A Latent Variable Model of Synchronous Parsing for Syntactic and Semantic Dependencies

James Henderson
Dept Computer Science
Univ Geneva
james.henderson@cui.unige.ch

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

Gabriele Musillo
Depts Linguistics and Computer Science
Univ Geneva
musillo@lettres.unige.ch

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

Abstract

We propose a solution to the challenge of the CoNLL 2008 shared task that uses a generative history-based latent variable model to predict the most likely derivation of a synchronous dependency parser for both syntactic and semantic dependencies. The submitted model yields 79.1% macro-average F1 performance, for the joint task, 86.9% syntactic dependencies LAS and 71.0% semantic dependencies F1. A larger model trained after the deadline achieves 80.5% macro-average F1, 87.6% syntactic dependencies LAS, and 73.1% semantic dependencies F1.

1 Introduction

Successes in syntactic tasks, such as statistical parsing and tagging, have recently paved the way to statistical learning techniques for levels of semantic representation, such as recovering the logical form of a sentence for information extraction and question-answering applications (e.g. (Wong and Mooney, 2007)) or jointly learning the syntactic structure of the sentence and the propositional argument-structure of its main predicates (Musillo and Merlo, 2006; Merlo and Musillo, 2008). In this vein, the CoNLL 2008 shared task sets the challenge of learning jointly both syntactic dependencies (extracted from the Penn Treebank (Marcus et al., 1993)) and semantic dependencies (extracted both from PropBank (Palmer et al., 2005) and NomBank (Meyers et al., 2004) under a unified representation.

We propose a solution that uses a generative history-based model to predict the most likely derivation of a synchronous dependency parser for both syntactic and semantic dependencies. Our 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 behaviour for both constituency (Titov and Henderson, 2007a) and dependency parsing (Titov and Henderson, 2007b). The ability of ISBNs to induce their features automatically enables us to extend this architecture to learning a synchronous parse of syntax and semantics without modification of the main architecture. By solving the problem with synchronous parsing, a probabilistic model is learnt which maximises the joint probability of the syntactic and semantic dependencies and thereby guarantees that the output structure is globally coherent, while at the same time building the two structures separately. This extension of the ISBN architecture is therefore applicable to other problems where two independent, but related, levels of representation are being learnt, such as statistical machine translation.

Currently the largest model we have trained achieves 80.5% macro-average F1 performance for the joint task, 87.6% syntactic dependencies LAS, and 73.1% semantic dependencies F1.

2 The Probability Model

Our probability model is a joint generative model of syntactic and semantic dependencies. The two dependency structures are specified as the sequence of actions for a synchronous parser, which requires each dependency structure to be projec-

[0]Authors in alphabetical order.

tivised separately.

2.1 Synchronous derivations

The derivations for syntactic dependency trees are the same as specified in (Titov and Henderson, 2007b), which are based on the shift-reduce style parser of (Nivre et al., 2006). The derivations use a stack and an input queue. There are actions for creating a leftward or rightward arc between the top of the stack and the front of the queue, for popping a word from the stack, and for shifting a word from the queue to the stack. The derivations for semantic dependency graphs use virtually the same set of actions, but impose fewer constraints on when they can be applied, due to the fact that a word in a semantic dependency graph can have more than one parent. An additional action *predicate*$_s$ was introduced to label a predicate with sense s.

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 need to specify how the two derivations are synchronised, and in particular make the important choice of the granularity of the synchronisation step. Linguistic intuition would perhaps suggest that syntax and semantics are connected at the clause level – a big step size – while a fully integrated system would synchronise at each parsing decision, thereby providing the most communication between these two levels. We choose to synchronise the construction of the two structures at every word – an intermediate step size. This choice is simpler, as it is based on the natural total order of the input, and it avoids the problems of the more linguistically motivated choice, where chunks corresponding to different semantic propositions would be overlapping.

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 $D_d^{b_d^t-1} = D_s^{b_s^t-1} = \mathit{shift}_{t-1}$ and $D_d^{e_d^t+1} = D_s^{e_s^t+1} = \mathit{shift}_t$. Then the actions of the synchronous derivations consist of quadruples $C^t = (c_d^t, \mathit{switch}, c_s^t, \mathit{shift}_t)$, 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.

$$\begin{aligned} P(T_d, T_s) &= P(C^1, \ldots, C^n) \\ &= \textstyle\prod_t P(C^t | C^1, \ldots, C^{t-1}) \end{aligned} \quad (1)$$

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.

$$\begin{aligned} P(C^t | C^1, \ldots, C^{t-1}) \;=\; & \\ & P(c_d^t | C^1, \ldots, C^{t-1}) \times \\ & P(\mathit{switch} | c_d^t, C^1, \ldots, C^{t-1}) \times \\ & P(c_s^t | \mathit{switch}, c_d^t, C^1, \ldots, C^{t-1}) \times \\ & P(\mathit{shift}_t | c_d^t, c_s^t, C^1, \ldots, C^{t-1}) \end{aligned} \quad (2)$$

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

The probability of c_d^t is decomposed into derivation action D^i probabilities, and likewise for c_s^t.

$$\begin{aligned} & P(c_d^t | C^1, \ldots, C^{t-1}) \\ & = \textstyle\prod_i P(D_d^i | D_d^{b_d^t}, \ldots, D_d^{i-1}, C^1, \ldots, C^{t-1}) \end{aligned} \quad (3)$$

The actions are also sometimes split into a sequence of elementary decisions $D^i = d_1^i, \ldots, d_n^i$, as discussed in (Titov and Henderson, 2007a).

2.2 Projectivisation of dependencies

These derivations can only specify projective syntactic or semantic dependency graphs. Exploratory data analysis indicates that many instances of non-projectivity in the complete graph are due to crossings of the syntactic and semantic graphs. The amount of non-projectivity of the joint syntactic-semantic graph is approximately 7.5% non-projective arcs, while summing the non-projectivity within the two separate graphs results in only roughly 3% non-projective arcs.

Because our synchronous derivations use two different stacks for the syntactic and semantic dependencies, respectively, we only require each individual graph to be projective. As with many dependency parsers (Nivre et al., 2006; Titov and Henderson, 2007b), we handle non-projective (i.e. crossing) arcs by transforming them into non-crossing arcs with augmented labels.[1] Because our syntactic derivations are equivalent to those of (Nivre et al., 2006), we use their HEAD methods to projectivise the syntactic dependencies.

Although our semantic derivations use the same set of actions as the syntactic derivations, they differ in that the graph of semantic dependencies need

[1]During testing, these projectivised structures are then transformed back to the original format for evaluation.

not form a tree. The only constraints we place on the set of semantic dependencies are imposed by the use of a stack, which excludes crossing arcs. Given two crossing arcs, we try to uncross them by changing an endpoint of one of the arcs. The arc (p, a), where p is a predicate and a is an argument, is changed to (p, h), where h is the syntactic head of argument a. Its label r is then changed to r/d where d is the syntactic dependency of a on h. This transformation may need to be repeated before the arcs become uncrossed. The choice of which arc to transform is done using a greedy algorithm and a number of heuristics, without doing any global optimisation across the data.

This projectivisation method is similar to the HEAD method of (Nivre et al., 2006), but has two interesting new characteristics. First, syntactic dependencies are used to projectivise the semantic dependencies. Because the graph of semantic roles is disconnected, moving across semantic arcs is often not possible. This would cause a large number of roles to be moved to ROOT. Second, our method changes the semantic argument of a given predicate, whereas syntactic dependency projectivisation changes the head of a given dependent. This difference is motivated by a predicate-centred view of semantic dependencies, as it avoids changing a predicate to a node which is not a predicate.

3 The Learning Architecture

The synchronous derivations described above are modelled with an Incremental Sigmoid Belief Network (ISBN) (Titov and Henderson, 2007a). ISBNs are dynamic Bayesian Networks which incrementally specify their model structure based on the partial structure being built by a derivation. They have previously been applied to constituency and dependency parsing. In both cases the derivations were based on a push-down automaton, but ISBNs can be directly applied to any automaton. We successfully apply ISBNs to a two-stack automaton, without changing the machine learning methods.

3.1 The Incremental Sigmoid Belief Networks

ISBNs use vectors of latent variables to represent properties of parsing history relevant to the next decisions. Latent variables do not need to be annotated in the training data, but instead get induced during learning. As illustrated by the vectors S^i in figure 1, the latent feature vectors are used to estimate the probabilities of derivation actions D^i.

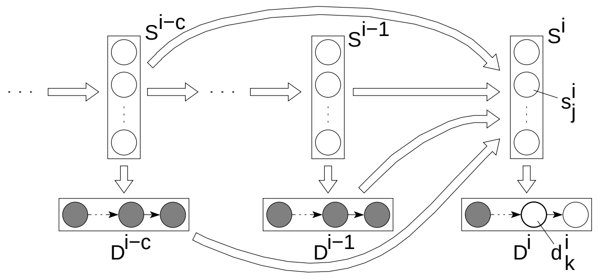

Figure 1: An ISBN for estimating $P(d^i_k|history(i,k))$ – one of the elementary decisions. Variables whose values are given in $history(i,k)$ are shaded, and latent and current decision variables are unshaded.

Latent variable vectors are connected to variables from previous positions via a pattern of edges determined by the previous decisions. Our ISBN model distinguishes two types of latent states: syntactic states, when syntactic decisions are considered, and semantic states, when semantic decision are made. Different patterns of interconnections are used for different types of states. We use the neural network approximation (Titov and Henderson, 2007a) to perform inference in our model.

As also illustrated in figure 1, the induced latent variables S^i at state i are statistically dependent on both pre-defined features of the derivation history $D^1, \ldots, D^{i-1}$ and the latent variables for a finite set of relevant previous states $S^{i'}, i' < i$. Choosing this set of relevant previous states is one of the main design decisions in building an ISBN model. By connecting to a previous state, we place that state in the local context of the current decision. This specification of the domain of locality determines the inductive bias of learning with ISBNs. Thus, we need to choose the set of local (i.e. connected) states in accordance with our prior knowledge about which previous decisions are likely to be particularly relevant to the current decision.

3.2 Layers and features

To choose previous relevant decisions, we make use of the partial syntactic and semantic dependency structures which have been decided so far in the parse. Specifically, the current latent state vector is connected to the most recent previous latent state vectors (if they exist) whose configuration shares a node with the current configuration, as specified in Table 1. The nodes are chosen because their properties are thought to be relevant to the current decision. Each row of the table indicates which nodes need to be identical, while each

Closest	Current	Syn-Syn	Srl-Srl	Syn-Srl
Input	Input	+	+	+
Top	Top	+	+	+
RDT	Top	+	+	
LDT	Top	+	+	
HT	Top	+	+	
LDN	Top	+	+	
Input	Top	+		

Table 1: Latent-to-latent variable connections. Input= input queue; Top= top of stack; RDT= rightmost right dependent of top; LDT= leftmost left dependent of top; HT= Head of top; LDN= leftmost dependent of next (front of input).

column indicates whether the latent state vectors are for the syntactic or semantic derivations. For example, the first row indicates edges between the current state and a state which had the same input as the current state. The three columns indicate that this edge holds within syntactic states, within semantic states, and from syntactic to semantic states. The fourth cell of the third row, for example, indicates that there is an edge between the current semantic state on top of the stack and the most recent semantic state where the rightmost dependent of the current top of the semantic stack was at the top of the semantic stack.

Each of these relations has a distinct weight matrix for the resulting edges in the ISBN, but the same weight matrix is used at each position where the relation applies. Training and testing times scale linearly with the number of relations.

The pre-defined features of the parse history which also influence the current decision are specified in table 2. The model distinguishes argument roles of nominal predicates from argument roles of verbal predicates.

3.3 Decoding

Given a trained ISBN as our probability estimator, we search for the most probable joint syntactic-semantic dependency structure using a beam search. Most pruning is done just after each *shift* operation (when the next word is predicted). Global constraints (such as label uniqueness) are not enforced by decoding, but can be learnt.

For the system whose results we submitted, we then do a second step to improve on the choice of syntactic dependency structure. Because of the lack of edges in the graphical model from semantic to syntactic states, it is easy to marginalise out the semantic structure, giving us the most probable syntactic dependency structure. This syntactic structure is then combined with the semantic structure from the first stage, to get our submitted results. This second stage does not maximise performance on the joint syntactic-semantic dependency structure, but it better fits the evaluation measure used to rank systems.

State	Stack	Syntactic step features		
		LEX	POS	DEP
Input		+	+	
Top	syn	+	+	
Top - 1	syn		+	
HT	syn	+		
RDT	syn			+
LDT	syn			+
LDN	syn			+

State	Stack	Semantic step features			
		LEX	POS	DEP	SENSE
Input		+	+		+
Top	sem	+	+		+
Top - 1	sem	+	+		
HT	sem	+		+	
RDT	sem			+	
LDT	sem			+	
LDN	sem			+	
A0-A5 of Top	sem		+		
A0-A5 of Input	sem		+		

Table 2: Pre-defined features. syn=syntactic stack; sem=semantic stack. Input= input queue; Top= top of stack; RDT= rightmost dependent of top; LDT= leftmost dependent of Top; HT= Head of top; LDN= leftmost dependent of next (front of input); A0-A5 of Top/Input= arguments of top of stack / input.

4 Experiments and Discussion

The experimental set-up common for all the teams is described in the introduction (Surdeanu et al., 2008). The submitted model has latent variable vectors of 60 units, and a word frequency cut-off of 100, resulting in a small vocabulary of 1083 words. We used a beam of size 15 to prune derivations after each *shift* operation to obtain the joint structure, and a beam of size 40 when performing the marginalisation. Training took approximately 2.5 days on a standard PC with 3.0 GHz Pentium4 CPU. It took approximately 2 hours to parse the entire testing set (2,824 sentences) and an additional 3 hours to perform syntactic parsing when marginalising out the semantic structures.[2] Shortly after the submission deadline, we trained a 'large' model with a latent variable vector of size 80, a word frequency cut-off of 20, and additional latent-to-latent connections from semantics to syntax of the same configuration as the last column

[2]A multifold speed-up with a small decrease in accuracy can be achieved by using a small beam.

	Syn	Semantic			Overall		
	LAS	P	R	F1	P	R	F1
Submitted							
D	86.1	78.8	64.7	71.1	82.5	75.4	78.8
W	87.8	79.6	66.2	72.3	83.7	77.0	80.2
B	80.0	66.6	55.3	60.4	73.3	67.6	70.3
WB	86.9	78.2	65.0	71.0	82.5	76.0	79.1
Joint inference							
D	85.5	78.8	64.7	71.1	82.2	75.1	78.5
Large, joint inference							
D	86.5	79.9	67.5	73.2	83.2	77.0	80.0
W	88.5	80.4	69.2	74.4	84.4	78.8	81.5
B	81.0	68.3	57.7	62.6	74.7	69.4	71.9
WB	87.6	79.1	67.9	73.1	83.4	77.8	80.5

Table 3: Scores on the development set and the final testing sets (percentages). D= development set; W=WSJ; B=Brown; WB=WSJ+Brown;

of table 1. This model took about 50% longer in training and testing.

In table 3, we report results for the marginalised inference ('submitted') and joint inference for the submitted model, and the results for joint inference with the 'large' model. The larger model improves on the submitted results by almost 1.5%, a significant improvement. If completed earlier, this model would have been fifth overall, second for syntactic LAS, and fifth for semantic F1.

To explore the relationship between the two components of the model, we removed the edges between the syntax and the semantics in the submitted model. This model's performance drops by about 3.5% for semantic role labelling, thereby indicating that the latent annotation of parsing states helps semantic role labelling. However, it also indicates that there is much room for improvement in developing useful semantic-specific features, which was not done for these experiments simply due to constraints on development time.

To test whether joint learning degrades the accuracy of the syntactic parsing model, we trained a syntactic parsing model with the same features and the same pattern of interconnections as used for the syntactic states in our joint model. The resulting labelled attachment score was non-significantly lower (0.2%) than the score for the marginalised inference with the joint model. This result suggests that, though the latent variables associated with syntactic states in the joint model were trained to be useful in semantic role labelling, this did not have a negative effect on syntactic parsing accuracy, and may even have helped.

Finally, an analysis of the errors on the development set for the submitted model paints a coherent picture. We find attachment of adjuncts particularly hard. For dependency labels, we make the most mistakes on modification labels, while for semantic labels, we find TMP, ADV, LOC, and PRN particularly hard. NomBank arcs are not learnt as well as PropBank arcs: we identify PropBank SRL arguments at F1 70.8% while Nombank arguments reach 58.1%, and predicates at accuracy 87.9% for PropBank and 74.9% for NomBank.

5 Conclusions

While still preliminary, these results indicate that synchronous parsing is an effective way of building joint models on separate structures. The generality of the ISBN design used so far suggests that ISBN's latent feature induction extends well to estimating very complex probability models, with little need for feature engineering. Nonetheless, performance could be improved by task-specific features, which we plan for future work.

Acknowledgements

This work was partly funded by European Community FP7 grant 216594 (CLASSiC, www.classic-project.org), Swiss NSF grant 114044, and Swiss NSF fellowships PBGE2-117146 and PBGE22-119276. Part of this work was done when G. Musillo was visiting MIT/CSAIL, hosted by Prof. Michael Collins.

References

Marcus, M., B. Santorini, and M.A. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19:313–330.

Merlo, P. and G. Musillo. 2008. Semantic parsing for high-precision semantic role labelling. In *Procs of CoNLL 2008*, Manchester, UK.

Meyers, A., R. Reeves, C. Macleod, R. Szekely, V. Zielinska, B. Young, and R. Grishman. 2004. The nombank project: An interim report. In Meyers, A., editor, *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*, 24–31, Boston, MA.

Musillo, G. and P. Merlo. 2006. Accurate semantic parsing of the Proposition Bank. In *Procs of NAACL 2006*, New York, NY.

Nivre, J., J. Hall, J. Nilsson, G. Eryigit, and S. Marinov. 2006. Pseudo-projective dependency parsing with support vector machines. In *Proc. of CoNNL*, 221–225, New York, USA.

Palmer, M., D. Gildea, and P. Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31:71–105.

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. In *Procs of CoNLL-2008*, Manchester,UK.

Titov, I. and J. Henderson. 2007a. Constituent parsing with incremental sigmoid belief networks. In *Procs of ACL'07*, pages 632–639, Prague, Czech Republic.

Titov, I. and J. Henderson. 2007b. A latent variable model for generative dependency parsing. In *Procs of IWPT'07*, Prague, Czech Republic.

Wong, Y.W. and R. Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *Procs of ACL'07*, 960–967, Prague, Czech Republic.