifier by introducing a virtual root for every predicates for the rest four languages[3]. Features used for sense determination are also selected following the same procedure in Section 5. The difference is only predicate related features are used for selection.

_	*p.lm.pos*
_	*p.rm.pos*
_	*p.lemma*
_	*p.lemma* + $p.lemma_1$
_	*p.lemma* + *p.children.dprel.noDup*
_	*p.lemma* + *p.currentSense*
_	*p.form*
_	$p.form_{-1}$ + *p.form*
_	*p.form* + $p.form_1$

Table 4: Feature set for English sense classification

7 Decoding

The decoding for four languages, Catalan, Czech, Japanese and Spanish is trivial, each word pairs will be checked one by one. The first word of the pair is the virtual root or the predicate, the second is the predicate or every argument candidates. Argument candidates are checked in the order of different syntactic relations to their predicate, which are enumerated by the pruning algorithms in Section 3, or from left to right for the same syntactic relation. After the sense of the predicate is determined, the label of each argument candidate will be directly classified, or, it is proved non-argument.

As for the rest languages, Chinese, English or German, after the sense classifier outputs its result, an optimal argument structure for each predicate is determined by the following maximal probability.

$$S_p = \text{argmax} \prod_i P(a_i|a_{i-1}, a_{i-2}, ...), \qquad (1)$$

where S_p is the argument structure, $P(a_i|a_{i-1}...)$ is the conditional probability to determine the label of the i-th argument candidate label. Note that

[2]Full feature lists and their explanation for all languages will be available at the website, http://bcmi.sjtu.edu.cn/~zhaohai.

[3]For Japanese, no senses for predicates are defined. Thus it is actually a trivial classification task in this case.

_	*p.currentSense* + *p.lemma*
_	*p.currentSense* + *p.pos*
_	*p.currentSense* + *a.pos*
_	p_{-1}.*FEAT1*
_	*p.FEAT2*
_	p_1.*FEAT3*
_	*p.semrm.semdprel*
_	*p.lm.dprel*
_	*p.form* + *p.children.dprel.bag*
	p.lemma$_n$ ($n = -1, 0$)
_	*p.lemma* + *p.lemma*$_1$
_	*p.pos*$_{-1}$ + *p.pos*
_	*p.pos*$_1$
_	*p.pos* + *p.children.dprel.bag*
_	*a.FEAT1* + *a.FEAT3* + *a.FEAT4* + *a.FEAT5* + *a.FEAT6*
_	a_{-1}.*FEAT2* + *a.FEAT2*
_	*a.FEAT3* + a_1.*FEAT3*
_	*a.FEAT3* + *a.h.FEAT3*
_	*a.children.FEAT1.noDup*
_	*a.children.FEAT3.bag*
_	*a.h.lemma*
_	*a.lm.dprel* + *a.form*
_	*a.lm.form*
_	*a.lm*$_{-1}$.*lemma*
_	*a.lm*$_n$.*pos* (n=0,1)
_	*a.noFarChildren.pos.bag* + *a.rm.form*
_	*a.pphead.lemma*
_	*a.rm.dprel* + *a.form*
_	*a.rm*$_{-1}$.*form*
_	*a.rm.lemma*
_	*a.rn.dprel* + *a.form*
_	*a.lowSupportVerb.lemma*
_	a_{-1}.*form*
_	*a.form* + a_1.*form*
_	*a.form* + *a.children.pos*
_	*a.lemma* + *a.h.form*
_	*a.lemma* + *a.pphead.form*
_	a_1.*lemma*
_	a_1.*pos* + *a.pos.seq*
_	*a.pos* + *a.children.dprel.bag*
_	*a.lemma* + *p.lemma*
_	*(a:p|dpPath.dprel)* + *p.FEAT1*
_	*a:p|linePath.distance*
_	*a:p|linePath.FEAT1.bag*
_	*a:p|linePath.form.seq*
_	*a:p|linePath.lemma.seq*
_	*a:p|linePath.dprel.seq*
_	*a:p|dpPath.lemma.seq*
_	*a:p|dpPath.lemma.bag*
_	*a:p|dpPathArgu.lemma.seq*
_	*a:p|dpPathArgu.lemma.bag*

Table 5: Feature set for Catalan argument classification

$P(a_i|a_{i-1}, ...)$ in equation (1) may be simplified as $P(a_i)$ if the input feature template set does not concerned with the previous argument label output. A beam search algorithm is used to find the parsing decision sequence.

8 Evaluation Results

Our evaluation is carried out on two computational servers, (1) **LEGA**, a 64-bit ubuntu Linux installed server with double dual-core AMD Opteron processors of 2.8GHz and 24GB memory. This server was also used for our previous participation in CoNLL-2008 shared task. (2) **MEGA**, a 64-bit ubuntu Linux installed server with six quad-core Intel Xeon processors of 2.33GHz and 128GB memory.

Altogether nearly 60,000 machine learning routines were run to select the best fit feature template sets for all seven languages within two months. Both LEGA and MEGA were used for this task. However, training and test for the final submission of Chinese, Czech and English run in MEGA, and the rest in LEGA. As we used multiple thread training and multiple routines run at the same time, the exact time cost for either training or test is hard to estimate. Here we just report the actual time and memory cost in Table 7 for reference.

The official evaluation results of our system are in Table 6. Numbers in bold in the table stand for the best performances for the specific languages. The results in development sets are also given. The first row of the table reports the results using golden input features.

Two facts as the following suggest that our system does output robust and stable results. The first is that two results for development and test sets in the same language are quite close. The second is about out-of-domain (OOD) task. Though for each OOD task, we just used the same model trained from the respective language and did nothing to strengthen it, this does not hinder our system to obtain top results in Czech and English OOD tasks.

In addition, the feature template sets from automatical selection procedure in this task were used for the joint task of this shared task, and also output top results according to the average score of semantic labeled F1 (Zhao et al., 2009).

	average	Catalan	Chinese	Czech	English	German	Japanese	Spanish
Development with Gold	81.24	81.52	78.32	86.96	84.19	77.75	78.67	81.32
Development	80.46	80.66	77.90	85.35	84.01	76.55	78.41	80.39
Test (official scores)	**80.47**	**80.32**	77.72	85.19	85.44	75.99	**78.15**	**80.46**
Out-of-domain	74.34			**85.44**	**73.31**	64.26		

Table 6: Semantic labeled F1

		Catalan	Chinese	Czech	English	German	Japanese	Spanish
Sense	Training memory (MB)		418.0		136.0	63.0		
	Training time (Min.)		11.0		2.5	1.7		
	Test time (Min.)		0.7		0.2	0.03		
Argument	Training memory (GB)	0.4	3.7	3.2	3.8	0.2	1.4	0.4
	Training time (Hours)	3.0	13.8	24.9	12.4	0.2	6.1	4.4
	Test time (Min.)	3.0	144.0	27.1	88.0	1.0	4.2	7.0

Table 7: Computational cost

9 Conclusion

As presented in the above sections, we have tackled semantic parsing for the CoNLL-2009 shared task as a word-pair classification problem. Incorporated with a proper argument candidate pruning strategy and a large scale feature engineering for each language, our system produced top results.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco.

Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *2004 Conference on Empirical Methods in Natural Language Processing (EMNLP-2004)*, pages 88–94, Barcelona, Spain, July 25-26.

Hai Zhao and Chunyu Kit. 2008. Parsing syntactic and semantic dependencies with two single-stage maximum entropy models. In *Twelfth Conference on Computational Natural Language Learning (CoNLL-2008)*, pages 203–207, Manchester, UK, August 16-17.

Hai Zhao, Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Multilingual dependency learning: Exploiting rich features for tagging syntactic and semantic dependencies. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

Multilingual Dependency Learning: Exploiting Rich Features for Tagging Syntactic and Semantic Dependencies

Hai Zhao(赵海)[†], Wenliang Chen(陈文亮)[‡],
Jun'ichi Kazama[‡], Kiyotaka Uchimoto[‡], and Kentaro Torisawa[‡]
[†]Department of Chinese, Translation and Linguistics
City University of Hong Kong
83 Tat Chee Avenue, Kowloon, Hong Kong, China
[‡]Language Infrastructure Group, MASTAR Project
National Institute of Information and Communications Technology
3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, Japan, 619-0289
haizhao@cityu.edu.hk, chenwl@nict.go.jp

Abstract

This paper describes our system about multilingual syntactic and semantic dependency parsing for our participation in the joint task of CoNLL-2009 shared tasks. Our system uses rich features and incorporates various integration technologies. The system is evaluated on in-domain and out-of-domain evaluation data of closed challenge of joint task. For in-domain evaluation, our system ranks the second for the average macro labeled F1 of all seven languages, 82.52% (only about 0.1% worse than the best system), and the first for English with macro labeled F1 87.69%. And for out-of-domain evaluation, our system also achieves the second for average score of all three languages.

1 Introduction

This paper describes the system of National Institute of Information and Communications Technology (NICT) and City University of Hong Kong (CityU) for the joint learning task of CoNLL-2009 shared task (Hajič et al., 2009)[1]. The system is basically a pipeline of syntactic parser and semantic parser. We use a syntactic parser that uses very rich features and integrates graph- and transition-based methods. As for the semantic parser, a group of well selected feature templates are used with n-best syntactic features.

The rest of the paper is organized as follows. The next section presents the technical details of our syntactic dependency parsing. Section 3 describes the details of the semantic dependency parsing. Section 4 shows the evaluation results. Section 5 looks into a few issues concerning our forthcoming work for this shared task, and Section 6 concludes the paper.

2 Syntactic Dependency Parsing

Basically, we build our syntactic dependency parsers based on the MSTParser, a freely available implementation[2], whose details are presented in the paper of McDonald and Pereira (2006). Moreover, we exploit rich features for the parsers. We represent features by following the work of Chen et al. (2008) and Koo et al. (2008) and use features based on dependency relations predicted by transition-based parsers (Nivre and McDonald, 2008). Chen et al. (2008) and Koo et al. (2008) proposed the methods to obtain new features from large-scale unlabeled data. In our system, we perform their methods on training data because the closed challenge does not allow to use unlabeled data. In this paper, we call these new additional features rich features.

2.1 Basic Features

Firstly, we use all the features presented by McDonald et al. (2006), if they are available in data. Then we add new features for the languages having FEAT information (Hajič et al., 2009). FEAT is a set of morphological-features, e.g. more detailed part of speech, number, gender, etc. We try to align different types of morphological-features. For example,

[1]Our thanks give to the following corpus providers, (Taulé et al., 2008; Palmer and Xue, 2009; Hajič et al., 2006; Surdeanu et al., 2008; Burchardt et al., 2006) and (Kawahara et al., 2002).

[2]http://mstparser.sourceforge.net

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 61–66,
Boulder, Colorado, June 2009.

we can obtain a sequence of gender tags of all words from a head h to its dependent d. Then we represent the features based on the obtained sequences.

Based on the results of development data, we perform non-projective parsing for Czech and German and perform projective parsing for Catalan, Chinese, English, Japanese, and Spanish.

2.2 Features Based on Dependency Pairs

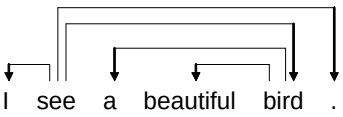

Figure 1: Example dependency graph.

Chen et al. (2008) presented a method of extracting short dependency pairs from large-scale auto-parsed data. Here, we extract all dependency pairs rather than short dependency pairs from training data because we believe that training data are reliable. In a parsed sentence, if two words have dependency relation, we add this word pair into a list named L and count its frequency. We consider the direction. For example, in figure 1, *a* and *bird* have dependency relation in the sentence "I see a beautiful bird.". Then we add word pair "a-bird:HEAD"[3] into list L and accumulate its frequency.

We remove the pairs which occur only once in training data. According to frequency, we then group word pairs into different buckets, with bucket LOW for frequencies 2-7, bucket MID for frequencies 8-14, and bucket HIGH for frequencies $15+$. We set these threshold values by following the setting of Chen et al. (2008). For example, the frequency of pair "a-bird:HEAD" is 5. Then it is grouped into bucket "LOW". We also add a virtual bucket "ZERO" to represent the pairs that are not included in the list. So we have four buckets. "ZERO", "LOW", "MID", and "HIGH" are used as bucket IDs.

Based on the buckets, we represent new features for a head h and its dependent d. We check word pairs surrounding h and d. Table 1 shows the word pairs, where h-word refers to the head word, d-word refers to the dependent word, h-word-1 refers to the word to the left of the head in the sentence, h-word+1 refers to the word to the right of the head, d-word-1 refers to the word to the left of the dependent, and d-word+1 refers the word to the right of the dependent. Then we obtain the bucket IDs of these word pairs from L.

We generate new features consisting of indicator functions for bucket IDs of word pairs. We call these features word-pair-based features. We also generate combined features involving bucket IDs and part-of-speech tags of heads.

h-word, d-word
h-word-1, d-word
h-word+1, d-word
h-word, d-word-1
h-word, d-word+1

Table 1: Word pairs for feature representation

2.3 Features Based on Word Clusters

Koo et al. (2008) presented new features based on word clusters obtained from large-scale unlabeled data and achieved large improvement for English and Czech. Here, word clusters are generated only from the training data for all the languages. We perform word clustering by using the clustering tool[4], which also was used by Koo et al. (2008). The cluster-based features are the same as the ones used by Koo et al. (2008).

2.4 Features Based on Predicted Relations

Nivre and McDonald (2008) presented an integrating method to provide additional information for graph-based and transition-based parsers. Here, we represent features based on dependency relations predicted by transition-based parsers for graph-based parser. Based on the results on development data, we choose the MaltParser for Catalan, Czech, German, and Spanish, and choose another MaxEnt-based parser for Chinese, English, and Japanese.

2.4.1 A Transition-based Parser: MaltParser

For Catalan, Czech, German, and Spanish, we use the MaltParser, a freely available implementa-

[3] HEAD means that *bird* is the head of the pair.

[4] http://www.cs.berkeley.edu/~pliang/software/brown-cluster-1.2.zip

tion[5], whose details are presented in the paper of Nivre (2003). More information about the parser can be available in the paper (Nivre, 2003).

Due to computational cost, we do not select new feature templates for the MaltParser. Following the features settings of Hall et al. (2007), we use their Czech feature file and Catalan feature file. To simply, we apply Czech feature file for German too, and apply Catalan feature file for Spanish.

2.4.2 Another Transition-based Parser: MaxEnt-based Parser

In three highly projective language, Chinese, English and Japanese, we use the maximum entropy syntactic dependency parser as in Zhao and Kit (2008). We still use the similar feature notations of that work. We use the same greedy feature selection of Zhao et al. (2009) to determine an optimal feature template set for each language. Full feature sets for the three languages can be found at website, http://bcmi.sjtu.edu.cn/~zhaohai.

2.4.3 Feature Representation

For training data, we use 2-way jackknifing to generate predicted dependency parsing trees by two transition-based parsers. Following the features of Nivre and McDonald (2008), we define features for a head h and its dependent d with label l as shown in table 2, where G_{Tran} refers to dependency parsing trees generated by the MaltParser or MaxEnt-base Parser and $*$ refers to any label. All features are conjoined with the part-of-speech tags of the words involved in the dependency.

Is $(h, d, *)$ in G_{Tran}?
Is (h, d, l) in G_{Tran}?
Is $(h, d, *)$ not in G_{Tran}?
Is (h, d, l) not in G_{Tran}?

Table 2: Features set based on predicted labels

3 n-best Syntactic Features for Semantic Dependency Parsing

Due to the limited computational resource that we have, we used the the similar learning framework as our participant in semantic-only task (Zhao et al., 2009). Namely, three languages, a single maximum entropy model is used for all identification and classification tasks of predicate senses or argument labels in four languages, Catalan, Czech, Japanese, or Spanish. For the rest three languages, an individual sense classifier still using maximum entropy is additionally used to output the predicate sense previously. More details about argument candidate pruning strategies and feature template set selection are described in Zhao et al. (2009).

	Normal	n-best	Matched
Ca	53	54	50
Ch	75	65	55
En	73	70	63

Table 3: Feature template sets:n-best vs. non-n-best

The same feature template sets as the semantic-only task are used for three languages, Czech, German and Japanese. For the rest four languages, we further use n-best syntactic features to strengthen semantic dependency parsing upon those automatically discovered feature template sets. However, we cannot obtain an obvious performance improvement in Spanish by using n-best syntactic features. Therefore, only Catalan, Chinese and English semantic parsing adopted these types of features at last.

Our work about n-best syntactic features still starts from the feature template set that is originally selected for the semantic-only task. The original feature template set is hereafter referred to 'the normal' or 'non-n-best'. In practice, only 2nd-best syntactic outputs are actually adopted by our system for the joint task.

To generate helpful feature templates from the 2nd-best syntactic tree, we simply let all feature templates in the normal feature set that are based on the 1st-best syntactic tree now turn to the 2nd-best one. Using the same notations for feature template representation as in Zhao et al. (2009), we take an example to show how the original n-best features are produced. Assuming $a.children.dprel.bag$ is one of syntactic feature templates in the normal set, this feature means that all syntactic children of the argument candidate (a) are chosen, and their dependant labels are collected, the duplicated labels are removed and then sorted, finally all these strings are concatenated as a feature. The cor-

[5]http://w3.msi.vxu.se/~nivre/research/MaltParser.html

Language	Features
Catalan	*p:2.lm.dprel*
_	*a.lemma + a:2.h.form*
_	*a.lemma + a:2.pphead.form*
_	*(a:2:p:2|dpPath.dprel.seq) + p.FEAT1*
Chinese	*a:2.h.pos*
_	*a:2.children.pos.seq + p:2.children.pos.seq*
_	*a:2:p:2|dpPath.dprel.bag*
_	*a:2:p:2|dpPathPred.form.seq*
_	*a:2:p:2|dpPath.pos.bag*
_	*(a:2:p:2|dpTreeRelation) + p.pos*
_	*(a:2:p:2|dpPath.dprel.seq) + a.pos*
English	*a:2:p:2|dpPathPred.lemma.bag*
_	*a:2:p:2|dpPathPred.pos.bag*
_	*a:2:p:2|dpTreeRelation*
_	*a:2:p:2|dpPath.dprel.seq*
_	*a:2:p:2|dpPathPred.dprel.seq*
_	*a.lemma + a:2.dprel + a:2.h.lemma*
_	*(a:2:p:2|dpTreeRelation) + p.pos*

Table 4: Features for n-best syntactic tree

responding 2nd-best syntactic feature will be a : $2.children.dprel.bag$. As all operations to generate the feature for $a.children.dprel.bag$ is within the 1st-best syntactic tree, while those for a : $2.children.dprel.bag$ is within the 2nd-best one. As all these 2nd-best syntactic features are generated, we use the same greedy feature selection procedure as in Zhao et al. (2009) to determine the best fit feature template set according to the evaluation results in the development set.

For Catalan, Chinese and English, three optimal n-best feature sets are obtained, respectively. Though dozens of n-best features are initially generated for selection, only few of them survive after the greedy selection. A feature number statistics is in Table 3, and those additionally selected n-best features for three languages are in Table 4. Full feature lists and their explanation for all languages will be available at the website, http://bcmi.sjtu.edu.cn/˜zhaohai.

4 Evaluation Results

Two tracks (closed and open challenges) are provided for joint task of CoNLL2009 shared task. We participated in the closed challenge and evaluated our system on the in-domain and out-of-domain evaluation data.

	avg.	Cz	En	Gr
Syntactic (LAS)	77.96	75.58	82.38	75.93
Semantic (Labeled F1)	75.01	82.66	74.58	67.78
Joint (Macro F1)	76.51	79.12	**78.51**	71.89

Table 6: The official results of our submission for out-of-domain task(%)

	Test		Dev	
	Basic	ALL	Basic	ALL
Catalan	82.91	85.88	83.15	85.98
Chinese	74.28	75.67	73.36	75.64
Czech	77.21	79.70	77.91	80.22
English	88.63	89.19	86.35	87.40
German	84.61	86.24	83.99	85.44
Japanese	92.31	92.32	92.01	92.85
Spanish	83.59	86.29	83.73	86.22
Average	83.32	85.04 (+1.72)	82.92	84.82 (+1.90)

Table 7: The effect of rich features for syntactic dependency parsing

4.1 Official Results

The official results for the joint task are in Table 5, and the out-of-domain task in Table 6, where numbers in bold stand for the best performances for the specific language. For out-of-domain (OOD) evaluation, we did not perform any domain adaptation. For both in-domain and out-of-domain evaluation, our system achieved the second best performance for the average Macro F1 scores of all the languages. And our system provided the first best performance for the average Semantic Labeled F1 score and the forth for the average Labeled Syntactic Accuracy score for in-domain evaluation.

4.2 Further results

At first, we check the effect of rich features for syntactic dependency parsing. Table 7 shows the comparative results of basic features and all features on test and development data, where "Basic" refers to the system with basic features and "ALL" refers to the system with basic features plus rich features. We found that the additional features provided improvement of 1.72% for test data and 1.90% for development data.

Then we investigate the effect of different training data size for semantic parsing. The learning

	average	Catalan	Chinese	Czech	English	German	Japanese	Spanish
Syntactic (LAS)	85.04	85.88	75.67	79.70	89.19	86.24	92.32	86.29
Semantic (Labeled F1)	79.96	80.10	76.77	82.04	86.15	76.19	78.17	80.29
Joint (Macro F1)	82.52	**83.01**	76.23	80.87	**87.69**	81.22	85.28	**83.31**

Table 5: The official results of our joint submission (%)

Data	Czech	Chinese		English	
		normal	n-best	normal	n-best
25%	80.71	75.12	75.24	82.02	82.06
50%	81.52	76.50	76.59	83.52	83.42
75%	81.90	76.92	77.01	84.21	84.30
100%	82.24	77.35	77.34	84.73	84.80

Table 8: The performance in development set (semantic labeled F1) vs. training corpus size

curves are drawn for Czech, Chinese and English. We use 25%, 50% and 75% training corpus, respectively. The results in development sets are given in Table 8. Note that in this table the differences between normal and n-best feature template sets are also given for Chinese and English. The results in the table show that n-best features help improve Chinese semantic parsing as the training corpus is smaller, while it works for English as the training corpus is larger.

5 Discussion

This work shows our further endeavor in syntactic and semantic dependency parsing, based on our previous work (Chen et al., 2008; Zhao and Kit, 2008).

Chen et al. (Chen et al., 2008) and Koo et al. (Koo et al., 2008) used large-scale unlabeled data to improve syntactic dependency parsing performance. Here, we just performed their method on training data. From the results, we found that the new features provided better performance. In future work, we can try these methods on large-scale unlabeled data for other languages besides Chinese and English.

In Zhao and Kit (2008), we addressed that semantic parsing should benefit from cross-validated training corpus and n-best syntactic output. These two issues have been implemented during this shared task. Though existing work show that re-ranking for semantic-only or syntactic-semantic joint tasks may bring higher performance, the limited computational resources does not permit us to do this for multiple languages.

To analyze the advantage and the weakness of our system, the ranks for every languages of our system's outputs are given in Table 9, and the performance differences between our system and the best one in Table 10[6]. The comparisons in these two tables indicate that our system is slightly weaker in the syntactic parsing part, this may be due to the reason that syntactic parsing in our system does not benefit from semantic parsing as the other joint learning systems. However, considering that the semantic parsing in our system simply follows the output of the syntactic parsing and the semantic part of our system still ranks the first for the average score, the semantic part of our system does output robust and stable results. It is worth noting that semantic labeled F1 in Czech given by our system is 4.47% worse than the best one. This forby gap in this language further indicates the advantage of our system in the other six languages and some latent bugs or learning framework misuse in Czech semantic parsing.

6 Conclusion

We describe the system that uses rich features and incorporates integrating technology for joint learning task of syntactic and semantic dependency parsing in multiple languages. The evaluation results show that our system is good at both syntactic and semantic parsing, which suggests that a feature-oriented method is effective in multiple language processing.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006.

[6]The difference for Chinese in the latter table is actually computed between ours and the second best system.

	average	Catalan	Chinese	Czech	English	German	Japanese	Spanish
Syntactic (LAS)	4	4	4	4	2	3	3	4
Semantic (Labeled F1)	1	1	3	4	1	2	2	1
Joint (Macro F1)	2	1	3	4	1	3	2	1

Table 9: Our system's rank within the joint task according to three main measures

	average	Catalan	Chinese	Czech	English	German	Japanese	Spanish
Syntactic (LAS)	0.73	1.98	0.84	0.68	0.69	1.24	0.25	1.35
Semantic (Labeled F1)	-	-	0.38	4.47	-	2.42	0.09	-
Joint (Macro F1)	0.12	-	0.15	2.40	-	1.22	0.37	-

Table 10: The performance differences between our system and the best one within the joint task according to three main measures

The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of LREC-2006*, Genoa, Italy.

Wenliang Chen, Daisuke Kawahara, Kiyotaka Uchimoto, Yujie Zhang, and Hitoshi Isahara. 2008. Dependency parsing with short dependency relations in unlabeled data. In *Proceedings of IJCNLP-2008*, Hyderabad, India, January 8-10.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL-2009*, Boulder, Colorado, USA.

Johan Hall, Jens Nilsson, Joakim Nivre, Gülsen Eryiğit, Beáta Megyesi, Mattias Nilsson, and Markus Saers. 2007. Single malt or blended? a study in multilingual parser optimization. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, pages 933–939, Prague, Czech, June.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of LREC-2002*, pages 2008–2013, Las Palmas, Canary Islands.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pages 595–603, Columbus, Ohio, USA, June.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *Proceedings of EACL-2006*, pages 81–88, Trento, Italy, April.

Ryan McDonald, Kevin Lerman, and Fernando Pereira. 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of CoNLL-X*, New York City, June.

Joakim Nivre and Ryan McDonald. 2008. Integrating graph-based and transition-based dependency parsers. In *Proceedings of ACL-08: HLT*, pages 950–958, Columbus, Ohio, June.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT 03)*, pages 149–160, Nancy, France, April 23-25.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

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 the CoNLL-2008*.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the LREC-2008*, Marrakesh, Morroco.

Hai Zhao and Chunyu Kit. 2008. Parsing syntactic and semantic dependencies with two single-stage maximum entropy models. In *Proceedings of CoNLL-2008*, pages 203–207, Manchester, UK, August 16-17.

Hai Zhao, Wenliang Chen, Chunyu Kit, and Guodong Zhou. 2009. Multilingual dependency learning: A huge feature engineering method to semantic dependency parsing. In *Proceedings of CoNLL-2009*, Boulder, Colorado, USA.

Efficient Parsing of Syntactic and Semantic Dependency Structures

Bernd Bohnet
International Computer Science Institute
1947 Center Street
Berkeley 94704, California
bohnet@icsi.Berkeley.edu

Abstract

In this paper, we describe our system for the 2009 CoNLL shared task for joint parsing of syntactic and semantic dependency structures of multiple languages. Our system combines and implements efficient parsing techniques to get a high accuracy as well as very good parsing and training time. For the applications of syntactic and semantic parsing, the parsing time and memory footprint are very important. We think that also the development of systems can profit from this since one can perform more experiments in the given time. For the subtask of syntactic dependency parsing, we could reach the second place with an accuracy in average of 85.68 which is only 0.09 points behind the first ranked system. For this task, our system has the highest accuracy for English with 89.88, German with 87.48 and the out-of-domain data in average with 78.79. The semantic role labeler works not as well as our parser and we reached therefore the fourth place (ranked by the macro F1 score) in the joint task for syntactic and semantic dependency parsing.

1 Introduction

Depedendency parsing and semantic role labeling improved in the last years significantly. One of the reasons are CoNLL shared tasks for syntactic dependency parsing in the years 2006, 2007 (Buchholz and Marsi, 2006; Nivre et al., 2007) and the CoNLL shared task for joint parsing of syntactic and semantic dependencies in the year 2008 and of cause this shared task in 2009, cf. (Surdeanu et al., 2008; Hajič et al., 2009). The CoNLL Shared Task 2009 is to parse syntactic and semantic dependencies of seven languages. Therefore, training and development data in form of annotated corpora for Catalan, Chinese, Czech, English, German, Japanese and Spanish is provided, cf. (Taulé et al., 2008; Palmer and Xue, 2009; Hajič et al., 2006; Surdeanu et al., 2008; Burchardt et al., 2006; Kawahara et al., 2002).

There are two main approaches to dependency parsing: Maximum Spanning Tree (MST) based dependency parsing and Transition based dependency parsing, cf. (Eisner, 1996; Nivre et al., 2004; McDonald and Pereira, 2006). Our system uses the first approach since we saw better chance to improve the parsing speed and additionally, the MST had so far slightly better parsing results. For the task of semantic role labeling, we adopted a pipeline architecture where we used for each step the same learning technique (SVM) since we opted for the possibility to build a synchronous combined parser with one score function.

2 Parsing Algorithm

We adopted the second order MST parsing algorithm as outlined by Eisner (1996). This algorithm has a higher accuracy compared to the first order parsing algorithm since it considers also siblings and grandchildren of a node. Eisner‘s second order approach can compute a projective dependency tree within cubic time (O(n^3)).

Both algorithms are bottom up parsing algorithms based on dynamic programming similar to the CKY chart parsing algorithm. The score for a dependency tree is the score of all edge scores. The following

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 67–72,
Boulder, Colorado, June 2009.

equation describes this formally.

$$score(S,t) = \sum_{\forall(i,j)\in E} score(i,j)$$

The score of the sentence S and a tree t over S is defined as the sum of all edge scores where the words of S are $w_0...w_1$. The tree consists of set of nodes N and set of edges $E = \langle N \times N \rangle$. The word indices $(0..n)$ are the elements of the node set N. The expression $(i,j) \in E$ denotes an edge which is going from the node i to the node j.

The edge score ($score(i,j)$) is computed as the scalar product of a feature vector representation of each edge $\overrightarrow{f_S}(i,j)$ with a weight vector $\overrightarrow{w}$ where i,j are the indices of the words in a sentence. The feature vector f_S might take not only into account the words with indices i and j but also additional values such as the words before and after the words w_i and w_j. The following equation shows the score function.

$$score(i,j) = \overrightarrow{f_S}(i,j) * \overrightarrow{w}$$

Many systems encode the features as strings and map the strings to a number. The number becomes the index of the feature in the feature vector and weight vector. In order to compute the weight vector, we reimplemented the support vector machine MIRA which implements online Margin Infused Relaxed Algorithm, cf. (Crammer et al., 2003).

3 Labeled Dependency Parsing

The second order parsing algorithm builds an unlabeled dependency tree. However, all dependency tree banks of the shared task provide trees with edge labels. The following two approaches are common to solve this problem. An additional algorithm labels the edges or the parsing algorithm itself is extended and the labeling algorithm is integrated into the parsing algorithm. McDonald et al. (2006) use an additional algorithm. Their two stage model has a good computational complexity since the labeling algorithm contributes again only a cubic time complexity to the algorithm and keeps therefore the joint algorithm still cubic. The algorithm selects the highest scored label due to the score function $score(w_i, label) + score(w_j, label)$ and inserts the highest scored label into a matrix. The scores are also used in the parsing algorithms and added to the edge scores which improves the overall parsing results as well. In the first order parsing scenario, this procedure is sufficient since no combination of edges are considered by the parsing algorithm. However, in the second order parsing scenario where more than one edge are considered by the parsing algorithm, combinations of two edges might be more accurate.

Johansson and Nugues (2008) combines the edge labeling with the second order parsing algorithm. This adds an additional loop over the edge labels. The complexity is therefore O(n^4). However, they could show that a system can gain accuracy of about 2-4% which is a lot.

4 Non-Projective Dependency Parsing

The dependency parser developed in the last years use two different techniques for non-projective dependency parsing.

Nivre and Nilsson (2005) uses tree rewriting which is the most common technique. With this technique, the training input to the parser is first projectivized by applying a minimal number of lifting operations to the non-projective edges and encoding information about these lifts in edge labels. After these operations, the trees are projective and therefore a projective dependency parser can be applied. During the training, the parser learns also to built trees with the lifted edges and so indirect to built non-projective dependency trees by applying the inverse operations to the lifting on the projective tree.

McDonald and Pereira (2006) developed a technique to rearrange edges in the tree in a postprocessing step after the projective parsing has taken place. Their *Approximate Dependency Parsing Algorithm* searches first the highest scoring projective parse tree and then it rearranges edges in the tree until the rearrangements does not increase the score for the tree anymore. This technique is computationally expensive for trees with a large number of non-projective edges since it considers to re-attach all edges to any other node until no higher scoring trees can be found. Their argument for the algorithm is that most edges in a tree even in language with lot of non-projective sentences, the portion of non-projective edges are still small and therefore by starting with the highest scoring projective tree, typ-

ically the highest scoring non-projective tree is only a small number of transformations away.

Our experiments showed that with the non-projective Approximate Dependency Parsing Algorithm and a threshold for the improvment of score higher than about 0.7, the parsing accuracy improves even for English slightly. With a threshold of 1.1, we got the highest improvements.

5 Learning Framework

As learning technique, we use Margin Infused Relaxed Algorithm (MIRA) as developed by Crammer et al. (2003) and applied to dependency parsing by McDonald et al. (2005). The online Algorithm in Figure 1 processes one training instance on each iteration, and updates the parameters accordingly.

Algorithm 1: MIRA

$\tau = \{S_x, T_x\}_{x=1}^{X}$ // The set of training data consists
// of sentences and the corresponding dependency trees
$\overrightarrow{w}^{(0)} = 0, \overrightarrow{v} = 0$
for n = 1 **to** N
  for x = 1 **to** X
    w^{i+1} = update w^i according to instance (S_x, T_x)
    $v = v + w^{i+1}$
    $i = i + 1$
  end for
end for
$w = v/(N * X)$

The inner loop iterates over all sentences x of the training set while the outer loop repeats the train i times. The algorithm returns an averaged weight vector and uses an auxiliary weight vector v that accumulates the values of w after each iteration. At the end, the algorithm computes the average of all weight vectors by dividing it by the number of training iterations and sentences. This helps to avoid overfitting, cf. (Collins, 2002).

The update function computes the update to the weight vector w^i during the training so that wrong classified edges of the training instances are possibly correctly classified. This is computed by increasing the weight for the correct features and decreasing the weight for wrong features of the vectors for the tree of the training set $\overrightarrow{f_{T_x}} * w^i$ and the vector for the predicted dependency tree $\overrightarrow{f_{T'_x}} * w^i$.

The update function tries to keep the change to the parameter vector w^i as small as possible for correctly classifying the current instance with a difference at least as large as the loss of the incorrect classifications.

6 Selected Parsing Features

Table 1, 4 and 2 give an overview of the selected features for our system. Similar to Johansson and Nugues (2008), we add the edge labels to each features. In the feature selection, we follow a bit more McDonald and Pereira (2006) since we have in addition the lemmas, morphologic features and the distance between the word forms.

For the parsing and training speed, most important is a fast feature extraction beside of a fast parsing algorithm.

Standard Features	
h-f/l	h-f/l, d-pos
h-pos	h-pos, d-f/l
d-f/l	h-f/l, d-f/l
d-pos	h-pos, d-pos
h-f/l,h-pos	h-f/l, d-f/l, h-pos
d-f/l,d-pos	h-f/l, d-f/l, d-pos
	h-pos, d-pos, h-f/l
	h-pos, d-pos, d-f/l
	h-pos, d-pos, h-f/l, d-f/l

Table 1: Selected standard parsing features. h is the abbrevation for head, d for dependent, g for grandchild, and s for sibling. Each feature contains also the edge label which is not listed in order to save some space. Additional features are build by adding the **dir**ection and the **dist**ance plus the direction. The direction is left if the dependent is left of the head otherwise right. The distance is the number of words between the head and the dependent, if $\leq$5, 6 if $>$5 and 11 if $>$10. $\oplus$ means that an additional feature is built with the previous part plus the following part. f/l represent features that are built once with the form and once with the lemma.

Selected morphologic parsing features.

$\forall$ h-morpheme $\in$ head-morphologic-feature-set **do**
  $\forall$ d-morpheme $\in$ dependent-morphologic-feature-set **do**
    build-feautre: h-pos, d-pos, h-morpheme, d-morpheme

7 Implementation Aspects

In this section, we provide implementation details considering improvements of the parsing and training time. The training of our system (parser) has

Linear Features	Grandchild Features	Sibling Features
h-pos, d-pos, h-pos + 1	h-pos, d-pos, g-pos, dir(h,d), dir(d,g)	d-f/l, s-f/l ⊕ dir(d,s) ⊕dist(d,s)
h-pos, d-pos, h-pos - 1	h-f/l, g-f/l, dir(h,d), dir(d,g)	d-pos, s-f/l ⊕ dir(d,s) ⊕dist(d,s)
h-pos, d-pos, d-pos + 1	d-f/l, g-f/l, dir(h,d), dir(d,g)	d-pos, s-f/l ⊕ dir(d,s)+ ⊕ dist(d,s)
h-pos, d-pos, d-pos - 1	h-pos, g-f/l, dir(h,d), dir(d,g)	d-pos, s-pos ⊕dir(d,s)⊕dist(d,s)
h-pos, d-pos, h-pos - 1, d-pos - 1	d-pos, g-f/l, dir(h,d), dir(d,g)	h-pos, d-pos, s-pos, dir(h,d), dir(d,s) ⊕dist(h,s)
h-f/l, g-pos, dir(h,d), dir(d,g)	h-f/l, s-f/l, dir(h,d), dir(d,s)⊕dist(h,s)	h-pos, s-f/l, dir(h,d), dir(d,s)⊕dist(h,s)
d-f/l, g-pos, dir(h,d), dir(d,g)	d-f/l, s-f/l, dir(h,d), dir(d,s) ⊕dist(h,s)	d-pos, s-f/l, dir(h,d), dir(d,s)⊕dist(h,s)
		h-f/l, s-pos, dir(h,d), dir(d,s)⊕dist(h,s)
		d-f/l, s-pos, dir(h,d), dir(d,s)⊕dist(h,s)

Table 2: Selected Features.

three passes. The goal of the first two passes is to collect the set of possible features of the training set. In order to determine the minimal description length, the feature extractor collects in the first pass all attributes that the features can contain. For each attribute (labels, part-of-speech, etc.), the extractor computes a mapping to a number which is continous from 1 to the count of elements without duplicates.

We enumerate in the same way the feature patterns (e.g. h-pos, d-pos) in order to distinguish the patterns. In the second pass, the extractor builds the features for all training examples which occur in the train set. This means for all edges in the training examples.

We create the features with a function that adds iteratively the attributes of a feature to a number represented with 64 bits and shifts it by the minimal number of bits to encode the attribute and then enumerates and maps these numbers to 32 bit numbers to save even more memory.

Beside this, the following list shows an overview of the most important implementation details to improve the speed:

1. We use as feature vector a custom array implementation with only a list of the features that means without double floating point value.

2. We store the feature vectors for $f(label, w_i, w_j)$, $f(label, w_i, w_j, w_g)$, $f(label, w_i, w_j, w_s)$ etc. in a **compressed** file since otherwise it becomes the bottleneck.

3. After the training, we store only the parameters of the support vector machine which are higher than a threshold of 1×10^{-7}

Table 3 compares system regarding their performance and memory usage. For the shared task, we use the system (3) with integrated labeling.

System	(1)	(2)	(3)
Type	2nd order	2nd order	2nd order
Labeling	separate	separate	integrated
System	baseline	this	this
Training	22 hours	3 hours	15 hours
	7GB	1.5 GB	3 GB
Parsing	2000 ms	50 ms	610 ms
		700 MB	1 GB
LAS	0.86	0.86	0.88

Table 3: Performance Comparison. For the baseline system (1), we used the system of McDonald and Pereira (2006) on a MacPro 2.8 Ghz as well for our implementation (2). For system (3), we use a computer with Intel i7 3.2 Ghz which is faster than the MacPro. For all systems, we use 10 training iterations for the SVM Mira.

8 Semantic Role Labeling

The semantic role labeler is implemented as a pipeline architecture. The components of the pipeline are predicate selection (PS), argument identification (AI), argument classification (AC), and word sense disambiguation (WSD).

In order to select the predicates, we look up the lemmas in the Prob Bank, Nom Bank, etc. if available, cf. (Palmer et al., 2005; Meyers et al., 2004). For all other components, we use the support vector machine MIRA to select and classify the semantic role labels as well as to disambiguate the word senese.

The AI component identifies the arguments of each predicate. It iterates over the predicates and over the words of a sentence. In the case that the score function is large or equal to zero the argument is added to the set of arguments of the predicate in question. Table 5 lists for the attribute identification and semantic role labeling.

The argument classification algorithm labels each

Language	Catalan	Chinese	Czech	English	German	Japanese	Spanish	Czech	English	German
Development Set										
LAS	86.69	76.77	80.75	87.97	86.46	92.37	86.53			
Semantic Unlabeled	93.92	85.09	94.05	91.06	91.61	93.90	93.87			
Semantic Labeled	74.98	75.94	78.07	78.79	72.66	72.86	73.01			
Macro (F1)	80.84	76.48	79.42	84.52	79.56	82.63	79.77			
Test Set								Out-of-domain data		
LAS	86.35	76.51	80.11	@89.88	@87.48	92.21	87.19	76.40	@82.64	@77.34
Semantic Labeled	74.53	75.29	79.02	80.39	75.72	72.76	74.31	78.01	68.44	63.36
Macro (F1)	80.44	75.91	79.57	85.14	81.60	82.51	80.75	77.20	75.55	70.35

Table 4: Syntactic and Semantic Scores. @ indicate values that are the highest scores of all systems.

Features with part-of-speech tags	Features with lemmas	Features with rels
arg, path-len	arg, p-lemma $\oplus$ dir $\oplus$ path-len	arg, a-rel $\oplus$ path-len
arg, p-pos	arg, a-lemma, path, dir	arg, a-pos, p-pos, p-rel $\oplus$ path-len
arg, sub-cat, a-pos, p-pos	arg, p-lemma - 1, a-pos, path-len, dir	arg, p-rel, a-pos, lms-lemma
arg, p-pos, a-pos, a-rmc-pos	arg, p-lemma + 1, a-lemma, path, dir	arg, a-pos, p-pos, a-rel
arg, p-pos, a-pos, a-lmc-pos	arg, p-lemma - 1, a-lemma, path, dir	arg, path-rel
arg, p-pos, a-pos, a-lemma-1	arg, p-lemma - 2, a-lemma, path, dir	arg, p-lemma, a-pos, path-rel
arg, sub-cat, a-lemma, dir, path-len	arg, p-lemma, a-lemma, pathPos, dir	
arg, a-pos, a-lemma + 1	arg, p-lemma, p-lemma + 1	
arg, a-pos, a-lemma + 2	arg, p-lemma, p-lemma - 1, a-pos $\oplus$ dir $\oplus$ path-len	
arg, a-pos, a-lemma-lmc	arg, p-lemma, a-lms-lemma, a-pos $\oplus$ dir $\oplus$ path-len	
arg, p-sub-cat, p-pos $\oplus$ dir	arg, p-lemma, path-len $\oplus$ dir	
arg, p-pos, path, p-parent-lemma	arg, p-lemma, path-len $\oplus$ dir	
arg, p-pos, path, p-parent-pos $\oplus$ dir	arg, a-pos, path	
arg, p-pos, a-pos, familyship(p,a)	arg, p-pos, p-lemma, familyship(p,a)	
arg, path-pos	arg, a-pos, p-lemma, familyship(p,a)	
	arg, p-pos, a-lemma, familyship(p,a)	

Table 5: Argument identification and semantic role labeling Features. p is the abbrivation for predicate and a for argument. For the AI component, the attribute arg is either the value *yes* and *no* and for the SRL component, ars is the role label. *path* is the path in terms of up‘s and down‘s. *pathPos* is a path plus the part-of-speech on the path. *dir* is *left*, if the argument is left of the predicate, *equal* if the predicate and argument are equal, otherwise right. *rmc* is the abbrivation for right most child, *lmc* for left most child, and lms left most sibling. *familiship(x,y)* is a function that computes the relation between two words: self, parent, child, ancestor, decendant and none.

identified argument with a semantic role label. The argument classification algorithm selects with a beam search algorithm the combination of arguments with the highest score.

The last component of our pipeline is the word sense disambiguation. We put this against the intuition at the end of our pipeline since experiments showed that other components could not profit from disambiguated word senses but on the other hand the word sense disambiguation could profit from the argument identification and argument classification. In order to disambiguate, we iterate over the words in the corpus that have more than one sense and take the sense with the highest score.

The average time to execute the SRL pipeline on a sentence is less than 0.15 seconds and the training time for all languages less than 2 hours.

9 Conclusion

We provided a fast implementation with good parsing time and memory footprint. Even if we traded off a lot of the speed improvement by using a more expensive decoder and more attributes to get a higher accuracy.

For some languages, features are not provided or the parser does not profit from using these features. For instance, the English parser does not profit from the lemmas and the Chinese as well as the Japanese

corpus does not have lemmas different from the word forms, etc. Therefore, a possible further accuracy and parsing speed improvement would be to select different features sets for different languages or to leave out some features.

Acknowledgments

This work was supported by the German Academic Exchange Service (DAAD). We gratefully acknowledge this support.

References

Sabine Buchholz and Erwin Marsi. 2006. CoNLL-X Shared Task on Multilingual Dependency Parsing. In *In Proc. of CoNLL*, pages 149–164.

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Pado, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *EMNLP*.

Koby Crammer, Ofer Dekel, Shai Shalev-Shwartz, and Yoram Singer. 2003. Online Passive-Aggressive Algorithms. In *Sixteenth Annual Conference on Neural Information Processing Systems (NIPS)*.

Jason Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pages 340–345, Copenhaen.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeňk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Llu'is Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Miahi Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. In *Proceedings of the 13th CoNLL-2009, June 4-5*, Boulder, Colorado, USA.

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *Proceedings of the Shared Task Session of CoNLL-2008*, Manchester, UK.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

Ryan McDonald and Fernando Pereira. 2006. Online Learning of Approximate Dependency Parsing Algorithms. In *In Proc. of EACL*, pages 81–88.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online Large-margin Training of Dependency Parsers. In *Proc. ACL*, pages 91–98.

Ryan McDonald, Kevin Lerman, Koby Crammer, and Fernando Pereira. 2006. Multilingual Dependency Parsing with a Two-Stage Discriminative Parser. In *Tenth Conference on Computational Natural Language Learning (CoNLL-X)*, pages 91–98.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The nombank project: An interim report. In A. Meyers, editor, *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*, pages 24–31, Boston, Massachusetts, USA, May 2 - May 7. Association for Computational Linguistics.

Joakim. Nivre and Jens Nilsson. 2005. Pseudo-Projective Dependency Parsing. In *In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 99–106.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2004. Memory-Based Dependency Parsing. In *Proceedings of the 8th CoNLL*, pages 49–56, Boston, Massachusetts.

Joakim Nivre, Johan Hall, Sandra Kübler, Rayn McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. The conll 2007 shared task on dependency parsing. In *Proc. of the CoNLL 2007 Shared Task. Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, June.

Martha Palmer and Nianwen Xue. 2009. Adding Semantic Roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

Martha Palmer, Paul Kingsbury, and Daniel Gildea. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. volume 31, pages 71–106.

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 the 12th CoNLL-2008*.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the LREC-2008*, Marrakesh, Morroco.

Exploring Multilingual Semantic Role Labeling

Baoli Li, Martin Emms, Saturnino Luz, Carl Vogel
Department of Computer Science
Trinity College Dublin
Dublin 2, Ireland
{baoli.li,mtemms,luzs,vogel}@cs.tcd.ie

Abstract

This paper describes the multilingual semantic role labeling system of Computational Linguistics Group, Trinity College Dublin, for the CoNLL-2009 SRLonly closed shared task. The system consists of two cascaded components: one for disambiguating predicate word sense, and the other for identifying and classifying arguments. Supervised learning techniques are utilized in these two components. As each language has its unique characteristics, different parameters and strategies have to be taken for different languages, either for providing functions required by a language or for meeting the tight deadline. The system obtained labeled F1 69.26 averaging over seven languages (Catalan, Chinese, Czech, English, German, Japanese, and Spanish), which ranks the system fourth among the seven systems participating the SRLonly closed track.

1 Introduction

Semantic role labeling, which aims at computationally identifying and labeling arguments of predicate words, has become a leading research problem in computational linguistics with the advent of various supporting resources (e.g. corpora and lexicons) (Màrquez et al., 2008). Word semantic dependencies derived by semantic role labeling are assumed to facilitate automated interpretation of natural language texts. Moreover, techniques for automatic annotation of semantic dependencies can also play an important role in adding metadata to corpora for the purposes of machine translation and speech processing. We are currently investigating such techniques as part of our research into integrated language technology in the Center for Next Generation Localization (CNGL, http://www.cngl.ie). The multilingual nature of the CoNLL-2009 shared task on syntactic and semantic dependency analysis, which includes Catalan, Chinese, Czech, English, German, Japanese, and Spanish (Hajič et al., 2009), makes it a good testbed for our research.

We decided to participate in the CoNLL-2009 shared task at the beginning of March, signed the agreement for getting the training data on March 2nd, 2009, and obtained all the training data (especially the part from LDC) on March 4th, 2009. Due to the tight time constraints of the task, we chose to use existing packages to implement our system. These time constraints also meant that we had to resort to less computationally intensive methods to meet the deadline, especially for some large datasets (such as the Czech data). In spite of these difficulties and resource limitations, we are proud to be among the 21 teams who successfully submitted the results[1].

As a new participant, our goals in attending the CoNLL-2009 SRLonly shared task were to gain more thorough knowledge of this line of research and its state-of-the-art, and to explore how well a system quickly assembled with existing packages can fare at this hard semantic analysis problem.

Following the successful approaches taken by the participants of the CoNLL-2008 shared task (Surdeanu et al., 2008) on monolingual syntactic and semantic dependency analysis, we designed and implemented our CoNLL-2009 SRLonly system with pipeline architecture. Two main components are cascaded in this system: one is for disambiguating predicate word sense[2], and the other for identifying and classifying arguments for

[1] According to our correspondence with Dr. Jan Hajič, totally 31 teams among 60 registered ones signed and got the evaluation data.

[2] As predicate words are marked in the CoNLL-2009 datasets, we don't need to identify predicate words.

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 73–78,
Boulder, Colorado, June 2009.

predicate words. Different supervised learning techniques are utilized in these two components. For predicate word sense disambiguation (WSD), we have experimented with three algorithms: SVM, kNN, and Naïve Bayes. Based on experimental results on the development datasets, we chose SVM and kNN to produce our submitted official results. For argument identification and classification, we used a maximum entropy classifier for all the seven datasets. As each language has its unique characteristics and peculiarities within the dataset, different parameters and strategies have to be taken for different languages (as detailed below), either for providing functions required by a language or for meeting the tight deadline. Our official submission obtained 69.26 labeled F1 averaging over the seven languages, which ranks our system fourth among the seven systems in the SRLonly closed track.

The rest of this paper is organized as follows. Section 2 discusses the first component of our system for predicate word sense disambiguation. Section 3 explains how our system detects and classifies arguments with respect to a predicate word. We present experiments in Section 4, and conclude in Section 5.

2 Predicate Word Sense Disambiguation

This component tries to determine the sense of a predicate word in a specific context. As a sense of a predicate word is often associated with a unique set of possible semantic roles, this task is also called role set determination. Based on the characteristics of different languages, we take different strategies in this step, but the same feature set is used for different languages.

2.1 Methods

Intuitively, each predicate word should be treated individually according to the list of its possible senses. We therefore designed an initial solution based on the traditional methods in WSD: represent each sense as a vector from its definition or examples; describe the predicate word for disambiguation as a vector derived from its context; and finally output the sense which has the highest similarity with the current context. We also considered using singular value decomposition (SVD) to overcome the data sparseness problem. Unfortunately, we found this solution didn't work well in our preliminary experiments. The main problem is that the definition of each sense of a predicate word is not available. What we have is just a few example contexts for one sense of a predicate word, and these contexts are often not informative enough for WSD. On the other hand, our limited computing resources could not afford SVD operation on a huge matrix.

We finally decided to take each sense tag as a class tag across different words and transform the disambiguation problem into a normal multi-class categorization problem. For example, in the English datasets, all predicates with "01" as a sense identifier were counted as examples for the class "01". With this setting, a predicate word may be assigned an invalid sense tag. It is an indirect solution, but works well. We think there are at least two possible reasons: firstly, most predicate words take their popular sense in running text. For example, in the English dataset (training and development), 160,477 of 185,406 predicate occurrences (about 86.55%) take their default sense "01". Secondly, predicates may share some common role sets, even though their senses may not be exactly the same, e.g. "tell" and "inform".

Unlike the datasets in other languages, the Japanese dataset doesn't have specialized sense tags annotated for each predicate word, so we simply copy the predicted lemma of a predicate word to its PRED field. For other datasets, we derived a training sample for each predicate word, whose class tag is its sense tag. Then we trained a model from the generated training data with a supervised learning algorithm, and applied the learned model for predicting the sense of a predicate word. This is our base solution.

When transforming the datasets, the Czech data needs some special processing because of its unique annotation format. The sense annotation for a predicate word in the Czech data does not take the form "LEMMA.SENSE". In most cases, no specialized sense tags are annotated. The PRED field of these words only contains "LEMMA". In other cases, the disambiguated senses are annotated with an internal representation, which is given in a predicate word lexicon. We decomposed the internal representation of each predicate word into two parts: word index id and sense tag. For example, from "zvýšení v-w10004f2" we know "v-w10004" is the index id of word "zvýšení", and "f2" is its sense tag. We then use these derived

sense tags as class tags and add a class tag "=" for samples without specialized sense tag.

For each predicate word, we derive a vector describing its context and attributes, each dimension of which corresponds to a feature. We list the feature types in the next subsection. Features appearing only once are removed. The TF*IDF weighting schema is used to calculate the weight of a feature.

Three different algorithms were tried during the development period: support vector machines (SVM), distance-weighted k-Nearest Neighbor (kNN) (Li et al., 2004), and Naïve Bayes with multinomial model (Mccallum and Nigam, 1998). As to the SVM algorithm, we used the robust LIBSVM package (Chang and Lin, 2001), with a linear kernel and default values for other parameters. The algorithms achieving the best results in our preliminary experiments are chosen for different languages: SVM for Catalan, Chinese, and Spanish; kNN for German (k=20).

We used kNN for English (k=20) and Czech (k=10) because we could not finish training with SVM on these two datasets in limited time. Even with kNN algorithm, we still had trouble with the English and Czech datasets, because thousands of training samples make the prediction for the evaluation data unacceptably slow. We therefore had to further constrain the search space for a new predicate word to those samples containing the same predicate word. If there are not samples containing the same predicate word in the training data, we will assign it the most popular sense tag (e.g. "01" for English).

How to use the provided predicate lexicons is a challenging issue. Lexicons for different languages take different formats and the information included in different lexicons is quite different. We derived a sense list lexicon from the original predicate lexicon for Chinese, Czech, English, and German. Each entry in a sense list lexicon contains a predicate word, its internal representation (especially for Czech), and a list of sense tags that the predicate can have. Then we obtained a variant of our base solution, which uses the sense list of a predicate word to filter impossible senses. It works as follows:

- Disambiguate a new predicate with the base solution;
- Choose the most possible sense from all the candidate senses obtained in step 1: if the base classifier doesn't output a vector of probabilities for classes, only check whether the predicted one is a valid sense for the predicate;
- If there is not a valid sense for a new predicate (including the cases where the predicate does not have an entry in the sense list lexicon), output the most popular sense tag;

Unfortunately, preliminary experiments on the German and Chinese datasets didn't support to include such a post-processing stage. The performance with this filtering became a little worse. Therefore, we decided not to use it generally, but one exception is for the Czech data.

With kNN algorithm, we can greatly reduce the time for training the Czech data, but we do have problem with prediction, as there are totally 469,754 samples in the training dataset. It's a time-consuming task to calculate the similarities between a new sample and all the samples in the training dataset to find its k nearest neighbors, thus we have to limit the search space to those samples that contain the predicate word for disambiguation. To process unseen predicate words, we used the derived sense list lexicon: if a predicate word for disambiguation is out of the sense list lexicon, we simply copy its predicted lemma to the PRED field; if no sample in the training dataset has the same predicate word, we take its first possible sense in the sense list lexicon. With this strategy, our system can process the huge Czech dataset in short time.

2.2 Features

The features we used in this step include[3]:

a. [Lemma | (Lemma with POS)] of all words in the sentence;
b. Attributes of predicate word, which is obtained from PFEAT field by splitting the field at symbol "|" and removing the invalid attribute of "*";
c. [Lemma | POS] bi-grams of predicate word and its [previous | following] one word;
d. [Lemma | POS] tri-grams of predicate word and its [previous | following] two words;
e. [Lemma | (Lemma with POS)] of its most [left | right] child;
f. [(Lemma+Dependency_Relation+Lemma) | (POS +Dependency_Relation+POS)] of predicate word and its most [left | right] child;

[3] We referred to those CoNLL-2008 participants' reports, e.g. (Ciaramita et al., 2008), when we designed the feature sets for the two components.

g. [Lemma | (Lemma with POS)] of the head of the predicate word;
h. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS)] of predicate word and its head;
i. [Lemma | (Lemma with POS)] of its [previous | following] two brothers;
j. [Lemma | POS | (Dependency relation)] bi-gram of predicate word and its [previous | following] one brother;
k. [Lemma | POS | (Dependency relation)] tri-gram of predicate word and its [previous | following] two brothers.

3 Argument Identification and Classification

The second component of our system is used to detect and classify arguments with respect to a predicate word. We take a joint solution rather than solve the problem in two consecutive steps: argument identification and argument classification.

3.1 Methods

By introducing an additional argument type tag "_" for non-arguments, we transformed the two tasks (i.e. argument identification and argument classification) into one multi-class classification problem. As a word can play different roles with respect to different predicate words and a predicate word can be an argument of itself, we generate a training set by deriving a training example from each word-predicate pair. For example, if a sentence with two predicates has 7 words, we will derive 7*2=14 training examples. Therefore, the number of training examples generated in this step will be around L times larger than that obtained in the previous step, where L is the average length of sentences.

We chose to use maximum entropy algorithm in this step because of its success in the CoNLL-2008 shared task (Surdeanu et al., 2008). Le Zhang's maximum entropy package (Zhang, 2006) is integrated in our system.

The Czech data cause much trouble again for us, as the training data derived by the above strategy became even larger. We had to use a special strategy for the Czech data: we selectively chose word-predicate pairs for generating the training dataset. In other words, not all possible combinations are used. We chose the following words with respect to each predicate: the first and the last two words of a sentence; the words between the predicate and any argument of it; two words before the predicate or any argument; and two words after the predicate or any argument.

In the Czech and Japanese data, some words may play multiple roles with respect to a predicate word. We thus have to consider multi-label classification problem (Tsoumakas and Katakis, 2007) for these two languages' data. We tried the following two solutions:

- Take each role type combination as a class and transform the multi-label problem to a single-label classification problem;
- Classify a word with a set of binary classifiers: consider each role type individually with a binary classifier; any possible role type will be output; if no role type is obtained after considering all the role types, the role type with the highest confidence value will be output; and, if "_" is output with any other role type, remove it.

We used the second solution in our official submission, but we finally found these two solutions perform almost the same. The performance difference is very small. We found the cases with multi-labels (actually at most two) in the training data are very limited: 690 of 414,326 in the Czech data and 113 of 46,663 in the Japanese data.

3.2 Features

The features we used in this step include:

a. Whether the current word is a predicate;
b. [Lemma | POS] of current word and its [previous | following] one word;
c. [Lemma | POS] bi-grams of current word and its [previous | following] one word;
d. POS tri-grams of current word, its previous word and its following word;
e. Dependency relation of current word to its head;
f. [Lemma | POS] of the head of current word;
g. [Lemma | POS] bi-grams of current word and its head;
h. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS)] of current word and its head;
i. [Lemma | POS] of its most [left | right] child;
j. [Lemma | POS] bi-grams of current word and its most [left | right] child;
k. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS) of current word and its most [left | right] child;
l. The number of children of the current word and the predicate word;
m. Attributes of the current word, which is obtained from PFEAT field by splitting the field at symbol "|" and removing the invalid attribute of "*";
n. The sense tag of the predicate word;

o. [Lemma | POS] of the predicate word and its head;
p. Dependency relation of the predicate word to its head;
q. [Lemma | POS] bi-grams of the predicate word and its head;
r. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS)] of the predicate word and its head;
s. [Lemma | POS] of the most [left | right] child of the predicate word;
t. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS)] of predicate word and its head;
u. [Lemma | POS] bi-gram of the predicate word and its most [left | right] child;
v. [(Lemma+Dependency_Relation+Lemma) | (POS+Dependency_Relation+POS)] of the predicate word and its most [left | right] child;
w. The relative position of the current word to the predicate one: before, after, or on;
x. The distance of the current word to the predicate one;
y. The relative level (up, down, or same) and level difference on the syntactic dependency tree of the current word to the predicate one;
z. The length of the shortest path between the current word and the predicate word.

4 Experiments

4.1 Datasets

The datasets of the CoNLL-2009 shared task contain seven languages: Catalan (CA), Chinese (CN), Czech (CZ), English (EG), German (GE), Japanese (JP), and Spanish (SP). The training and evaluation data of each language (Taulé et al., 2008; Xue et al., 2008; Hajič et al., 2006; Palmer et al., 2002; Burchardt et al., 2006; Kawahara et al., 2002) have been converted to a uniform CoNLL Shared Task format. Each participating team is required to process all seven language datasets.

Lanuage	CA	CN	CZ	EN	GE	JP	SP
Size (KB)	48974	41340	**94284**	58155	41091	8948	52430
# of Sentences	14924	24039	**43955**	40613	38020	4643	15984
# of Predicate words	42536	110916	**469754**	185404	17988	27251	48900
Avg. # of Predicates per sentence	2.85	4.61	**10.69**	4.57	0.47	5.87	3.06
popular sense tag	a2 (37%)	01 (90%)	= (81%)	01 (87%)	1 (75%)	= (100%)	a2 (39%)

Table 1. Statistical information of the seven language datasets (training and development).

Table 1 shows some statistical information of both training and development data for each language. The total size of the uncompressed original data without lexicons is about 345MB. The Czech dataset is the largest one containing 43,955 sentences and 469,754 predicate words, while the Japanese dataset the smallest one. On average, 10.69 predicate words appear in a Czech sentence, while only 0.47 predicate words exist in a German sentence. The most popular sense tag in the Czech datasets is "=", which means the PRED field has the same value as the PLEMMA field or the FORM field. About 81% of Czech predicate words take this value.

4.2 Experimental Results

F1 is used as the main evaluation metric in the CoNLL-2009 shared task. As to the SRLonly track, a joint semantic labeled F1, which considers predicate word sense disambiguation and argument labeling equally, is used to rank systems.

Avg.	CA	CN	CZ	EG	GE	JP	SP
69.26	**74.06**	70.37	57.46	69.63	67.76	72.03	73.54

Table 2. Official results of our system.

Table 2 gives the official results of our system on the evaluation data. The system obtained the best result (74.06) on the Catalan data, but performed very poor (57.46) on the Czech data. Except the Czech data, our system performs quite stable on the other six language data with mean of 71.23 and standard deviation of 2.42.

	Avg.	CA	CN	CZ	EG	GE	JP	SP
Overall F1	**69.47**	**74.12**	**70.52**	**57.57**	**70.24**	**67.97**	**72.17**	**73.68**
Pred. WSD F1	**86.9**	**84.42**	**94.54**	**72.23**	**92.98**	**81.09**	**99.07**	**83.96**
Arg I&C F1	**57.24**	**69.29**	**57.71**	**33.19**	**58.25**	**60.64**	**52.72**	**68.86**
Arg I&C PR	69.77	73.43	72.48	62.14	70.14	66.63	69.37	74.23
Arg I&C RE	49.77	65.6	47.94	22.64	49.81	55.64	42.52	64.21

Table 3. Results of our system after fixing a minor bug.

After submitting the official results, we found and fixed a minor bug in the implementation of the second component. Table 3 presents the results of our system after fixing this bug. The overall performance doesn't change much. We further analyzed the bottlenecks by checking the performance of different components.

At the predicate WSD part, our system works reasonable with labeled F1 86.9, but the performance on the Czech data is lower than that of a baseline system that constantly chooses the most popular sense tag. If we use this baseline solution,

we can get predicate WSD F1 78.66, which further increases the overall labeled F1 on the Czech data to 61.68 from 57.57 and the overall labeled F1 over the seven languages to 70.05 from 69.47.

From table 3, we can see our system performs relatively poorly for argument identification and classification (57.24 vs. 86.9). The system seems too conservative for argument identification, which makes the recall very lower. We explored some strategies for improving the performance of the second component, e.g. separating argument identification and argument classification, and using feature selection (with DF threshold) techniques, but none of them helps much. We are thinking the features currently used may not be effective enough, which deserves further study.

5 Conclusion and Future Work

In this paper, we describe our system for the CoNLL-2009 shared task -- SRLonly closed track. Our system was built on existing packages with a pipeline architecture, which integrated two cascaded components: predicate word sense disambiguation and argument identification and classification. Our system performs well at disambiguating the sense of predicate words, but poorly at identifying and classifying arguments. In the future, we plan to explore much effective features for argument identification and classification.

Acknowledgments

This research was funded by Science Foundation Ireland under the CNGL grant. We used the IITAC Cluster in our initial experiments. We thank IITAC, the HEA, the National Development Plan and the Trinity Centre for High Performance Computing for their support. We are also obliged to John Keeney for helping us running our system on the CNGL servers.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó and Manfred Pinkal. 2006. The SALSA Corpus: a German Corpus Resource for Lexical Semantics. *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006).* Genoa, Italy.

Chih-Chung Chang and Chih-Jen Lin. 2001. LIBSVM: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Massimiliano Ciaramita, Giuseppe Attardi, Felice Dell'Orletta, and Mihai Surdeanu. 2008. DeSRL: A Linear-Time Semantic Role Labeling System. *Proceedings of the CoNLL-2008.*

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antonia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue and Yi Zhang. 2009. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009).* Boulder, Colorado, USA.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová and Zdeněk Žabokrtský. 2006. *The Prague Dependency Treebank 2.0.* Linguistic Data Consortium, USA. ISBN 1-58563-370-4.

Daisuke Kawahara, Sadao Kurohashi and Koiti Hasida. 2002. Construction of a Japanese Relevance-tagged Corpus. *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002).* Las Palmas, Spain.

Baoli Li, Qin Lu and Shiwen Yu. 2004. An Adaptive k-Nearest Neighbor Text Categorization Strategy. *ACM Transactions on Asian Language Information Processing*, 3(4): 215-226.

Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski and Suzanne Stevenson. 2008. Semantic Role Labeling: An Introduction to the Special Issue. *Computational Linguistics*, 34(2):145-159.

Andrew Mccallum and Kamal Nigam. 1998. A Comparison of Event Models for Naive Bayes Text Classification. *Proceedings of AAAI/ICML-98 Workshop on Learning for Text Categorization*.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluis Marquez and Joakim Nivre. 2008. The CoNLL-2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies. *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL-2008).*

Mariona Taulé, Maria Antònia Martí and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008).* Marrakech, Morocco.

Grigorios Tsoumakas and Ioannis Katakis. 2007. Multi-Label Classification: An Overview. *International Journal of Data Warehousing and Mining*, 3(3):1-13.

Nianwen Xue and Martha Palmer. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering,* 15(1):143-172.

Le Zhang. 2006. Maximum Entropy Modeling Toolkit for Python and C++. Software available at http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html.

A Second-Order Joint Eisner Model for Syntactic and Semantic Dependency Parsing

Xavier Lluís **Stefan Bott** **Lluís Màrquez**
TALP Research Center – Software Department (LSI)
Technical University of Catalonia (UPC)
{xlluis,sbott,lluism}@lsi.upc.edu

Abstract

We present a system developed for the CoNLL-2009 Shared Task (Hajič et al., 2009). We extend the Carreras (2007) parser to jointly annotate syntactic and semantic dependencies. This state-of-the-art parser factorizes the built tree in second-order factors. We include semantic dependencies in the factors and extend their score function to combine syntactic and semantic scores. The parser is coupled with an on-line averaged perceptron (Collins, 2002) as the learning method. Our averaged results for all seven languages are 71.49 macro F_1, 79.11 LAS and 63.06 semantic F_1.

1 Introduction

Systems that jointly annotate syntactic and semantic dependencies were introduced in the past CoNLL-2008 Shared Task (Surdeanu et al., 2008). These systems showed promising results and proved the feasibility of a joint syntactic and semantic parsing (Henderson et al., 2008; Lluís and Màrquez, 2008).

The Eisner (1996) algorithm and its variants are commonly used in data-driven dependency parsing. Improvements of this algorithm presented by McDonald et al. (2006) and Carreras (2007) achieved state-of-the-art performance for English in the CoNLL-2007 Shared Task (Nivre et al., 2007). Johansson and Nugues (2008) presented a system based on the Carreras' extension of the Eisner algorithm that ranked first in the past CoNLL-2008 Shared Task. We decided to extend the Carreras (2007) parser to jointly annotate syntactic and semantic dependencies.

The present year Shared Task has the incentive of being multilingual with each language presenting their own particularities. An interesting particularity is the direct correspondence between syntactic and semantic dependencies provided in Catalan, Spanish and Chinese. We believe that these correspondences can be captured by a joint system. We specially look at the syntactic-semantic alignment of the Catalan and Spanish datasets.

Our system is an extension of the Lluís and Màrquez (2008) CoNLL-2008 Shared Task system. We introduce these two following novelties:

- An extension of the second-order Carreras (2007) algorithm to annotate semantic dependencies.
- A combined syntactic-semantic scoring for Catalan and Spanish to exploit the syntactic-semantic mappings.

The following section outlines the system architecture. The next sections present in more detail the system novelties.

2 Architecture

The architecture consists on four main components: 1) Preprocessing and feature extraction. 2) Syntactic preparsing. 3) Joint syntactic-semantic parsing. 4) Predicate classification.

The preprocessing and feature extraction is intended to ease and improve the performance of the parser precomputing a binary representation of

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 79–84,
Boulder, Colorado, June 2009.

each sentence features. These features are borrowed from existing and widely-known systems (Xue and Palmer, 2004; McDonald et al., 2005; Carreras et al., 2006; Surdeanu et al., 2007).

The following step is a syntactic pre-parse. It is only required to pre-compute additional features (e.g., syntactic path, syntactic frame) from the syntax. These new features will be used for the semantic role component of the following joint parser.

The joint parser is the core of the system. This single algorithm computes the complete parse that optimizes a score according to a function that depends on both syntax and semantics. Some of the required features that could be unavailable or expensive to compute at that time are provided by the previous syntactic pre-parse.

The predicate sense classification is performed as the last step. Therefore no features representing the predicate sense are employed during the training. The predicates are labeled with the most frequent sense extracted from the training corpus.

No further postprocessing is applied.

3 Second-order Eisner model

The Carreras' extension of the Eisner inference algorithm is an expensive $O(n^4)$ parser. The number of assignable labels for each dependency is a hidden multiplying constant in this asymptotic cost.

We begin describing a first-order dependency parser. It receives a sentence x and outputs a dependency tree y. A dependency, or first-order factor, is defined as $f_1 = \langle h, m, l \rangle$. Where h is the head token, m the modifier and l the syntactic label. The score for this factor f_1 is computed as:

$$\text{score}_1(f_1, x, \mathbf{w}) = \phi(h, m, x) \cdot \mathbf{w}^{(l)}$$

Where $\mathbf{w}^{(l)}$ is the weight vector for the syntactic label l and ϕ a feature extraction function.

The parser outputs the best tree y^* from the set $\mathcal{T}(x)$ of all projective dependency trees.

$$y^*(x) = \underset{y \in \mathcal{T}(x)}{\text{argmax}} \sum_{f_1 \in y} score(f_1, x, \mathbf{w})$$

The second-order extension decomposes the dependency tree in factors that include some children of the head and modifier. A second-order factor is:

$$f_2 = \langle h, m, l, c_h, c_{mo}, c_{mi} \rangle$$

where c_h is the daughter of h closest to m within the tokens $[h, \ldots, m]$; c_{mo} is the outermost daughter of m outside $[h, \ldots, m]$; and $c_m i$ is the furthest daughter of m inside $[h, \ldots, m]$.

The score for these new factors is computed by

$$\begin{aligned}\text{score}_2(f_2, x, \mathbf{w}) = \phi(h, m, x) \cdot \mathbf{w}^{(l)} + \\ \phi(h, m, c_h, x) \cdot \mathbf{w}^{(l)}_{c_h} + \\ \phi(h, m, c_{mi}, x) \cdot \mathbf{w}^{(l)}_{c_{mi}} + \\ \phi(h, m, c_{mo}, x) \cdot \mathbf{w}^{(l)}_{c_{mo}}\end{aligned}$$

The parser builds the best-scoring projective tree factorized in second-order factors. The score of the tree is also defined as the sum of the score of its factors.

3.1 Joint second-order model

We proceeded in an analogous way in which the Lluís and Màrquez (2008) extended the first-order parser. That previous work extended a first-order model by including semantic labels in first-order dependencies.

Now we define a second-order joint factor as:

$$f_{2\text{syn-sem}} = \langle h, m, l, c_h, c_{mo}, c_{mi}, l_{semp_1}, \ldots, l_{semp_q} \rangle$$

Note that we only added a set of semantic labels $l_{semp_1}, \ldots, l_{semp_q}$ to the second-order factor. Each one of these semantic labels represent, if any, one semantic relation between the argument m and the predicate p_i. There are q predicates in the sentence, labeled $p_1, \ldots, p_q$.

The corresponding joint score to a given joint factor is computed by adding a semantic score to the previously defined *score*$_2$ second-order score function:

$$\begin{aligned}\text{score}_{2\text{syn-sem}}(f_{2\text{syn-sem}}, x, \mathbf{w}) = \\ \text{score}_2(f_2, x, \mathbf{w}) + \\ \sum_{p_i} \frac{\text{score}_{\text{sem}}(h, m, p_i, l_{semp_i}, x, \mathbf{w})}{q}\end{aligned}$$

where,

$$\begin{aligned}\text{score}_{\text{sem}}(h, m, p_i, l_{sem}, x, \mathbf{w}) = \\ \phi_{sem}(h, m, p_i, x) \cdot \mathbf{w}^{(l_{sem})}\end{aligned}$$

We normalize the semantic score by the number of predicates q. The semantic score is computed as a score between m and each sentence predicate p_i. No second-order relations are considered in these score functions. The search of the best c_h, c_{mo} and c_{mi} is independent of the semantic components of the factor. The computational cost of the algorithm is increased by one semantic score function call for every m, h, and p_i combination. The asymptotic cost of this operation is $O(q \cdot n^2)$ and it is sequentially performed among other $O(n^2)$ operations in the main loop of the algorithm.

Algorithm 1 Extension of the Carreras (2007) algorithm

$C[s][t][d][m] \leftarrow 0, \forall s, t, d, m$
$O[s][t][d][l] \leftarrow 0, \forall s, t, d, l$
for $k = 1, \ldots, n$ **do**
  for $s = 0, \ldots, n - k$ **do**
    $t \leftarrow s + k$
    $\forall l \quad O[s][t][\leftarrow][l] = \max_{r,cmi,ch}$
      $C[s][r][\rightarrow][cmi] + C[r+1][t][\leftarrow][ch]$
      $+\text{score}(t, s, l) + \text{score}_{cmi}(t, s, c_{mi}, l) +$
      $\text{score}_{ch}(t, s, l, c_h) +$
      $\sum_{p_i} \max_{l_{sem}} \text{scoresem}(t, s, p_i, l_{sem})/q$

    $\forall l \quad O[s][t][\rightarrow][l] = \max_{r,cmi,ch}$
      $C[s][r][\rightarrow][ch] + C[r+1][t][\leftarrow][cmi] +$
      $\text{score}(s, t, l) + \text{score}_{cmi}(s, t, c_{mi}, l) +$
      $\text{score}_{ch}(s, t, l, c_h) +$
      $\sum_{p_i} \max_{l_{sem}} \text{scoresem}(t, s, p_i, l_{sem})/q$

    $\forall m \quad C[s][t][\leftarrow][m] = \max_{l,cmo}$
      $C[s][m][\leftarrow][cmo] + O[m][t][\leftarrow][l] +$
      $\text{score}_{cmo}(s, m, l, c_{mo})$

    $\forall m \quad C[s][t][\rightarrow][m] = \max_{l,cmo}$
      $O[s][m][\rightarrow][l] + C[m][t][\rightarrow][cmo] +$
      $\text{score}_{cmo}(m, t, l, c_{mo})$
  end for
end for

Our implementation slightly differs from the original Carreras algorithm description. The main difference is that no specific features are extracted for the second-order factors. This allows us to reuse the feature extraction mechanism of a first-order parser.

Algorithm 1 shows the Carreras' extension of the Eisner algorithm including our proposed joint semantic scoring.

The tokens s and t represent the start and end tokens of the current substring, also called span. The direction $d \in \{\leftarrow, \rightarrow\}$ defines whether t or s is the head of the last dependency built inside the span. The score functions $score_{ch}$,$score_{cmi}$ and $score_{cmo}$ are the linear functions that build up the previously defined second-order global score, e.g., $score_{ch} = \phi(h, m, c_h, x) \cdot \mathbf{w}_{c_h}^{(l)}$. The two tables C and O maintain the dynamic programming structures.

Note that the first steps of the inner loop are applied for all l, the syntactic label, but the semantic score function does not depend on l. Therefore the best semantic label can be chosen independently.

For simplicity, we omitted the weight vectors required in each score function and the backpointers tables to save the local decisions. We also omitted the definition of the domain of some variables. Moreover, the filter of the set of assignable labels is not shown. A basic filter regards the POS of the head and modifier to filter out the set of possible arguments for each predicate. Another filter extract the set of allowed arguments for each predicate from the frames files. These last filters were applied to the English, German and Chinese.

3.2 Catalan and Spanish joint model

The Catalan and Spanish datasets (Taulé et al., 2008) present two interesting properties. The first property, as previously said, is a direct correspondence between syntactic and semantic labels. The second interesting property is that all semantic dependencies exactly overlap with the syntactic tree. Thus the semantic dependency between a predicate and an argument always has a matching syntactic dependency between a head and a modifier. The Chinese data also contains direct syntactic-semantic mappings. But due to the Shared Task time constraints we did not implemented a specific parsing method for this language.

The complete overlap between syntax and semantics can simplify the definition of a second-order joint factor. In this case, a second-order factor will only have, if any, one semantic dependency. We only allow at most one semantic relation l_{sem} between the head token h and the modifier m. Note that h must be a sentence predicate and m its argument if

l_{sem} is not null. We extend the second-order factors with a single and possibly null semantic label, i.e., $f_{2\text{syn-sem}} = \langle h, m, l, c_h, c_{mo}, c_{mi}, l_{sem} \rangle$. This slightly simplifies the scoring function:

$$\begin{aligned} \text{score}_{2\text{syn-sem}}(f_{2\text{syn-sem}}, x, \mathbf{w}) = \\ \text{score}_2(f_2, x, \mathbf{w}) + \\ \alpha \cdot \text{score}_{\text{sem}}(h, m, x, \mathbf{w}) \end{aligned}$$

where α is an adjustable parameter of the model and,

$$\text{score}_{\text{sem}}(h, m, x, \mathbf{w}) = \phi_{sem}(h, m, x) \cdot \mathbf{w}^{(l_{sem})}$$

The next property that we are intended to exploit is the syntactic-semantic mappings. These mappings define the allowed combinations of syntactic and semantic labels. The label combinations can only be exploited when there is semantic dependency between the head h and the modifier m of a factor. An argument identification classifier determines the presence of a semantic relation, given h is a predicate. In these cases we only generate factors that are compliant with the mappings. If a syntactic label has many corresponding semantic labels we will score all of them and select the combination with the highest score.

The computational cost is not significantly increased as there is a bounded number of syntactic and semantic combinations to score. In addition, the only one-argument-per-factor constraint reduces the complexity of the algorithm with respect to the previous joint extension.

We found some inconsistencies in the frames files provided by the organizers containing the correspondences between syntax and semantics. For this reason we extracted them directly from the corpus. The extracted mappings discard the 7.9% of the correct combinations in the Catalan development corpus that represent a 1.7% of its correct syntactic dependencies. The discarded semantic labels are the 5.14% for Spanish representing the 1.3% of the syntactic dependencies.

4 Results and discussion

Table 1 shows the official results for all seven languages, including out-of-domain data labeled as *ood*. The high computational cost of the second-order models prevented us from carefully tuning the system parameters. After the shared task evaluation deadline, some bug were corrected, improving the system performance. The last results are shown in parenthesis.

The combined filters for Catalan and Spanish hurt the parsing due to the discarded correct labels but we believe that this effect is compensated by an improved precision in the cases where the correct labels are not discarded. For example, in Spanish these filters improved the syntactic LAS from 85.34 to 86.77 on the development corpus using the gold syntactic tree as the pre-parse tree.

Figure 1 shows the learning curve for the English and Czech language. The results are computed in the development corpus. The semantic score is computed using gold syntax and gold predicate sense classification. We restricted the learning curve to the first epoch. Although the this first epoch is very close to the best score, some languages showed improvements until the fourth epoch. In the figure we can see better syntactic results for the joint system with respect to the syntactic-only parser. We should not consider this improvement completely realistic as the semantic component of the joint system uses gold features (i.e., a gold pre-parse). Nonetheless, it points that a highly accurate semantic component could improve the syntax.

Table 2 shows the training time for a second-order syntactic and joint configurations of the parser. Note that the time per instance is an average and some sentences could require a significantly higher time. Recall that our parser is $O(n^4)$ dependant on the sentence length. We discarded large sentences during training for efficiency reasons. We discarded sentences with more than 70 words for all languages except for Catalan and Spanish where the threshold was set to 100 words in the syntactic parser. This larger number of sentences is aimed to improve the syntactic performance of these languages. The shorter sentences used in the joint parsing and the pruning of the previously described filters reduced the training time for Catalan and Spanish. The amount of main memory consumed by the system is 0.5–1GB. The machine used to perform the computations is an *AMD64 Athlon 5000+*.

	avg	cat	chi	cze	eng	ger	jap	spa
macro F_1	71.49 (74.90)	56.64 (73.21)	66.18 (70.91)	75.95	81.69	72.31	81.76	65.91 (68.46)
syn LAS	79.11 (82.22)	64.21(84.20)	70.53 (70.90)	75.00	87.48	81.94	91.55	83.09 (84.48)
semantic F_1	63.06 (67.41)	46.79 (61.68)	59.72 (70.88)	76.90	75.86	62.66	71.60	47.88 (52.30)
ood macro F_1	71.92	-	-	74.56	73.91	67.30	-	-
ood syn LAS	75.09	-	-	72.11	80.92	72.25	-	-
ood sem F_1	68.74	-	-	77.01	66.88	62.34	-	-

Table 1: Overall results. In parenthesis post-evaluation results.

	cat	chi	cze	eng	ger	jap	spa
syntax only (s/sentence)	18.39	8.07	3.18	2.56	1.30	1.07	15.31
joint system (s/sentence)	10.91	9.49	3.99	3.13	2.36	1.25	12.29

Table 2: Parsing time per sentence.

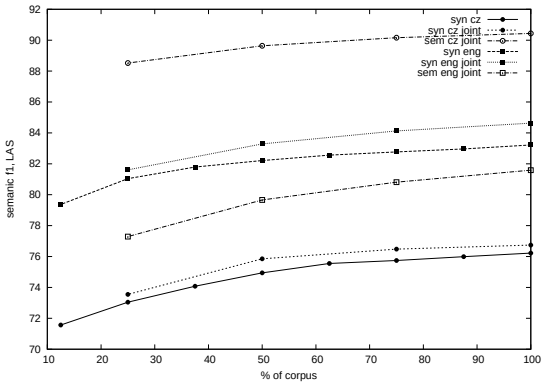

Figure 1: Learning curves for the syntactic-only and joint parsers in Czech and English.

5 Conclusion

We have shown that a joint syntactic-semantic parsing can be based on the state-of-the-art Carreras (2007) parser at an expense of a reasonable cost. Our second-order parser still does not reproduce the state-of-the art results presented by similar systems (Nivre et al., 2007). Although we achieved mild results we believe that a competitive system based in our model can be built. Further tuning is required and a complete set of new second-order features should be implemented to improve our parser.

The multilingual condition of the task allows us to evaluate our approach in seven different languages. A detailed language-dependent evaluation can give us some insights about the strengths and weaknesses of our approach across different languages. Unfortunately we believe that this objective was possibly not accomplished due to the time constraints.

The Catalan and Spanish datasets presented interesting properties that could be exploited. The mapping between syntax and semantics should be specially useful for a joint system. In addition the semantic dependencies for these languages are aligned with the projective syntactic dependencies, i.e., the predicate-argument pairs exactly match syntactic dependencies. This is a useful property to simultaneously build joint dependencies.

6 Future and ongoing work

Our syntactic and semantic parsers, as many others, is not exempt of bugs. Furthermore, very few tuning and experimentation was done during the development of our parser due to the Shared Task time constraints. We believe that we still did not have enough data to fully evaluate our approach. Further experimentation is required to asses the improvement of a joint architecture vs. a pipeline architecture. Also a careful analysis of the system across the different languages is to be performed.

Acknowledgments

We thank the corpus providers (Taulé et al., 2008; Palmer and Xue, 2009; Hajič et al., 2006; Surdeanu et al., 2008; Burchardt et al., 2006; Kawahara et al., 2002) for their effort in the annotation and conversion of the seven languages datasets.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

Xavier Carreras, Mihai Surdeanu, and Lluís Màrquez. 2006. Projective dependency parsing with perceptron. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-2006)*.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proceedings of the 11th Conference on Computational Natural Language Learning (CoNLL-2007)*.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing*.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

James Henderson, Paola Merlo, Gabriele Musillo, and Ivan Titov. 2008. A latent variable model of synchronous parsing for syntactic and semantic dependencies. In *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*, Manchester, UK.

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic–semantic analysis with propbank and nombank. In *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*, Manchester, UK.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

Xavier Lluís and Lluís Màrquez. 2008. A joint model for parsing syntactic and semantic dependencies. In *Proceedings of the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*, Manchester, UK.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-2006)*.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*.

J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL 2007 shared task on dependency parsing.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

Mihai Surdeanu, Lluís Màrquez, Xavier Carreras, and Pere R. Comas. 2007. Combination strategies for semantic role labeling. *Journal of Artificial Intelligence Research*.

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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco.

Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP-2004)*.

Multilingual Semantic Role Labelling with Markov Logic

Ivan Meza-Ruiz[*] **Sebastian Riedel**[†‡]
[*]School of Informatics, University of Edinburgh, UK
[†]Department of Computer Science, University of Tokyo, Japan
[‡]Database Center for Life Science, Research Organization of Information and System, Japan
[*]I.V.Meza-Ruiz@sms.ed.ac.uk [†] sebastian.riedel@gmail.com

Abstract

This paper presents our system for the CoNLL 2009 Shared Task on Syntactic and Semantic Dependencies in Multiple Languages (Hajič et al., 2009). In this work we focus only on the Semantic Role Labelling (SRL) task. We use Markov Logic to define a joint SRL model and achieve the third best average performance in the closed Track for SRLOnly systems and the sixth including for both SRLOnly and Joint systems.

1 Markov Logic

Markov Logic (ML, Richardson and Domingos, 2006) is a Statistical Relational Learning language based on First Order Logic and Markov Networks. It can be seen as a formalism that extends First Order Logic to allow formulae that can be violated with some penalty. From an alternative point of view, it is an expressive template language that uses First Order Logic formulae to instantiate Markov Networks of repetitive structure.

In the ML framework, we model the SRL task by first introducing a set of logical predicates[1] such as *word(Token,Ortho)* or *role(Token,Token,Role)*. In the case of *word/2* the predicate represents a word of a sentence, the type *Token* identifies the position of the word and the type *Ortho* its orthography. In the case of *role/3*, the predicate represents a semantic role. The first token identifies the position of the predicate, the second the syntactic head of the argument and finally the type Role signals the semantic role label. We will refer to predicates such as *word/2* as *observed* because they are known in advance. In contrast, *role/3* is *hidden* because we need to infer it at test time.

With the ML predicates we specify a set of weighted first order formulae that define a distribution over sets of ground atoms of these predicates (or so-called *possible worlds*). A set of weighted formulae is called a *Markov Logic Network* (MLN). Formally speaking, an MLN M is a set of pairs (ϕ, w) where ϕ is a first order formula and w a real weight. M assigns the probability

$$p(\mathbf{y}) = \frac{1}{Z} \exp\left(\sum_{(\phi,w)\in M} w \sum_{\mathbf{c}\in C^{\phi}} f^{\phi}_{\mathbf{c}}(\mathbf{y})\right) \quad (1)$$

to the possible world $\mathbf{y}$. Here C^{ϕ} is the set of all possible bindings of the free variables in ϕ with the constants of our domain. $f^{\phi}_{\mathbf{c}}$ is a feature function that returns 1 if in the possible world $\mathbf{y}$ the *ground formula* we get by replacing the free variables in ϕ by the constants in $\mathbf{c}$ is true and 0 otherwise. Z is a normalisation constant. Note that this distribution corresponds to a Markov Network (the so-called *Ground Markov Network*) where nodes represent ground atoms and factors represent ground formulae.

In this work we use 1-best MIRA (Crammer and Singer, 2003) Online Learning in order to train the weights of an MLN. To find the SRL assignment with maximal *a posteriori* probability according to an MLN and observed sentence, we use Cutting Plane Inference (CPI, Riedel, 2008) with ILP base solver. This method is used during both test time and the MIRA online learning process.

[1]In the cases were is not obvious whether we refer to SRL or ML predicates we add the prefix SRL or ML, respectively.

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 85–90,
Boulder, Colorado, June 2009.

2 Model

In order to model the SRL task in the ML framework, we propose four hidden predicates. Consider the example of the previous section:

argument/1 indicates the phrase for which its head is a specific position is an SRL argument. In our example *argument(2)* signals that the phrase for which the word in position 2 is its head is an argument (i.e., *Ms. Haag*).

hasRole/2 relates a SRL predicate to a SRL argument. For example, *hasRole(3,2)* relates the predicate in position 3 (i.e., *play*) to the phrase which head is in position 2 (i.e., *Ms. Haag*).

role/3 identifies the role for a predicate-argument pair. For example, *role(3,2,ARG0)* denotes the role $ARG0$ for the SRL predicate in the position 2 and the SRL argument in position 3.

sense/2 denotes the sense of a predicate at a specific position. For example, *sense(3,02)* signals that the predicate in position 3 has the sense *02*.

We also define three sets of observable predicates. The first set represents information about each token as provided in the shared task corpora for the closed track: *word* for the word form (e.g. *word*(3,plays)); *plemma/2* for the lemma; *ppos/2* for the POS tag; *feat/3* for each feature-value pair; *dependency/3* for the head dependency and relation; *predicate/1* for tokens that are predicates according to the "FILLPRED" column. We will refer to these predicates as the *token* predicates.

The second set extends the information provided in the closed track corpus: *cpos/2* is a coarse POS tag (first letter of actual POS tag); *possibleArg/1* is true if the POS tag the token is a potential SRL argument POS tag (e.g., PUNC is not); *voice/2* denotes the voice for verbal tokens based on heuristics that use syntactic information, or based on features in the FEAT column of the data. We will refer to these predicates as the *extended* predicates.

Finally, the third set represents dependency information inspired by the features proposed by Xue and Palmer (2004). There are two types of predicates in this set: *paths* and *frames*. Paths capture the dependency path between two tokens, and frames the subcategorisation frame for a token or a pair of tokens. There are directed and undirected versions of paths, and labelled (with dependency relations) and unlabelled versions of paths and frames. Finally, we have a frame predicate with the distance from the predicate to its head. We will refer to the paths and most of the frames predicates as the *path* predicates, while we will consider the *frame* predicates for a unique token part *token* predicates.

The ML predicates here presented are used within the formulae of our MLN. We distinguish between two types of formula: local and global.

2.1 Local formulae

A formula is local if its groundings relate any number of observed ground atoms to exactly one hidden ground atom. For example, a grounding of the local formula

$$lemma(p, +l_1) \wedge lemma(a, +l_2) \Rightarrow hasRole(p, a)$$

connects a hidden *hasRole/2* ground atom to two observed *plemma/2* ground atoms. This formula can be interpreted as the feature for the predicate and argument lemmas in the argument identification stage of a pipeline SRL system. Note that the "+" prefix indicates that there is a different weight for each possible pair of lemmas (l_1, l_2).

We divide our local formulae into four sets, one for each hidden predicate. For instance, the set for *argument/1* only contains formulae in which the hidden predicate is *argument/1*.

The sets for *argument/1* and *sense/2* predicates have similar formulae since each predicate only involves one token at time: the SRL argument or the SRL predicate token. The formulae in these sets are defined using only *token* or *extended* observed predicates.

There are two differences between the *argument/1* and *sense/2* formulae. First, the *argument/1* formulae use the *possibleArg/1* predicate as precondition, while the sense formulae are conditioned on the *predicate/1* predicate. For instance, consider the *argument/1* formula based on word forms:

$$word(a, +w) \wedge possibleArg(a) \Rightarrow argument(a),$$

and the equivalent version for the *sense/2* predicate:

$$word(p, +w) \wedge predicate(p) \Rightarrow sense(p, +s).$$

This means we only apply the *argument/1* formulae if the token is a potential SRL argument, and the *sense/2* formulae if the token is a SRL predicate.

The second difference is the fact that for the *sense/2* formulae we have different weights for each possible sense (as indicated by the $+s$ term in the second formula above), while for the *argument/1* formulae this is not the case. This follows naturally from the fact that *argument/1* do not explicitly consider senses.

Table 1 presents templates for the local formuale of *argument/1* and *sense/2*. Templates allow us to compactly describe the FOL clauses of a ML. The template column shows the body of a clause. The last two columns of the table indicate if there is a clause with the given body and $argument(i)$ (I) or $sense(i, +s)$ (S) head, respectively. For example, consider the first row: since the last two columns of the row are marked, this template expands into two formulae: $word(i, +w) \Rightarrow argument(i)$ and $word(i, +w) \Rightarrow sense(i, +s)$. Including the preconditions for each hidden predicate we obtain the following formulae:

$$possibleArg(i) \wedge word(i, +w) \Rightarrow argument(i)$$

and

$$predicate(i) \wedge word(i, +w) \Rightarrow sense(i, +s).$$

In the case of the template marked with a "*" sign, the parameters $\mathbf{P}$ and $\mathbf{I}$, where $\mathbf{P} \in \{ppos, plemma\}$ and $\mathbf{I} \in \{-2, -1, 0, 1, 2\}$, have to be replaced by any combination of possible values. Since we generate $argument$ and $sense$ formulae for this template, the row corresponds to 20 formulae in total.

Table 2 shows the local formuale for *hasRole/2* and *role/3* predicates, for these formulae we use *token*, *extended* and *path* predicates. In this case, these templates have as precondition the formula $predicate(p) \wedge possibleArg(a)$. This ensures that the formulae are only applied for SRL predicates and potential SRL arguments. In the table we include the values to replace the template parameters with. Some of these formulae capture a notion of distance between SRL predicate and SRL argument and are implicitely conjoined with a $distance(p, a, +d)$ atom. If a formulae exists both with and without $distance$ atom, we write *Both* in the "Dist" column; if it only exists with the $distance$ atom, we write *Only*, otherwise *No*.

Note that Tables 1 and 2 do not mention the feature information provided in the corpora because this information was not available for every language. We therefore group the formulae which consider the *feature/3* predicate into another a set we call *feature* formulae. This is the summary of these formulae:

$feat(p, +f, +v) \Rightarrow sense(p, +s)$
$feat(p, +f, +v) \Rightarrow argument(a)$
$feat(p, +f, +v1) \wedge feat(p, f, +v2) \Rightarrow hasRole(p, a)$
$feat(p, +f, +v1) \wedge feat(p, f, +v2) \Rightarrow role(p, a, +r)$

Additionally, we define a set of language specific formulae. They are aimed to capture the relations between argument and its siblings for the *hasRole/2* and *role/3* predicates. In practice in turned out that these formulae were only beneficial for the Japanese language. This is a summary of such formulae which we called *argument siblings*:

$dep(a, h, \_) \wedge dep(h, c, \_) \wedge ppos(a, +p1) \wedge ppos(c, +p2) \Rightarrow hasRole(p, a)$
$dep(a, h, \_) \wedge dep(h, c, \_) \wedge ppos(a, +p1) \wedge ppos(c, +p2) \Rightarrow role(p, a, +r)$
$dep(a, h, \_) \wedge dep(h, c, \_) \wedge plemma(a, +p1) \wedge ppos(c, +p2) \Rightarrow hasRole(p, a)$
$dep(a, h, \_) \wedge dep(h, c, \_) \wedge plemma(a, +p1) \wedge ppos(c, +p2) \Rightarrow role(p, a, +r)$

With these sets of formulae we can build specific MLNs for each language in the shared task. We group the formulae into the modules: *argument/1*,

Template	I	S
$word(i, +w)$	X	X
$\mathbf{P}(i + \mathbf{I}, +v)$*	X	X
$cpos(i + 1, +c_1) \wedge cpos(i - 1, +c_2)$	X	X
$cpos(i + 1, +c_1) \wedge cpos(i - 1, +c_2) \wedge cpos(i + 2, +c_3) \wedge cpos(i - 2, +c_4)$	X	X
$dep(i, \_, +d)$	X	X
$dep(\_, i, +d)$	X	X
$ppos(i, +o) \wedge dep(i, j, +d)$	X	X
$ppos(i, +o_1) \wedge ppos(j, +o_2) \wedge dep(i, j, +d)$	X	X
$ppos(j, +o_1) \wedge ppos(k, +o_2) \wedge dep(j, k, \_) \wedge dep(k, i, +d)$	X	X
$plemma(i, +l) \wedge dep(j, i, +d)$	X	X
$frame(i, +f)$	X	X
(Empty Body)		X

Table 1: Templates of the local formulae for *argument/1* and *sense/2*. I: head of clause is $argument(i)$, S: head of clause is $sense(i, +s)$

Template	Parameters	Dist.	H	R
$\mathbf{P}(p, +v)$	$\mathbf{P} \in S_1$	Both	X	X
$plemma(p, +l) \wedge ppos(a, +o)$		No	X	
$ppos(p, +o) \wedge plemma(a, +l)$		No	X	
$plemma(p, +l_1) \wedge plemma(a, +l_2)$		Only	X	X
$ppos(p, +o_1) \wedge ppos(a, +o_2)$		Only	X	
$ppos(p, +o_1) \wedge ppos(a + \mathbf{I}, +o_2)$	$\mathbf{I} \in \{-1, 0, 1\}$	Only	X	
$plemma(p, +l)$		Only		X
$voice(p, +e) \wedge lemma(a, +l)$		Only		X
$cpos(p, +c_1) \wedge cpos(p + \mathbf{I}, +c_2) \wedge cpos(a, +c_3) \wedge cpos(a + \mathbf{J}, c_4)$	$\mathbf{I}, \mathbf{J} \in \{-1, 1\}^2$	No	X	X
$ppos(p, +v_1) \wedge ppos(a, \text{IN}) \wedge dep(a, m, \_) \wedge \mathbf{P}(m, +v_2)$	$\mathbf{P} \in S_1$	No	X	X
$plemma(p, +v_1) \wedge ppos(a, \text{IN}) \wedge dep(a, m, \_) \wedge ppos(m, +v_2)$		No	X	X
$\mathbf{P}(p, a, +v)$	$\mathbf{P} \in S_2$	No	X	X
$\mathbf{P}(p, a, +v) \wedge plemma(p, +l)$	$\mathbf{P} \in S_3$	No	X	X
$\mathbf{P}(p, a, +v) \wedge plemma(p, +l_1) \wedge plemma(a, +l_2)$	$\mathbf{P} \in S_4$	No	X	X
$pathFrame(p, a, +t) \wedge plemma(p, +l) \wedge voice(p, +e)$		No	X	X
$pathFrameDist(p, a, +t)$		Only	X	X
$pathFrameDist(p, a, +t) \wedge voice(p, +e)$		Only	X	X
$pathFrameDist(p, a, +t) \wedge plemma(p, +l)$		Only	X	X
$\mathbf{P}(p, a, +v) \wedge plemma(a, +l)$	$\mathbf{P} \in S_5$	Only	X	X
$\mathbf{P}(p, a, +v) \wedge ppos(p, +o)$	$\mathbf{P} \in S_5$	Only	X	X
$pathFrameDist(p, a, +t) \wedge ppos(p, +o_1) \wedge ppos(a, +o_2)$		Only	X	X
$path(p, a, +t) \wedge plemma(p, +l) \wedge cpos(a, +c)$		Only	X	X
$dep(\_, a, +d)$		Only	X	X
$dep(\_, a, +) \wedge voice(p, +e)$		Only	X	X
$dep(\_, a, +d_1) \wedge dep(\_, p, +d_2)$		Only	X	X
$(EmptyBody)$		No	X	X

Table 2: Templates of the local formulae for *hasRole/2* and *role/3*. H: head of clause is $hasRole(p, a)$, R: head of clause is $role(p, a, +r)$ and $S_1 = \{ppos, plemma\}, S_2 = \{frame, unlabelFrame, path\}, S_3 = \{frame, pathFrame\}, S_4 = \{frame, pathFrame, path\}, S_5 = \{pathFrameDist, path\}$

hasRole/2, *role/3*, *sense/3*, *feature* and *argument siblings*. Table 3 shows the different configurations of such modules that we used for the individual languages. We omit to mention the *argument/1*, *hasRole/2* and *role/3* modules because they are present for all languages.

A more detailed description of the formulae can be found in our MLN model files.[2] They can be used both as a reference and as input to our Markov Logic Engine,[3] and thus allow the reader to easily reproduce our results.

Set	Feature	*sense/2*	Argument siblings
Catalan	Yes	Yes	No
Chinese	No	Yes	No
Czech	Yes	No	No
English	No	Yes	No
German	Yes	Yes	No
Japanese	Yes	No	Yes
Spanish	Yes	Yes	No

Table 3: Different configuration of the modules for the formulae of the languages.

2.2 Global formulae

Global formulae relate several hidden ground atoms. We use them for two purposes: to ensure consis-

[2] `http://thebeast.googlecode.com/svn/mlns/conll09`

[3] `http://thebeast.googlecode.com`

tency between the decisions of all SRL stages and to capture some of our intuition about the task. We will refer to formulae that serve the first purpose as *structural constraints*. For example, a structural constraint is given by the (deterministic) formula

$$role(p, a, r) \Rightarrow hasRole(p, a)$$

which ensures that, whenever the argument a is given a label r with respect to the predicate p, this argument must be an argument of a as denoted by *hasRole(p,a)*.

The global formulae that capture our intuition about the task itself can be further divided into two classes. The first one uses deterministic or *hard* constraints such as

$$role(p, a, r_1) \wedge r_1 \neq r_2 \Rightarrow \neg role(p, a, r_2)$$

which forbids cases where distinct arguments of a predicate have the same role unless the role describes a modifier.

The second class of global formulae is *soft* or non-deterministic. For instance, the formula

$$lemma(p, +l) \wedge ppos(a, +p)$$
$$\wedge hasRole(p, a) \Rightarrow sense(p, +f)$$

is a soft global formula. It captures the observation that the sense of a verb or noun depends on the type of its arguments. Here the type of an argument token is represented by its POS tag.

Table 4 presents the global formulae used in this model.

Structural constraints
$hasRole(p, a) \Rightarrow argument(a)$ $role(p, a, r) \Rightarrow hasRole(p, a)$
$argument(a) \Rightarrow \exists p.hasRole(p, a)$ $hasRole(p, a) \Rightarrow \exists r.role(p, a, r)$
Hard constraints
$role(p, a, r_1) \wedge r_1 \neq r_2 \Rightarrow \neg role(p, a, r_2)$ $sense(p, s_1) \wedge s_1 \neq s_2 \Rightarrow \neg sense(p, r_2)$ $role\,(p, a_1, r) \wedge \neg mod\,(r) \wedge a_1 \neq a_2 \Rightarrow \neg role\,(p, a_2, r)$
Soft constraints
$role\,(p, a_1, r) \wedge \neg mod\,(r) \wedge a_1 \neq a_2 \Rightarrow \neg role\,(p, a_2, r)$ $plemma(p, +l) \wedge ppos(a, +p) \wedge hasRole(p, a) \Rightarrow sense(p, +f)$ $plemma(p, +l) \wedge role(p, a, +r) \Rightarrow sense(p, +f)$

Table 4: Global formulae for ML model

Language	Devel	Test	Train time	Test time
Average	77.25%	77.46%	11h 29m	23m
Catalan	78.10%	78.00%	6h 11m	14m
Chinese	77.97%	77.73%	36h 30m	34m
Czech	75.98%	75.75%	14h 21m	1h 7m
English	82.28%	83.34%	12h 26m	16m
German	72.05%	73.52%	2h 28m	7m
Japanese	76.34%	76.00%	2h 17m	4m
Spanish	78.03%	77.91%	6h 9m	16m

Table 5: F-scores for in-domain in corpora for each language.

3 Results

For our experiments we use the corpora provided in the SRLonly track of the shared task. Our MLN is tested on the following languages: Catalan and Spanish (Taulé et al., 2008) , Chinese (Palmer and Xue, 2009), Czech (Hajič et al., 2006),[4] English (Surdeanu et al., 2008), German (Burchardt et al., 2006), Japanese (Kawahara et al., 2002).

Table 5 presents the F1-scores and training/test times for the development and in-domain corpora. Clearly, our model does better for English. This is in part because the original model was developed for English.

To put these results into context: our SRL system is the third best in the *SRLOnly* track of the Shared Task, and it is the sixth best on both *Joint* and *SRLOnly* tracks. For five of the languages the difference to the F1 scores of the best system is 3%. However, for German it is 6.19% and for Czech 10.76%. One possible explanation for the poor performance on Czech data will be given below. Note that in comparison our system does slightly better in terms of precision than in terms of recall (we have the fifth best average precision and the eighth average recall).

Table 6 presents the F1 scores of our system for the out of domain test corpora. We observe a similar tendency: our system is the sixth best for both *Joint* and *SRLOnly* tracks. We also observe similar large differences between our scores and the best scores for German and Czech (i.e., $> 7.5\%$), while for English the difference is relatively small (i.e., $< 3\%$).

[4]For training we use only sentences shorter than 40 words in this corpus.

Language	Czech	English	German
F-score	77.34%	71.86%	62.37%

Table 6: F-scores for out-domain in corpora for each language.

Finally, we evaluated the effect of the *argument siblings* set of formulae introduced for the Japanese MLN. Without this set the F-score is 69.52% for the Japanese test set. Hence *argument siblings* formulae improve performance by more than 6%.

We found that the MLN for Czech was the one with the largest difference in performance when compared to the best system. By inspecting our results for the development set, we found that for Czech many of the errors were of a rather technical nature. Our system would usually extract frame IDs (such as "play.02") by concatenating the lemma of the token and outcome of the *sense/2* prediction (for the "02" part). However, in the case of Czech some frame IDs are not based on the lemma of the token, but on an abstract ID in a vocabulary (e.g., "v-w1757f1"). In these cases our heuristic failed, leading to poor results for frame ID extraction.

4 Conclusion

We presented a Markov Logic Network that performs joint multi-lingual Semantic Role Labelling. This network achieves the third best semantic F-scores in the closed track among the SRLOnly systems of the CoNLL-09 Shared Task, and sixth best semantic scores among SRLOnly and Joint systems for the closed task.

We observed that the inclusion of features which take into account information about the siblings of the argument were beneficial for SRL performance on the Japanese dataset. We also noticed that our poor performance with Czech are caused by our frame ID heuristic. Further work has to be done in order to overcome this problem.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of LREC-2006*, Genoa, Italy, 2006.

Koby Crammer and Yoram Singer. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991, 2003. ISSN 1533-7928.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. Prague dependency treebank 2.0, 2006.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL-2009)*, Boulder, Colorado, USA, 2009.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the LREC-2002*, pages 2008–2013, Las Palmas, Canary Islands, 2002.

Martha Palmer and Nianwen Xue. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172, 2009.

Matt Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62:107–136, 2006.

Sebastian Riedel. Improving the accuracy and efficiency of map inference for markov logic. In *UAI '08: Proceedings of the Annual Conference on Uncertainty in AI*, 2008.

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

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of LREC-2008*, Marrakesh, Morroco, 2008.

Nianwen Xue and Martha Palmer. Calibrating features for semantic role labeling. In *EMNLP '04: Proceedings of the Annual Conference on Empirical Methods in Natural Language Processing*, 2004.

The Crotal SRL System : a Generic Tool Based on Tree-structured CRF*

Erwan Moreau
LIPN - CNRS UMR 7030 & Univ. Paris 13
Erwan.Moreau@lipn.univ-paris13.fr

Isabelle Tellier
LIFO - Univ. Orléans
Isabelle.Tellier@univ-orleans.fr

Abstract

We present the Crotal system, used in the CoNLL09 Shared Task. It is based on XCRF, a highly configurable CRF library which can take into account hierarchical relations. This system had never been used in such a context thus the performance is average, but we are confident that there is room for progression.

1 Introduction

In this paper we present the Crotal Semantic Role Labelling (SRL) system, which has been used in the CoNLL 2009 Shared Task (Hajič et al., 2009)[1]. This system is based on Conditional Random Fields (CRF) (Lafferty et al., 2001; Sutton and McCallum, 2006): our idea is that we can use the provided dependency structure as the skeleton of a graphical model expressing independence asumptions in a CRF model. CRF are a powerful machine learning technique that has been successfully applied to a large number of natural language tasks, mainly to tag sequences. Compared to classification techniques, CRF can easily take into account dependencies among annotations: it is therefore possible to represent tree-like structures in the input of the algorithm. Recently, CRF using tree structures were used in (Finkel et al., 2008) in the case of parsing.

Before participating to this Shared Task, our prototype had only been used to annotate function tags in a French Treebank: these data were drastically smaller, and the task was simpler. Therefore CoNLL 2009 ST is the first time the Crotal System is run for a quite complex task, with so many data as input, and seven different languages (Catalan, Spanish (Taulé et al., 2008), Chinese (Palmer and Xue, 2009), Czech (Hajič et al., 2006), English (Surdeanu et al., 2008), German (Burchardt et al., 2006) and Japanese (Kawahara et al., 2002)). In this context, the performance we obtained seems reasonable: our average F1-measure is 66.49% (evaluation dataset).

One of the advantages we want to emphasise about our system is its genericity: the system does not need a lot of information as input (we mainly use *pos* and *deprel* columns, and the frame sets have not been used), and it was able to achieve satisfying results for the seven different languages using nearly the same parameters (differences were essentially due to the volume of data, since it was sometimes necessary to reduce the processing time). Of course, we hope to improve this prototype thanks to this experience: it may become necessary to lose in genericity in order to gain in performance, but our goal is to maintain as much as possible this advantage.

In section 2 we explain the general architecture for Crotal, then we explain how features are selected in our system in section 3, and finally we detail and discuss the results in section 4.

2 The Crotal System Architecture

2.1 General principle

The system we propose is based on the public library XCRF (Gilleron et al., 2006; Jousse, 2007), which

*This work has been funded by the French National project ANR-07-MDCO-03 "CRoTAL".

[1] We have participated in the SRL-only category.

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 91–96,
Boulder, Colorado, June 2009.

implements CRF model(s) to learn to annotate trees represented by XML documents. Of course, its performance depends on the way it is used, and especially on how *features* are chosen to reliably represent the labeled data. In order to keep the system as generic as possible, features are generated automatically and only a few parameters may vary. The global process has been divided into a sequence of steps, by creating clusters (one for each predicate, except the less frequent ones). Indeed, one expects that the behaviour of the arguments for a given predicate is more regular than for all predicates put together. Moreover, the size of the training set for all seven languages allows such a clustering, and it would even be difficult to process the whole set of predicates due to time and memory limitations. Thus the global process is[2]:

1. Data conversion from CoNLL format to XCRF format:
   - For each sentence containing n predicates, generate n different XML trees[3].
   - The tree is simply built following the dependencies (as provided by the *head* column). Therefore the possible non-projectivity of a tree is ignored, though the order of words is of course prefered whenever possible. An artificial root node is always added (useful for languages where several roots are possible).
   - In each such XML tree, there is only one (marked) predicate, and in the annotated version its arguments (extracted from the corresponding column) and only them are reported in the corresponding nodes.

   Figure 1 shows the labeled XML tree obtained for a (part of) example sentence.

2. Clustering by *lemma*: all dependency trees having the same lemma as predicate are put together if the number of such trees is at least a given threshold (generally 3, also tested with 2 to 5). There is a special cluster for less frequent lemmas[4]. Then, for each cluster, in training mode the process consists of:
   (a) Generation of features for the arguments training step.
   (b) The CRF model for arguments is trained with XCRF.
   (c) Generation of features for the senses training step.
   (d) The CRF model for senses[5] is trained with XCRF.

   In annotation mode, the CRF model for arguments is first applied to the input tree, then the CRF model for senses (if possible, an individual evaluation is also computed).

3. Back conversion from XCRF format to CoNLL format (in annotation mode).

In the framework of this task, features generation is crucial for improving performance. That is why we will mainly focus on that point in the remaining of this paper.

2.2 The XCRF Library

XCRF (Gilleron et al., 2006; Jousse, 2007) is a public library which has been applied successfully to *HTML* documents in order to extract information or translate the tree structure into *XML* (Jousse, 2007). More recently we have applied it to annotate function tags in a French Treebank.

In a CRF model, a feature is a function (usually providing a boolean result) whose value depends on the annotations present in a special *clique* of the graph, and on the value of the observed data. In our system, each feature is defined by a pair (C, T), where:

- C is the set of annotations present in a given clique, i.e. a completely connected subgraph of the graphical structure between annotations.

[2] Remark: unless stated otherwise, we will use terms "lemma", "POS tag" "dependency relation" or "head" to refer to the information contained in the corresponding "P-columns" for each word. It is worth noticing that performance would be better using the "real" columns, but we have followed the instructions given by the organizers.

[3] Thus sentences with no predicate are skipped and several trees possibly correspond to the same sentence.

[4] This special cluster is used as a default case. In particular, if an unknown lemma is encoutered during annotation, it will be annotated using the model learned for this default cluster.

[5] Steps 2c and 2d are skipped if the lemma has only one possible sense (or no sense is needed, like in Japanese data and for some Czech predicates).

Several solutions are possible to choose this graph. In most of our experiments, we have chosen a graph where only the node-parent relationship between nodes is taken into account (denoted FT2), as illustrated by Figure 2. XCRF is also able to deal with simple one-node cliques (no dependency between annotation, denoted FT1) and node-parent-sibling relationship (denoted FT3).

- $T = \{t_1, \ldots, t_n\}$ is a (possibly empty) set of boolean tests on the observation (i.e. not depending on the annotations). Each t_i is an atomic test[6]: for example, the test "*pos* attribute for first left sibling is NNS" is satisfied for node 3 in fig. 1. T is the conjunction of all t_i.

For example, let us define the following FT2 feature (C, T), that would be true for node 4 in fig. 1: C is $\{apred_{parent} = \text{PRED} \wedge apred_{current} = \text{C-A1}\}$ and T is $\{pos_{child_1} = \text{VB} \wedge deprel_{parent} = \text{VC}\}$.

3 Selecting Features

Our goal is somehow to "learn" features from the training set, in the sense that we do not explicitly define them but generate them from the corpus. The main parameters we use for generating a set of features are the following:

- The feature type n, with $n \leq 3$. All FT n', with $n' \leq n$, are also considered, because some function tags possibly appear in FT n and not (or more rarely) in FT $n+1$.
- Various kind of accessible information (decomposed through two distinct parameters *information* and *neighbourhood*):
  - Information: form, lemma, POS tags, dependency relation and various secondary attributes (column *features*) are available for all nodes (i.e. word), in every tree extracted from the corpus.
  - Neighbourhood: Given a current node, the "neighbourhood" defines the set of nodes

[6] A test is provided to XCRF as an XPath expression, which will be applied to the current node in the XML tree corresponding to the sentence.

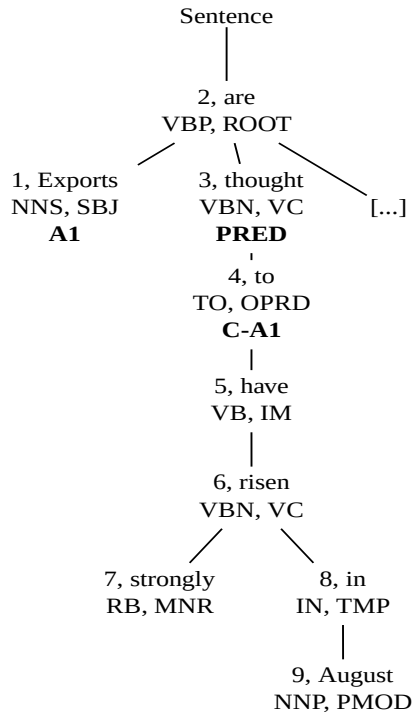

Figure 1: a labeled example for the (part of) sentence "Exports are thought to have risen strongly in August [...]": the nodes are represented with their POS tags, and in bold face the corresponding annotation associated with the predicate "thought" (label PRED was added during preprocessing, see 3.1)

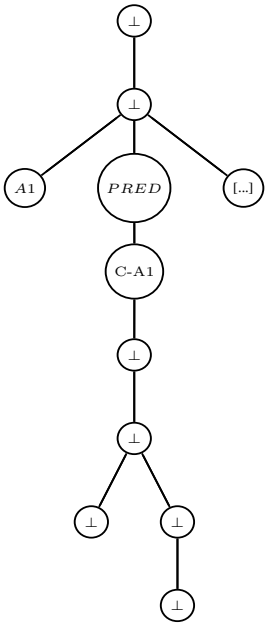

Figure 2: graph for a FT2-CRF for the annotation of the sentence of Figure 1 (where ⊥ means "no annotation")

that will be observed to help deduce its annotation: only this node, or also its parent, possibly its siblings, etc.

- The maximum number of (atomic) tests in the set T for these nodes: combining several tests makes features more precise (conjunction), but also more numerous.

A few other parameters may be added to speed up learning:

- minimum proportion for an argument label which is present in the data to be taken into account,
- minimum proportion for a feature which is present in the data to be included in the model,
- and maximum number of sentences to process by XCRF in the training step.

We try to use as less linguistic knowledge as possible, because we are interested in testing to what extent the model is able to learn such knowledge by itself. Moreover, we observe that using too many features and/or examples as input in XCRF requires a lot of time and memory (sometimes too much), so we have to restrict the selection to the most relevant kind of information in order to get a tractable machinery. This is why we use only POS tags (*pos*) and dependency relations (*deprel*) (as one can see in fig. 1). Finally the process of generating features consists in parsing the training data in the following way: for each encoutered clique, all the possible (combinations of) tests concerning the given neighbourhood are generated, and each of them forms a feature together with the observed clique.

3.1 Learning Argument Roles

In our system, the arguments and the sense of a predicate are trained (or annotated) one after the other: the former is always processed before the latter, thus the dependency holds only in the direction from arguments to sense. Therefore the training of arguments only relies on the observed trees (actually only the neighbourhood considered and the arguments cliques). In order to help the learner locate the right arguments, a special label *PRED* is added as "argument" to the node corresponding to the target predicate: by this way cliques can more easily take the tree structure into account in the neighbourhood of the predicate.

After some tests using the development set as test set, we observed that the following parameters were the best suited to build a reliable CRF model (for the arguments) in a reasonable time (and thus used them to learn the final models): the neighbourhood consists in the node itself, its parent and grand-parent, first and second siblings on both sides and first child; the FT2 model performs quite correctly (FT3 has been discarded because it would have taken too much time), and at most two tests are included in a feature.

3.2 Learning Predicate Senses

The step of predicting senses can use the arguments that have been predicted in the previous step. In particular, the list of all arguments that have been found is added and may be used as a test in any feature. We did not use at all the frame sets provided with the data: our system is based only on the sentences. This choice is mainly guided by our goal to build a generic system, thus does not need a lot of input information in various formats. The lemma part of the predicate is simply copied from the *lemma* column (this may cause a few errors due to wrong lemmas, as observed in the English data).

The fact that sentences have been classified by lemma makes it convenient to learn/annotate senses: of course lemmas which can not have more than one sense are easily processed. In the general case, we also use XCRF to learn a model to assign senses for each lemma, using the following parameters: there is no need to use another model than FT1, since in each tree there is only one (clearly identified) node to label; a close neighbourhood (parent, first left and right siblings and first child) and only two tests are enough to obtain satisfactory results.

4 Results and Discussion

4.1 General Results

Due to limited time and resources, we had to relax some time-consuming constraints for some clusters of sentences (concerning mainly the biggest training sets, namely Czech and English): in some cases, the

threshold for a feature to be selected has been increased, resulting in a probably quite lower performance for these models. Ideally we would also have done more tests with all languages to fine-tune parameters. Nevertheless, we have obtained quite satisfying results for such a generic approach: the average F1-measure is 66.49%, ranging from 57.75% (Japanese) to 72.14% (English). These results show that the system is generic enough to work quite correctly with all seven languages[7].

4.2 Internal Evaluation

Here we report detailed results obtained in annotating the development set. Since we process the task in two distinct steps, we can evaluate both separately: for the arguments step, the F1-measure ranges from 56.0% (Czech) to 61.8% (German), except for Japanese data where it is only 27%. For the senses step, the F1-measure is generally better: it ranges from 61.5% for the Czech case[8] to 93.3% for Chinese.

It is also interesting to observe the difference between using "real" indicators (i.e. *lemma*, *pos*, *deprel* and *head* columns) versus predicted ones (i.e. *P*-columns): for example, with German data (respectively Catalan data) the F1-measure reaches 73.6% (resp. 70.8%) in the former case, but only 61.8% (resp. 60.6%) in the latter case (for the argument labeling step only).

4.3 Impact of Parameters

At first we intended to use the most precise CRF model (namely FT3), but the fact that it generates many more features (thus taking too much time) together with the fact that it does not improve performance a lot made impossible to use it for the whole data. More precisely, it was possible but only by setting restrictive values for other parameters (neighbourhood, thresholds), which would have decreased performance. This is why we had to use FT2 as a compromise, thus making possible to use better values for the other parameters. We have also tested using 3 tests instead of only 2, but it does not improve performance, or not enough to compensate for the huge number of generated features, which requires excessive time and/or memory for XCRF learning step.

One of the most important parameters is the neighbourhood, since it specifies the location (and consequently the amount) of the information taken into account in the features. We have tried different cases for both the argument labeling step and the sense disambiguation step: in the former case, observing children nodes is useless, whereas observing the parent and grand-parent nodes together with two siblings in both left and right handside improves the model. On the contrary, in the senses step observing more than close nodes is useless. These facts are not surprising, since arguments are generally hierarchically lower than predicates in the dependency trees.

We have also studied the problem of finding an optimal threshold for the minimum number of sentences by cluster (all sentences in a given cluster having the same lemma for predicate): if this threshold is too low some clusters will not contain enough examples to build a reliable model, and if it is too high a lot of sentences will fall in the default cluster (for which the model could be less precise). But surprisingly the results did not show any significant difference between using a threshold of 2, 3 or 5: actually individual results differ, but the global performance remains the same.

Finally a word has to be said about "efficiency parameters": the most important one is the minimum proportion for a generated feature to be included in the final set of features for the model. Clearly, the lower this threshold is, the better the performance is. Nevertheless, in the framework of a limited time task, it was necessary to set a value of 0.0005% in most cases, and sometimes a higher value (up to 0.001%) for the big clusters: these values seem low but prevent including a lot of features (and probably sometimes useful ones).

5 Problems, Discussion and Future Work

Since there was a time limit and the system was used for the first time for such a task, we had to face

[7] Actually detailed evaluation shows that the system does not deal very well with Japanese, since locating arguments is harder in this language.

[8] Counting only "real senses": it is worth noticing that Czech data were a bit different from the other languages concerning senses, since most predicates do not have senses (not counted here and easy to identify) and the set of possible senses is different for each lemma.

several unexpected problems and solve them quite rapidly. Therefore one may suppose that our system could perform better, provided more tests are done to fine-tune parameters, especially to optimize the balance between efficiency and performance. Indeed, there is a balance to find between the amount of information (number of features and/or examples) and the time taken by XCRF to process the training step. Generally speaking, performance increases with the amount of information, but practically XCRF can not handle a huge number of features and/or examples in a reasonable time. This is why selecting the "right" features as soon as possible is so important.

Among various possible ways to improve the system, we should benefit from the fact that CRF do not need a lot of examples as input to learn quite correctly. Informally, the XCRF library seems to have some kind of "optimal point": before this point the model learned could be better, but beyond this point time and/or memory are excessive. Thus one can try for example to apply an iterative process using a sufficiently low number of features at each step, to select the more useful ones depending on the weight XCRF assigns to them.

Since the Crotal system obtained reasonable results in this "non ideal" context, we are quite confident in the fact that it can be significantly improved. The CoNLL 09 Shared Task has been a good opportunity to validate our approach with a non trivial problem. Even if the performance is not excellent, several important points are satisfying: this experience shows that the system is able to handle such a task, and that it is generic enough to deal with very different languages.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

Jenny Rose Finkel, Alex Kleeman, and Christopher D. Manning. 2008. Efficient, feature-based, conditional random field parsing. In *Proceedings of ACL-08:HLT*, pages 959–967, Columbus, Ohio. Association for Computational Linguistics.

Rémi Gilleron, Florent Jousse, Isabelle Tellier, and Marc Tommasi. 2006. Conditional random fields for xml trees. In *Proceeding of ECML workshop on Mining and Learning in Graphs*.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. *Prague Dependency Treebank 2.0*. Linguistic Data Consortium, Philadelphia, Pennsylvania, USA. URL: http://ldc.upenn.edu. Cat. No. LDC2006T01, ISBN 1-58563-370-4.

Florent Jousse. 2007. *Transformations d'Arbres XML avec des Modèles Probabilistes pour l'Annotation*. Ph.D. thesis, Université Charles de Gaulle - Lille 3, October.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML'01: Proceedings of the 18th International Conf. on Machine Learning*, pages 282–289.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*.

Charles Sutton and Andrew McCallum. 2006. An introduction to conditional random fields for relational learning. In Lise Getoor and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco.

Parsing Syntactic and Semantic Dependencies for Multiple Languages with A Pipeline Approach

Han Ren, Donghong Ji
School of Computer Science
Wuhan University
Wuhan 430079, China
cslotus@mail.whu.edu.cn
donghong_ji@yahoo.com

Jing Wan, Mingyao Zhang
Center for Study of Language & Information
Wuhan University
Wuhan 430079, China
{jennifer.wanj, my.zhang}@gmail.com

Abstract

This paper describes a pipelined approach for CoNLL-09 shared task on joint learning of syntactic and semantic dependencies. In the system, we handle syntactic dependency parsing with a transition-based approach and utilize MaltParser as the base model. For SRL, we utilize a Maximum Entropy model to identify predicate senses and classify arguments. Experimental results show that the average performance of our system for all languages achieves 67.81% of macro F1 Score, 78.01% of syntactic accuracy, 56.69% of semantic labeled F1, 71.66% of macro precision and 64.66% of micro recall.

1 Introduction

Given a sentence with corresponding part-of-speech for each word, the task of syntactic and semantic dependency parsing contains two folds: (1) identifying the syntactic head of each word and assigning the dependency relationship between the word and its head; (2) identifying predicates with proper senses and labeling semantic dependencies for them.

For data-driven syntactic dependency parsing, many approaches are based on supervised learning using treebank or annotated datasets. Currently, graph-based and transition-based algorithms are two dominating approaches that are employed by many researchers, especially in previous CoNLL shared tasks. Graph-based algorithms (Eisner, 1996; McDonald et al., 2005) assume a series of dependency tree candidates for a sentence and the goal is to find the dependency tree with highest score. Transition-based algorithms (Yamada and Matsumoto, 2003; Nivre et al., 2004) utilize transition histories learned from dependencies within sentences to predict next state transition and build the optimal transition sequence. Although different strategies were considered, two approaches yielded comparable results at previous tasks.

Semantic role labeling contains two problems: identification and labeling. Identification is a binary classification problem, and the goal is to identify annotated units in a sentence; while labeling is a multi-class classification problem, which is to assign arguments with appropriate semantic roles. Hacioglu (2004) utilized predicate-argument structure and map dependency relations to semantic roles. Liu et al. (2005) combined two problems into a classification one, avoiding some annotated units being excluded due to some incorrect identification results. In addition, various features are also selected to improve accuracy of SRL.

In this paper, we propose a pipelined approach for CoNLL-09 shared task on joint learning of syntactic and semantic dependencies, and describe our system that can handle multiple languages. In the system, we handle syntactic dependency parsing with a transition-based approach. For SRL, we utilize Maximum Entropy model to identify predicate senses and classify arguments.

The remain of the paper is organized as follows. In Section 2, we discuss the processing mechanism containing syntactic and semantic dependency parsing of our system in detail. In Section 3, we give the evaluation results and analysis. Finally, the conclusion and future work are given in Section 4.

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 97–102,
Boulder, Colorado, June 2009.

2 System Description

The system, which is a two-stage pipeline, processes syntactic and semantic dependencies respectively. To reduce the difficulties in SRL, predicates of each sentence in all training and evaluation data are labeled, thus predicate identification can be ignored.

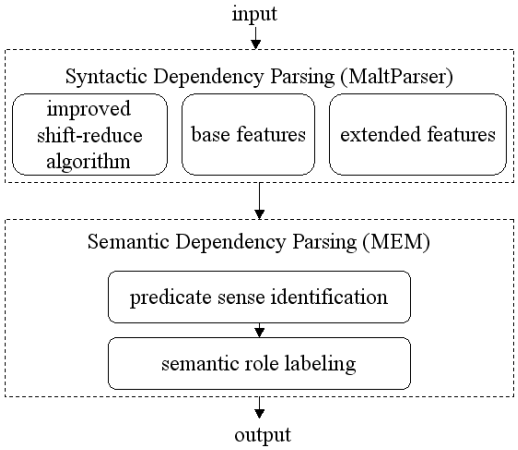

Figure 1. System Architectures

For syntactic dependencies, we employ a state-of-the-art dependency parser and basic plus extended features for parsing. For semantic dependencies, a Maximum Entropy Model is used both in predicate sense identification and semantic role labeling. Following subsections will show components of our system in detail.

2.1 Syntactic Dependency Parsing

In the system, MaltParser[1] is employed for syntactic dependency parsing. MaltParser is a data-driven deterministic dependency parser, based on a Support Vector Machine classifier. An extensive research (Nivre, 2007) parsing with 9 different languages shows that the parser is language-independent and yields good results.

MaltParser supports two kinds of parsing algorithms: Nivre's algorithms and Covington's incremental algorithms. Nivre's algorithms, which are deterministic algorithms consisting of a series of shift-reduce procedures, defines four operations:

• **Right**. For a given triple <*t*|*S*, *n*|*I*, *A*>, *S* represents STACK and *I* represents INPUT. If dependency relation $t \rightarrow n$ exists, it will be pendency relation $t \rightarrow n$ exists, it will be appended into *A* and *t* will be removed from *S*.

• **Left**. For a given triple <*t*|*S*, *n*|*I*, *A*>, if dependency relation $n \rightarrow t$ exists, it will be appended into *A* and *n* will be pushed into *S*.

• **Reduce**. If dependency relation $n \rightarrow t$ does not exist, and the parent node of *t* exists left to it, t will be removed from *S*.

• **Shift**. If none of the above satisfies, *n* will be pushed into *S*.

The deterministic algorithm simplifies determination for Reduce operation. As a matter of fact, some languages, such as Chinese, have more flexible word order, and some words have a long distance with their children. In this case, *t* should not be removed from *S*, but be handled with Shift operation. Otherwise, dependency relations between *t* and its children will never be identified, thus sequential errors of dependency relations may occur after the Reduce operation.

For syntactic dependencies with long distance, an improved Reduce strategy is: if the dependency relation between *n* and *t* does not exist, and the parent node of *t* exists left to it and the dependency relation between the parent node and *n*, *t* will be removed from *S*. The Reduce operation is projective, since it doesn't influence the following parsing procedures. The Improved algorithm is described as follows:

(1) one of the four operations is performed according to the dependency relation between *t* and *n* until EOS; if only one token remains in *S*, go to (3).

(2) continue to select operations for remaining tokens in *S*; when Shift procedure is performed, push *t* to *S*; if only one token remains in *S* and *I* contains more tokens than only EOS, goto (1).

(3) label all odd tokens in *S* as ROOT, pointing to EOS.

We also utilize history-based feature models implemented in the parser to predict the next action in the deterministic derivation of a dependency structure. The parser provides some default features that is general for most languages: (1) part-of-speech features of TOP and NEXT and following 3 tokens; (2) dependency features of TOP containing leftmost and rightmost dependents, and of NEXT containing leftmost dependents; (3) Lexical

[1] http://w3.msi.vxu.se/~jha/maltparser/

features of TOP, head of TOP, NEXT and following one token. We also extend features for multiple languages: (1) count of part-of-speech features of following tokens extend to 5; (2) part-of-speech and dependent features of head of TOP.

2.2 Semantic Dependency Parsing

Each defacto predicate in training and evaluation data of CoNLL09 is labeled with a sign 'Y', which simplifies the work of semantic dependency parsing. In our system, semantic dependency parsing is a pipeline that contains two parts: predicate sense identification and semantic role labeling. For predicate sense identification, each predicate is assigned a certain sense number. For semantic role labeling, local and global features are selected. Features of each part are trained by a classification algorithm. Both parts employ a Maximum Entropy Tool MaxEnt in a free package OpenNLP[2] as a classifier.

2.2.1 Predicate Sense Identification

The goal of predicate sense identification is to decide the correct frame for a predicate. According to PropBank (Palmer, et al., 2005), predicates contain one or more rolesets corresponding to different senses. In our system, a classifier is employed to identify each predicate's sense.

Suppose $C = \{01, 02, \ldots, N_L\}$ is the sense set (N_L is the count of categories corresponding to the language L, eg., in Chinese training set N_L = 10 since predicates have at most 10 senses in the set), and t_i is the ith sense of word w in sentence s. The model is implemented to assign each predicate to the most probatilistic sense.

$$t = argmax_{i \in C} P(w \mid s,\ t_i) \qquad (1)$$

Features for predicate sense identification are listed as follows:

• WORD, LEMMA, DEPREL: The lexical form and lemma of the predicate; the dependency relation between the predicate and its head; for Chinese and Japanese, WORD is ignored.

• HEAD_WORD, HEAD_POS: The lexical form and part-of-speech of the head of the predicate.

• CHILD_WORD_SET, CHILD_POS_SET, CHILD_DEP_SET: The lexical form, part-of-speech and dependency relation of dependents of the predicate.

• LSIB_WORD, LSIB_POS, LSIB_DEPREL, RSIB_WORD, RSIB_POS, RSIB_DEPREL: The lexical form, part-of-speech and dependency relation of the left and right sibling token of the predicate. Features of sibling tokens are adopted, because senses of some predicates can be inferred from its left or right sibling.

For English data set, we handle verbal and nominal predicates respectively; for other languages, we handle all predicates with one classifier. If a predicate in the evaluation data does not exist in the training data, it is assigned the most frequent sense label in the training data.

2.2.2 Semantic Role Labeling

Semantic role labeling task contains two parts: argument identification and argument classification. In our system the two parts are combined as one classification task. Our reason is that those argument candidates that potentially become semantic roles of corresponding predicates should not be pruned by incorrect argument identification. In our system, a predicate-argument pair consists of any token (except predicates) and any predicate in a sentence. However, we find that argument classification is a time-consuming procedure in the experiment because the classifier spends much time on a great many of invalid predicate-argument pairs. To reduce useless computing, we add a simple pruning method based on heuristic rules to remove invalid pairs, such as punctuations and some functional words.

Features used in our system are based on (Hacioglu, 2004) and (Pradhan et al, 2005), and described as follows:

• WORD, LEMMA, DEPREL: The same with those mentioned in section 2.2.1.

• VOICE: For verbs, the feature is Active or Passive; for nouns, it is null.

• POSITION: The word's position corresponding to its predicate: Left, Right or Self.

• PRED: The lemma plus sense of the word.

• PRED_POS: The part-of-speech of the predicate.

[2] http://maxent.sourceforge.net/

• LEFTM_WORD, LEFTM_POS, RIGHTM_WORD, RIGHTM_POS: Leftmost and rightmost word and their part-of-speech of the word.

• POS_PATH: All part-of-speech from the word to its predicate, including Up, Down, Left and Right, eg. "NN ↑ VV ↓ CC ↓ VV".

• DEPREL_PATH: Dependency relations from the word to its predicate, eg. "COMP ↑ RELC ↑ COMP ↓ ".

• ANC_POS_PATH, ANC_DEPREL_PATH: Similar to POS_PATH and DEPREL_PATH, part-of-speech and dependency relations from the word to the common ancestor with its predicate.

• PATH_LEN: Count of passing words from the word to its predicate.

• FAMILY: Relationship between the word and its predicate, including Child, Parent, Descendant, Ancestor, Sibling, Self and Null.

• PRED_CHD_POS, PRED_CHD_DEPREL: Part-of-speech and dependency relations of all children of the word's predicate.

For different languages, some features mentioned above are invalid and should be removed, and some extended features could improve the performance of the classifier. In our system we mainly focus on Chinese, therefore, WORD and VOICE should be removed when processing Chinese data set. We also adopt some features proposed by (Xue, 2008):

• POS_PATH_BA, POS_PATH_SB, POS_PATH_LB: BA and BEI are functional words that impact the order of arguments. In PropBank, BA words have the POS tag BA, and BEI words have two POS tags: SB (short BEI) and LB (long BEI).

3 Experimental Results

Our experiments are based on a PC with a Intel Core 2 Duo 2.1G CPU and 2G memory. Training and evaluation data (Taulé et al., 2008; Xue et al., 2008; Hajič et al., 2006; Palmer et al., 2002; Burchardt et al., 2006; Kawahara et al., 2002) have been converted to a uniform CoNLL Shared Task format. In all experiments, SVM and ME model are trained using training data, and tested with development data of all languages.

The system for closed challenge is designed as two parts. For syntactic dependency training and parsing, we utilize the projective model in MaltParser for data sets. We also follow default settings in MaltParser, such as assigned parameters for LIBSVM and combined prediction strategy, and utilize improved approaches mentioned in section 2. For semantic dependency training and parsing, we choose the count of iteration as 100 and cutoff value as 10 for the ME model. Table 1 shows the training time for syntactic and semantic dependency of all languages. Parsing time for syntactic is not more than 30 minutes, and for semantic is not more than 5 minutes of each language.

	syn	prd	sem
English	7h	12min	47min
Chinese	8h	18min	61min
Japanese	7h	14min	46min
Czech	13h	46min	77min
German	6h	16min	54min
Spanish	6h	15min	55min
Catalan	6h	15min	50min

Table 1. Training cost for all languages. syn, prd and sem mean training time for syntactic dependency, predicate identification and semantic dependency.

3.1 Syntactic Dependency Parsing

We utilize MaltParser with improved algorithms mentioned in section 2.1 for syntactic dependency parsing, and the results are shown in Table 2.

	LAS	UAS	label-acc.
English	87.57	89.98	92.19
Chinese	79.17	81.22	85.94
Japanese	91.47	92.57	97.28
Czech	57.30	75.66	65.39
German	76.63	80.31	85.97
Spanish	76.11	84.40	84.69
Catalan	77.84	86.41	85.78

Table 2. Performance of syntactic dependency parsing

Table 2 indicates that parsing for Japanese and English data sets has a better performance than other languages, partly because determinative algorithm and history-based grammar are more suited for these two languages. To compare the performance of our approach of improved deterministic algorithm and extended features, we make another experiment that utilize original arc-standard algorithm and base features for syntactic experiments. Due to time limitation, the experiments are only based on Chinese training and evaluation data. The results show that LAS and UAS drops about 2.7% and 2.2% for arc-standard algorithm, 1.6% and 1.2% for base features. They indicate that our de-

terministic algorithm and the extend features can help to improve syntactic dependency parsing. We also notice that the results of Czech achieve a lower performance than other languages. It mainly because the language has more rich morphology, usually accompanied by more flexible word order. Although using a large training set, linguistic properties greatly influence the parsing result. In addition, extended features are not suited for this language and the feature model should be optimized individually.

For all of the experiments we mainly focus on the language of Chinese. When parsing Chinese data sets we find that the focus words where most of the errors occur are almost punctuations, such as commas and full stops. Apart from errors of punctuations, most errors occur on prepositions such as the Chinese word 'at'. Most of these problems come from assigning the incorrect dependencies, and the reason is that the parsing algorithm concerns the form rather than the function of these words. In addition, the prediction of dependency relation ROOT achieves lower precision and recall than others, indicating that MaltParser overpredicts dependencies to the root.

3.2 Semantic Dependency Parsing

MaxEnt is employed as our classifier to train and parse semantic dependencies, and the results are shown in Table 3, in which all criterions are labeled.

	P	R	F1
English	76.57	60.45	67.56
Chinese	75.45	69.92	72.58
Japanese	91.93	43.15	58.73
Czech	68.83	57.78	62.82
German	62.96	47.75	54.31
Spanish	40.11	39.50	39.80
Catalan	41.34	40.66	41.00

Table 3. Performance of semantic dependency parsing

As shown in Table 3, the scores of the latter five languages are quite lower than those of the former two languages, and the main reason could be inferred from the scores of Table 2 that the drop of the performance of semantic dependency parsing comes from the low performance of syntactic dependency parsing. Another reason is that, morphological features are not be utilized in the classifier. Our post experiments after submission show that average performance could improve the performance after adding morphological and some combined features. In addition, difference between precision and recall indicates that the classification procedure works better than the identification procedure in semantic role labeling.

For Chinese, semantic role of some words with part-of-speech VE have been mislabeled. It's mainly because that these words in Chinese have multiple part-of-speech. The errors of POS and PRED greatly influence the system to perform these words. Another main problem occurs on the pairs NN + A0/A1. Identification of the two pairs are much lower than VA/VC/VE/VV + A0/A1 pairs. The reason is that the identification of nominal predicates have more errors than that of verbal predicates due to the combination of SRL for these two kinds of predicates. For further study, verbal predicates and nominal predicates should be handled respectively so that the overall performance can be improved.

3.3 Overall Performance

The average performance of our system for all languages achieves 67.81% of macro F1 Score, 78.01% of syntactic accuracy, 56.69% of semantic labeled F1, 71.66% of macro precision and 64.66% of micro recall.

4 Conclusion

In this paper, we propose a pipelined approach for CoNLL-09 shared task on joint learning of syntactic and semantic dependencies, and describe our system that can handle multiple languages. Our system focuses on improving the performance of syntactic and semantic dependency respectively. Experimental results show that the overall performance can be improved for multiple languages by long distance dependency algorithm and extended history-based features. Besides, the system fits for verbal predicates than nominal predicates and the classification procedure works better than identification procedure in semantic role labeling. For further study, respective process should be handled between these two kinds of predicates, and argument identification should be improved by using more discriminative features for a better overall performance.

Acknowledgments

This work is supported by the Natural Science Foundation of China under Grant Nos.60773011, 90820005, and Independent Research Foundation of Wuhan University.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó and Manfred Pinkal. 2006. The SALSA Corpus: a German Corpus Resource for Lexical Semantics. *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006).* Genoa, Italy.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pp.340–345.

Kadri Hacioglu. 2004. Semantic Role Labeling Using Dependency Trees. In *Proceedings of the International Conference on Computational Linguistics (COLING)*.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová and Zdeněk Žabokrtský. 2006. *The Prague Dependency Treebank 2.0.* CD-ROM. Linguistic Data Consortium, Philadelphia, Pennsylvania, USA. ISBN 1-58563-370-4. LDC Cat. No. LDC2006T01. URL: http://ldc.upenn.edu.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antonia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue and Yi Zhang. 2009. The CoNLL 2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009).* Boulder, Colorado, USA. June 4-5. pp.3-22.

Daisuke Kawahara, Sadao Kurohashi and Koiti Hasida. 2002. Construction of a Japanese Relevance-tagged Corpus. *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002).* Las Palmas, Spain. pp.2008-2013.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pp.91–98.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CoNLL)*, pp.49–56.

Joakim Nivre. 2004. Incrementality in Deterministic Dependency Parsing. In Incremental Parsing: Bringing Engineering and Cognition Together. *Workshop at ACL-2004*, Barcelona, Spain, pp.50-57.

Joakim Nivre and Johan Hall. 2005. MaltParser: A language-independent system for data-driven dependency parsing. In *Proceedings of the Fourth Workshop on Treebanks and Linguistic Theories (TLT)*.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gulsen Eryigit, Sandra Kubler, Svetoslav Marinov and Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural language Engineering*, Volume 13, Issue 02, pp.95-135.

Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H. Martin and Daniel Jurafsky. 2005. Support Vector Learning for Semantic Argument classification. *Machine Learning Journal*, 2005, 60(3): 11−39.

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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*.

Mariona Taulé, Maria Antònia Martí and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008).* Marrakech, Morocco.

Liu Ting, Wanxiang Che, Sheng Li, Yuxuan Hu, and Huaijun Liu. 2005. Semantic role labeling system using maximum entropy classifier. In *Proceedings of the 8th Conference on Computational Natural Language Learning (CoNLL)*.

Nianwen Xue. 2008. Labeling Chinese Predicates with Semantic roles. *Computational Linguistics*, 34(2): 225-255.

Nianwen Xue and Martha Palmer. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering,* **15**(1):143-172.

Hiroyasu Yamada and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pp.195–206.

Multilingual semantic parsing with a pipeline of linear classifiers

Oscar Täckström
Swedish Institute of Computer Science
SE-16429, Kista, Sweden
oscar@sics.se

Abstract

I describe a fast multilingual parser for semantic dependencies. The parser is implemented as a pipeline of linear classifiers trained with support vector machines. I use only first order features, and no pair-wise feature combinations in order to reduce training and prediction times. Hyper-parameters are carefully tuned for each language and sub-problem.

The system is evaluated on seven different languages: Catalan, Chinese, Czech, English, German, Japanese and Spanish. An analysis of learning rates and of the reliance on syntactic parsing quality shows that only modest improvements could be expected for most languages given more training data; Better syntactic parsing quality, on the other hand, could greatly improve the results. Individual tuning of hyper-parameters is crucial for obtaining good semantic parsing quality.

1 Introduction

This paper presents my submission for the semantic parsing track of the CoNLL 2009 shared task on syntactic and semantic dependencies in multiple languages (Hajič et al., 2009). The submitted parser is simpler than the submission in which I participated at the CoNLL 2008 shared task on joint learning of syntactic and semantic dependencies (Surdeanu et al., 2008), in which we used a more complex committee based approach to both syntax and semantics (Samuelsson et al., 2008). Results are on par with our previous system, while the parser is orders of magnitude faster both at training and prediction time and is able to process natural language text in Catalan, Chinese, Czech, English, German, Japanese and Spanish. The parser depends on the input to be annotated with part-of-speech tags and syntactic dependencies.

2 Semantic parser

The semantic parser is implemented as a pipeline of linear classifiers and a greedy constraint satisfaction post-processing step. The implementation is very similar to the best performing subsystem of the committee based system in Samuelsson et al. (2008).

Parsing consists of four steps: *predicate sense disambiguation*, *argument identification*, *argument classification* and *predicate frame constraint satisfaction*. The first three steps are implemented using linear classifiers, along with heuristic filtering techniques. Classifiers are trained using the support vector machine implementation provided by the LIBLINEAR software (Fan et al., 2008). MALLET is used as a framework for the system (McCallum, 2002).

For each classifier, the c-parameter of the SVM is optimised by a one dimensional grid search using threefold cross validation on the training set. For the identification step, the c-parameter is optimised with respect to F_1-score of the positive class, while for sense disambiguation and argument labelling the optimisation is with respect to accuracy. The regions to search were identified by initial runs on the development data. Optimising these parameters for each classification problem individually proved to be crucial for obtaining good results.

2.1 Predicate sense disambiguation

Since disambiguation of predicate sense is a multiclass problem, I train the classifiers using the method of Crammer and Singer (2002), using the implementation provided by LIBLINEAR. Sense labels do not generalise over predicate lemmas, so one classifier is trained for each lemma occurring in the training data. Rare predicates are given the most common sense of the predicate. Predicates occurring less than

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 103–108,
Boulder, Colorado, June 2009.

7 times in the training data were heuristically determined to be considered rare. Predicates with unseen lemmas are labelled with the most common sense tag in the training data.

2.1.1 Feature templates

The following feature templates are used for predicate sense disambiguation:

PREDICATEWORD
PREDICATE[POS/FEATS]
PREDICATEWINDOWBAGLEMMAS
PREDICATEWINDOWPOSITION[POS/FEATS]
GOVERNORRELATION
GOVERNOR[WORD/LEMMA]
GOVERNOR[POS/FEATS]
DEPENDENTRELATION
DEPENDENT[WORD/LEMMA]
DEPENDENT[POS/FEATS]
DEPENDENTSUBCAT.

The *WINDOW feature templates extract features from the two preceding and the two following tokens around the predicate, with respect to the linear ordering of the tokens. The *FEATS templates are based on information in the PFEATS input column for the languages where this information is provided.

2.2 Argument identification and labelling

In line with most previous pipelined systems, identification and labelling of arguments are performed as two separate steps. The classifiers in the identification step are trained with the standard L_2-loss SVM formulation, while the classifiers in the labelling step are trained using the method of Crammer and Singer.

In order to reduce the number of candidate arguments in the identification step, I apply the filtering technique of Xue and Palmer (2004), trivially adopted to the dependency syntax formalism. Further, a filtering heuristic is applied in which argument candidates with rare predicate / argument part-of-speech combinations are removed; *rare* meaning that the argument candidate is actually an argument in less than 0.05% of the occurrences of the pair. These heuristics greatly reduce the number of instances in the argument identification step and improve performance by reducing noise from the training data.

Separate classifiers are trained for verbal predicates and for nominal predicates, both in order to save computational resources and because the frame structures do not generalise between verbal and nominal predicates. For Czech, in order to reduce training time I split the argument identification problem into three sub-problems: *verbs*, *nouns* and *others*, based on the part-of-speech of the predicate. In hindsight, after solving a file encoding related bug which affected the separability of the Czech data set, a split into verbal and nominal predicates would have sufficed. Unfortunately I was not able to rerun the Czech experiments on time.

2.2.1 Feature templates

The following feature templates are used both for argument identification and argument labelling:

PREDICATELEMMASENSE
PREDICATE[POS/FEATS]
POSITION
ARGUMENT[POS/FEATS]
ARGUMENT[WORD/LEMMA]
ARGUMENTWINDOWPOSITIONLEMMA
ARGUMENTWINDOWPOSITION[POS/FEATS]
LEFTSIBLINGWORD
LEFTSIBLING[POS/FEATS]
RIGHTSIBLINGWORD
RIGHTSIBLING[POS/FEATS]
LEFTDEPENDENTWORD
RIGHTDEPENDENT[POS/FEATS]
RELATIONPATH
TRIGRAMRELATIONPATH
GOVERNORRELATION
GOVERNORLEMMA
GOVERNOR[POS/FEATS]

Most of these features, introduced by Gildea and Jurafsky (2002), belong to the folklore by now. The TRIGRAMRELATIONPATH is a "soft" version of the RELATIONPATH template, which treats the relation path as a bag of triplets of directional labelled dependency relations. Initial experiments suggested that this feature slightly improves performance, by overcoming local syntactic parse errors and data sparseness in the case of small training sets.

2.2.2 Predicate frame constraints

Following Johansson and Nugues (2008) I impose the CORE ARGUMENT CONSISTENCY and CON-

TINUATION CONSISTENCY constraints on the generated semantic frames. In the cited work, these constraints are used to filter the candidate frames for a re-ranker. I instead perform a greedy search in which only the core argument with the highest score is kept when the former constraint is violated. The latter constraint is enforced by simply dropping any continuation argument lacking its corresponding core argument. Initial experiments on the development data indicates that these simple heuristics slightly improves semantic parsing quality measured with labelled F_1-score. It is possible that the improvement could be greater by using L_2-regularised logistic regression scores instead of the SVM scores, since the latter can not be interpreted as probabilities. However, logistic regression performed consistently worse than the SVM formulation of Crammer and Singer in the argument labelling step.

2.2.3 Handling of multi-function arguments

In Czech and Japanese an argument can have multiple relations to the same predicate, i.e. the semantic structure needs sometimes be represented by a multi-graph. I chose the simplest possible solution and treat these structures as ordinary graphs with complex labels. This solution is motivated by the fact that the palette of multi-function arguments is small, and that the multiple functions mostly are highly interdependent, such as in the ACT|PAT complex which is the most common in Czech.

3 Results

The semantic parser was evaluated on in-domain data for Catalan, Chinese, Czech, English, German, Japanese and Spanish, and on out-of-domain data for Czech, English and German. The respective data sets are described in Taulé et al. (2008), Palmer and Xue (2009), Hajič et al. (2006), Surdeanu et al. (2008), Burchardt et al. (2006) and Kawahara et al. (2002).

My official submission scores are given in table 1, together with post submission labelled and unlabelled F_1-scores. The official submissions were affected by bugs related to file encoding and hyperparameter search. After resolving these bugs, I obtained an improvement of mean F_1-score of almost 10 absolute points compared to the official scores.

	Lab F_1	Lab F_1	Unlab F_1
Catalan	57.11	67.14	93.31
Chinese	63.41	74.14	82.57
Czech	71.05	78.29	89.20
English	67.64	78.93	88.70
German	53.42	62.98	89.64
Japanese	54.74	61.44	66.01
Spanish	61.51	69.93	93.54
Mean	61.27	70.41	86.14
Czech†	71.59	78.77	87.13
English†	59.82	68.96	86.23
German†	50.43	47.81	79.52
Mean†	60.61	65.18	84.29

Table 1: Semantic labelled and unlabelled F_1-scores for each language and domain. Left column: official labelled F_1-score. Middle column: post submission labelled F_1-score. Right column: post submission unlabelled F_1-score. † indicates out-of-domain test data.

Clearly, there is a large difference in performance for the different languages and domains. As could be expected the parser performs much better for the languages for which a large training set is provided. However, as discussed in the next section, simply adding more training data does not seem to solve the problem.

Comparing unlabelled F_1-scores with labelled F_1-scores, it seems that argument identification and labelling errors contribute almost equally to the total errors for Chinese, Czech and English. For Catalan, Spanish and German argument identification scores are high, while labelling scores are in the lower range. Japanese stands out with exceptionally low identification scores. Given that the quality of the predicted syntactic parsing was higher for Japanese than for any other language, the bottleneck when performing semantic parsing seems to be the limited expressivity of the Japanese syntactic dependency annotation scheme.

Interestingly, for Czech, the result on the out-of-domain data set is better than the result on the in-domain data set, even though the unlabelled result is slightly worse. For English, on the other hand the performance drop is in the order of 10 absolute labelled F_1 points, while the drop in unlabelled F_1-score is comparably small. The result on German out-of-domain data seems to be an outlier, with post-submission results even worse than the official sub-

	10%	25%	50%	75%	100%
Catalan	54.86	60.52	65.22	66.35	67.14
Chinese	72.93	73.40	73.77	74.08	74.14
Czech	75.42	76.90	77.69	78.00	78.29
English	75.75	77.56	78.37	78.71	78.93
German	47.77	54.74	58.94	61.02	62.98
Japanese	59.82	60.34	60.99	61.37	61.44
Spanish	58.80	64.32	68.35	69.34	69.93
Mean	63.62	66.83	69.05	69.84	70.41
Czech†	76.51	77.48	78.41	78.59	78.77
English†	66.04	67.54	68.37	69.00	68.96
German†	41.65	45.94	46.24	47.45	47.81
Mean†	61.40	63.65	64.34	65.01	65.18

Table 2: Semantic labelled F_1-scores w.r.t. training set size. † indicates out-of-domain test data.

	10%	25%	50%	75%	100%
Catalan	93.12	93.18	93.28	93.35	93.31
Chinese	82.37	82.45	82.54	82.55	82.57
Czech	89.03	89.12	89.17	89.21	89.20
English	87.96	88.38	88.52	88.67	88.70
German	88.23	89.02	89.63	89.53	89.64
Japanese	65.64	65.75	65.88	66.02	66.01
Spanish	93.52	93.49	93.52	93.53	93.54
Mean	85.70	85.91	86.08	86.12	86.14
Czech†	86.76	87.02	87.16	87.08	87.13
English†	85.67	86.14	86.22	86.20	86.23
German†	77.35	78.31	79.09	79.10	79.52
Mean†	83.26	83.82	84.16	84.13	84.29

Table 3: Semantic unlabelled F_1-scores w.r.t. training set size. † indicates out-of-domain test data.

mission results. I suspect that this is due to a bug.

3.1 Learning rates

In order to assess the effect of training set size on semantic parsing quality, I performed a learning rate experiment, in which the proportion of the training set used for training was varied in steps between 10% and 100% of the full training set size.

Learning rates with respect to labelled F_1-scores are given in table 2. The improvement in scores are modest for Chinese, Czech, English and Japanese, while Catalan, German and Spanish stand out by vast improvements with additional training data. However, the improvement when going from 75% to 100% of the training data is only modest for all languages. With the exception for English, for which the parser achieves the highest score, the relative labelled F_1-scores follow the relative sizes of the training sets.

Looking at learning rates with respect to unlabelled F_1-scores, given in table 3, it is evident that adding more training data only has a minor effect on the identification of arguments.

From table 4, one can see that predicate sense disambiguation is the sub-task that benefits most from additional training data. This is not surprising, since the senses does not generalise, and hence we cannot hope to correctly label the senses of unseen predicates; the only way to improve results with the current formalism seems to be by adding more training data.

The limited power of a pipeline of local classifiers shows itself in the exact match scores, given in table 5. This problem is clearly not remedied by additional training data.

	10%	25%	50%	75%	100%
Catalan	30.61	40.29	53.83	55.83	58.95
Chinese	94.06	94.37	94.71	95.10	95.26
Czech	83.24	84.75	85.78	86.21	86.60
English	92.18	93.68	94.83	95.35	95.60
German	34.91	47.27	58.18	62.18	66.55
Japanese	99.07	99.07	99.07	99.07	99.07
Spanish	38.53	50.22	59.59	62.01	66.26
Mean	67.51	72.81	78.00	79.39	81.18
Czech†	89.05	89.88	91.06	91.38	91.56
English†	83.64	84.27	84.83	85.70	85.94
German†	33.64	43.36	42.59	44.44	45.22
Mean†	68.78	72.51	72.83	73.84	74.24

Table 4: Predicate sense disambiguation F_1-scores w.r.t. training set size. † indicates out-of-domain test data.

3.2 Dependence on syntactic parsing quality

Since I only participated in the semantic parsing task, the results reported above rely on the provided predicted syntactic dependency parsing. In order to investigate the effect of parsing quality on the current system, I performed the same learning curve experiments with gold standard parse information. These results, shown in tables 6 and 7, give an upper bound on the possible improvement of the current system by means of improved parsing quality, given that the same syntactic annotation formalism is used.

Labelled F_1-scores are greatly improved for all languages except for Japanese, when using gold

	10%	25%	50%	75%	100%
Catalan	6.77	9.08	11.39	11.17	12.24
Chinese	17.02	17.33	17.61	17.76	17.68
Czech	9.33	9.59	9.97	9.95	10.11
English	12.01	12.76	12.96	13.13	13.17
German	76.95	78.50	78.95	79.20	79.50
Japanese	1.20	1.40	1.80	1.60	1.60
Spanish	8.23	10.20	12.93	13.39	13.16
Mean	18.79	19.84	20.80	20.89	21.07
Czech†	2.53	2.79	2.79	2.87	2.87
English†	19.06	19.53	19.76	20.00	20.00
German†	15.98	19.24	17.82	19.94	20.08
Mean†	12.52	13.85	13.46	14.27	14.32

Table 5: Percentage of exactly matched predicate-argument frames w.r.t. training set size. † indicates out-of-domain test data.

	10%	25%	50%	75%	100%
Catalan	62.65	72.50	75.39	77.03	78.86
Chinese	82.59	83.23	83.90	83.94	84.03
Czech	79.15	80.62	81.46	81.91	82.24
English	79.84	81.74	82.65	83.01	83.25
German	52.15	60.66	65.12	65.71	68.36
Japanese	60.85	61.76	62.55	62.85	63.23
Spanish	66.40	72.47	75.70	77.73	78.38
Mean	69.09	73.28	75.25	76.03	76.91
Czech†	78.64	80.07	80.77	81.01	81.20
English†	73.05	74.18	74.99	75.28	75.81
German†	52.06	52.77	54.72	56.22	56.35
Mean†	67.92	69.01	70.16	70.84	71.12

Table 6: Semantic labelled F_1-scores w.r.t. training set size, using gold standard syntactic and part-of-speech tag annotation. † indicates out-of-domain test data.

standard syntactic and part-of-speech annotations. For Catalan, Chinese and Spanish the improvement is in the order of 10 absolute points. For Japanese the improvement is a meagre 2 absolute points. This is not surprising given that the quality of the provided syntactic parsing was already very high for Japanese, as discussed previously.

Results with respect to unlabelled F_1-scores follow the same pattern as for labelled F_1-scores. Again, with Japanese the semantic parsing does not benefit much from better syntactic parsing quality. For Catalan and Spanish on the other hand, the identification of arguments is almost perfect with gold standard syntax. The poor labelling quality for these languages can thus not be attributed to the syntactic parse quality.

	10%	25%	50%	75%	100%
Catalan	99.94	99.98	99.99	99.99	99.99
Chinese	92.55	92.67	92.72	92.63	92.62
Czech	91.21	91.27	91.30	91.30	91.31
English	92.34	92.61	92.85	92.89	92.95
German	93.46	93.59	94.08	93.85	94.14
Japanese	66.98	67.20	67.58	67.62	67.74
Spanish	99.99	99.99	100.00	100.00	100.00
Mean	90.92	91.04	91.22	91.18	91.25
Czech†	89.00	89.22	89.34	89.38	89.36
English†	92.71	92.56	92.91	93.06	93.04
German†	90.54	90.23	90.77	90.86	90.99
Mean†	90.75	90.67	91.01	91.10	91.13

Table 7: Semantic unlabelled F_1-scores w.r.t. training set size, using gold standard syntactic and part-of-speech tag annotation. † indicates out-of-domain test data.

3.3 Computational requirements

Training and prediction times on a 2.3 GHz quad-core AMD Opteron™system are given in table 8. Since only linear classifiers and no pair-wise feature combinations are used, training and prediction times are quite modest. Verbal and nominal predicates are trained in parallel, no additional parallelisation is employed. Most of the training time is spent on optimising the c parameter of the SVM. Training times are roughly ten times as long as compared to training times with no hyper-parameter optimisation. Czech stands out as much more computationally demanding, especially in the sense disambiguation training step. The reason is the vast number of predicates in Czech compared to the other languages. The majority of the time in this step is, however, spent on writing the SVM training problems to disk.

Memory requirements range between approximately 1 Gigabytes for the smallest data sets and 6 Gigabytes for the largest data set. Memory usage could be lowered substantially by using a more compact feature dictionary. Currently every feature template / value pair is represented as a string, which is wasteful since many feature templates share the same values.

4 Conclusions

I have presented an effective multilingual pipelined semantic parser, using linear classifiers and a simple

	Sense	ArgId	ArgLab	Tot	Pred
Catalan	7m	11m	33m	51m	13s
Chinese	7m	13m	22m	42m	15s
Czech	10h	1h	1.5h	12.5h	34.5m
English	16m	14m	28m	58m	14.5s
German	4m	2m	5m	13m	3.5s
Japanese	1s	1m	4m	5m	4s
Spanish	10m	16m	40m	1.1h	13s

Table 8: Training times for each language and subproblem and approximate prediction times. Columns: training times for sense disambiguation (Sense), argument identification (ArgId), argument labelling (ArgLab), total training time (Tot), and total prediction time (Pred). Training times are measured w.r.t. to the union of the official training and development data sets. Prediction times are measured w.r.t. to the official evaluation data sets.

greedy constraint satisfaction heuristic. While the semantic parsing results in these experiments fail to reach the best results given by other experiments, the parser quickly delivers quite accurate semantic parsing of Catalan, Chinese, Czech, English, German, Japanese and Spanish.

Optimising the hyper-parameters of each of the individual classifiers is essential for obtaining good results with this simple architecture. Syntactic parsing quality has a large impact on the quality of the semantic parsing; a problem that is not remedied by adding additional training data.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006)*, Genoa, Italy.

Koby Crammer and Yoram Singer. 2002. On the learnability and design of output codes for multiclass problems. *Machine Learning*, 47(2):201–233, May.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0. CD-ROM, Cat. No. LDC2006T01, ISBN 1-58563-370-4, Linguistic Data Consortium, Philadelphia, Pennsylvania, USA.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*, Boulder, Colorado, USA.

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *Proceedings of the Shared Task Session of CoNLL-2008*, Manchester, UK.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013, Las Palmas, Canary Islands.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

Yvonne Samuelsson, Oscar Täckström, Sumithra Velupillai, Johan Eklund, Mark Fishel, and Markus Saers. 2008. Mixing and blending syntactic and semantic dependencies. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 248–252, Manchester, England, August. Coling 2008 Organizing Committee.

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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008)*, Manchester, Great Britain.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morroco.

Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 88–94, Barcelona, Spain, July. Association for Computational Linguistics.

A Joint Syntactic and Semantic Dependency Parsing System based on Maximum Entropy Models

Buzhou Tang[1] **Lu Li**[2] **Xinxin Li**[1] **Xuan Wang**[2] **Xiaolong Wang**[2]
Shenzhen Graduate School
Harbin Institute of Technology
Shenzhen,518055, China
[1]{tangbuzhou,lixxin2}@gmail.com
[2]{lli,wangxuan,wangxl}@insun.hit.edu.cn

Abstract

A joint syntactic and semantic dependency parsing system submitted to the CoNLL-2009 shared task is presented in this paper. The system is composed of three components: a syntactic dependency parser, a predicate classifier and a semantic parser. The first-order MSTParser is used as our syntactic dependency pasrser. Projective and non-projective MSTParsers are compared with each other on seven languages. Predicate classification and semantic parsing are both recognized as classification problem, and the Maximum Entropy Models are used for them in our system. For semantic parsing and predicate classifying, we focus on finding optimized features on multiple languages. The average Macro F1 Score of our system is 73.97 for joint task in closed challenge.

1 Introduction

The task for CoNLL-2009 is an extension of the CoNLL-2008 shared task to multiple languages: English (Surdeanu et al., 2008), Catalan plus Spanish (Mariona Taulé et al., 2008), Chinese (Martha Palmer et al., 2009), Czech (Jan Hajič et al., 2006), German (Aljoscha Burchardt et al., 2006) and Japanese (Daisuke Kawahara et al., 2002). Compared to the CoNLL-2008 shared task, the predicates are given for us in semantic dependencies task. Therefore, we have only need to label the semantic roles of nouns and verbs, and the frames of predicates.

In this paper, a joint syntactic and semantic dependency parsing system submitted to the CoNLL-2009 shared task is presented. The system is composed of three components: a syntactic dependency parser, a predicate classifier and a semantic parser. The first-order MSTParser is used as our syntactic dependency parser. Projective and non-projective MSTParsers are compared with each other on seven languages. The predicate classifier labeling the frames of predicates and the semantic parser labeling the semantic roles of nouns and verbs for each predicate are both recognized as classification problem, and the Maximum Entropy Models (MEs) are used for them in our system. Among three components, we mainly focus on the predicate classifier and the semantic parser.

For semantic parsing and predicate classifying, features of different types are selected to our system. The effect of them on multiple languages will be described in the following sections in detail.

2 System Description

Generally Speaking, a syntactic and semantic dependency parsing system is usually divided into four separate subtasks: syntactic parsing, predicate identification, predicate classification, and semantic role labeling. In the CoNLL-2009 shared task, the predicate identification is not required, since the predicates are given for us. Therefore, the system we present is only composed of three components: a syntactic dependency parser, a predicate classifier and a semantic parser. The syntactic dependencies are processed with the MSTParser 0.4.3b. The predicates identification and semantic role label are processed with MEs-based classifier respectively. Unlike conventional systems, the predicates identifica-

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 109–113,
Boulder, Colorado, June 2009.

tion and the semantic parser are independent with each other. Figure 1 is the architecture of our system.

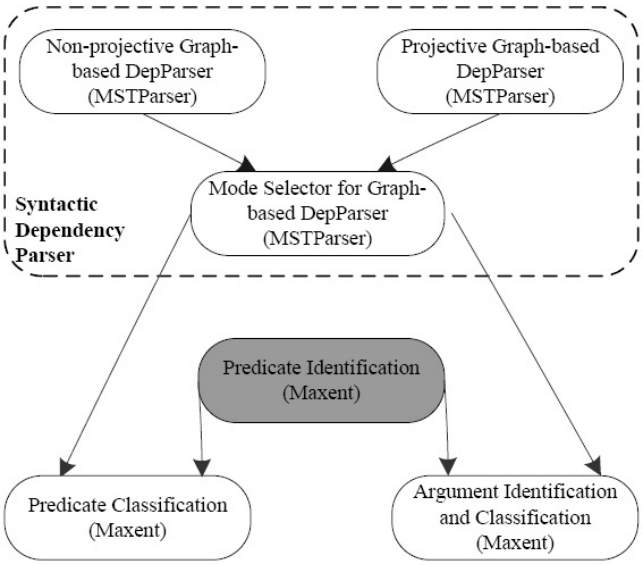

Figure 1: System Architecture

No	Features	No	Features
1	w_0	20	$Lemma$
2	p_0	21	DEPREL
3	p_{-1}	22	CHD_POS
4	p_1	23	CHD_POS_U
5	$p_{-1}p_0$	24	CHD_REL
6	p_0p_1	25	CHD_REL_U
7	$p_{-2}p_0$	26	SIB_REL
8	p_0p_2	27	SIB_REL_U
9	$p_{-3}p_0$	28	SIB_POS
10	p_0p_3	29	SIB_POS_U
11	$p_{-1}p_0p_1$	30	VERB_V
12	w_0p_0	31	4+11
13	$w_0p_{-1}p_0$	32	Indegree
14	$w_0p_0p_1$	33	Outdegree
15	$w_0p_{-2}p_0$	34	Degree
16	$w_0p_0p_2$	35	ARG_IN
17	$w_0p_{-3}p_0$	36	ARG_OUT
18	$w_0p_0p_3$	37	ARG_Degree
19	$w_0p_{-1}p_0p_1$	38	Span

Table 1: Features for Predicate Classification.

In our system, we firstly select an appropriate mode (projective or non-projective) of Graph-based Parser (MSTParser) for each language, then construct the MEs-based predicates classification and the MEs-based semantic parser with syntactic dependency relationships and predicate classification respectively.

2.1 Syntactic Dependency Parsing

MSTParser (McDonald, 2008) is used as our syntactic dependency parser. It is a state-of-the-art dependency parser that searches for maximum spanning trees (MST) over directed graph. Both of projective and non-projective are supported by MSTParser. Our system employs the first-order framework with projective and non-projective modes on seven given languages.

2.2 Predicate Classification

In this phase, we label the sense of each predicate and the MEs are adopted for classification. Features of different types are extracted for each predicate, and an optimized combination of them is adopted in our final system. Table 1 lists all features. 1-20 are the features used in Li's system (Lu Li et al., 2008), and 21-31 are a part of the optimized features presented in Che's system (Wanxiang Che et al., 2008)

In Table 1, "w" denotes the word and "p" denotes POS of the words. Features in the form of part1_part2 denote the part2 of the part1, while features in the form of part1+part2 denote the combination of the part1 and part2. "CHD" and "SIB" denote a sequence of the child and the sibling words respectively, "REL" denotes the type of relations, "U" denotes the result after reducing the adjacent duplicate tags to one, "V" denotes whether the part is a voice, "In" and "OUT" denote the in_degree and out_degree, which denotes how many dependency relations coming into this word and going away from this word,and "ARG" denotes the semantic roles of the predicate. The "Span" denotes the maximum length between the predicate and its arguments. The final optimized feature combination is :1-31 and 33-37.

2.3 Semantic Role Labeling

The semantic role labeling usually contains two subtasks: argument identification and argument classification. In our system, we perform them in a single

stage through one classifier, which specifies a particular role label to the argument candidates directly and assigns "NONE" label to the argument candidates with no role. MEs are also adopted for classification. For each word in a sentence, MEs gives each candidate label (including semantic role labels and none label) a probability for the predicate. The features except for the feature (lemma plus sense number of the predicate in (Lu Li et al., 2008)) and the features 32-38 in Table 1 are selected in our system.

3 Experiments and Results

We train the first-order MSTParser [1] with projective and non-projective modes in terms of default parameters respectively. Our maximum entropy classifiers are implemented with the Maximum Entropy Modeling Toolkit [2] . The default classifier parameters are used in our system except for iterations. All models are trained using all training data, and tested on the whole development data and test data, with 64-bit 3.00GHz Intel(R) Pentium(R) D CPU and 4.0G memory.

3.1 Syntactic Dependency Parsing

Table 2 is a performance comparison between projective parser and non-projective parser on the development data of seven languages. In Table 2, "LAS", "ULAS" and "LCS" denote as Labeled attachment score, Unlabeled attachment score and Label accuracy score respectively.

The experiments show that Catalan, Chinese and Spanish have projective property and others have non-projective property.

3.2 Predicate Classification

To get the optimized system, three group features are used for comparison.

- group 1: features 1-20 in Table 1.
- group 2: features 1-31 in Table 1.
- group 3: all features in Table 1.

The performance of predicate classification on the development data of the six languages, which contain this subtask, are given in Table 3. The results show that Che's features and the degrees of the predicate and its arguments are useful for all languages, the former improves the labeled F1 measure by 0.3% to 5.4%, and the latter by 0.3% to 1.7%.

[1]http://sourceforge.net/projects/mstparser.

[2]http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html.

	LAS(%)	ULAS(%)	LCS(%)
Catalan	84.18	88.18	91.76
	83.69	87.74	91.59
Chinese	72.58	77.06	82.07
	62.85	69.47	73.00
Czech	72.79	81.40	80.93
	73.18	81.86	81.30
English	86.89	90.29	91.50
	86.88	90.34	91.58
German	83.43	86.89	90.24
	84.00	87.40	90.61
Japanese	92.23	93.16	98.38
	92.23	93.14	98.45
Spanish	83.88	87.93	91.36
	83.46	87.46	91.37

Table 2: Performance of Syntactic Dependency Parsing with different modes. The above line is the performance of projective mode, while the below one is the performance of non-projective mode for each language.

	group 1	group 2	group 3
Catalan	75.51	80.90	82.23
Chinese	93.79	94.99	94.75
Czech	91.83	91.77	91.86
English	92.12	92.48	93.20
German	74.49	74.14	75.85
Spanish	74.01	76.22	76.53

Table 3: Performance of predicate classification (F1 scores) for different group features on the development data of the six languages.

3.3 Semantic Role Labeling

In this phase, feature selection and performance lose caused by P-columns are studied. Firstly, we compare the following two group features:

- group 1: The features except for the lemma plus sense number of the predicate in (Lu Li et al., 2008).

	LF1	ULF1	PF1
Catalan	73.25	92.69	38.41
	72.71	91.93	35.22
	83.23	100.00	61.88
Chinese	69.60	82.15	28.35
	71.49	81.71	29.41
	85.44	95.21	58.20
Czech	80.62	92.49	70.04
	79.10	91.44	68.34
	85.42	96.93	77.78
English	73.91	87.26	33.16
	76.10	88.58	36.28
	79.35	91.74	43.32
German	64.85	88.05	27.21
	65.36	88.63	26.70
	72.78	94.54	41.50
Japanese	69.43	82.79	29.27
	69.87	83.31	29.69
	72.80	87.13	34.96
Spanish	73.49	93.15	39.64
	78.18	91.68	33.57
	81.96	99.98	59.20

Table 4: Performance of Semantic Role Labeling (F1 score) with different features.

- group 2: group1+the degrees of the predicate and its arguments presented in the last section.

Secondly, features extracted from golden-columns and P-columns are both used for testing.

The performance of them are given in Table 4, where "LF1", "ULF1" and "PF1" denote as Labeled F1 score, Unlabeled F1 score and Proposition F1 score respectively. The above line is the F1 scores of Semantic Role Labeling with different features. The uppermost line is the result of group1 features, the middle line is the result of group2 features extracted from P-columns, and the downmost one is the result of group2 features extracted from golden-columns for each language.

The results show that the features of degree also improves the labeled F1 measure by 3.4% to 15.8%, the different labeled F1 between golden-columns and P-columns is about 2.9%–13.9%.

	LAS	LF1	M_LF1
Catalan	84.18	72.71	81.46
	75.68	66.95	71.32
Chinese	72.58	71.49	72.20
	63.95	67.06	65.53
Czech	73.18	79.10	76.37
	72.60	79.08	75.85
Czech-ood	69.81	79.80	74.81
English	86.88	76.10	82.89
	86.61	77.17	81.92
English-ood	80.09	67.21	73.69
German	84.00	65.36	83.06
	79.85	61.98	70.93
German-ood	71.86	61.83	66.86
Japanese	92.23	69.87	83.77
	91.26	69.58	80.49
Spanish	83.88	71.18	80.74
	77.21	66.23	71.72

Table 5: Overall performance of our final joint system.

3.4 Overall Performance

In the final system, we select the optimized feature subset discussed in the former sections. The overall performance of the system on the development data , test data and Out-of-domain data are shown in Table 5 (all features are extracted from P-columns). The average Macro F1 Scores of our system are 73.97 on test data and 71.79 on Out-of-domain data.

In Table 5, "LAS", "LF1" and "M_LF1" denote as Labeled accuracy score for Syntactic Dependency Parsing, Labeled F1 score for Semantic Role Labeling, and Overall Macro Labeled F1 score respectively. The topmost line is the result on the development data, the middle one is the result on the test data for each language and the downmost one is the result on the Out-of-domain data if the data exist.

4 Conclusion and Discussion

We present a joint syntactic and semantic dependency parsing system for CoNLL2009 Shared Task, which composed of three components: a syntactic dependency parser, a predicate classifier and a semantic parser. All of them are built with some state-of-the-art methods. For the predicate classifier and the semantic parser, a new kind of features—

degrees, which reflect the activeness of the words in a sentence improves their performance. In order to improve the performance further, we will study new machine learning methods for semantic dependency parsing, especially the joint learning methods, which can avoid the information loss problem of our system.

Acknowledgments

We would like to thank McDonald for providing the MSTParser program, to Zhang Le for providing the Maxent program. This research has been partially supported by the National Natural Science Foundation of China(No.60703015) and the National 863 Program of China (No.2006AA01Z197, No.2007AA01Z194).

References

Jan Hajič and Massimiliano Ciaramita and Richard Johansson and Daisuke Kawahara and Maria Antònia Martí and Lluís Màrquez and Adam Meyers and Joakim Nivre and Sebastian Padó and Jan Štěpánek and Pavel Straňák and Miahi Surdeanu and Nianwen Xue and Yi Zhang. 2009. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009), June 4-5*. Boulder, Colorado, USA.

Mariona Taulé and Maria Antònia Martí and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008)*. Marrakesh, Morroco.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1),pages 143–172.

Jan Hajič and Jarmila Panevová and Eva Hajičová and Petr Sgall and Petr Pajas and Jan Štěpánek and Jiří Havelka and Marie Mikulová and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0. *CD-ROM*, Cat. No. LDC2006T01, ISBN 1-58563-370-4. Linguistic Data Consortium, Philadelphia, Pennsylvania, USA. URL: http://ldc.upenn.edu.

Surdeanu, Mihai and Johansson, Richard and Meyers, Adam and Màrquez, Lluís and Nivre, Joakim. 2008. The CoNLL-2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies. *Proceedings of the 12th Conference on Computational Natural Language Learning(CoNLL-2008)*.

Aljoscha Burchardt and Katrin Erk and Anette Frank and Andrea Kowalski and Sebastian Padó and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. *Proceedings of the 5rd International Conference on Language Resources and Evaluation (LREC-2006)*, pages 2008–2013. Genoa, Italy.

Daisuke Kawahara and Sadao Kurohashi and Kôiti Hasida. 2002. Construction of a Japanese Relevance-tagged Corpus. *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002)*, pages 2008–2013. Las Palmas, Canary Islands.

McDonald and Ryan. 2006. *Discriminative Learning and Spanning Tree Algorithms for Dependency Parsing*, Ph.D. thesis. University of Pennsylvania.

Lu Li, Shixi Fan, Xuan Wang, XiaolongWang. 2008. Discriminative Learning of Syntactic and Semantic Dependencies. *CoNLL 2008: Proceedings of the 12th Conference on Computational Natural Language Learning*, pages 218–222. Manchester.

Wanxiang Che, Zhenghua Li, Yuxuan Hu, Yongqiang Li, Bing Qin, Ting Liu, Sheng Li. 2008. A Cascaded Syntactic and Semantic Dependency Parsing System. *CoNLL 2008: Proceedings of the 12th Conference on Computational Natural Language Learning*, pages 238–242. Manchester.

Multilingual Syntactic-Semantic Dependency Parsing with Three-Stage Approximate Max-Margin Linear Models

Yotaro Watanabe, Masayuki Asahara and Yuji Matsumoto
Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, Japan, 630-0192
{yotaro-w, masayu-a, matsu}@is.naist.jp

Abstract

This paper describes a system for syntactic-semantic dependency parsing for multiple languages. The system consists of three parts: a state-of-the-art higher-order projective dependency parser for syntactic dependency parsing, a predicate classifier, and an argument classifier for semantic dependency parsing. For semantic dependency parsing, we explore use of global features. All components are trained with an approximate max-margin learning algorithm.

In the closed challenge of the CoNLL-2009 Shared Task (Hajič et al., 2009), our system achieved the 3rd best performances for English and Czech, and the 4th best performance for Japanese.

1 Introduction

In recent years, joint inference of syntactic and semantic dependencies has attracted attention in NLP communities. Ideally, we would like to choose the most plausible syntactic-semantic structure among all possible structures in that syntactic dependencies and semantic dependencies are correlated. However, solving this problem is too difficult because the search space of the problem is extremely large. Therefore we focus on improving performance for each subproblem: dependency parsing and semantic role labeling.

In the past few years, research investigating higher-order dependency parsing algorithms has found its superiority to first-order parsing algorithms. To reap the benefits of these advances, we use a higher-order projective dependency parsing algorithm (Carreras, 2007) which is an extension of the span-based parsing algorithm (Eisner, 1996), for syntactic dependency parsing.

In terms of semantic role labeling, we would like to capture global information about predicate-argument structures in order to accurately predict the correct predicate-argument structure. Previous research dealt with such information using *re-ranking* (Toutanova et al., 2005; Johansson and Nugues, 2008). We explore a different approach to deal with such information using *global features*. Use of global features for structured prediction problem has been explored by several NLP applications such as sequential labeling (Finkel et al., 2005; Krishnan and Manning, 2006; Kazama and Torisawa, 2007) and dependency parsing (Nakagawa, 2007) with a great deal of success. We attempt to use global features for argument classification in which the most plausible semantic role assignment is selected using both local and global information. We present an approximate max-margin learning algorithm for argument classifiers with global features.

2 Dependency Parsing

As in previous work, we use a linear model for dependency parsing. The score function used in our dependency parser is defined as follows.

$$s(\mathbf{y}) = \sum_{(h,m)\in\mathbf{y}} F(h, m, \mathbf{x}) \qquad (1)$$

where h and m denote the head and the dependent of the dependency edge in $\mathbf{y}$, and $F(h, m, \mathbf{x})$ is a *Factor* that specifies dependency edge scores.

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 114–119,
Boulder, Colorado, June 2009.

We used a second-order factorization as in (Carreras, 2007). The second-order factor F is defined as follows.

$$F(h, m, \mathbf{x}) = \mathbf{w} \cdot \Phi(h, m, \mathbf{x}) + \mathbf{w} \cdot \Phi(h, m, c_h, \mathbf{x}) + \mathbf{w} \cdot \Phi(h, m, c_{mi}, \mathbf{x}) + \mathbf{w} \cdot \Phi(h, m, c_{mo}, \mathbf{x}) \quad (2)$$

where $\mathbf{w}$ is a parameter vector, Φ is a feature vector, c_h is the child of h in the span $[h...m]$ that is closest to m, c_{mi} is the child of m in the span $[h...m]$ that is farthest from m and c_{mo} is the child of m outside the span $[h...m]$ that is farthest from m. For more details of the second-order parsing algorithm, see (Carreras, 2007).

For parser training, we use the Passive Aggressive Algorithm (Crammer et al., 2006), which is an approximate max-margin variant of the perceptron algorithm. Also, we apply an efficient parameter averaging technique (Daumé III, 2006). The resulting learning algorithm is shown in Algorithm 1.

Algorithm 1 A Passive Aggressive Algorithm with parameter averaging

input Training set $\mathcal{T} = \{\mathbf{x}_t, \mathbf{y}_t\}_{t=1}^{T}$, Number of iterations N and Parameter C
$\mathbf{w} \leftarrow \mathbf{0}, \mathbf{v} \leftarrow \mathbf{0}, c \leftarrow 1$
for $i \leftarrow 0$ to N **do**
  for $(\mathbf{x}_t, \mathbf{y}_t) \in \mathcal{T}$ **do**
    $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \mathbf{w} \cdot \Phi(\mathbf{x}_t, \mathbf{y}) + \rho(\mathbf{y}_t, \hat{\mathbf{y}})$
    $\tau_t = \min\left(C, \frac{\mathbf{w} \cdot \Phi(\mathbf{x}_t, \hat{\mathbf{y}}) - \mathbf{w} \cdot \Phi(\mathbf{x}_t, \mathbf{y}_t) + \rho(\mathbf{y}_t, \hat{\mathbf{y}})}{||\Phi(\mathbf{x}_t, \mathbf{y}_t) - \Phi(\mathbf{x}_t, \hat{\mathbf{y}})||^2}\right)$
    $\mathbf{w} \leftarrow \mathbf{w} + \tau_t(\Phi(\mathbf{x}_t, \mathbf{y}_t) - \Phi(\mathbf{x}_t, \hat{\mathbf{y}}))$
    $\mathbf{v} \leftarrow \mathbf{v} + c\tau_t(\Phi(\mathbf{x}_t, \mathbf{y}_t) - \Phi(\mathbf{x}_t, \hat{\mathbf{y}}))$
    $c \leftarrow c + 1$
  end for
end for
return $\mathbf{w} - \mathbf{v}/c$

We set $\rho(\mathbf{y}_t, \hat{\mathbf{y}})$ as the number of incorrect head predictions in the $\hat{\mathbf{y}}$, and C as 1.0.

Among the 7 languages of the task, 4 languages (Czech, English, German and Japanese) contain non-projective edges (13.94 %, 3.74 %, 25.79 % and 0.91 % respectively), therefore we need to deal with non-projectivity. In order to avoid losing the benefits of higher-order parsing, we considered applying pseudo-projective transformation (Nivre and Nilsson, 2005). However, growth of the number of dependency labels by pseudo-projective transformation increases the dependency parser training time, so we did not adopt transformations. Therefore, the parser ignores the presence of non-projective edges in the training and the testing phases.

The features used for our dependency parser are based on those listed in (Johansson, 2008). In addition, distance features are used. We use shorthand notations in order to simplify the feature representations: 'h', 'd', 'c', 'l', 'p', '-1' and '$+1$' correspond to head, dependent, head's or dependent's child, lemma , POS, left position and right position respectively.

First-order Features

Token features: hl, hp, hl+hp, dl, dp and dl+dp.

Head-Dependent features: hp+dp, hl+dl, hl+dl, hl+hp+dl, hl+hp+dp, hl+dl+dp, hp+dl+dp and hl+hp+dl+dp.

Context features: hp+hp_{+1}+dp_{-1}+dp, hp_{-1}+hp+dp_{-1}+dp, hp+hp_{+1}+dp+dp_{+1} and hp_{-1}+hp+dp+dp_{+1}.

Distance features: The number of tokens between the head and the dependent.

Second-order Features

Head-Dependent-Head's or Dependent's Child: hl+cl, hl+cl+cp, hp+cl, hp+cp, hp+dp+cp, dp+cp, dp+cl+cp, dl+cp, dl+cp+cl

3 Semantic Role Labeling

Our SRL module consists of two parts: a predicate classifier and an argument classifier. First, our system determines the word sense for each predicate with the predicate classifier, and then it detects the highest scored argument assignment using the argument classifier with global features.

3.1 Predicate Classification

The first phase of SRL in our system is to detect the word sense for each predicate. WSD can be formalized as a multi-class classification problem given lemmas. We created a linear model for each lemma and used the Passive Aggressive Algorithm with parameter averaging to train the models.

3.1.1 Features for Predicate Classification

Word features: Predicted lemma and the predicted POS of the predicate, predicate's head, and its conjunctions.

Dependency label: The dependency label between the predicate and the predicate's head.

Dependency label sequence: The concatenation of the dependency labels of the predicate dependents.

Since effective features for predicate classification are different for each language, we performed greedy forward feature selection.

3.2 Argument Classification

In order to capture global clues of predicate-argument structures, we consider introducing global features for linear models. Let $\mathcal{A}(p)$ be a joint assignment of role labels for argument candidates given the predicate p. Then we define a score function $s(\mathcal{A}(p))$ for argument label assignments $\mathcal{A}(p)$.

$$s(\mathcal{A}(p)) = \sum_k F_k(\mathbf{x}, \mathcal{A}(p)) \tag{3}$$

We introduce two factors: *Local Factor* F_L and *Global Factor* F_G defined as follows.

$$F_L(\mathbf{x}, a(p)) = \mathbf{w} \cdot \Phi_L(\mathbf{x}, a(p)) \tag{4}$$

$$F_G(\mathbf{x}, \mathcal{A}(p)) = \mathbf{w} \cdot \Phi_G(\mathbf{x}, \mathcal{A}(p)) \tag{5}$$

where Φ_L, Φ_G denote feature vectors for the local factor and the global factor respectively. F_L scores a particular role assignment for each argument candidate individually, and F_G treats global features that capture what structure the assignment $\mathcal{A}$ has. Resulting scoring function for the assignment $\mathcal{A}(p)$ is as follows.

$$s(\mathcal{A}(p)) = \sum_{a(p) \in \mathcal{A}(p)} \mathbf{w} \cdot \Phi_L(\mathbf{x}, a(p)) + \mathbf{w} \cdot \Phi_G(\mathbf{x}, \mathcal{A}(p)) \tag{6}$$

Use of global features is problematic, because it becomes difficult to find the highest assignment efficiently. In order to deal with the problem, we use a simple approach, n-best relaxation as in (Kazama and Torisawa, 2007). At first we generate n-best assignments using only the local factor, and then add the global factor score for each n-best assignment, finally select the best scoring assignment from them. In order to generate n-best assignments, we used a beam-search algorithm.

3.2.1 Learning the Model

As in dependency parser and predicate classifier, we train the model using the PA algorithm with parameter averaging. The learning algorithm is shown in Algorithm 2. In this algorithm, the weights correspond to local factor features Φ_L and global factor features Φ_G are updated simultaneously.

Algorithm 2 Learning with Global Features for Argument Classification

input Training set $\mathcal{T} = \{\mathbf{x_t}, \mathcal{A}_t\}_{t=1}^{T}$, Number of iterations N and Parameter C
$\mathbf{w} \leftarrow \mathbf{0}$, $\mathbf{v} \leftarrow \mathbf{0}$, $c \leftarrow 1$
for $i \leftarrow 0$ to N **do**
  for $(\mathbf{x}_t, \mathcal{A}_t) \in \mathcal{T}$ **do**
    let $\Phi(\mathbf{x}_t, \mathcal{A}) = \sum_{a \in \mathcal{A}} \Phi_L(\mathbf{x}_t, a) + \Phi_G(\mathbf{x}_t, \mathcal{A})$
    generate n-best assignments $\{\mathcal{A}^n\}$ using F_L
    $\hat{\mathcal{A}} = \arg\max_{\mathcal{A} \in \{\mathcal{A}^n\}} \mathbf{w} \cdot \Phi(\mathbf{x}_t, \mathcal{A}) + \rho(\mathcal{A}_t, \mathcal{A})$
    $\tau_t = \min\left(C, \frac{\mathbf{w} \cdot \Phi(\mathbf{x}_t, \hat{\mathcal{A}}) - \mathbf{w} \cdot \Phi(\mathbf{x}_t, \mathcal{A}_t) + \rho(\mathcal{A}_t, \hat{\mathcal{A}})}{||\Phi(\mathbf{x}_t, \mathcal{A}_t) - \Phi(\mathbf{x}_t, \hat{\mathcal{A}})||^2}\right)$
    $\mathbf{w} \leftarrow \mathbf{w} + \tau_t(\Phi(\mathbf{x}_t, \mathcal{A}_t) - \Phi(\mathbf{x}_t, \hat{\mathcal{A}}))$
    $\mathbf{v} \leftarrow \mathbf{v} + c\tau_t(\Phi(\mathbf{x}_t, \mathcal{A}_t) - \Phi(\mathbf{x}_t, \hat{\mathcal{A}}))$
    $c \leftarrow c + 1$
  end for
end for
return $\mathbf{w} - \mathbf{v}/c$

We set the margin value $\rho(\mathcal{A}, \hat{\mathcal{A}})$ as the number of incorrect assignments plus $\delta(\mathcal{A}, \hat{\mathcal{A}})$, and C as 1.0. The delta function returns 1 if at least one assignment is different from the correct assignment and 0 otherwise.

The model is similar to re-ranking (Toutanova et al., 2005; Johansson and Nugues, 2008). However in contrast to re-ranking, we only have to prepare one model. The re-ranking approach requires other training datasets that are different from the data used in local model training.

3.2.2 Features for Argument Classification

The local features used in our system are the same as our previous work (Watanabe et al., 2008) except for language dependent features. The global features that used in our system are based on (Johansson and Nugues, 2008) that used for re-ranking.

Local Features

Word features: Predicted lemma and predicted POS of the predicate, predicate's head, argument candidate, argument candidate's head, leftmost/rightmost dependent and leftmost/rightmost sibling.

Dependency label: The dependency label of predicate, argument candidate and argument candidate's dependent.

Family: The position of the argument candicate with respect to the predicate position in the dependency tree (e.g. child, sibling).

	Average	Catalan	Chinese	Czech	English	German	Japanese	Spanish
Macro F1 Score	78.43	75.91	73.43	81.43	86.40	69.84	84.86	77.12
	(78.00*)	(74.83*)	(73.43*)	(81.38*)	(86.40*)	(68.39*)	(84.84*)	(76.74*)
Semantic Labeled F1	75.65	72.35	74.17	84.69	84.26	63.66	77.93	72.50
	(75.17*)	(71.05*)	(74.17*)	(84.66*)	(84.26*)	(61.94*)	(77.91*)	(72.25*)
Labeled Syntactic Accuracy	81.16	79.48	72.66	78.17	88.54	75.85	91.69	81.74
	(80.77*)	(78.62*)	(72.66*)	(78.10*)	(88.54*)	(74.60*)	(91.66*)	(81.23*)
Macro F1 Score	84.30	84.79	81.63	83.08	87.93	83.25	85.54	83.94
Semantic Labeled F1	81.58	80.99	79.99	86.67	85.09	79.46	79.03	79.85
Labeled Syntactic Accuracy	87.02	88.59	83.27	79.48	90.77	87.03	91.96	88.04

Table 1: Scores of our system.

Position: The position of the head of the dependency relation with respect to the predicate position in the sentence.

Pattern: The left-to-right chain of the predicted POS/dependency labels of the predicate's children.

Path features: Predicted lemma, predicted POS and dependency label paths between the predicate and the argument candidate.

Distance: The number of dependency edges between the predicate and the argument candidate.

Global Features

Predicate-argument label sequence: The sequence of the predicate sense and argument labels in the predicate-argument strucuture.

Presence of labels defined in frame files: Whether the semantic roles defined in the frame present in the predicate-argument structure (e.g. MISSING:A1 or CONTAINS:A1.)

3.2.3 Argument Pruning

We observe that most arguments tend to be not far from its predicate, so we can prune argument candidates to reduce search space. Since the characteristics of the languages are slightly different, we apply two types of pruning algorithms.

Pruning Algorithm 1: Let S be an argument candidate set. Initially set $S \leftarrow \phi$ and start at predicate node. Add dependents of the node to S, and move current node to its parent. Repeat until current node reaches to ROOT.

Pruning Algorithm 2: Same as the Algorithm 1 except that added nodes are its grandchildren as well as its dependents.

The pruning results are shown in Table 2. Since we could not prune arguments in Japanese accurately using the two algorithms, we pruned argument candidates simply by POS.

	algorithm	coverage (%)	reduction (%)
Catalan	1	100	69.1
Chinese	1	98.9	69.1
Czech	2	98.5	49.1
English	1	97.3	63.1
German	1	98.3	64.3
Japanese	POS	99.9	41.0
Spanish	1	100	69.7

Table 2: Pruning results.

4 Results

The submitted results on the test data are shown in the upper part of Table 1. Due to a bug, we mistakenly used the gold lemmas in the dependency parser. Corrected results are shown in the part marked with *. The lower part shows the post evaluation results with the gold lemmas and POSs.

For some of the 7 languages, since the global model described in Section 3.2 degraded performance compare to a model trained with only F_L, we did **NOT** use the model for all languages. We used the global model for only three languages: Chinese, English and Japanese. The remaining languages (Catalan, Czech, German and Spanish) used a model trained with only F_L.

4.1 Dependency Parsing Results

The parser achieved relatively high accuracies for Czech, English and Japanese, and for each language, the difference between the performance with correct POS and predicted POS is not so large. However, in Catalan, Chinese German and Spanish, the parsing accuracies was seriously degraded by replacing correct POSs with predicted POSs (6.3 - 11.2 %). This is likely because these languages have relatively low predicted POS accuracies (92.3 - 95.5 %) ; Chinese

	F_L	F_L+F_G	(ΔP, ΔR)
Catalan	**85.80**	85.68	(+0.01, -0.26)
Chinese	86.58	**87.39**	(+0.24, +1.36)
Czech	**89.63**	89.05	(-0.87, -0.28)
English	85.66	**85.74**	(-0.87, +0.98)
German	**80.82**	77.30	(-7.27, +0.40)
Japanese	79.87	**81.01**	(+0.17, +1.88)
Spanish	**84.38**	83.89	(-0.42, -0.57)

Table 3: Effect of global features (semantic labeled F1). ΔP and ΔR denote the differentials of labeled precision and labeled recall between F_L and F_L+F_G respectively.

has especially low accuracy (92.3%). The POS accuracy may affect the parsing performances.

4.2 SRL Results

In order to highlight the effect of the global features, we compared two models. The first model is trained with only the local factor F_L. The second model is trained with both the local factor F_L and the global factor F_G. The results are shown in Table 3. In the experiments, we used the development data with gold parse trees. For Chinese and Japanese, significant improvements are obtained using the global features (over +1.0% in labeled recall and the slightly better labeled precision). However, for Catalan, Czech, German and Spanish, the global features degraded the performance in labeled F1. Especially, in German, the precision is substantially degraded (-7.27% in labeled F1). These results indicate that it is necessary to introduce language dependent features.

4.3 Training, Evaluation Time and Memory Requirements

Table 4 and 5 shows the training/evaluation times and the memory consumption of the second-order dependency parsers and the global argument classifiers respectively. The training times of the predicate classifier were less than one day, and the testing times were mere seconds.

As reported in (Carreras, 2007; Johansson and Nugues, 2008), training and inference of the second-order parser are very expensive. For Chinese, we could only complete 2 iterations.

In terms of the argument classifier, since N-best generation time account for a substantial proportion of the training time (in this work $N = 100$), changing N affects the training and evaluation times significantly.

	iter	hrs./iter	sent./min.	mem.
Catalan	9	14.6	9.0	9.6 GB
Chinese	2	56.5	3.7	16.2 GB
Czech	8	14.6	20.5	12.6 GB
English	7	22.0	13.4	15.1 GB
German	4	12.3	59.1	13.1 GB
Japanese	7	11.2	21.8	13.0 GB
Spanish	7	19.5	7.3	17.9 GB

Table 4: Training, evaluation time and memory requirements of the second-order dependency parsers. The 'iter' column denote the number of iterations of the model used for the evaluations. Catalan, Czech and English are trained on Xeon 3.0GHz, Chinese and Japanese are trained on Xeon 2.66GHz, German and Spanish are trained on Opteron 2.3GHz machines.

	train (hrs.)	sent./min.	mem.
Chinese	6.5	453.7	2.0 GB
English	13.5	449.8	3.2 GB
Japanese	3.5	137.6	1.1 GB

Table 5: Training, evaluation time and memory requirements of the global argument classifiers. The classifiers are all trained on Opteron 2.3GHz machines.

All modules of our system are implemented in Java. The required memory spaces shown in Table 4 and 5 are calculated by subtracting free memory size from the total memory size of the Java VM. Note that we observed that the value fluctuated drastically while measuring memory usage, so the value may not indicate precise memory requirements of our system.

5 Conclusion

In this paper, we have described our system for syntactic and semantic dependency analysis in multilingual. Although our system is not a *joint* approach but a *pipeline* approach, the system is comparable to the top system for some of the 7 languages.

A further research direction we are investigating is the application of various types of global features. We believe that there is still room for improvements since we used only two types of global features for the argument classifier.

Another research direction is investigating joint approaches. To the best of our knowledge, three

types of joint approaches have been proposed: N-best based approach (Johansson and Nugues, 2008), synchronous joint approach (Henderson et al., 2008), and a joint approach where parsing and SRL are performed simultaneously (Lluís and Màrquez, 2008). We attempted to perform N-best based joint approach, however, the expensive computational cost of the 2nd-order projective parser discouraged it. We would like to investigate syntactic-semantic joint approaches with reasonable time complexities.

Acknowledgments

We would like to thank Richard Johansson for his advice on parser implementation, and the CoNLL-2009 organizers (Hajič et al., 2009; Taulé et al., 2008; Palmer and Xue, 2009; Hajič et al., 2006; Surdeanu et al., 2008; Burchardt et al., 2006; Kawahara et al., 2002; Taulé et al., 2008).

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó, and Manfred Pinkal. 2006. The SALSA corpus: a German corpus resource for lexical semantics. In *Proc. of LREC-2006*, Genoa, Italy.

Xavier Carreras. 2007. Experiments with a higher-order projective dependency parser. In *Proc. of EMNLP-CoNLL 2007*.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *JMLR*, 7:551–585.

Hal Daumé III. 2006. *Practical Structured Learning Techniques for Natural Language Processing*. Ph.D. thesis, University of Southern California, Los Angeles, CA, August.

Jason Eisner. 1996. Three new probabilistic models for dependency parsing. In *Proc. of ICCL 1996*.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proc. of ACL 2005*.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová, and Zdeněk Žabokrtský. 2006. Prague Dependency Treebank 2.0.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proc. of CoNLL-2009*, Boulder, Colorado, USA.

James Henderson, Paola Merlo, Gabriele Musillo, and Ivan Titov. 2008. A latent variable model of synchronous parsing for syntactic and semantic dependencies. In *Proc. of CoNLL 2008*.

Richard Johansson and Pierre Nugues. 2008. Dependency-based syntactic-semantic analysis with propbank and nombank. In *Proc. of CoNLL 2008*.

Richard Johansson. 2008. *Dependency-based Semantic Analysis of Natural-language Text*. Ph.D. thesis, Lund University.

Daisuke Kawahara, Sadao Kurohashi, and Kôiti Hasida. 2002. Construction of a Japanese relevance-tagged corpus. In *Proc. of LREC-2002*, pages 2008–2013, Las Palmas, Canary Islands.

Jun'Ichi Kazama and Kentaro Torisawa. 2007. A new perceptron algorithm for sequence labeling with non-local features. In *Proc. of EMNLP-CoNLL 2007*.

Vijay Krishnan and Christopher D. Manning. 2006. An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *Proc. of ACL-COLING 2006*.

Xavier Lluís and Lluís Màrquez. 2008. A joint model for parsing syntactic and semantic dependencies. In *Proc. of CoNLL 2008*.

Tetsuji Nakagawa. 2007. Multilingual dependency parsing using global features. In *Proc. of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*.

Joakim Nivre and Jens Nilsson. 2005. Pseudo-projective dependency parsing. In *Proc. of ACL 2005*.

Martha Palmer and Nianwen Xue. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering*, 15(1):143–172.

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 *Proc. of CoNLL-2008*.

Mariona Taulé, Maria Antònia Martí, and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. In *Proc. of LREC-2008*, Marrakesh, Morroco.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2005. Joint learning improves semantic role labeling. In *Proc. of ACL 2005*.

Yotaro Watanabe, Masakazu Iwatate, Masayuki Asahara, and Yuji Matsumoto. 2008. A pipeline approach for syntactic and semantic dependency parsing. In *Proc. of CoNLL 2008*.

A Simple Generative Pipeline Approach to Dependency Parsing and Semantic Role Labeling

Daniel Zeman
Ústav formální a aplikované lingvistiky
Univerzita Karlova v Praze
Malostranské náměstí 25, Praha, CZ-11800, Czechia
zeman@ufal.mff.cuni.cz

Abstract

We describe our CoNLL 2009 Shared Task system in the present paper. The system includes three cascaded components: a generative dependency parser, a classifier for syntactic dependency labels and a semantic classifier. The experimental results show that the labeled macro F1 scores of our system on the joint task range from 43.50% (Chinese) to 57.95% (Czech), with an average of 51.07%.

1 Introduction

The CoNLL 2009 shared task is an extension of the tasks addressed in previous years: unlike the English-only 2008 task, the present year deals with seven languages; and unlike 2006 and 2007, semantic role labeling is performed atop the surface dependency parsing.

We took part in the closed challenge of the joint task.[1] The input of our system contained gold standard lemma, part of speech and morphological features for each token. Tokens which were considered *predicates* were marked in the input data. The system was required to find the following information:

- parent (syntactic dependency) for each token
- label for each syntactic dependency (token)
- label for every predicate
- for every token (predicate or non-predicate) *A* and every predicate *P* in the sentence, say whether there is a semantic relation between *P* and *A* (*A* is an argument of *P*) and if so, provide a label for the relation (role of the argument)

The organizers of the shared task provided training and evaluation data (Hajič et al., 2006; Surdeanu et al., 2008; Burchardt et al., 2006; Taulé et al., 2008; Kawahara et al., 2002; Xue and Palmer, 2009) converted to a uniform CoNLL Shared Task format.

2 System Description

The system is a sequence of three components: a surface syntactic parser, a syntactic tagger that assigns labels to the syntactic dependencies and a semantic classifier (labels both the predicates and the roles of their arguments). We did not attempt to gain advantage from training a joint classifier for all the subtasks. We did not have time to do much beyond putting together the basic infrastructure. The components 2 and 3 are thus fairly primitive.

2.1 Surface Dependency Parser

We use the parser described by Zeman (2004). The parser takes a generative approach. It has a model of dependency statistics in which a dependency is

[1] For more details on the two tasks and challenges, see Hajič et al. (2009).

Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task, pages 120–125,
Boulder, Colorado, June 2009.

specified by the lemma and tag of the parent and the child nodes, by direction (left or right) and adjacency. The core of the algorithm can be described as repeated greedy selecting of best-weighted *allowed* dependencies and adding them to the dependency tree.

There are other components which affect the dependency selection, too. They range from supporting statistical models to a few hard-coded rules. However, some features of the parser are designed to work with Czech, or even with the Prague Dependency Treebank. For instance, there is a specialized model for coordinative constructions. The model itself is statistical but it depends on the PDT annotation guidelines in various ways. Most notably, the training component recognizes coordination by the `Coord` dependency label, which is not present in other treebanks. Other rules (e.g. the constraints on the set of allowed dependencies) rely on correct interpretation of the part-of-speech tags.

In order to make the parser less language-dependent in the multilingual environment of the shared task, we disabled most of the abovementioned treebank-bound features. Of course, it led to decreased performance on the Czech data.[2]

2.2 Assignment of Dependency Labels

The system learns surface dependency labels as a function of the part-of-speech tags and features of the parent and the child node. Almost no back-off is applied. The most frequent label for the given pair of tags (and feature structures) is always selected. If the pair of tags is unknown, the label is based on the features of the child node, and if it is unknown, too, the most frequent label of the training data is selected.

Obviously, both the training and the labeling procedures have to know the dependencies. Gold standard dependencies are examined during training while parser-generated dependencies are used for real labeling.

2.3 Semantic Classifier

The semantic component solves several tasks. First, all predicates have to be labeled. Tokens that are considered predicates in the particular treebank are marked on input, so this is a simple classification problem. Again, we took the path of least resistance and trained the PRED labels as a function of gold-standard lemmas.

Second, we have to find semantic dependencies. Any token (predicate or not) can be the argument of one or more predicates. These relations may or may not be parallel to a syntactic dependency. For each token, we need to find out 1. which predicates it depends on, and 2. what is the label of its semantic role in this relation?

The task is complex and there are apparently no simple solutions to it. We learn the semantic role labels as a function of the gold-standard part of speech of the argument, the gold-standard lemma of the predicate and the flag whether there is a syntactic dependency between the two nodes or not. This approach makes it theoretically possible to make one token semantically dependent on more than one predicate. However, we have no means to control the number of the dependencies.

3 Results

The official results of our system are given in Table 1. The system made the least syntactic errors (attachment and labels) for Japanese. The Japanese treebank seems to be relatively easy to parse, as many other systems achieved very high scores on this data. At the other end of the rating scale, Chinese seems to be the syntactically hardest language. Our second-worst syntactic score was for Czech, most likely owing to the turning off all language-dependent (and Czech-biased) features of the parser.

An obvious feature of the table is the extremely poor semantic scores (in contrast to the accuracy of surface dependency attachment and labels). While the simplicity of the additional models does not seem to hurt too much the dependency labeling, it apparently is too primitive for semantic role labeling. We analyze the errors in more detail in Section 4.

The system is platform-independent;[3] we have been running all the experiments under Linux on an AMD Opteron 848 processor, 2 GHz, with 32 GB RAM. The running times and memory requirements are shown in Table 2.

[2] However, the parser – without adaptation – would not do well on Czech anyway because the PDT tags are presented in a different format in the shared task data.

[3] It is written entirely in Perl.

Language	Average	Cs	En	De	Es	Ca	Ja	Zh
Labeled macro F1	51.07	**57.95**	50.27	49.57	48.90	49.61	57.69	43.50
OOD lab mac F1	43.67	**54.49**	48.56	27.97				
Labeled syn accur	64.92	57.06	61.82	69.79	65.98	67.68	**82.66**	49.48
Unlab syn accur	70.84	66.04	70.68	72.91	71.22	73.81	**83.36**	57.87
Syn labeling accur	79.20	69.10	74.24	84.63	81.83	82.46	**95.98**	66.13
OOD lab syn acc	50.20	51.45	**62.83**	36.31				
OOD unl syn acc	58.08	60.56	**71.78**	41.90				
OOD syn labeling	69.65	65.64	**75.22**	68.08				
Semantic lab F1	32.14	**58.13**	36.05	16.44	25.36	24.19	30.13	34.71
OOD sem lab F1	32.86	**56.83**	31.77	9.98				

Table 1. The official results of the system. ISO 639-1 language codes are used (cs = Czech, en = English, de = German, es = Spanish, ca = Catalan, ja = Japanese, zh = Chinese). "OOD" means "out-of-domain test data".

Language	Cs	En	De	Es	Ca	Ja	Zh
Training sentences	43955	40613	38020	15984	14924	4643	24039
Training tokens	740532	991535	680710	477810	443317	119144	658680
Average sentence length	17	24	18	30	30	26	27
Training minutes	9:21	10:41	8:28	6:17	5:42	1:24	7:01
Training sentences per secnd	78	63	75	42	44	55	57
Training tokens per second	1320	1547	1340	1267	1296	1418	1565
Training rsize memory	3.9 GB	2.2 GB	2.7 GB	2.7 GB	2.4 GB	416 MB	1.5 GB
Test sentences	4213	2399	2000	1725	1862	500	2556
Test tokens	70348	57676	31622	50630	53355	13615	73153
Parsing minutes	6:36	3:11	2:24	5:47	6:05	0:46	5:45
Parsing sentences per second	10.6	12.6	13.9	5.0	5.1	10.9	7.4
Parsing tokens per second	178	302	220	146	146	296	212
Parsing rsize memory	980 MB	566 MB	779 MB	585 MB	487 MB	121 MB	444 MB

Table 2. Time and space requirements of the syntactic parser.

To assess the need for data, Table 3 presents selected points on the learning curve of our system. The system has been retrained on 25, 50 and 75% of the training data for each language (the selection process was simple: the first N% of sentences of the training data set were used).

Generally, our method does not seem very data-hungry. Even for Japanese, with the smallest training data set, reducing training data to 25% of the original size makes the scores drop less than 1% point. The drop for other languages lies mostly between 1 and 2 points. The exceptions are (unlabeled) syntactic attachment accuracies of Czech and Spanish, and labeled semantic F1 of Spanish and Chinese. The Chinese learning curve also contains a nonmonotonic anomaly of syntactic dependency labeling between data sizes of 50 and 75% (shown in boldface). This can be probably explained by uneven distribution of the labels in training data.

As to the comparison of the various languages and corpora, Japanese seems to be the most specific (relatively high scores even with such small data). Spanish and Catalan are related languages, their treebanks are of similar size, conform to similar guidelines and were prepared by the same team. Their scores are very similar.

4 Discussion

In order to estimate sources of errors, we are now going to provide some analysis of the data and the errors our system does.

4.1 DEPREL Coverage

The syntactic tagger (assigns DEPREL syntactic labels) and the semantic tagger (assigns PRED and APRED labels) are based on simple statistical models without sophisticated back-off techniques.

Score	TrSize	Average	Cs	En	De	Es	Ca	Ja	Zh
UnLab Syn Attach	**25%**	69.38	63.72	69.70	71.36	68.99	72.41	82.58	56.90
	50%	70.14	64.96	70.13	72.11	70.37	72.83	82.99	57.58
	75%	70.51	65.50	70.37	72.50	70.83	73.47	83.17	57.73
	100%	70.84	66.04	70.68	72.91	71.22	73.81	83.36	57.87
Syn Label	**25%**	78.47	68.28	73.79	84.21	80.67	81.92	95.70	64.71
	50%	78.94	68.68	74.08	84.44	81.59	81.99	95.86	**65.94**
	75%	79.03	68.87	74.14	84.51	81.67	82.19	95.97	**65.83**
	100%	79.20	69.10	74.24	84.63	81.83	82.46	95.98	66.13
Labeled Sem F1	**25%**	30.10	56.29	34.47	15.51	22.78	22.14	28.91	30.58
	50%	**33.85**	57.24	35.34	16.03	24.46	23.13	29.60	33.31
	75%	**31.76**	57.76	35.85	16.29	24.96	23.77	29.96	33.71
	100%	32.14	58.13	36.05	16.44	25.36	24.19	30.13	34.71
Labeled Macro F1	**25%**	49.19	55.87	49.06	48.10	46.22	47.76	56.66	40.64
	50%	50.28	56.99	49.66	48.90	47.97	48.53	57.23	42.66
	75%	50.68	57.53	50.01	49.26	48.47	49.21	57.52	42.73
	100%	51.07	57.95	50.27	49.57	48.90	49.61	57.69	43.50

Table 3. The learning curve of the principal scores.

Sparse data could pose a serious problem. So how sparse are the data? Some cue could be drawn from Table 3. However, we should also know how often the labels had to be assigned to an unknown set of input features.

DEPREL (syntactic dependency label) is estimated based on morphological tag (i.e. POS + FEAT) of both the child and parent. If the pair of tags is unknown, then it is based on the tag of the child, and if it is unknown, too, the most frequent label is chosen. Coverage is high: 93 (Czech) to 97 % (Chinese) of the pairs of tags in test data were known from training data. Moreover, the error rate on the unknown pairs is actually much *lower* than on the whole data![4]

4.2 PRED Coverage

PRED (predicate sense label) is estimated based on lemma. For most languages, this seems to be a good selection. Japanese predicate labels are always identical to lemmas; elsewhere, there are by average 1.05 (Chinese) to 1.48 (Spanish) labels per lemma; the exception is German with a label-lemma ratio of 2.33.

Our accuracy of PRED label assignment ranges from 71% (German) to 100% (Japanese). We always assign the most probable label for the given lemma; if the lemma is unknown, we copy the lemma to the PRED column. Coverage is not an issue here. It goes from 94% (Czech) to almost 100% (German).[5] The accuracy on unknown lemmas could probably be improved using the subcategorization dictionaries accompanying the training data.

Language	Lemma	PREDs
Cs	1. mít	77
	2. přijmout	8
En	1. take	20
	2. go	18
De	1. kommen	28
	2. nehmen	25
Es	1. pasar	10
	1. dar	10
	3. llevar	9
	3. hacer	9
Ca	1. fer	11
	2. pasar	9
Ja	*Always 1 PRED per lemma*	
Zh	1. 要 (yào)	8
	1. 有 (yǒu)	8
	1. 打 (dǎ)	8

Table 4. Most homonymous predicates.

[4] This might suggest that the input features are chosen inappropriately and that the DEPREL label should be based just on the morphology of the child.

[5] The coverage of Japanese is 88% but since Japanese PRED labels are exact copies of lemmas, even unknown lemmas yield 100%-correct labels.

Language	Cs	En	De	Es	Ca	Ja	Zh
Potential APRED slots	1287545	195029	12066	192103	197976	57394	329757
Filled in APREDs	87934	32968	10480	49904	52786	6547	49047
Feature pair coverage (%)	46.05	40.04	14.99	29.34	29.89	18.31	38.08
Non-empty APRED accuracy	73.19	64.65	67.37	56.90	57.89	59.20	68.77
Unlabeled precision	34.94	26.86	10.88	21.71	20.25	9.13	25.66
Unlabeled recall	62.61	63.86	97.52	93.40	92.72	22.10	67.82
Unlabeled F	44.86	37.81	19.57	35.23	33.24	12.93	37.23
Labeled precision	25.58	17.36	7.33	12.35	11.72	5.41	17.64
Labeled recall	45.83	41.28	65.70	53.15	53.67	13.08	46.64
Labeled F	32.83	24.44	13.19	20.05	19.24	7.65	25.60

Table 5. APRED detailed analysis. Non-empty APRED accuracy includes only APRED cells that were non-empty both in gold standard and system output. Feature-pair coverage includes all cells filled by the system. Unlabeled precision and recall count non-empty vs. empty APREDs without respect to their actual labels. Counted on development data with gold-standard surface syntax.

4.3 APRED Assignment Analysis

The most complicated part of the task is the assignment of the APRED labels. In a sense, APRED labeling is dependency parsing on a deeper level. It consists of several sub-problems:

- Is the node an argument of any predicate at all?
- If so, how many predicates is the node argument of? Should the predicate be, say, coordination, then the node would semantically depend on all members of the coordination.
- In what way is the semantic dependency related to the syntactic dependency between the node and its syntactic parent? In majority of cases, syntactic and semantic dependencies go parallel; however, there are still a significant number of semantic relations for which this assumption does not hold.[6]
- Once we know that there is a semantic relation (an APRED field should not be empty), we still have to figure out the correct APRED label. This is the semantic role labeling (or tagging) proper.

Our system always makes semantic roles parallel to surface syntax. It even does not allow for empty APRED if there is a syntactic dependency—this turned out to be one of the major sources of errors.[7]

The role labels are estimated based on the lemma of the predicate and the part of speech of the argument. Low coverage of this pair of features in the training data turns to be another major source of errors. If the pair is not known from training data, the system selects the most frequent APRED in the given treebank. Table 5 gives an overview of the principal statistics relevant to the analysis of APRED errors.

5 Post-evaluation Experiments

Finally, we performed some preliminary experiments focused on the syntactic parser. As mentioned in Section 2.1, many features of the parser have to be turned off unless the parser understands the part-of-speech and morphological features. We used DZ Interset (Zeman, 2008) to convert Czech and English CoNLL POS+FEAT strings to PDT-like positional tags. Then we switched back on the parser options that use up the tags and re-ran parsing. The results (Table 6) confirm that the tag manipulation significantly improves Czech parsing while it does not help with English.

[6] Nearly all Spanish and Catalan semantic dependencies are parallel to syntactic ones (but not all syntactic dependencies are also semantic); in most other languages, about two thirds of semantic relations match syntax. Japanese is the only language in which this behavior does not prevail.

[7] This is a design flaw that we overlooked. Most likely, making empty APRED one of the predictable values would improve accuracy.

	Cs	En
Before	65.81	69.48
After	71.76	68.92

Table 6. Unlabeled attachment accuracy on development data before and after tagset conversion.

6 Conclusion

We described one of the systems that participated in the CoNLL 2009 Shared Task. We analyzed the weaknesses of the system and identified possible room for improvement. The most important point to focus on in future work is specifying where APRED should be filled in. The links between syntactic and semantic structures have to be studied further. Subcategorization frames could probably help improve these decisions, too—our present system ignores the subcategorization dictionaries that accompany the participating treebanks.

Acknowledgments

This research has been supported by the Ministry of Education of the Czech Republic, project No. MSM0021620838.

References

Aljoscha Burchardt, Katrin Erk, Anette Frank, Andrea Kowalski, Sebastian Padó and Manfred Pinkal. 2006. The SALSA Corpus: a German Corpus Resource for Lexical Semantics. *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC-2006).* Genova, Italy.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antonia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue and Yi Zhang. 2009. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009).* June 4-5. pp. 3-22. Boulder, Colorado, USA.

Jan Hajič, Jarmila Panevová, Eva Hajičová, Petr Sgall, Petr Pajas, Jan Štěpánek, Jiří Havelka, Marie Mikulová and Zdeněk Žabokrtský. 2006. *The Prague Dependency Treebank 2.0.* CD-ROM. Linguistic Data Consortium, Philadelphia, Pennsylvania, USA. ISBN 1-58563-370-4. LDC Cat. No. LDC2006T01. URL: http://ldc.upenn.edu/.

Daisuke Kawahara, Sadao Kurohashi and Koiti Hasida. 2002. Construction of a Japanese Relevance-tagged Corpus. *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC-2002).* pp. 2008-2013. Las Palmas, Spain.

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 the 12th Conference on Computational Natural Language Learning (CoNLL-2008).* August 16 – 17. Manchester, UK.

Mariona Taulé, Maria Antònia Martí and Marta Recasens. 2008. AnCora: Multilevel Annotated Corpora for Catalan and Spanish. *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008).* Marrakech, Morocco.

Nianwen Xue and Martha Palmer. 2009. Adding semantic roles to the Chinese Treebank. *Natural Language Engineering,* **15**(1):143-172.

Daniel Zeman. 2004. *Parsing with a Statistical Dependency Model* (PhD thesis). Univerzita Karlova, Praha, Czechia. URL: http://ufal.mff.cuni.cz/~zeman/projekty/parser/index.html

Daniel Zeman. 2008. Reusable Tagset Conversion Using Tagset Drivers. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC-2008).* ISBN 2-9517408-4-0. Marrakech, Morocco.

Author Index

Asahara, Masayuki, 114

Björkelund, Anders, 43
Bohnet, Bernd, 67
Bott, Stefan, 79

Che, Wanxiang, 49
Chen, Enhong, 19
Chen, Wenliang, 55, 61
Ciaramita, Massimiliano, 1

Dai, Qifeng, 19

Emms, Martin, 73

Gesmundo, Andrea, 37
Guo, Yuhang, 49

Hafdell, Love, 43
Hajič, Jan, 1
Henderson, James, 37

Ji, Donghong, 97
Johansson, Richard, 1

Kawahara, Daisuke, 1
Kazama, Jun'ichi, 61
Kity, Chunyu, 55

Li, Baoli, 73
Li, Lu, 109
Li, Xinxin, 109
Li, Yongqiang, 49
Li, Zhenghua, 49
Liu, Ting, 49
Lluís, Xavier, 79
Luz, Saturnino, 73

Màrquez, Lluís, 1, 79
Martí, Maria Antònia, 1
Matsumoto, Yuji, 114

Merlo, Paola, 37
Meyers, Adam, 1
Meza-Ruiz, Ivan, 85
Morante, Roser, 25
Moreau, Erwan, 91

Nivre, Joakim, 1
Nugues, Pierre, 43

Oepen, Stephan, 31

Padó, Sebastian, 1

Qin, Bing, 49

Ren, Han, 97
Riedel, Sebastian, 85

Shi, Liu, 19
Štěpánek, Jan, 1
Straňák, Pavel, 1
Surdeanu, Mihai, 1

Täckström, Oscar, 103
Tang, Buzhou, 109
Tellier, Isabelle, 91
Titov, Ivan, 37
Torisawa, Kentaro, 61

Uchimoto, Kiyotaka, 61

Van Asch, Vincent, 25
van den Bosch, Antal, 25
Vogel, Carl, 73

Wan, Jing, 97
Wang, Rui, 31
Wang, Xiaolong, 109
Wang, Xuan, 109
Watanabe, Yotaro, 114

Xue, Nianwen, 1

Zeman, Daniel, 120
Zhang, Mingyao, 97
Zhang, Yi, 1, 31
Zhao, Hai, 55, 61
Zhou, Guodong, 55