Computational Linguistics in the Netherlands Journal 2 (2012) 97–117 Submitted 09/2012; Published 12/2012

Joint learning of dependency parsing and semantic role labeling

Antal van den Bosch*, **Roser Morante****, **and Sander Canisius*****

* *Centre for Language Studies, Radboud University Nijmegen*

** *Computational Linguistics and Psycholinguistics Research Centre, University of Antwerp*

*** *Netherlands Cancer Institute*

Abstract

When natural language processing tasks overlap in their linguistic input space, they can be technically merged. Applying machine learning algorithms to the new joint task and comparing the results of joint learning with disjoint learning of the original tasks may bring to light the linguistic relatedness of the two tasks. We present a joint learning experiment with dependency parsing and semantic role labeling of Catalan and Spanish. The developed systems are based on local memory-based classifiers predicting constraints on the syntactic and semantic dependency relations in the resulting graph based on the same input features. In a second global phase, a constraint satisfaction inference procedure produces a dependency graph and semantic role label assignments for all predicates in a sentence. The comparison between joint and disjoint learning shows that dependency parsing is better learned in a disjoint setting, while semantic role labeling benefits from joint learning. We explain the results by providing an analysis of the output of the systems.

1. Introduction

In this article we present a series of comparative experiments in which a single natural language processing architecture is trained to perform dependency parsing and semantic role labeling both in isolation and jointly. In the joint system, the two labeling tasks are learned as a single joint task, by merging the syntactic and semantic dependencies. The goal of these experiments is to investigate the performance of the same architecture in a joint and disjoint setting.

Semantic role labeling and dependency parsing are two well defined tasks as shows the fact that several CoNLL Shared Tasks have been organised for each of the tasks (Carreras and Màrquez 2004, Carreras and Màrquez 2005, Nivre et al. 2007) and for the two tasks in combination (Surdeanu et al. 2008, Hajič et al. 2009). In their currently popular definitions, dependency parsing and semantic role labeling are partially overlapping tasks. The definitions state that they map to differently structured output spaces: dependency graphs span sentences, while semantic role assignments center around predicates. Yet, the spaces overlap; in a dependency graph verbal predicates will tend to have dependency relations with the same modifiers that have a semantic role as argument of that predicate. In general, even though the labels are different, syntactic dependencies between two words often co-occur with certain semantic roles. The "subject" dependency relation, for example, often co-occurs with the "A0" label that denotes the agentive role in the PropBank annotation scheme (Palmer et al. 2005).

The co-occurrence of semantic roles and syntactic functions is a well studied phenomenon from a linguistic perspective. According to Linking Theory (Hale and Keyser 1993, Levin and Rappaport 1995, Hale and Keyser 2002, Randall 2009) regular mappings can be established from the semantic to the syntactic level. The theory assumes a hierarchy of semantic roles which can be mapped to a hierarchy of syntactic positions. Theories of argument realization (Levin and Rappaport 2005) assume a close connection between verb meaning and syntactic structure and analyze how semantic roles map onto syntactic positions. Transposing these insights to the present-day paradigm of

machine learning of natural language processing, the connection between the semantic and syntactic component suggests the possibility of jointly learning the two labeling tasks as if they were one.

The learning of semantic and syntactic representations as a single task has been explored before. Musillo and Merlo (2006) and Merlo and Musillo (2008) present experiments with an enriched statistical parser that identifies and labels nodes with semantic role labels as well as syntactic labels. The CoNLL Shared Tasks of 2008 (Surdeanu et al. 2008) and 2009 (Hajič et al. 2009) combined dependency parsing and the task of identifying and labeling semantic arguments of predicates. In contrast with Musillo and Merlo (2006) and Merlo and Musillo (2008), the originality of our approach lies in the fact that a single architecture and single sets of features and parameter settings are used to perform the two tasks in isolation and jointly, which allows to make a direct comparison of the joint and non-joint results, all other conditions being equal. Thus we do not focus on producing optimized results for each of the tasks, but rather on performing a comparative analysis. Another difference is that we do not need an extra step to build the argument structure of a predicate, since this information is present in the dependency tree that our system generates. As compared to the joint systems that participated in the CoNLL Shared Tasks, our joint system concatenates syntactic and semantic edge labels so that the learning is a single learning process, rather than a two-step learning process with a non-learning intermediate step.

Setting up semantic role labeling and dependency parsing as a joint task sharing the same output space implies that the number of labels increases, and the average number of examples per label decreases. This does not rule out the application of a machine learning classifier to the joint task, but the classifier should not be too sensitive to a fragmented class space with many labels. This is the main reason our system relies on memory-based classifiers: they are practically insensitive in terms of training and processing efficiency to the number of class labels (Daelemans and Van den Bosch 2005).

Memory-based algorithms have been previously applied to processing semantic and syntactic dependencies separately. As for semantic role labeling, Morante (2008) describes a memory-based semantic role labeling system for Spanish based on gold standard dependency syntax; Morante et al. (2008) report on a semantic role labeling system for English based on syntactic dependencies produced by the MaltParser system of Nivre (2006), a typical pipeline architecture. As for dependency parsing, MaltParser uses memory-based learning as one of its optional local classifiers. Canisius et al. (2006b) present another type of memory-based dependency parser, extended later in (Canisius and Tjong Kim Sang 2007) to a constraint satisfaction-based dependency parser. The latter parser combines local memory-based classification with a sentence-global optimization method based on weighted constraint-satisfaction inference, where the local classifiers estimate syntactic relations between pairs of words, the direction of the relation from children to parents, and the relations that parents have with children. The present study adopts this system and extends it to semantic role labeling and the joint task.

We discuss the issue of joint learning of two tasks in Section 2. The data used for the experiments are introduced in Section 3. The system combining local memory-based classifiers with weighted constraint satisfaction is described in Section 4. Section 5 presents the results, in Section 6 we comment on the results, and in Section 7 we put forward some conclusions and formulate points for 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. For example, Sejnowski and Rosenberg (1987) train a back-propagation multi-layered perceptron network on the joint task of mapping a letter in its context within an English word onto a joint label representing its phonemic mapping and a marker indicating stress on that phoneme. Van den Bosch (1997) demonstrates that the joint learning of these two tasks indeed produces superior generalization performance as compared to learning the letter-phoneme

task and the stress marker assignment task separately. Buchholz (2002) transposes this idea to shallow parsing, and shows that POS tagging and base phrase chunking could be learned as a single task without any significant performance loss. Wang et al. (2008) describe a system that learns Chinese word segmentation, named entity recognition, and part-of-speech tagging jointly, outperforming a pipeline architecture baseline. Finkel and Manning show that joint learning of parsing and named entity recognition produce mildly improved performance for both tasks (Finkel and Manning 2009, Finkel and Manning 2010). Kate and Mooney (2010) present a system that extracts entities and relations jointly using a method based on a "card-pyramid" graph. All possible entities and relations are encoded in a graph so that the extraction task is reduced to jointly labeling the nodes of the graph.

Merlo and Musillo (2008) approach the learning of semantic and syntactic representations jointly in two steps. First, they generate trees that contain syntactic and semantic information by extending a statistical parser. In a second step, they recover the predicate-argument structure of each verb from the parse tree, because the parse tree does not associate semantic roles to predicates. Their system is trained and tested on the PropBank corpus of English. They show that the performance of the parser on the syntactic labels degrades in the joint learning condition (84.4% F1 versus 89.1% F1), and that the semantic role labeling task can be solved at a level of performance compared to other approaches. Their experiments do not provide a comparison of the performance of an isolated semantic role labeler versus a joint one with the same architecture.

Li et al. (2010) perform joint syntactic and semantic parsing of Chinese. Their system integrates semantic information generated by a semantic parser into the syntactic parser. They show that incorporating semantic role information into the syntactic parsing model improves the results of both syntactic and semantic parsing.

Joint learning of syntactic and semantic dependencies has been the attention of the CoNLL Shared Tasks 2008 and 2009, although, as indicated in the general overview papers (Hajič et al. 2009), most participating systems did not employ joint approaches. Of the 20 participants in the 2009 edition, only four applied some type of joint learning (Dai et al. 2009, Gesmundo et al. 2009, Lluís et al. 2009, Morante et al. 2009), and only one (Morante et al. 2009) applied a truly joint learning formulation by merging syntactic and semantic edge labels in the output space. The current article expands on this approach.

The merging of two tasks will typically lead to an increased granularity of discernible class labels, and thus to a more complex class space. In the worst case, the number of classes in the new class space is the product of the number of classes in the original tasks. In practice, if two combined tasks are to some extent related, the increase will tend to be limited, as class labels from the original tasks will tend to correlate. For instance, the POS tag for "determiner" will typically co-occur with the chunk marker for "beginning of noun phrase", and less so, or not at all, with other chunk markers. Yet, even a mild increase of the number of classes leads to a further separation 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 separation of the class space is counter-balanced by an unharmed, or even improved learnability. The latter is the primary reason for doing joint learning in the first place: certain parts of either of the combined tasks may be learned with more ease and with better success when it is co-learned with a part of another task. Learning this new separate part of the class space may in theory be easier than learning that particular part of the larger unseparated class space of either of the composing tasks.

In this article we consider the joint learning of semantic role labeling and dependency parsing. This means that our systems generate labels that carry information on semantic roles and on syntactic dependencies. Figure 1 displays a partial Catalan sentence in which it is obvious that some relations in the dependency graph, shown on top of the words, have a counterpart relation in the semantic role labeling graph centering around the sentence's main verb. Our system will be trained to assign joint syntactic and semantic labels when they coincide, such as the *suj* (subject) syntactic dependency and the *arg0-agt* (agentive) semantic role between the modifier word *treballadors* and

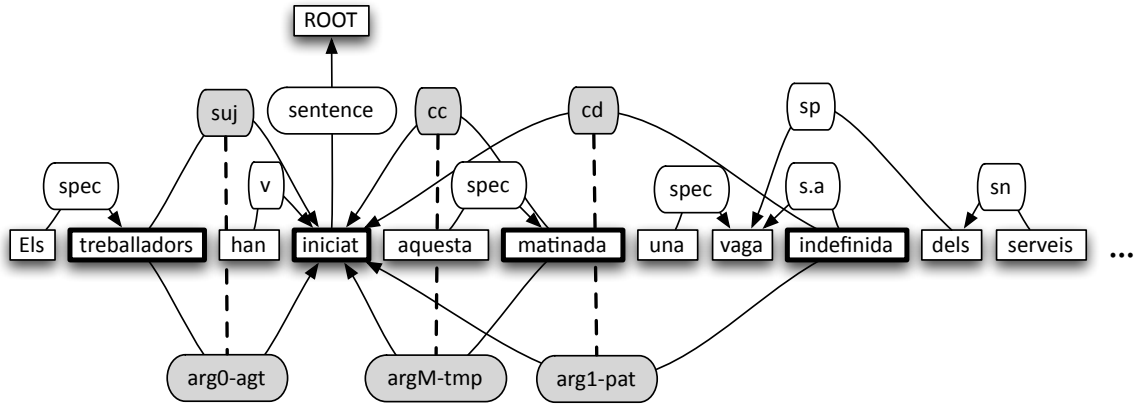

Figure 1: The relations between the dependency graph (top) and the semantic role labeling graph (bottom) of a partial Catalan example sentence.

the main verb *iniciat*. The notation we adopt in this article for joint labels uses a colon as the connecting delimiter, as in *suj:arg0-agt*.

The figure indicates that there are three more cases where syntactic and semantic relations coincide, among which *cd* (direct object) and *arg1-pat* (argument with the thematic role *patient*). All three are fairly typical co-occurrences, although common pairs such as *suj* and *arg0-agt* do not occur exclusively with each other, as we detail in the next section.

3. Data

The example sentence in Figure 1 is taken from AnCora[1], a manually (and partly semi-automatically) annotated corpus of Spanish and Catalan news text that uses the same tag sets for the two languages (Taulé et al. 2008). In the remainder of this article we describe systems trained on data from the AnCora corpus that were prepared for the CoNLL 2009 shared task (Hajič et al. 2009). This preparation process involved the conversion of the original constituent-based syntactic annotation to dependency graphs (Civit et al. 2006), the addition of automatically generated linguistic layers for part-of-speech tagging using Freeling[2], and a split into training data and test data. The Catalan corpus (AnCora-Ca) is split into a 390,302 token training set and a 53,355 token test set; the Spanish corpus (AnCora-Es) is split into a 427,442 token training set and a 50,630 token test set.

As Hajic *et al.* point out, semantic dependencies in AnCora completely match syntactic dependency structures, i.e., no new edges are introduced by the semantic structure. Furthermore, dependency trees in the AnCora data are projective; no word can be the argument of more than one predicate in a sentence; only verbal predicates are annotated; and multi-words expressions are tokenized together (Hajič et al. 2009). We chose the Catalan and Spanish corpora instead of the corpora of other languages from the CoNLL Shared Task because these characteristics facilitate the adaptation of the parser that we use to the semantic role labeling only task. We consider this setup to be valid for a first approximation to the problem, to be further explored for other languages in future research. Dealing with non-projective trees and non-overlapping semantic role and dependency graphs, although not trivial, should be possible within the framework presented here.

The syntactic tag set consists of 50 syntactic functions. At the semantic side, 45 combinations of eight semantic arguments and 21 thematic roles (such as *agt*, agent, *pat*, patient, and *cau*, cause)

1. `http://clic.ub.edu/ancora`
2. `http://www.lsi.upc.es/~nlp/freeling`

occur in joint semantic labels[3]. The total number of unique combinations of syntactic and semantic labels occurring in the corpus is 130, considerably less than the theoretical upper bound of $50 \times 45 = 2,250$. Many syntactic dependencies never co-occur with a semantic role, as they denote relations outside the part of the syntactic graph that connects predicates to arguments. At the same time, a subset of predicative syntactic dependencies always co-occur with a semantic role: *cc* (adverbial complement), *suj* (subject), *cd* (direct object), *ci* (indirect object), *creg* (prepositional complement), *atr* (attribute), and *cpred* (predicative complement). In all these cases, there are favorite combinations, often mutually. For instance, the syntactic dependency *suj* occurs 62,652 times, and the semantic role *arg0-agt* occurs 34,147 times; their combination occurs 32,010 times, which is 49.8% of all occurrences of *suj* and 93.7% of all occurrences of *arg0-agt*, in both cases the most frequent combination. Similar cases are *cd:arg1-pat* (direct objects), *cc:argm-tmp* (temporal adjuncts), and *atr:arg2-atr* (attributive objects). Yet, many of the other combinations occur quite infrequently, and are thus supported by markedly few training examples.

4. System architecture and experimental design

In this section we outline our system which was originally designed for dependency parsing by Canisius and Tjong Kim Sang (2007), and which is based on local memory-based classifiers and a constraint satisfaction inference solver. We are able to fully reuse this system for the joint syntactic–semantic task as the semantic role graphs in the AnCora corpus fully overlap with the dependency graphs. The joint task then becomes an enriched parsing task for which the same dependency parser can be used. Processing the corpora of languages for which this overlap does not hold would require further adaptation of the parser. In this section we first describe the architecture of the system in Subsection 4.1, and then describe how we set up a comparative experiment in which the same system is applied to the two tasks of syntactic dependency parsing and semantic role labeling in isolation, and in combination, in Subsection 4.2.

4.1 Constraint satisfaction inference for dependency structures

Our dependency parser is an adaptation for dependency structures of the constraint satisfaction inference method for sequential output structures proposed by Canisius et al. (2006a). The technique uses standard classifiers to predict a weighted constraint satisfaction problem, the solution to which is a complete dependency structure. The predicted constraints each cover a small part of the complete structure. Overlap between them ensures that global output structure is taken into account, even though the classifiers only make local predictions in isolation of each other.

A weighted constraint satisfaction problem (W-CSP) is a tuple (X, D, C, W). Here, $X = \{x_1, x_2, \ldots, x_n\}$ is a finite set of variables. $D(x)$ is a function that maps each variable to its domain, and C is a set of constraints on the values assigned to the variables. For a traditional (non-weighted) constraint satisfaction problem, a valid solution is an assignment of values to the variables that (1) are a member of the corresponding variable's domain, and (2) satisfy *all* constraints in the set C. Weighted constraint satisfaction, however, relaxes this requirement. Instead, constraints are assigned weights that may be interpreted as reflecting the importance of satisfying that constraint. The optimal solution to a W-CSP is the solution that assigns those values that maximize the sum of the weights of satisfied constraints.

To adapt this framework to predicting a dependency structure for a sentence, we construct a constraint satisfaction problem by first introducing one variable x_i for each token of the sentence. This variable's value corresponds to the dependency relation that token is the modifier of, i.e. it should specify a relation type and a head token. The constraints of the CSP are predicted by a

3. The fact that the semantic labels are composed of semantic argument types and thematic roles indicates that at the outset of our study we are already adopting a merged semantic label set. In Part-of-Speech tagging it is also typical to merge different types of morpho-syntactic information in single tags.

classifier, where the weight for a constraint corresponds to the classifier's confidence estimate for the prediction.

For the current study, we trained three classifiers to predict three different types of constraints:

1. $C_{dep}(head, modifier, relation)$, signifying a dependency between *head* and $modifier$. The resulting dependency structure should have a dependency arc from *head* to $modifier$ labeled with type *relation*. For the example structure in Figure 1, among others the constraint $C_{dep}(head = iniciat, modifier = treballadors, relation = suj)$ should be predicted.

2. $C_{dir}(modifier, direction)$, the relative position of the head of $modifier$. The structure in Figure 1 will give rise to constraints such as $C_{dir}(modifier = matinada, direction = LEFT)$.

3. $C_{mod}(head, relation)$, representing a modifying relation of *head*. In the dependency structure, *head* should be modified by a relation of type *relation*. The constraints generated for the word *vaga* in Figure 1 would be $C_{mod}(head = vaga, relation = spec)$, $C_{mod}(head = vaga, relation = sp)$, and $C_{mod}(head = vaga, relation = s.a)$.

Predicting constraints of type C_{dep} is essentially what is done by Canisius et al. (2006b); a classifier is trained to predict a relation label, or a symbol signaling the absence of a relation, for each pair of tokens in a sentence[4]. The training data for this classifier consists of positive examples of constraints to generate, e.g. $iniciat, treballadors, suj$, and negative examples of constraints *not* to generate, using "NONE" as the class label for these cases, e.g. $iniciat, Els, NONE$ or $treballadors, iniciat, NONE$. In the aforementioned paper, it is shown that down-sampling the negative class in the classifier's training data improves the recall for predicted constraints. The fact that improved recall comes at the cost of a reduced precision is compensated for by our choice for the weighted constraint satisfaction framework: an overpredicted constraint may still be left unsatisfied if other conflicting constraints outweigh its weight.

In addition to giving rise to a set of constraints, this classifier differs from the other two in the sense that it is also used to predict the domains of the variables, i.e. any dependency relation not predicted by this classifier will not be considered for inclusion in the output structure. In other words, the C_{dep} classifier could work in isolation and produce full output structures. The other two classifiers, C_{dir} and C_{mod}, add constraints that can potentially outweigh and correct constraints predicted by the C_{dep} classifier.

While the C_{dep} classifier classifies instances for each pair of words, the classifiers for C_{dir} and C_{mod} only classify individual tokens. The two classifiers differ in the classes they predict. For C_{dir}, there are three possible classes: `LEFT`, `RIGHT`, `ROOT`. For C_{mod} there is a rather large class space; a class label reflects all modifying relations for the token simultaneously, e.g. `cd+suj`. Multiple labels for a single token are uniquely and alphabetically sorted. From this label, as many constraints are generated as there are different relation types in the label.

We opt for a feature space that is both relatively straightforward and similar across classification tasks. For all tasks we draw on the lexical identity and the morpho-syntactic tags of the focus words and their immediately neighboring words. The following list describes the features for the pairwise C_{dep} task on the one hand, and for the token-by-token C_{dir} and C_{mod} tasks on the other hand:

- For C_{dep}, given a pairwise instance focusing on a pair of tokens, the feature space contains
  - the identity of both focus tokens and their immediate left and right neighboring tokens (i.e. six tokens in total);
  - the main Part-of-Speech tags of these six tokens;

4. To avoid having too many negative instances in the training data, thus to limit the majority effect of the "no relation" class and to boost the recall of other relational class labels, we follow the approach of Canisius et al. (2006b) of limiting the maximum distance between a potential head and modifier; we adopt the value of 20 tokens as the distance limit.

  - the bigram of the main Part-of-Speech tags of the two focus tokens;
  - the bigram of the morphosyntactic subtags of the two focus tokens;
  - whether the second token is positioned to the right or to the left of the first;
  - the absolute distance in tokens between the positions of the two tokens.

- for C_{dir} and C_{mod}, given a focus token, the feature space contains
  - the identity of the focus token and its immediate two left and right neighboring words (i.e. five tokens in total);
  - the main Part-of-Speech tags of these five tokens;
  - bigrams of the tokens and the main Part-of-Speech tags of the five tokens;
  - bigrams of the main Part-of-Speech tags of the focus token and its left and right neighbor tokens;
  - the morphosyntactic subtags of the focus token and its left and right neighbor tokens.

Each task is performed by a memory-based classifier based on the k-nearest neighbor classification rule[5], running a feature-weighted version of the IB1 algorithm (Aha et al. 1991), where the weighting and other hyperparameters are estimated for the dependency parsing task on heldout material through wrapped progressive sampling, an automated heuristic hyperparameter method (Van den Bosch 2004). These settings are kept constant throughout all further experiments. In detail, they include the use of the MVDM distance metric (Cost and Salzberg 1993), gain-ratio feature weighting, linearly-inverse distance weighting, and $k = 9$. The output of each classification is a weighted distribution of classes occurring in the nearest neighbor set, where each class receives a score that is the sum of inverse distances of the nearest neighbors labeled with that class.

When a single sentence has been converted to instances of the three tasks, and when these instances have all been classified with their respective memory-based classifiers, each providing weighted distributions of output labels, a weighted constraint satisfaction problem can be formulated that, when solved, describes a dependency structure. As we formulated our problem as a constraint satisfaction problem, any off-the-shelf W-CSP solver could be used to obtain the best dependency parse. However, in general such solvers have a time complexity exponential in the number of variables, and thus in the length of the sentence. As a more efficient alternative we chose to use the CKY algorithm for dependency parsing (Eisner 2000) for computing the best solution, which has only cubic time complexity, but comes with the limitation of only considering projective trees as candidate solutions—but as noted earlier, all dependency trees in the AnCora treebank are projective.

4.2 A single architecture for two isolated and one joint task

As the goal of this study is to assess the efficacy of the joint learning of dependency parsing and semantic role labeling, the means is to design a comparative experiment between the disjoint and the joint learning condition. In order to keep the comparison as tight as possible, so that differences in performance must derive from the task and not from another factor, we use the same overall architecture for all three tasks, namely the dependency parser architecture described in the previous subsection, based on three local memory-based classifiers and a weighted constraint satisfaction inference process. It is straightforward to see the joint task as an enriched type of dependency parsing, and thus as comparable to the disjoint dependency parsing task. It is less straightforward, though, to see semantic role labeling as a dependency parsing task, as its output is not a single dependency graph per sentence, but rather one or more predicate-centered subgraphs per sentence. We solve this by defining the semantic role labeling task as a full dependency parsing task at the

5. Available from `http://ilk.uvt.nl/timbl`

sentence level, where all dependencies, the semantic as well as the non-semantic ones, need to be established; however, all non-semantic relations receive a dummy label. Thus, the output of the disjoint task of semantic role labeling is a full dependency graph per sentence, where only the semantically labeled relations should receive a non-dummy semantic role label.

We set up a matrix of experiments. For both languages, Catalan and Spanish, we develop three systems: a disjoint dependency parser, a disjoint semantic role labeler, and a joint dependency parser and semantic role labeler. All 2×3 systems are based on three local memory-based classifiers, $C_{dep}, C_{dir}, C_{mod}$. The only difference among the two disjoint and the joint system can be found in the class labels. Taking the example sentence of Figure 1 as a reference, we exemplify these differences by focusing on the subject and the main verb tokens, *treballadors* and *iniciat*:

- For the joint C_{dep} classifier, the relation between modifier *treballadors* and head *iniciat* is *suj:arg0-agt*. For the disjoint dependency parser, this relation is *suj*; for the semantic role labeler, this relation is *arg0-agt*. As mentioned already in Section 2, the semantic part of the joint label is often empty, as many dependency relations (e.g. between non-verbal tokens) only have a syntactic type.

- For the C_{dir} classifier, which operates token by token, the task is the same for all three systems. The head of *treballadors* is to the right of the token, regardless of the type of relation, hence the class label is *RIGHT* in all three cases.

- For the C_{mod} classifier, also operating token by token, the joint classifier should generate for the word *iniciat* a compound class label of joint labels representing all syntactic and semantic relations of which *iniciat* is the head, i.e. *cc:argm-tmp*|*cd:arg1-pat*|*f:_*|*suj:arg0-agt*|*v:_*, where | is the delimiter symbol between joint labels. The disjoint syntactic classifier should generate a compound label of syntactic relations, i.e. *cc*|*cd*|*f*|*suj*|*v*, while the disjoint semantic classifier should generate *dummy:_*|*dummy:arg0-agt*|*dummy:arg1-pat*|*dummy:argm-tmp*. Recall that syntactic relations should be labeled with a dummy value by the semantic role labeler, and that the compound label uniquely sorts all occurring relations of a head token. As *iniciat* is the head of two non-semantic relations they are uniquely sorted under the single *dummy:_* part of the compound label.

Each experiment results in a trained system which is subsequently applied to unseen test data. We adopted the particular splits into training and test data made by the CoNLL-2009 shared task organizers (Hajič et al. 2009). To evaluate the performance of our system, we used the CoNLL-2009 shared task evaluation scripts[6] to enable the comparison of joint versus disjoint learning. Generalization performance on the joint task is measured in terms of macro precision and recall by averaging the labeled precision and recall for semantic dependencies with the Labelled Attachment Score (LAS) for syntactic dependencies. The macro labeled F1 is computed as the harmonic mean of the labeled macro precision and the labeled macro recall. Dependency parsing is evaluated in terms of labeled attachment score, which is a token-based score that measures the relative amount of tokens that is assigned both a correct head and the correct dependency relation with that head, of unlabeled attachment score (where the correctness of the dependency relation is ignored), and of label accuracy (where the correctness of the head is ignored). Semantic role labeling is evaluated in terms of the precision, recall, and F-score ($\beta = 1$) of correctly identified and labeled semantic roles.

Besides reporting the "joint" score provided by the evaluation software of the joint system, we also report the separate evaluation of dependency parsing and semantic role labeling. Thus, in the next section we present separate comparisons of the scores on dependency parsing of the joint system versus the disjoint dependency parser, and the scores on semantic role labeling of the joint system versus the disjoint semantic role labeler.

6. Available from `http://ufal.mff.cuni.cz/conll2009-st/scorer.html`

To gain more insight into the relative effect of the size of the training set, which for both languages is reasonably large (about 400 thousand tokens), we performed learning curve experiments of each of the three classifiers in each of the six systems, so that we are able to measure system performance at increasing training set sizes. Counting in training sentences, we ran pseudo-exponential learning curves with $100, 200, 500, 1,000, 2,000, 5,000$, and $10,000$ sentences, and ended the learning curves with the maximum available $13,200$ (Catalan) and $14,329$ (Spanish) sentences.

5. Results

We start by comparing the generalization performance on dependency parsing of the joint versus disjoint systems for both languages as shown in Table 1.

	Catalan			Spanish		
System	LAS	UAS	LA	LAS	UAS	LA
Joint	82.14	85.12	90.64	80.82	83.83	89.80
Disjoint	**82.63**	**85.65**	**91.04**	**81.50**	**84.51**	**90.38**

Table 1: Dependency parsing generalization performance in terms of labeled attachment score (LAS), unlabeled attachment score (UAS), and label accuracy (LA), by the joint (syntactic and semantic) systems versus the disjoint dependency parsing systems.

Table 1 displays the various scores on the maximal training set size. The results show that in terms of labeled attachment score, the disjoint systems for Catalan and Spanish dependency parsing outperform their joint learning counterparts with a difference of about half a percentage point (a relative error reduction of 2.7% for Catalan, and 3.5% for Spanish). The results on unlabeled attachment score and label accuracy follow the same trend.

	Catalan			Spanish		
System	LPrec	LRec	LF1	LPrec	LRec	LF1
Joint	78.09	**73.22**	**75.58**	**77.01**	**72.25**	**74.55**
Disjoint	**78.35**	71.88	74.98	76.37	69.42	72.73

Table 2: Semantic role labeling generalization performance in terms of labeled precision (LPrec), labeled recall (LRec), and labeled F-score with $\beta = 1$ (LF1), by the joint (syntactic and semantic) systems versus the disjoint semantic role labeling systems.

Table 2 lists the results on semantic role labeling for disjoint versus joint learning for both languages. Apart from the fact that the disjoint Catalan system attains a higher precision, in terms of labeled F-score the joint learning systems for Catalan and Spanish perform better than their counterpart disjoint systems. For Catalan, the difference is about half a percentage point (or 2.4% error reduction) in labeled F-score; for Spanish, the advantage of joint learning is larger: 1.8 percentage points, or 6.7% error reduction.

Figure 2 compares the learning curves of the joint and the disjoint systems for Catalan dependency parsing, in terms of labeled attachment score; the curves are the top two lines in the figure. The results show a positive correlation between the amount of training data (the logarithmic x-axis) and generalization performance. The relation is not entirely log-linear; the performance increases tend to decrease slightly with more data. Figure 2 also shows the learning curves of the joint and disjoint systems for semantic role labeling (the bottom two lines), indicating that the generalization performances of the joint and disjoint systems follow the same upward trend as those of the dependency parsing systems, except for the fact that the curves have swapped positions.

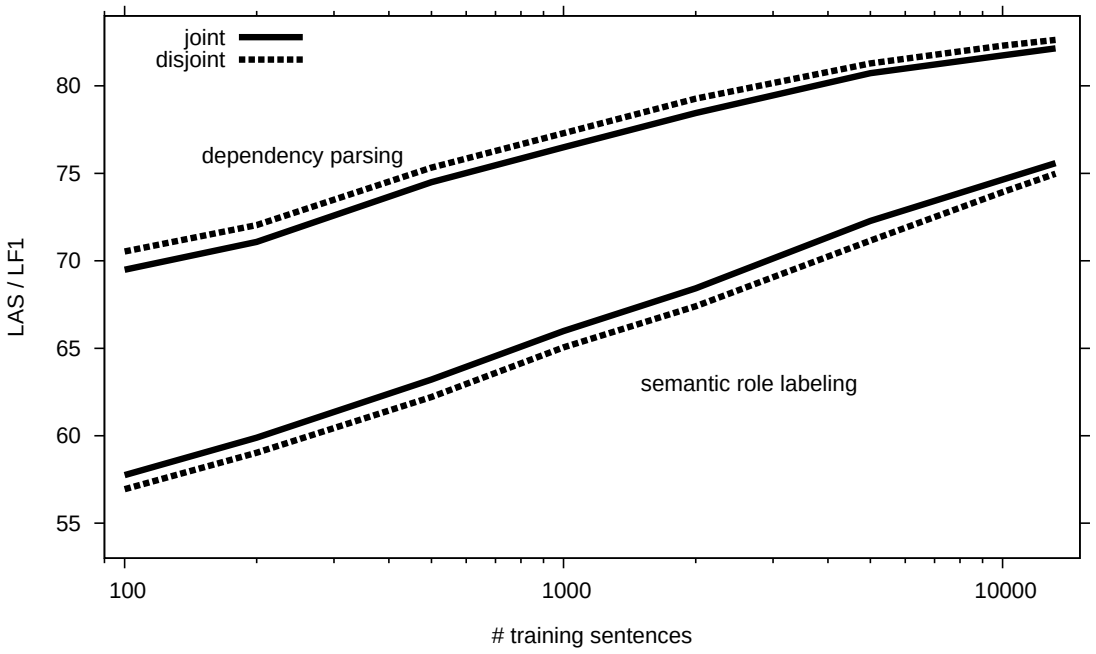

Figure 2: Learning curves of dependency parsing in terms of labeled attachment score, and of semantic role labeling in terms of labeled F-score, of the joint systems (continuous line) versus the disjoint systems (dashed line) for Catalan.

As the experiments were performed on data used in the CoNLL-2009 shared task, we compare the performance of our joint learning system against the performances of the overall best performing systems (Dai et al. 2009, Gesmundo et al. 2009, Che et al. 2009), as well as with an alternative memory-based joint system (Morante et al. 2009), in Table 3. None of the three best systems performs joint learning in the sense that we use it in this study, with a joint task with unified labels. Dai et al. (2009) perform the individual tasks disjointly, and iterate each task with input from other tasks in the previous iteration. Gesmundo et al. (2009) present a system that initially generates separate derivations for semantic and syntactic dependencies and then synchronizes these derivations for each word based on a generative latent variable model of joint syntactic–semantic dependency parsing. Che et al. (2009) implement a relatively straightforward pipeline of a dependency parser, a predicate sense classifier, and a semantic role labeler. Although the main point of this contribution is to present an all-else-being-equal comparison between joint and disjoint learning of semantic role labeling and dependency parsing, it is clear that our system is outperformed by the best three CoNLL-2009 shared task systems: joint learning does not seem to be a winning strategy.

	Catalan			Spanish		
System	LAS	LF1	LmF1	LAS	LF1	LmF1
Dai et al. (2009)	85.88	80.10	83.01	86.29	80.29	83.31
Gesmundo et al. (2009)	87.86	77.44	82.66	87.64	77.19	82.43
Che et al. (2009)	86.56	77.10	81.84	87.33	76.47	81.90
Joint	82.14	75.58	78.88	80.82	74.55	77.71
Morante et al. (2009)	77.33	70.14	73.75	73.07	68.48	70.78

Table 3: Scores on dependency parsing (LAS), semantic role labeling (LF1), and the joint task (LmF1, for LmacroF1) for the overall three best-performing systems in the CoNLL-2009 shared task, compared to our joint system and an alternative memory-based system.

Morante et al. (2009) is a similar system to the one presented here; it does not make use of constraint satisfaction inference, but rather builds on the output of a joint C_{dep} classifier, an unlabeled variant of the C_{dep} classifier, and a variant of the C_{mod} classifier, and adopts local re-ranking heuristics to select an optimal joint graph. The improvement of the current joint system over the one of Morante et al. (2009) indicates the relative effectiveness of the constraint satisfaction inference method.

6. Analysis of results

In this section we report on a closer analysis of the results of the systems for Catalan dependency parsing and semantic role labeling, trained on the maximal amount of data. Our goal is to explain what is the major cause for the differences in performance between the joint and disjoint systems.

6.1 Joint versus disjoint dependency parsing

As shown in Section 5, disjoint learning of syntactic dependency parsing outperforms joint learning. An intuitive explanation for this is that the disjoint dependency parsing task discerns 50 different relation types, while the joint task combines these 50 types with semantic relation types into a total of 130 relation types. Learning a task with more relations and consequently less examples per relation should be harder. Here we make an attempt to analyze in more detail where this difference leads to the largest differences.

The first place to seek differences between the two approaches is the three memory-based classifiers, C_{dep}, C_{dir}, and C_{mod}. Table 4 displays the classifier accuracy scores, i.e. the percentage of correctly classified test instances, of the three classifiers in the joint versus disjoint condition. The disjoint classifiers are clearly more accurate than their joint learning counterparts. This difference must lie at the basis of the better performance of the disjoint system in terms of labeled attachment score.

System	C_{dep}	C_{dir}	C_{mod}
Joint	89.33	92.27	81.90
Disjoint	95.74	94.12	85.77

Table 4: Classifier accuracy scores on test data of the C_{dep}, C_{dir}, and C_{mod} classifiers trained on joint and disjoint Catalan dependency parsing.

Entering a finer level of analysis, we analyze the F1 scores of the systems per dependency relation. Table 5 contains the F1 scores for dependency relations with more than two hundred occurrences

in the test corpus. For most relations the scores of the disjoint system are slightly higher (< 1.0), and for three relations the scores are markedly higher: *cc* (+4.06), *cpred* (+3, 89) and *sadv* (+4.36). Only in one case, *conj*, the joint system outperforms the disjoint system.

DEPREL	#	F1 J	F1 D	F1 D-J
sn	8828	92.82	92.81	-0.01
spec	7911	98.94	98.96	0.02
f	6291	64.94	65.06	0.12
sp	4843	78.55	80.33	1.78
suj	3834	86.08	86.81	0.73
cc	3083	61.08	65.14	4.06
cd	2536	78.09	78.10	0.01
S	2484	66.02	65.95	-0.07
s.a	2291	96.16	96.18	0.02
v	1872	96.75	97.11	0.36
ROOT	1862	76.84	77.71	0.87
coord	1575	62.26	62.58	0.32
conj	788	90.70	89.75	-0.95
creg	703	67.79	68.48	0.69
atr	534	82.14	83.64	1.50
grup.nom	489	72.30	72.28	-0.02
mod	430	88.43	88.70	0.27
pass	375	93.26	93.26	0.00
morf.pron	355	96.00	96.40	0.40
d	293	96.20	96.20	0.00
ao	270	33.91	35.21	1.30
ci	239	53.21	53.79	0.58
sadv	206	62.47	66.83	4.36
cpred	205	66.67	70.56	3.89

Table 5: Number of occurrences (#) per dependency relation (DEPREL), F1 of the joint (J) and disjoint (D) dependency parsers, and difference in F1 (D-J) for classes with more than 200 occurrences, measured on test data.

The score with the highest impact in the difference in performance of the joint and disjoint systems is that of the relatively frequent *cc* dependency relation, which we analyze in detail. *cc* stands for adverbial adjunct. An adjunct is an optional complement of the predicate that usually expresses the circumstances under which the event expressed by the predicate takes place. In 1 we show two examples of *cc* that perform a temporal role (*cc:argm-tmp*), and one example that performs a location role (*cc:argm-loc*). They express the circumstances associated with the event *faran*.

Example 1 *Les conferències es faran [tots els divendres* $_{cc:argm-tmp}$*], [entre les set i les deu* $_{cc:argm-tmp}$*], [també a la Torre LLuvià* $_{cc:argm-loc}$*]. English: 'Conferences will take place on Friday, between seven and ten, also in Torre LLuvià' .*

One of the characteristics of the dependency relation *cc* is that semantically it combines with a wide variety of semantic roles. Additionally, it can occupy different positions as adjunct of the verb. This means that while in the disjoint setting the system has to learn one class, in the joint setting the system has to learn as many as 20 classes[7], with markedly fewer examples per class. As listed

7. Classes in which the dependency relation *cc* is involved in the joint setting: cc:_, cc:arg1-null, cc:arg2-atr, cc:arg2-fin, cc:arg2-loc, cc:arg2-null, cc:arg3-ein, cc:arg3-loc, cc:arg3-ori, cc:arg4-des, cc:arg4-efi, cc:argL-null, cc:argm-adv, cc:argm-cau, cc:argm-ext, cc:argm-fin, cc:argm-ins, cc:argm-loc, cc:argm-mnr, cc:argm-tmp.

in more detail in Table 6, the balance between precision and recall scores on *cc* classes is different between the two systems. The joint system is more precise (+5.57), while the disjoint system has a better coverage (+10.64). It might be the case that for some applications a system that produces less false positives is more adequate than a system that produces more. Overall, these results confirm the intuitive hypothesis that a disjoint setting outperforms the joint setting, especially where the class space becomes very fragmented in the joint setting. The disjoint system generalizes better for the *cc* class, though it also produces more false positives.

System	# *cc* classes	TP	FP	Precision	Recall
Joint	20	1625	613	72.61	52.71
Disjoint	1	1953	960	67.04	63.35

Table 6: Total number of joint classes with the *cc* dependency relation, true positives (TP), false positives (FP), precision and recall of the joint and disjoint dependency parsers.

In order to further test the hypothesis, we analyze the results of other dependency relations that have a relatively fragmented class space in the joint setting. These are: *cd* (direct object) with six labels; *ci* (indirect object) with three; *cpred* (predicative complement) with four; *creg* (prepositional complement) with ten; and *suj* (subject) with seven. For all these dependency relations the disjoint system scores better. For the *cpred* relation the advantage of the disjoint system is almost as high (+3.89) as for *cc*, despite the fact that the class space for *cpred* is less fragmented in the joint setting (4 versus 20 classes). For the *cd* relation the advantage of the disjoint system is negligible, despite the fact that the class space in the joint setting is fragmented into six classes. This would suggest that fragmentation, though important, is not the only factor that plays a role in the different performance of the systems.

In order to find other potential factors we analyze the differences in the predictions of the joint and disjoint systems for *cpred* and *cd*. As for *cpred*, a translated typical example sentence for this relation is 2, where *tired* is the argument of the predicate *arrived*.

Example 2 *The president arrived [tired* $_{cpred:arg2-atr}$*].*

Although the disjoint system outperforms the joint system (+3.89 in terms of F1), the actual difference between the joint and disjoint systems for this relation is that the joint system has six more false negatives and six more false positives than the disjoint system; the high increase in F1 is based only on a small number of token-level differences. Overall, both systems make similar errors on *cpred*. The most common ones are assigning *sp* (prepositional phrase) and *atr* (attribute) instead of *cpred*. *Sp* is assigned when there are attachment errors as well, such as in 3, where the systems attach the prepositional phrase *de "perfecte example"* to *segrest* instead of to the predicate *qualificava*.

Example 3 *La nota qualificava el segrest [de "perfecte exemple"* $_{cpred:arg2-atr}$*]. English: 'The note qualified the kidnapping as a perfect example'.*

Atr is assigned frequently instead of *cpred*, which is to be expected, since these dependency relations participate in constructions that are similar. The difference is that *atr* is assigned to complements of the verbs *ser* and *estar* ('to be'). A translated typical example sentence for *atr* is 4, where *tired* is the argument of the predicate *is*.

Example 4 *The president is [tired* $_{atr:arg2-atr}$*].*

The dependency relation *cd* denotes the direct object. A typical example is shown in 5, where *his speech* is the *cd* of the predicate *reads*.

Example 5 *The president reads [his speech* $_{cd:arg1-tem}$*].*

There can only be one *cd* relation per predicate. In the joint system it splits into six classes with different distributions: *cd:arg1-ext* (5), *cd:arg1-pat* (1,998), *cd:arg1-tem* (250), *cd:arg2-atr* (223), *cd:arg2-ext* (27), and *cd:argl-null* (33). Despite the fragmentation, the scores for the joint and disjoint system are similar. An explanation for this is that of the five *cd* classes in the joint system one of them (*cd:arg1-pat*) is much more frequent, which might boost the global scores of *cd* for the joint system. Of the 2,536 occurrences of *cd*, 2,025 are predicted correctly by both systems, 80 by the disjoint system and 85 by the joint system. Of the correct 2,025 cases generated by both systems, 1,670 have the role *arg1-pat*, which accounts for 83.6% of the occurrences of *cd:arg1-pat*. The fact that both systems generalize well for this class would suggest that examples of the class are consistent, which compensates for the fragmentation in the class space. The most frequent common error that both systems commit is assigning *suj* instead of *cd*, which happens in 45.1% of the cases.

For example, in 6 both systems assign *suj* to *els seus estudis musicals* and attach it to *vulguin* instead of assigning *cd* to it and attaching it to *acabar*.

Example 6 *Els alumnes que vulguin acabar [els seus estudis musicals* $_{cd:arg1-pat}$*] a Reus o a Tortosa [...]. Engish: 'The students who would like to finish their music studies in Reus or Tortosa'.*

Similarly, in 7 both systems attach complement *veritats científiques* to *contradir*, but assign to it *suj* instead of *cd*.

Example 7 *[...] els interessos polítics no poden contradir [veritats científiques* $_{cd:arg1-pat}$*] [...]. English: '[...] political interests cannot contradict scientific certainties [...]'.*

Until now we have explored the hypothesis that the fragmentation in the class space of the joint system might be the cause of its lower performance. However, not all dependency relations that get a lower performance in the joint system are fragmented. For example, as shown in Table 5, *sadv* scores 4.36 higher in the disjoint system, whereas the class space is not fragmented, since in both systems there is only one class.

System	# *sadv* classes	TP	FP	Precision	Recall
Joint	1	134	89	60.09	65.05
Disjoint	1	134	61	68.72	65.05

Table 7: Total number of joint classes with *sadv* as dependency relation, true positives (TP), false positives (FP), precision and recall of the joint and disjoint dependency parsers.

Table 7 shows that the cause of the disadvantage for the joint system is a lower precision. In both systems, the false positives are mostly caused by assigning *sadv* instead of *cc*, but this error is more frequent in the joint system. So, even though the *sadv* class is not fragmented, the scores of this class are partly linked to the scores of the *cc* class, which is very fragmented; it seems that the relative higher frequency of *sadv* versus the fragmented *cc* classes in the joint system causes the latter system to overpredict the confusable *sadv* class.

Another category that deserves attention is *conj*, since it is the only category on which the joint system fares better. The cause for the slightly better performance of the joint system for this category is better attachment. Both systems predict the label at the same performance level if attachment is not taken into account, but the results favor the disjoint system when attachment is taken into account, as shown in Table 8.

	Label+Attachment		Label	
System	Precision	Recall	Precision	Recall
Joint	91.05	90.36	98.59	97.84
Disjoint	90.04	89.47	98.60	97.97

Table 8: Precision and recall scores for the *conj* dependency relation of the joint and disjoint dependency parsers with and without evaluating attachment.

6.2 Joint versus disjoint semantic role labeling

Whereas the dependency parsing results show that the disjoint setting is more favorable than the joint setting, the semantic role labeling results show that this task in fact profits from the joint setting. In order to find out why this happens, we analyze the results in more detail.

Contrary to what happens with the dependency parser, a comparison of the performance of the three memory-based classifiers, C_{dep}, C_{dir}, and C_{mod} does not explain the advantage of the joint approach in the semantic role labeling task. Table 9 displays the classifier accuracy scores of the three classifiers in the joint versus disjoint condition. As in the dependency parsing experiments, the disjoint classifiers are more accurate than their joint learning counterparts. Yet, the joint setting produces a better performance of the final semantic role labeling system.

System	C_{dep}	C_{dir}	C_{mod}
Joint	89.33	92.27	81.90
Disjoint	94.26	94.12	91.69

Table 9: Classifier accuracy scores on test data of the C_{dep}, C_{dir}, and C_{mod} classifiers trained on joint and disjoint Catalan semantic role labeling.

An alternative explanation can be explored by comparing the F1 scores of the disjoint and joint semantic role labelers. Table 10 lists the F1 scores for semantic roles with more than two hundred occurrences in the test corpus. The results per semantic role show that the joint system, while overall better, does not outperform the disjoint system in all classes. The frequent semantic roles with the most marked advantage for the joint system are *v+argm-tmp*, *v+argm-loc*, *v+argm-adv*, and *v+argm-fin*. In what follows we analyze the scores per semantic role from the perspective of class fragmentation in order to check whether fragmentation plays a role as it did in dependency parsing.

In Section 6.1 we showed that some dependency relations were considerably fragmented in the joint setting. However, this does not apply to semantic roles in the joint setting. Whereas a syntactic function can be associated with several semantic roles, a semantic role is mostly associated to one syntactic function. Of the 32 semantic roles, only 11 are fragmented, and from these only three are fragmented in more than two classes: *arg2-atr* splits into *cc:arg2-atr, cpred:arg2-atr, cd:arg2-atr*, and *atr:arg2-atr*; *arg2-loc* splits into *cc:arg2-loc* and *creg:arg2-loc*; and *argL-null* splits into *suj:argL-null, cpred:argL-null, creg:argL-null, cc:argL-null*, and *cd:argL-null*. The joint system scores higher for the three, despite the fact that the classes to be learned are more fragmented for this system. So, fragmentation does not hinder the performance of the joint system.

Another aspect to explore is whether the better perfomance of the joint system on particular classes is due to a better precision or to a better recall. The global results presented in Table 2 indicate that the joint system attains a relatively higher recall. Table 11 presents the precision and recall measures for the aforementioned four semantic roles. For the role *v+argM-fin* the advantage is equally spread over precision and recall, while for the other three roles the advantage is somewhat more pronounced in recall.

POS pred + sem role	#	F1 J	F1 D	F1 J-D
v + arg1-pat	2336	73.27	72.72	0.55
v + arg0-agt	1845	76.47	75.53	0.94
v + arg1-tem	1568	68.93	69.95	-1.02
v + argM-tmp	868	64.02	60.46	3.56
v + arg2-atr	841	73.86	73.75	0.11
v + argM-loc	616	42.83	36.35	6.48
v + argM-adv	580	29.03	23.69	5.34
v + arg2-null	322	58.74	60.10	-1.36
v + arg2-ben	203	50.27	50.00	0.27
v + arg1-null	191	62.19	62.99	-0.80
v + argM-cau	170	11.06	8.90	2.16
v + argM-fin	161	42.86	19.13	23.73
v + argM-mnr	143	50.22	44.96	5.26
v + arg2-loc	139	33.98	39.64	-5.66
a + arg0-agt	122	78.76	73.13	5.63
v + arg0-cau	100	35.29	40.85	-5.56

Table 10: Number of occurrences (#) per semantic role , F1 of the joint (J) and disjoint (D) semantic role labeler, and difference in F1 (J-D) for classes with more than 100 occurrences on test data.

Semantic role	Prec J	Prec D	J-D	Rec J	Rec D	J-D
v+argm-fin	54.29	31.88	22.41	35.40	13.66	21.74
v+argm-loc	45.75	41.54	4.21	40.26	32.31	7.95
v+argm-tmp	68.01	69.57	-1.56	60.48	53.46	7.02
v+argm-adv	37.64	34.90	2.74	23.62	17.93	5.69

Table 11: Precision (Prec) and recall (Rec) of four semantic roles in the joint (J) and disjoint (D) systems.

An error analysis per semantic role does not yield more information about why the recall of the joint system is slightly higher than the recall in the disjoint system. The types of errors made by the two systems are similar. The most common errors for the *v+argM-fin* semantic role are confusing it with the $NONE$ class or with the *v+argM-adv* role. The *v+argM-tmp* role gets confused mostly with *v+arg1-pat*, *v+arg2-atr*, *v+argM-loc*, and *v+arg1-tem*. For the *v+argM-loc* and the *v+argM-adv* roles the errors are more diverse.

A lack of a clear explanation leads us to explore a factor that is more technical in nature that might explain the difference in recall scores: the F scores of the $NONE$ and *dummy:_* class of the C_{dep} classifier. Recall that $NONE$ is the label associated with pairs that have no semantic or syntactic relation whatsoever, and *dummy:_* is the label for pairs that do have a syntactic relation, but no semantic relation. We hypothesize that the disjoint C_{dep} semantic dependency classifier will tend to overpredict these classes more often, while underpredicting and thus yielding a lower recall for the other classes. The reason why the disjoint semantic dependency classifier would overpredict them is that without the syntactic dependencies, the remaining semantic dependency subgraphs are sparser. In other words, there are relatively more pairwise examples of words that have no relation in the disjoint C_{dep} semantic dependency classifier than in the case of the same joint classifier, where many pairs will have at least a syntactic dependency. In the disjoint case, this means that the

majority class representing no relation will be more dominant, causing the classifier to defer to the majority more often.

This intuition is confirmed by the results. The test set of the C_{dep} disjoint classifier contains 40,847 instances with class *dummy:_*, 1,395,547 with class $NONE$, and 11,163 instances with a semantic role. This means that in the C_{dep} joint classifier these instances have a combined label where the left part is a syntactic dependency relation. Semantic roles in the disjoint system are often accidentally misclassified for *dummy:_*, causing their recall to be lower.

For example, for the sentence presented in Figure 1 the instance that represents the relation between *Els* and *treballadors* has class *spec:_* in the joint setting, and class *dummy:_* in the disjoint setting; the instance that represents the relation between *vaga* and *dels* will have class NONE both in the joint and disjoint setting.

Table 12 shows the numbers of true positives, false positives and false negatives for the *dummy:_* and $NONE$ classes produced by the C_{dep} classifier of the joint and disjoint systems. The joint classifier produces a higher number of true positives and a lower number of false positives and false negatives, confirming the hypothesis that the scores of these two classes determine the higher recall of the joint system.

Relation	TP J	TP D	FP J	FP D	FN J	FN D
dummy:_	39128	38833	1719	2014	44333	63730
NONE	1338477	1318000	57070	77547	1880	2346

Table 12: True positives (TP), false positives (FP), and false negatives (FN) of the *dummy:_* and $NONE$ classes for the joint (J) and disjoint (D) C_{dep} classifier.

Overall, our analysis suggests that on semantic roles the recall of the joint system is higher due to the distributions of the *dummy:_* and NONE classes in the instances of the C_{dep} classifier.

7. Conclusions

In this case study with two languages, Catalan and Spanish, and two complex natural language processing tasks, syntactic dependency parsing and semantic role labeling, we have shown that the joint learning of these two tasks as one single task can yield systems that are competitive in generalization performance, compared to their counterpart systems in which the tasks are learned and performed separately (all else being equal). We observe two opposed differences: joint learning performs slightly worse than disjoint learning in dependency parsing (about a half percentage point in terms of labeled attachment), while at the same time joint learning performs slightly better than disjoint learning on semantic role labeling (again, about a half percentage point in terms of labeled F-score).

We relate the disadvantage of the joint system on dependency parsing to the relatively high class fragmentation that joint learning causes. This effect is absent in the semantic role labeling task; here, the joint system attains a higher overall performance mostly because of a higher recall. The disjoint semantic role labeler, which does not jointly predict syntactic dependencies, is hurt slightly by the fact that the majority class, representing the absence of a relation, is more frequent and overbearing, rendering the disjoint semantic role labeler too conservative.

We venture to conclude that this case study is yet another illustration that complex natural language processing tasks that have an identical or overlapping input feature space as well as an overlapping output space, such as dependency graphs and semantic role labeling graphs, can be formulated as single tasks, to be learned by machine learning algorithms. It remains an empirical question whether this combination performs on par with the disjoint systems, as we explored here in the context of dependency parsing and semantic role labeling. The outcomes of our study add

to the growing number of observations that have shown relatively stable, or occasionally better performances by joint learning (Sejnowski and Rosenberg 1987, Van den Bosch 1997, Buchholz 2002, Wang et al. 2008, Finkel and Manning 2009, Kate and Mooney 2010, Li et al. 2010). Whether the joint learning approach is a match for pipeline systems, cascaded systems, or iterative versions of these systems where the output of modules is piped into the input of other modules, remains an empirical question as well as a departure point for future research, where we suggest that experiments are set up in which the experimental circumstances are kept as constant as possible, as illustrated in this study.

Linguistically speaking, our results support the Linking Theory according to which regular mappings can be established from the semantic to the syntactic level. The syntactic-semantic correspondence is reflected in the fact that both tasks can be learned jointly. Syntactic dependency analyses and semantic role analyses are but different sides of the same coin. They draw on the same linguistic elements, and largely overlap and correlate on important parts of their analyses. We express the hope that studies both in dependency parsing and semantic role labeling build forth on this insight, and henceforth treat both tasks as one. Learnability and generalization performance is only partly and mildly hampered by the merge, and when combined, the two sources of information can be available at no extra cost.

The experiments that we describe are based on corpora annotated with projective dependencies. As further research, we plan to adapt our framework to process non-projective dependencies as well, in order to account for relations in which syntactic dependencies and semantic roles do not have a direct correspondence.

Acknowledgements

This study was made possible through financial support from the University of Antwerp (GOA project BIOGRAPH), and from the Netherlands Organisation for Scientific Research (NWO Vici project "Implicit Linguistics"). The authors wish to thank Walter Daelemans and Vincent van Asch for comments and earlier joint work.

References

Aha, D. W., D. Kibler, and M. Albert (1991), Instance-based learning algorithms, *Machine Learning* **6**, pp. 37–66.

Buchholz, S. (2002), *Memory-Based Grammatical Relation Finding*, PhD thesis, University of Tilburg.

Canisius, S., A. Van den Bosch, and W. Daelemans (2006a), Constraint satisfaction inference: Non-probabilistic global inference for sequence labelling, *Proceedings of the EACL 2006 Workshop on Learning Structured Information in Natural Language Applications*, Trento, Italy, pp. 9–16.

Canisius, S. and E. Tjong Kim Sang (2007), A constraint satisfaction approach to dependency parsing, *Proc. of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, Prague, Czech Republic, pp. 1124–1128.

Canisius, S., T. Bogers, A. Van den Bosch, J. Geertzen, and E. Tjong Kim Sang (2006b), Dependency parsing by inference over high-recall dependency predictions, *Proceedings of the Tenth Conference on Computational Natural Language Learning, CoNLL-X*, New York, NY, pp. 176–180.

Carreras, X. and L. Màrquez (2004), Introduction to the conll-2004 shared task: Semantic role labe ling, *Proceedings of CoNLL-2004*, Boston, MA.

Carreras, X. and L. Màrquez (2005), Introduction to the CoNLL-2005 shared task: Semantic role labeling, *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, Association for Computational Linguistics, Ann Arbor, Michigan, pp. 152–164.

Che, W., Z. Li, Y. Li, Y. Guo, B. Qin, and T. Liu (2009), Multilingual dependency-based syntactic and semantic parsing, *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, Association for Computational Linguistics, Boulder, Colorado, pp. 49–54. http://www.aclweb.org/anthology/W09-1207.

Civit, M., M.A. Martí, and N. Bufí (2006), *Advances in Natural Language Processing (LNAI, 4139)*, Springer Verlag, Berlin, chapter Cat3LB and Cast3LB: from constituents to dependencies, pp. 141–153.

Cost, S. and S. Salzberg (1993), A weighted nearest neighbour algorithm for learning with symbolic features, *Machine Learning* **10**, pp. 57–78.

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

Dai, Q., E. Chen, and L. Shi (2009), An iterative approach for joint dependency parsing and semantic role labeling, *Proc. of the CoNLL 2009: Shared Task*, Boulder, CO, pp. 19–24.

Eisner, J. (2000), Bilexical grammars and their cubic-time parsing algorithms, *in* Bunt, H. and A. Nijholt, editors, *Advances in Probabilistic and Other Parsing Technologies*, Kluwer Academic Publishers, Norwell, MA, USA, pp. 29–62.

Finkel, J. R. and C. D. Manning (2009), Joint parsing and named entity recognition, *Proc. of Human Language Technologies: The 2009 Annual Conference of the NAACL*, ACL, Boulder, Colorado, pp. 326–334.

Finkel, J. R. and C. D. Manning (2010), Hierarchical joint learning: Improving joint parsing and named entity recognition with non-jointly labeled data, *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Uppsala, Sweden, pp. 720–728.

Gesmundo, A., J. Henderson, P. Merlo, and I. Titov (2009), A latent variable model of synchronous syntactic-semantic parsing for multiple languages, *Proc. of the CoNLL 2009: Shared Task*, Boulder, CO, pp. 37–42.

Hajič, J., M. Ciaramita, R. Johansson, D. Kawahara, M. A. Martí, L. Màrquez, A. Meyers, J. Nivre, S. Padó, J. Štěpánek, P. Straňák, M. Surdeanu, N. Xue, and Y. Zhang (2009), The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages, *Proc. of CoNLL-2009: Shared Task*, Boulder, Colorado, USA, pp. 1–18.

Hale, K. and J. Keyser (1993), On argument structure and the lexical representation of syntactic relations, *in* Hale, K. and J. Keyser, editors, *The View from Building 20*, MIT Press, Cambridge, MA, pp. 53–110.

Hale, K. and J. Keyser (2002), *Prolegomenon to a theory of argument structure*, MIT Press, Cambridge, MA.

Kate, R. J. and R. Mooney (2010), Joint entity and relation extraction using card-pyramid parsing, *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, Association for Computational Linguistics, Uppsala, Sweden, pp. 203–212.

Levin, B. and M. Rappaport (1995), *Unaccusativity*, MIT Press, Cambridge, MA.

Levin, B. and M. Rappaport (2005), *Argument realization*, Cambridge University Press, Cambridge, UK.

Li, J., G. Zhou, and H.T. Ng (2010), Joint syntactic and semantic parsing of chinese, *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Uppsala, Sweden, pp. 1108–1117.

Lluís, X., S. Bott, and Ll. Márquez (2009), A second-order joint eisner model for syntactic and semantic dependency parsing, *Proc. of the CoNLL 2009: Shared Task*, Boulder, CO, pp. 79–86.

Merlo, P. and G. Musillo (2008), Semantic parsing for high precision semantic role labelling, *Proceedings of the Twelfth Conference on Computational Natural Language Learning, CoNLL-2008*, Manchester, pp. 1–8.

Morante, R. (2008), Semantic role labeling tools trained on the Cast3LB-CoNLL-SemRol corpus, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco.

Morante, R., V. Van Asch, and A. Van den Bosch (2009), Joint memory-based learning of syntactic and semantic dependencies in multiple languages, *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL): Shared Task,*, Boulder, CO, USA, pp. 25–30.

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

Musillo, G. and P. Merlo (2006), Accurate parsing of the Proposition Bank, *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*, New York, pp. 101–104.

Nivre, J. (2006), *Inductive Dependency Parsing*, Springer.

Nivre, J, J Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret (2007), The CoNLL 2007 shared task on dependency parsing, *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*, Association for Computational Linguistics, Prague, Czech Republic, pp. 915–932. http://www.aclweb.org/anthology/D/D07/D07-1096.

Palmer, M., D. Gildea, and P. Kingsbury (2005), The proposition bank: An annotated corpus of semantic roles, *Computational Linguistics* **31** (1), pp. 71–105.

Randall, J.H. (2009), *Linking: The geometry of argument structure*, Springer, Amsterdam.

Sejnowski, T.J. and C.S. Rosenberg (1987), Parallel networks that learn to pronounce English text, *Complex Systems* **1**, pp. 145–168.

Surdeanu, M., R. Johansson, A. Meyers, L. Màrquez, and J. Nivre (2008), The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies, *Proc. of CoNLL-2008*.

Taulé, M., M. A. Martí, and M. 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.

Van den Bosch, A. (1997), *Learning to pronounce written words: A study in inductive language learning*, PhD thesis, Universiteit Maastricht, Cadier en Keer, the Netherlands.

Van den Bosch, A. (2004), Wrapped progressive sampling search for optimizing learning algorithm parameters, *in* Verbrugge, R., N. Taatgen, and L. Schomaker, editors, *Proceedings of the Sixteenth Belgian-Dutch Conference on Artificial Intelligence*, Groningen, The Netherlands, pp. 219–226.

Wang, X., J. Nie, D. Luo, and X. Wu (2008), A joint segmentation and labeling approach for Chinese lexical analysis, *Proceedings of the European Conference on Machine Learning and Knowledge Discovery*, Springer Verlag, Berlin, pp. 538–549.