Efficient Inference and Structured Learning for Semantic Role Labeling

Oscar Täckström
Google
New York
oscart@google.com

Kuzman Ganchev
Google
New York
kuzman@google.com

Dipanjan Das
Google
New York
dipanjand@google.com

Abstract

We present a dynamic programming algorithm for efficient constrained inference in semantic role labeling. The algorithm tractably captures a majority of the structural constraints examined by prior work in this area, which has resorted to either approximate methods or off-the-shelf integer linear programming solvers. In addition, it allows training a globally-normalized log-linear model with respect to constrained conditional likelihood. We show that the dynamic program is several times faster than an off-the-shelf integer linear programming solver, while reaching the same solution. Furthermore, we show that our structured model results in significant improvements over its local counterpart, achieving state-of-the-art results on both PropBank- and FrameNet-annotated corpora.

1 Introduction

Semantic role labeling (henceforth, SRL) is the task of identifying the semantic arguments of predicates in natural language text. Pioneered by Gildea and Jurafsky (2002), this task has been widely investigated by the NLP community. There have been two shared tasks at CoNLL 2004 and 2005 focusing on this problem, using PropBank conventions to identify the phrasal arguments of verbal predicates (Palmer et al., 2005; Carreras and Màrquez, 2004, 2005). Since then, there has been work on SRL for nominal predicates (Meyers et al., 2004; Gerber and Chai, 2010) and variants that investigated the prediction of semantic dependencies rather than phrasal arguments (Surdeanu et al., 2008; Hajič et al., 2009).

Here, we present an inference method for SRL, addressing the problem of phrasal argument structure prediction (as opposed to semantic dependencies). In contrast to most prior semantic role labeling work focusing on PropBank conventions, barring notable exceptions such as Meza-Ruiz and Riedel (2009), our framework first performs **frame identification**, the subtask of disambiguating the predicate frame; this makes our analysis more interpretable. The focus of this paper, however, is the subtask of **semantic role labeling**, wherein we take a set of (potentially overlapping) candidate sentential phrases and identify and label them with the semantic roles associated with the predicted frame. This treatment is commonly used in frame semantic parsing (Das et al., 2014; Hermann et al., 2014) and our two-stage framework is able to model both PropBank and FrameNet conventions.

Previous work focusing on semantic role labeling imposed several structural constraints warranted by the annotation conventions of the task and other linguistic considerations, such as avoiding overlapping arguments and repeated core roles in the final prediction. Such global inference often leads to improved results and more meaningful predictions compared to local unconstrained methods (Màrquez et al., 2008). A popular framework for imposing these constraints has been integer linear programming (ILP), wherein the inference problem is specified declaratively (Punyakanok et al., 2008). However, ILP-based inference methods often rely on generic off-the-shelf solvers that fail to exploit problem-specific structure (Martins et al., 2011). Instead, we present a dynamic program (DP) that exactly enforces most of the constraints examined by Punyakanok et al. (2008); remaining constraints are enforced by reverting to k-best inference if needed. We show that this technique solves the inference problem more than four times faster than a state-of-the-art off-the-shelf ILP solver, while

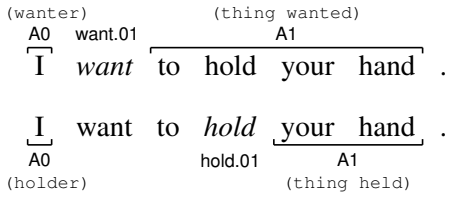


Figure 1: Example semantic role annotations for the two verbs in the sentence "I want to hold your hand.", according to PropBank. The annotations on top show the frame structure corresponding to *want*, while the ones below reflect the annotations for *hold*. Note that the agent role (A0) is realized as the same word ("I"), but with the meaning *wanter* in one case and *holder* in the other.

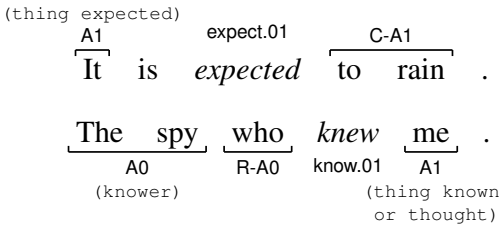


Figure 2: Examples showing continuation and reference roles according to PropBank. The role prefix C- indicates continuation of an argument, while the prefix R- indicates reference to another overt argument of the same predicate.

being guaranteed to achieve identical results.

In addition to being relatively slow, ILP-based methods only solve the maximum a posteriori (MAP) inference problem, which prevents the computation of marginals and feature expectations. The proposed DP, on the other hand, allows us to train a globally-normalized log-linear model, enforcing the structural constraints during training. Empirically, we show that such a structured model consistently performs better than training separate classifiers and incorporating the constraints only at inference time. We present results on the Wall Street Journal development and test sets, as well as the Brown test set from the CoNLL 2005 shared task for verbal SRL; these show that our structured model — which uses a single dependency parse and no model averaging or reranking — outperforms other strong single-model systems and rivals state-of-the-art ensemble-based methods. We further present results on the OntoNotes 5.0 corpora annotated with semantic roles for both verbal and nominal predicates (Weischedel et al., 2011) and strongly outperform the prior state of the art (Pradhan et al., 2013). Finally, we present results on FrameNet 1.5 data, again achieving state-of-the-art results.

2 Task Overview

We seek to predict the semantic argument structure of predicates in text. For brevity and practical reasons, the exposition and empirical study is primarily focused on PropBank-style annotations (Palmer et al., 2005). However, our approach applies directly to FrameNet-style annotations as well (Baker et al., 1998) and as shown empirically in §6, a similar trend holds across both types of annotation.

In both cases, we are provided with a frame lexicon that contains *type*-level information for *lexical units* (a lemma conjoined with a coarse-grained part-of-speech tag).[1] For each lexical unit, a list of senses, or *frames*, are provided, where each frame comes with a set of semantic roles that constitute the various participants in the frame. These roles can be either *core* or *non-core* to the frame.

In PropBank, a set of seven generic core role labels are defined (A0-A5 and AA) that take on different semantics for each frame; each frame associates with a subset of these core roles. In addition there are 21 non-core role labels that serve as adjuncts, such as the temporal role AM-TMP and the locative role AM-LOC; these are shared across frames and assume similar meaning.

FrameNet similarly specifies a set of frames and roles, with two key differences. First, the semantics of the small set of core role labels in PropBank are local to each frame. In contrast, the several hundred role labels in FrameNet are shared across frames and they take on similar semantics in the frames in which they participate. Second, while frames in PropBank are just coarse-grained lemma-specific senses, the frame repository in FrameNet is shared across lemmas. See Hermann et al. (2014) for examples of these differences.

Both PropBank- and FrameNet annotated data consist of sentence-level annotations that instantiate the respective frame lexicon with each predicate disambiguated to its frame, as well as the phrasal arguments of each predicate labeled with their semantic roles. Figure 1 shows an example sentence with two verbs annotated according to PropBank conventions.

[1] The CoNLL 2005 dataset is restricted to verbal predicates.

In addition to such basic semantic role annotation, the PropBank-annotated data sets from the CoNLL 2004 and 2005 shared tasks and OntoNotes 5.0, represent discontiguous arguments across multiple spans. These are annotated such that the first span is labeled with one of the 28 semantic role labels, while subsequent spans have the *continuation* prefix C- attached to the role. The first sentence in Figure 2 shows such an annotation. Moreover, these data sets feature *reference* roles for arguments, primarily relative pronouns, that refer to other overt arguments of the predicate. These roles are annotated by attaching the prefix R- to the role of the co-referent argument. For example, in the second sentence of Figure 2, the relative pronoun *who* refers to the argument *The spy* and is labeled R-A0. FrameNet annotations, on the other hand, contain neither continuation or reference roles according to conventions adopted by prior work.

3 Model

Before delving into the details of the structural constraints enforced in the SRL task, we describe its two subtasks. Akin to most previous work, these subtasks are solved as separate steps in a cascaded fashion.

3.1 Classifier Cascade

To predict annotations such as those described in the previous section, we take a preprocessed sentence and first attempt to disambiguate the frame of each predicate (*frame identification*). In this work, as part of preprocessing, we use a part-of-speech tagger and a dependency parser to syntactically analyze the sentence; this diverges from most prior work on semantic argument prediction, which rely on constituency parses. Next, we take each disambiguated frame and look up the core and non-core (or adjunct) roles that can associate with the frame. Given the predicate token, we (over-)generate a set of candidate spans in the sentence, that are then labeled with roles from the set of core roles, from the set of adjunct roles, or with the null role $\varnothing$ (*role labeling*).[2] Our system thus comprises a cascade of two statistical models. Note that most prior work on PropBank data only considered the latter task, remaining agnostic to the frame. Moreover, the semantic role labeling step has typically been divided into two stages: first identifying the spans that serve as semantic arguments and then labeling them with their roles (Màrquez et al., 2008). In contrast, we approach the semantic role labeling subproblem using a single statistical model.

[2]This setup differs from the related line of work that only predicts semantic *dependencies* between the predicate and the head words of semantic arguments; the latter task is arguably more straightforward (Surdeanu et al., 2008; Hajič et al., 2009).

3.2 Frame Identification

Given a preprocessed sentence x and a marked predicate t with lemma ℓ, we seek to predict the frame f instantiated by the predicate. To this end, we use different models in the PropBank and FrameNet settings. In case of PropBank, we define the probability of a frame f under a conditional log-linear model:

$$p(f \mid x, t, \ell) \propto \exp\left(\boldsymbol{\psi} \cdot \boldsymbol{h}(f, x, t, \ell)\right) ,$$

where $\boldsymbol{\psi}$ denotes the model parameters and $\boldsymbol{h}(\cdot)$ is the feature function (see Table 1 for details on the features employed). The model's partition function sums over all frames for the lemma ℓ in the lexicon and we estimate the model parameters by maximizing regularized conditional log-likelihood.

In the case of FrameNet, to make our results directly comparable to the recent state-of-the-art results of Hermann et al. (2014), we instead use their embeddings-based WSABIE model (Weston et al., 2011) for the frame identification step.

3.3 Unconstrained Semantic Role Labeling

Given an identified frame f in a sentence x of n words $(w_1, \ldots, w_n)$, we seek to predict a set of argument spans labeled with their semantic roles. We assume that there is a set of candidate spans $\mathcal{S}$ that could potentially serve as arguments of t. Specifically, we derive $\mathcal{S}$ with a high-recall rule-based algorithm that looks at the (dependency) syntactic context of the predicate word t, as described in §6.3.

Let one candidate span be $s \in \mathcal{S}$. The set of possible roles $\mathcal{R}$ is composed of core roles $\mathcal{R}_C$ associating with f, adjunct roles $\mathcal{R}_A$ and the *null* role $\varnothing$. In addition, in the PropBank setting, we have a set of continuation roles $\mathcal{R}_N$ and reference roles $\mathcal{R}_R$; thus, $\mathcal{R} = \mathcal{R}_C \cup \mathcal{R}_A \cup \mathcal{R}_N \cup \mathcal{R}_R \cup \{\varnothing\}$. We assume a model that assigns a real-valued compatibility score $g(s, r)$ to each pair of span and role $(s, r) \in \mathcal{S} \times \mathcal{R}$; the precise nature of the model and its estimation is described in §5. With no consistency constraints

between the span-role pairs, prediction amounts to selecting the optimal role for each span. This gives us a global score which is a sum over all spans:

$$\sum_{s \in \mathcal{S}} \max_{r \in \mathcal{R}} g(s, r)\,, \tag{1}$$

with the solution being the corresponding $\arg\max$.

3.4 Semantic Role Labeling as an ILP

We can represent any prediction for the individual classifiers with a set of indicator variables $\mathbf{z} = \{z_{s,r}\}$ with one variable for each span s and role r. An equivalent formulation to Equation (1) is then:

$$\begin{aligned} \max_{\mathbf{z}} \; & \sum_{s \in \mathcal{S}} \sum_{r \in \mathcal{R}} z_{s,r} \times g(s, r) \\ \text{s.t. } & \mathbf{z} \in \{0, 1\}^{|\mathcal{S}||\mathcal{R}|} \\ & \sum_{r \in \mathcal{R}} z_{s,r} = 1 \quad \forall s \in \mathcal{S}\,, \end{aligned} \tag{2}$$

where we have constrained the indicator variables to take on binary values, and required that we choose exactly one role (including the $\varnothing$ role) for each span. To further guide the inference, we add the following constraints to the ILP in Equation (2), as originally proposed by Punyakanok et al. (2008):[3]

No Span Overlap Let $\mathcal{S}_i$ be the set of spans covering token w_i. We want to ensure that at most one of the spans in $\mathcal{S}_i$ have an overt role assignment:

$$\forall i \in [1, n]\,, \quad \sum_{s \in \mathcal{S}_i} \sum_{r \neq \varnothing} z_{s,r} \leq 1\,.$$

Unique Core Roles Each core role $r \in \mathcal{R}_C$ can be overt in at most one of the spans in $\mathcal{S}$:

$$\forall r \in \mathcal{R}_C\,, \quad \sum_{s \in \mathcal{S}} z_{s,r} \leq 1\,.$$

Continuation Roles A *continuation* role, may only be assigned if the corresponding base (i.e. non-continuation, non-reference) role is assigned to an earlier span. To express this, we define $s \leq s'$ to mean that s starts before s'. For a continuation role $r \in \mathcal{R}_N$, let $\text{base}(r) \in \mathcal{R}_C \cup \mathcal{R}_A$ be the corresponding base role. Then the constraint is:

$$\forall r \in \mathcal{R}_N\,,\ \forall s \in \mathcal{S}\,, \quad z_{s,r} \leq \sum_{s' \leq s} z_{s',\text{base}(r)}\,.$$

[3]Note that the *continuation roles* and *reference roles* constraints below are only applicable to PropBank annotations, as these roles are not present in FrameNet annotations.

Reference Roles Similar to continuation roles, a span can only be labeled with a *reference* role $r \in \mathcal{R}_R$ if another span is labeled with the corresponding base role, $\text{base}(r) \in \mathcal{R}_C \cup \mathcal{R}_A$:

$$\forall r \in \mathcal{R}_R\,,\ \forall s \in \mathcal{S}\,,\ z_{s,r} \leq \sum_{s' \in \mathcal{S}} z_{s',\text{base}(r)}\,.$$

4 Dynamic Program Formulation

An advantage of the formulation in the previous section is that the constrained MAP inference problem can be solved with an off-the-shelf ILP solver. Unfortunately, these solvers typically fail to exploit the problem-specific structure of the set of admissible solutions, which often leads to slow inference. As an alternative, we propose a dynamic program that takes advantage of the sequential and local nature of the problem, while directly enforcing all but the non-core *continuation roles* constraint and the *reference roles* constraint; the remaining constraints can be efficiently enforced by a straightforward search over the k-best solutions of the dynamic program. The resulting inference procedure is guaranteed to find the same optimal solution as the corresponding ILP (modulo rounding and tie breaking), while being substantially faster. In addition, the forward-backward algorithm can be applied to compute marginals over the indicator variables, taking the constraints into account. This facilitates computation of confidence scores, as well as learning with a constrained globally normalized log-linear model, as described in §5.

We encode the dynamic program as a weighted lattice $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ is the set of (weighted) edges, such that the shortest path through the lattice corresponds to the optimal ILP solution. The core of the lattice is the encoding of the *no span overlap* constraint; additional constraints are later added on top of this backbone.

4.1 No Span Overlap

We first describe the structure and then the weights of the dynamic program lattice. For ease of exposition, Figure 3 shows an example sentence with three argument candidates corresponding to "It", "to rain" and "rain", with the possible span-role assignments: "It":A1/$\varnothing$, "to rain":A0/C-A1/$\varnothing$ and "rain":A0/$\varnothing$. Our goal is to construct a dynamic program such that the length of the optimal path is equal to the score

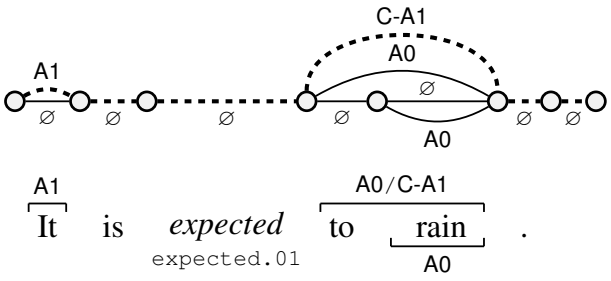


Figure 3: Lattice corresponding to the dynamic program for the *no span overlap* constraint. The path of the correct argument assignment is indicated with dashed edges.

of mapping "It" to A1, "to rain" to C-A1 and "rain" to ∅. The dynamic program needs to ensure that regardless of the scores, either "to rain" or "rain" must be labeled with ∅ since they overlap in the sentence. We satisfy the latter by using a semi-Markov model formally described below. In order to ensure that the ∅ role assignment scores are included correctly, they are given a special treatment in the scoring function.[4]

Lattice Structure The set of vertices $\mathcal{V} = \{v_j : j \in [0, n+1]\}$ contains a vertex between every pair of consecutive words. The edges $\mathcal{E}$ are divided into *null edges* (between consecutive vertices) and *argument edges* (which connect the vertices corresponding to the argument span endpoints). We will use the notation $e_{j,j+1,\varnothing}$ for the null edge from v_j to v_{j+1}. For each span and non-null role pair $(s, r), r \neq \varnothing$, we add an argument edge $e_{s,r}$ between v_{i-1} and v_j where the span s is from word i to j. Figure 3 illustrates the structure of the lattice. In this example, we assume that there are two possible roles (A0/C-A1) for the phrase "to rain"; consequently, there are two argument edges corresponding to this phrase.

A path through the lattice corresponds to a global assignment of roles to spans by assigning role r to span s for every edge $e_{s,r}$ in the path, and assigning the ∅ role to all other spans. The length of a path is given by the sum of the weights of its edges.

Lattice Weights The idea behind our weighting scheme is to include all the null scores $g(s, \varnothing)$ at the start, and then subtract them whenever we assign a role to a candidate span. Let us augment the lattice described above with a special node v_{-1} and a special edge $e_{*,\varnothing}$ between v_{-1} and v_0. Set the weight of $e_{*,\varnothing}$ to $c_*^{\varnothing} = \sum_{s \in \mathcal{S}} g(s, \varnothing)$. We then set the weight of the null edges $e_{j,j+1,\varnothing}$ to 0 and the weight of the argument edges $e_{s,r}$ to $c_s^r = g(s, r) - g(s, \varnothing)$.

Proposition 1. *There is a one-to-one correspondence between paths in the lattice and global role assignments with non-overlapping arguments. Furthermore, the length of a path is equivalent to the ILP score of the corresponding assignment.*

Proof Sketch We already described how to construct an assignment from any path through the lattice. For any role assignment without overlaps we can include all these edges in a single left-to-right path, and complete the path with null edges. Since there are no overlaps, we will not need to include incompatible edges. So there is a one-to-one correspondence between paths and valid assignments. To see that the score is the same as the path length, we can use induction on the number of non-null edges in the path. *Base case:* If there are no selected arguments, then the length of the path is just $c_*^{\varnothing}$ which is exactly the ILP score. *Inductive step:* We add an overt argument to the solution. In the path, we replace a sequence of null edges with an edge $e_{s,r}$. The change in path length is $c_s^r = g(s, r) - g(s, \varnothing)$. In the assignment, we need to change $z_{s,\varnothing}$ from 1 to 0 and $z_{s,r}$ from 0 to 1. Thus, the change in ILP score is also $g(s, r) - g(s, \varnothing)$. □

The above construction can be further simplified. Note that while the special edge $e_{*,\varnothing}$ is needed for the direct correspondence with the ILP score, its weight is constant across variable assignments. Thus, this edge only adds a constant offset of $c_*^{\varnothing}$ to the ILP solution. For the same reason, its presence has no influence on the $\arg\max$ or marginal computations and we therefore drop it in our implementation.

4.2 Unique Core Roles

To incorporate the *unique core roles* constraint, we add state signatures to the vertices in the lattice and restrict the edges accordingly. This increases the size of the lattice by $O(2^{|\mathcal{R}_C|})$, where $\mathcal{R}_C$ is the set of core roles. Our approach is similar to that of Tromble and Eisner (2006), but whereas they suggest incorporating the uniqueness constraints incrementally, we apply them all at once. This is necessary since we

[4] The resulting lattice corresponds to that of an *aggressively pruned* semi-Markov sequence model, modulo the special care given to the ∅ role in our case.

seek to train a structured *probabilistic* model, which requires the marginals with respect to the full set of constraints.[5] While the number of signatures is exponential in $|\mathcal{R}_C|$, in practice this is a modest constant as each frame only has a small number of possible core roles (two or three for many frames).[6] Furthermore, since many of the potential edges are pruned by the constraints, as described below, the added computational complexity is further reduced.

Lattice Structure The set of vertices are now $\mathcal{V} = \{v_0, v_{n+1}, v_j^k : j \in [1,n]\,,\ k \in \{0,1\}^{|\mathcal{R}_C|}\}$, where v_0 and v_{n+1} are the start and end vertices. The remaining vertices v_j^k are analogous to the ones in §4.1 but are annotated with a bit vector encoding the subset of core roles that have been used so far. The rth bit in the superscript k is set iff the rth core role has been assigned at v_j^k. The null edges $e_{j,j+1,\varnothing}^k$ connect each node v_j^k to its successor v_{j+1}^k. Since a null edge does not affect the core role assignment, the signature k remains unchanged between v_j^k and v_{j+1}^k.

Figure 4 shows an example lattice, which in addition to the *no span overlap* and *unique core roles* constraints encodes the *core continuation roles* constraint (see §4.3). For efficiency, we exclude vertices and edges not on any path from v_0 to v_{n+1}. For example, v_1^k exist only for $|k| \leq 1$, since v_0 corresponds to no core roles being selected and a single span can add at most one core role. Argument edges $e_{s,r}^k$ connecting vertices v_{i-1}^k and $v_j^{k'}$ corresponds to assigning role r to the span $s = w_i, \ldots, w_j$. If $r \in \mathcal{R}_C$ then $k \neq k'$, otherwise $k = k'$. The edge is only included if the role r is non-core, or if $k_r \neq 1$, to guarantee uniqueness of core roles. By this construction, once a core role has been assigned at a vertex v_j^k, it cannot be assigned on any future path reachable from v_j^k.

Lattice Weights The edges are weighted in the same way as in §4.1. It is easy to verify that the structure enforces unique core roles, but is otherwise equivalent to that in §4.1. Since the weights are identical, the proof of Proposition 1 carries over directly.

[5] We note that the approach of Riedel and Smith (2010) could potentially be used to compute the marginals in an incremental fashion similar to Tromble and Eisner (2006).

[6] In the OntoNotes 5.0 development set, there are on average 10.4 core-role combinations per predicate frame.

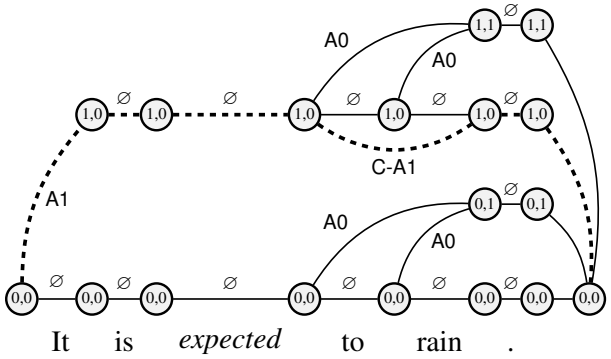


Figure 4: Lattice corresponding to the *no span overlap*, *unique core roles* and *core continuation roles* constraints. Each vertex is labeled with its signature $k \in \{0,1\}^{|\mathcal{R}_C|}$; in this example, "$0,1$" equals $\{\mathsf{A0}\}$. This represents the subset of core-roles assigned on the path up to and including the vertex. Dashed edges indicate the correct path.

4.3 Core Continuation Roles

Recall that the constraint for continuation roles is that they must occur after their corresponding base role. We enforce this constraint for core roles by not including argument edges $e_{s,r}^k$ with $r \in \mathcal{R}_N$ from a configuration k which does not have the corresponding base role set ($k_{\text{base}(r)} \neq 1$). Figure 4 shows an example; here the edge corresponding to "to rain" with label C-A1 is included since the vertex signature $k = \{1,0\}$ has $k_{\mathsf{A1}} = 1$, but there is no corresponding edge for $k' = \{0,0\}$ since $k'_{\mathsf{A1}} = 0$.

4.4 Remaining Constraints

Unfortunately, enforcing the *reference roles* constraint, and the *continuation roles* constraint for non-core roles, directly in the dynamic program is not practical, due to combinatorial explosion. First, while the *continuation roles* constraint almost only applies to core roles,[7] every role in $\mathcal{R}_C \cup \mathcal{R}_A$ may have a corresponding reference role. Second, even if we restrict the constraints to core reference roles, the lack of ordering between the spans in the constraint means that we would have to represent all subsets of $\mathcal{R}_C \times \{r \mid r \in \mathcal{R}_R\,, \text{base}(r) \in \mathcal{R}_C\}$.

However, these constraints are rarely violated in practice. As we will see in §6, these remaining constraints can be enforced efficiently with k-best inference in the constrained dynamic program from

[7] Less than 2% of continuation roles correspond to non-core roles in the OntoNotes 5.0 development set.

Frame identification features	
• the predicate t	• tag of t
• the lemma ℓ	• children words of t
• tag of t's children	• tag of t's parent
• parent word of t	• subcat. frame of t
• dep. label of t	• dep. labels of t's children
• word to the left of t	• word to the right of t
• tag to the left of t	• tag to the right of t
• word cluster of t	• word clusters of t's children

Table 1: Frame identification features. By *subcategorization frame*, we refer to the sequence of dependency labels of t's children in the dependency tree.

the previous section, using the algorithm of Huang and Chiang (2005) and picking the best solution that satisfies all the constraints.

5 Local and Structured Learning

To train our models, we assume a training set where each predicate t (with lemma ℓ) in sentence x has been identified and labeled with its semantic frame f, as well as with each candidate span and role pair $(s, r) \in \mathcal{S} \times \mathcal{R}$. We first consider a local log-linear model. Let the local score of span s and role r be given by $g(s, r) = \boldsymbol{\theta} \cdot \boldsymbol{f}(r, s, x, t, \ell, f)$, where $\boldsymbol{\theta}$ denotes the vector of model parameters and $\boldsymbol{f}(\cdot)$ the feature function (see Table 2 for the specific features employed). We treat the local scores as the potentials in a multiclass logistic regression model, such that $p(r \mid s, x, t, \ell, f) \propto \exp(g(s, r))$, and estimate the parameters by maximizing the regularized conditional likelihood of the training set.

A downside of estimating the parameters locally is that it "wastes" model capacity, in the sense that the learning seeks to move probability mass away from annotations that violate structural constraints but can never be predicted at inference time. With the dynamic program formulation from the previous section, we can instead use a globally normalized probabilistic model that takes the constraints from §4.1-§4.3 into account during learning. To achieve this, we model the probability of a joint assignment

Features additionally conjoined with the frame	
• starting word of s	• tag of the starting word of s
• ending word of s	• tag of the ending word of s
• head word of s	• tag of the head word of s
• bag of words in s	• bag of tags in s
• a bias feature	• cluster of s's head
• dependency path between s's head and t	
• the set of dependency labels of t's children	
• dependency path conjoined with the tag of s's head	
• dep. path conjoined with the cluster of s's head	
• *position* of s w.r.t. t (*before, after, overlap or same*)	
• *position* conjoined with distance from s to t	
• subcategorization frame of s	
• predicate use voice (*active*, *passive*, or *unknown*)	
• whether the subject of t is missing (*missingsubj*)	
• *missingsubj*, conjoined with the dependency path between s's head and t	
• *missingsubj*, conjoined with the dependency path between s's head and the verb dominating t	
Features only conjoined with the role	
• cluster of s's head conjoined with cluster of t	
• dep. path conjoined with the cluster of the head of s	
• word of s's head conj. with words of its children	
• tag of s's head conj. with words of its children	
• cluster of s's head conj. with cluster of its children	
• cluster of t's head conj. with cluster of s's head	
• word, tag, dependency label and cluster of the words immediately to the left and right of s	
• six features that each conjoin position and distance with one of the following: tag, dependency label and cluster of s's head, tag, dependency label and cluster of t's head	

Table 2: Semantic role labeling features. The argument span is denoted by s, while t denotes the predicate token. All features are conjoined with the role r. Features in the top part have two versions, one conjoined with the role r and one conjoined with both the role r and the frame f.

$\mathbf{z}$, subject to the constraints, as

$$p(\mathbf{z} \mid x, t, \ell, f) \propto \exp\left(\sum_{s \in \mathcal{S}} \sum_{r \in \mathcal{R}} g(s, r) \times z_{s,r}\right)$$
$$\text{s.t. } \mathbf{z} \in \{0, 1\}^{|\mathcal{S}||\mathcal{R}|}, \quad A\mathbf{z} \leq \boldsymbol{b},$$

where $A\mathbf{z} \leq \boldsymbol{b}$ encodes the subset of linear constraints from §3.4 that can be tractably enforced in the dynamic program. In effect, $p(\mathbf{z} \mid x, t, \ell, f) = 0$ for any $\mathbf{z}$ that violates the constraints. We estimate the parameters of this globally normalized model by maximizing the regularized conditional likelihood of

the training set, using the standard forward-backward algorithm on the dynamic program lattice to compute the required normalizer and feature expectations.

There have been several studies of the use of constrained MAP inference for semantic role labeling on top of the predictions of local classifiers (Tromble and Eisner, 2006; Punyakanok et al., 2008; Das et al., 2012), as well as on ensembles for combining the predictions of separate systems using integer linear programming (Surdeanu et al., 2007; Punyakanok et al., 2008).[8] Meza-Ruiz and Riedel (2009) further used a Markov Logic Network formulation to incorporate a subset of these constraints during learning. Another popular approach has been to apply a reranking model, which can incorporate soft structural constraints in the form of features, on top of the k-best output of local classifiers (Toutanova et al., 2008; Johansson and Nugues, 2008). However, none of these methods provide any means to perform efficient marginal inference and this work is the first to use a globally normalized probabilistic model with structural constraints for this task.

6 Empirical Study

We next present our experimental setup, datasets used, preprocessing details and empirical results.

6.1 Datasets and Evaluation

We measure experimental results on three datasets. First, we use the CoNLL 2005 shared task data annotated according to PropBank conventions with the standard training, development and test splits (Carreras and Màrquez, 2005). These were originally constructed from sections 02-21, section 24 and section 23 of the Wall Street Journal (WSJ) portion of the Penn Treebank (Marcus et al., 1993). The PropBank I resource was used to construct the verb frame lexicon for the CoNLL 2005 experiments.

Second, we perform experiments on a substantially larger data set annotated according to PropBank conventions, using the recent OntoNotes 5.0 corpus (Weischedel et al., 2011), with the CoNLL 2012 training, development and test splits from Pradhan et al. (2013). The frame lexicon for these experiments is derived from the OntoNotes frame files. This corpus consists of nominal predicate-argument structure annotations in addition to verbs. Specifically, we use version 12 downloaded from `http://cemantix.org/data/ontonotes.html`, for which some errors from the initial release used by Pradhan et al. (2013) have been corrected.

Finally, we present results on FrameNet-annotated data, where our setup mirrors that of Hermann et al. (2014), who used the full-text annotations of the FrameNet 1.5 release.[9] We use the same training, development and test splits as Hermann et al., which consists of 39, 16 and 23 documents, respectively.

For evaluation on PropBank, we use the script from the CoNLL 2005 shared task that measures role labeling precision, recall and F1-score, as well as the full argument structure accuracy.[10] In the FrameNet setting, we use a reimplementation of the SemEval 2007 shared task evaluation script that measures joint frame-argument precision, recall and F1-score (Baker et al., 2007). For consistency, we use a stricter measure of full structure accuracy than with PropBank that gives credit only when both the predicted frame and all of its arguments are correct.

The statistical significance of the observed differences between our different models is assessed with a paired bootstrap test (Efron and Tibshirani, 1994), using 1000 bootstrap samples. For brevity, we only provide the p-values for the difference between our best and second best models on the test set, as well as between our second and third best models.

6.2 Preprocessing

All corpora were preprocessed with a part-of-speech tagger and a syntactic dependency parser, both of which were trained on the CoNLL 2012 training split extracted from OntoNotes 5.0 (Pradhan et al., 2013); this training data has no overlap with any of the development or test corpora used in our experiments. The constituency trees in OntoNotes were converted to Stanford dependencies before training our parser (de Marneffe and Manning, 2013).

The part-of-speech tagger employs a second-order conditional random field (Lafferty et al., 2001) with the following features. **Emission features**: bias, the

[8] While the dynamic program in §4 could be used to efficiently implement such ensembles, since it solves the equivalent ILP, our focus in this work is on learning a single accurate model.

[9] `http://framenet.icsi.berkeley.edu`.

[10] `http://www.lsi.upc.edu/~srlconll/srl-eval.pl`

word, the cluster of the word, suffixes of lengths 1 to 4, the capitalization shape of the word, whether the word contains a hyphen and the identity of the last word in the sentence. **Transition features**: the tag bigram, the tag bigram conjoined with, respectively, the clusters of the current and the previous words, the tag trigram and the tag trigram conjoined with, respectively, the clusters of the current and previous word, as well as with the word two positions back.

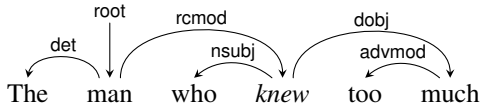
For syntactic dependencies, we use the parser and features described by Zhang and McDonald (2014), which exploits structural diversity in cube-pruning to improve higher-order graph-based inference. On the WSJ development set (section 22), the labeled attachment score of the parser is 90.9% while the part-of-speech tagger achieves an accuracy of 97.2% on the same dataset. On the OntoNotes development set, the corresponding scores are 90.2% and 97.3%.

Both the tagger and the parser, as well as the frame identification and role labeling models (see Tables 1 and 2), have features based on word clusters. Specifically, we use the clusters with 1000 classes described by Turian et al. (2010), which are induced with the Brown algorithm (Brown et al., 1992).

6.3 Candidate Argument Extraction

We use a rule-based heuristic to extract candidate arguments for role labeling. Most prior work on PropBank-style semantic role labeling have relied on constituency syntax for candidate argument extraction. Instead, we rely on dependency syntax, which allows faster preprocessing and potential extension to the many languages for which only dependency annotations are available. To this end, we adapt the constituency-based candidate argument extraction method of Xue and Palmer (2004) to dependencies.

In gold PropBank annotations, syntactic constituents serve as arguments in all constructions. However, extracting constituents from a dependency tree is not straightforward. The full dependency subtree under a particular head word often merges syntactic constituents. For example, in the tree fragment



the dependency tree has the full clause as the subtree headed by *man*, making it non-trivial to extract a partial subtree underneath it that could serve as a valid argument (for example, *The man*).

In our candidate argument extraction algorithm, first, we select all the children subtrees of a given predicate as potential arguments; if a child word is connected via the conj (conjunction) or the prep (preposition) label, we also select the corresponding grand-children subtrees. Next, we climb up to the predicate's syntactic parent and add any partial subtrees headed by it that could serve as constituents in the corresponding phrase-structure tree. To capture such constructions, we select partial subtrees for a head word by first adding the head word, then adding contiguous child subtrees from the head word's rightmost left child towards the leftmost left child until we either reach the predicate word or an *offensive* dependency label.[11] This procedure is then symmetrically applied to the head word's right children. Once a partial subtree has been added, we add the parent word's children subtrees — and potentially grandchildren subtrees in case of children labeled as conj or prep — to the candidate list, akin to the first step. We apply this parent operation recursively for all the ancestors of the predicate. Finally, we consider the predicate's syntactic parent word as a candidate argument if the predicate is connected to it via the amod label.

The candidates are further filtered to only keep those where the role of the argument, conjoined with the path from its head to the predicate, has been observed in the training data. This algorithm obtains an unlabeled argument recall of 88.2% on the OntoNotes 5.0 development data, with a precision of 38.2%.

For FrameNet, we use the extraction method of Hermann et al. (2014, §5.4), which is also inspired by Xue and Palmer (2004). On the FrameNet development data, this method obtains an unlabeled argument recall of 72.6%, with a precision of 25.1%.[12]

6.4 Baseline Systems

We compare our local and structured models to the top performing constituency-based systems from the

[11] All but the following labels are treated as offensive: advmod, amod, appos, aux, auxpass, cc, conj, dep, det, mwe, neg, nn, npadvmod, num, number, poss, preconj, predet, prep, prt, ps, quantmod and tmod.

[12] The low recall on FrameNet suggests that a deeper analysis of missed arguments is necessary. However, to allow a fair comparison with prior work, we leave this for future work.

	Development				WSJ Test				Brown Test			
Method	Prec.	Recall	F1	Comp.	Prec.	Recall	F1	Comp.	Prec.	Recall	F1	Comp.
Local/Local	80.0	75.2	77.5	51.5	81.6	76.6	79.0	53.1	73.7	68.1	70.8**	39.1
Local/DP	**81.3**	74.8	77.9	52.4	**82.6**	76.4	79.3*	54.3*	74.0	66.8	70.2	38.4
Structured/DP	81.2	**76.2**	**78.6**	54.4	82.3	**77.6**	**79.9***	56.0*	**74.3**	**68.6**	**71.3**	**39.8**
Prior work	Prec.	Recall	F1	Comp.	Prec.	Recall	F1	Comp.	Prec.	Recall	F1	Comp.
Surdeanu	–	–	–	–	79.7	74.9	77.2	52.0	–	–	–	–
Punyakanok	–	–	–	–	77.1	75.5	76.3	–	–	–	–	–
Toutanova	–	–	77.9	**57.2**	–	–	79.7	**58.7**	–	–	67.8	39.4
Ensembles	Prec.	Recall	F1	Comp.	Prec.	Recall	F1	Comp.	Prec.	Recall	F1	Comp.
Surdeanu	–	–	–	–	87.5	74.7	80.6	51.7	81.8	61.3	70.1	34.3
Punyakanok	80.1	74.8	77.4	50.7	82.3	76.8	79.4	53.8	73.4	62.9	67.8	32.3
Toutanova	–	–	78.6	58.7	81.9	78.8	80.3	60.1	–	–	68.8	40.8

Table 3: Semantic role labeling results on the CoNLL 2005 data set. The method labels are *training/inference*. For example, Local/DP means training with the local model, but inference with the dynamic program. Bold font indicates the best system using a single model and a single parse, while the best scores among all systems are underlined. Statistical significance was assessed for F1 and Comp. on the WSJ and Brown test sets with $p < 0.01^*$ and $p < 0.05^{**}$.

literature on the CoNLL 2005 datasets. To facilitate a more nuanced comparison, we distinguish between prior work based on single systems, which use a single input parse and no model combination, and ensemble-based systems. For single systems, our first baseline is the strongest non-ensemble system presented by Surdeanu et al. (2007) that treats the SRL problem as a sequential tagging task (see §4.1 of the cited paper). Next, we consider the non-ensemble system presented by Punyakanok et al. (2008) that trains local classifiers and uses an ILP to satisfy the structural constraints; this system is most similar to our approach, but is trained locally. Finally, our third single system baseline is the model of Toutanova et al. (2008) that uses a tree structured dynamic program that assumes that all candidate spans are nested; this system relies on global features in a reranking framework (see row 2 of Figure 19 of the cited paper). These authors also report ensemble-based variants that combine the outputs of multiple SRL systems in various ways; as observed in other NLP problems, the ensemble systems outperformed the single-system counterparts, and are state of the art. To situate our models with these ensemble-based approaches, we include them in Table 3.

For the OntoNotes datasets, we compare our models to Pradhan et al. (2013), who report results with a variant of the (non-ensemble) ASSERT system (Pradhan et al., 2005). These are the only previously reported results for the SRL problem on this dataset.

Finally, for the FrameNet experiments, our baseline is the state-of-the-art system of Hermann et al. (2014), which combines a frame-identification model based on WSABIE (Weston et al., 2011) with a log-linear role labeling model.

6.5 Hyperparameters

The l_1 and l_2 regularization weights for the frame identification and role labeling models for all experiments were tuned on the OntoNotes development data. For frame identification, the regularization weights are set to 0 and 0.1, while for semantic role labeling they are set to 0.1 and 1.0, respectively.

6.6 Results

Table 3 shows our results on the CoNLL 2005 development set as well as the WSJ and Brown test sets.[13] Our structured model achieves the highest F1-score among the non-ensemble systems, outperforming even the ensemble systems on the Brown

[13] We also experimented with a parser trained only on the WSJ training set. This results in a drop in role labeling F1-score of 0.3% (absolute) averaged across models on the CoNLL 2005 development set. The corresponding drop for the structured model is 0.6%, which suggests that it benefits more from parser improvements compared to the local models.

	Development			
Method	Prec.	Recall	F1	Comp.
Local/Local	79.5	77.0	78.2	57.4
Local/DP	**80.6**	77.1	78.8	59.0
Structured/DP	80.5	**77.8**	**79.1**	**60.1**
	CoNLL 2012 Test			
Method	Prec.	Recall	F1	Comp.
Local/Local	79.8	77.7	78.7	59.5
Local/DP	80.9	77.7	79.2*	60.9*
Structured/DP	80.6	**78.2**	**79.4***	**61.8***
Pradhan	**81.3**	70.5	75.5	51.7
Pradhan (revised)	78.5	76.6	77.5	55.8

Table 4: Semantic role labeling results on the OntoNotes 5.0 development and test sets from CoNLL 2012. "Pradhan" is the *Overall* results from Table 5 of Pradhan et al. (2013). "Pradhan (revised)" are corrected results from personal communication with Pradhan et al. (see footnote 14 for details). Statistical significance was assessed for F1 and Comp. on the test set with $p < 0.01^*$.

	Development			
Method	Prec.	Recall	F1	Comp.
Local/Local	80.5	62.7	70.5	31.3
Local/DP	**80.7**	62.9	70.7	31.2
Structured/DP	79.6	64.1	**71.0**	**32.6**
Hermann	78.3	**64.5**	70.8	–
	Test			
Method	Prec.	Recall	F1	Comp.
Local/Local	75.9	64.5	69.7	32.8
Local/DP	**76.1**	64.9	70.1*	33.0
Structured/DP	75.4	65.8	**70.3****	**33.8*****
Hermann	74.3	**66.0**	69.9	–

Table 5: Full structure prediction results (joint frame identification and semantic role labeling performance) for FrameNet. All systems use the WSABIE model from Hermann et al. (2014) for the frame identification step. "Hermann" is the *Wsabie Embedding* results from Table 3 of Hermann et al. (2014). Statistical significance was assessed for F1 and Comp. on the test set with $p < 0.01^*$, $p < 0.05^{**}$ and $p < 0.075^{***}$.

test set, while performing at par on the development set. Overall, using structured learning improves recall at a slight expense of precision when compared to local learning. This leads to a higher F1-score and a substantial increase in complete argument structure accuracy (*Comp.* in the tables). The increase in recall is to be expected, since during training the structured model can rely on the constraints to eliminate some hypotheses. This has the effect of alleviating some of the label imbalance seen in the training data (recall that the model encounters roughly four times as many null roles as non-null role assignments). While the results on the WSJ test set are highly statistically significant, the small size of the Brown test set give rise to a larger variance; results here are only significant at a level of $p \approx 0.1$ for F1 and $p \approx 0.2$ for Comp.

Table 4 shows the semantic role labeling results on the OntoNotes data. We observe the same trend as we did on the CoNLL 2005 data from Table 3. Adding constraints at inference time notably improves precision at virtually no cost to recall. Structured learning additionally increases recall at a small cost to precision and yields the best results both in terms of F1- and complete analysis scores. These results are all highly statistically significant. Compared to the results of Pradhan et al. (2013), our structured model yields a 15% relative error reduction in terms of F1-score and a 20% reduction in terms of complete analysis score.[14] The frame identification accuracies on the OntoNotes development and test set are 94.5% and 94.9%, respectively, whereas Pradhan et al. (2013) report an accuracy of 92.8% on the test set; this represents almost a 30% relative error reduction.

Finally, Table 5 shows the results on the FrameNet data. While structured learning helps less here compared to the PropBank setting, our model outperforms the prior state-of-the-art model of Hermann et al. (2014) and we obtain a modest improvement in complete analysis score compared to local training. Due to the small size of the FrameNet test set, similarly to the Brown test set, we observe a larger variance across bootstrap samples, but in this case the results are statistically significant to a larger degree.

Table 6 relates the speed of the various inference algorithms to the number of constraint violations. The time is relative to local inference; it excludes the

[14] Unfortunately, these results are not strictly comparable, due to errors in the original release of the data that was used by Pradhan et al. (2013). Results with Pradhan et al.'s system on the corrected release, obtained from personal communication with Pradhan et al., are included in Table 4 as "Pradhan (revised)".

time of feature extraction and computation of $g(s, r)$, which is the same across inference methods. Similar to Tromble and Eisner (2006), for all algorithms, we first use the local solution without constraints and only apply the constraints in the case of a violation. Removing this optimization results in a slowdown across the board by a factor of about 5 and does not change the ranking of the methods. Since the structured model has identical parameterization to the local model, optimality is guaranteed even when using this scheme with the former. We report the results of two ILP solvers: SCIP[15] and Gurobi.[16] SCIP is a factor of 8 slower than Gurobi for this problem, while Gurobi is a further factor of about 4 slower than our dynamic program. The penultimate line of Table 6 shows the result of using an LP-relaxation instead of the ILP. This does not come with optimality guarantees, but is included for completeness.

Finally, when using k-best inference to satisfy the *reference roles* and non-core *continuation roles* constraints in the dynamic program (§4.4), the maximum value of k is 80 on the OntoNotes development set. Across data points for which such k-best inference is necessary, the average k is found to be 1.8. If we allow ourselves to ignore these constraints, we can avoid k-best inference and achieve a further speedup, as shown in the last line of Table 6. The heuristics of Toutanova et al. (2008) could potentially be used as an alternative way of satisfying these constraints.

		Number of constraint violations			
Method	Time	*overlap*	*unique*	*cont.*	*ref.*
ILP-SCIP	198.7	0	0	0	0
ILP-Gurobi	25.0	0	0	0	0
DP k-best	6.2	0	0	0	0
Local	1.0	162	1725	63	297
LP-Gurobi	23.0	6	0	0	0
DP no k-best	4.0	0	0	0	272

Table 6: Speed and constraint violation results on the OntoNotes 5.0 development set. Exact and approximate methods are shown above and below the line, respectively.

7 Conclusions

We described a dynamic program for constrained inference in semantic role labeling that efficiently enforces a majority of structural constraints, given potentially overlapping candidate arguments. The dynamic program provably finds the optimal solutions of a corresponding ILP and in practice requires a fraction of the computational cost compared to an highly optimized off-the-shelf ILP solver, which has typically been used for this problem. Furthermore, the dynamic program facilitates learning with a globally normalized log-linear model and provides a probabilistic measure of confidence in predictions. Empirically, we showed a four-fold speedup in inference time compared to a state-of-the-art ILP solver and by using structured learning our model outperforms all comparable non-ensemble baselines on both PropBank and FrameNet data sets.

[15] `http://scip.zib.de/`

[16] `http://www.gurobi.com/`

Acknowledgments

We thank Ryan McDonald, Emily Pitler, Slav Petrov and Fernando Pereira for their detailed comments. In particular, Ryan pointed out a simplification that improved on our original dynamic program formulation. We also thank Sameer Pradhan for his corrections to the OntoNotes data. Finally, we thank André Martins for numerous discussions on this subject and the anonymous reviewers for their insightful comments.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of ACL*.

Collin Baker, Michael Ellsworth, and Katrin Erk. 2007. Semeval-2007 task 19: Frame semantic structure extraction. In *Proceedings of SemEval*.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4).

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of CoNLL*.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of CoNLL*.

Dipanjan Das, André F. T. Martins, and Noah A. Smith. 2012. An exact dual decomposition algorithm for shallow semantic parsing with constraints. In *Proceedings of *SEM*.

Dipanjan Das, Desai Chen, André F. T. Martins, Nathan Schneider, and Noah A. Smith. 2014. Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56.

Marie-Catherine de Marneffe and Christopher D. Manning, 2013. *Stanford typed dependencies manual*.

Bradley Efron and Robert J Tibshirani. 1994. *An introduction to the bootstrap*. CRC press.

Matthew Gerber and Joyce Y. Chai. 2010. Beyond NomBank: A study of implicit arguments for nominal predicates. In *Proceedings of ACL*.

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

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

Karl Moritz Hermann, Dipanjan Das, Jason Weston, and Kuzman Ganchev. 2014. Semantic frame identification with distributed word representations. In *Proceedings of ACL*.

Liang Huang and David Chiang. 2005. Better k-best parsing. In *Proceedings of IWPT*.

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

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*.

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

Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski, and Suzanne Stevenson. 2008. Semantic role labeling: An introduction to the special issue. *Computational Linguistics*, 34(2):145–159.

André F. T. Martins, Noah A. Smith, Pedro M. Q. Aguiar, and Mário A. T. Figueiredo. 2011. Dual decomposition with many overlapping components. In *Proceedings of EMNLP*.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank project: An interim report. In *Proceedings of NAACL/HLT Workshop on Frontiers in Corpus Annotation*.

Ivan Meza-Ruiz and Sebastian Riedel. 2009. Jointly identifying predicates, arguments and senses using Markov Logic. In *Proceedings of NAACL-HLT*.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2005. Support vector learning for semantic argument classification. *Machine Learning*, 60(1-3):11–39.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Tou Hwee Ng, Anders Björkelund, Olga Uryupina, Yuchen Zhang, and Zhi Zhong. 2013. Towards robust linguistic analysis using OntoNotes. In *Proceedings of CoNLL*.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Sebastian Riedel and David A. Smith. 2010. Relaxed marginal inference and its application to dependency parsing. In *Proceedings of NAACL-HLT*.

Mihai Surdeanu, Lluís Màrquez, Xavier Carreras, and Pere R. Comas. 2007. Combination strategies for semantic role labeling. *Journal of Artificial Intelligence Research*, 29(1):105–151.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of CoNLL*.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2008. A global joint model for semantic role labeling. *Computational Linguistics*, 34(2):161–191.

Roy W. Tromble and Jason Eisner. 2006. A fast finite-state relaxation method for enforcing global constraints on sequence decoding. In *Proceedings of NAACL-HLT*.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of ACL*.

Ralph Weischedel, Eduard Hovy, Martha Palmer, Mitch Marcus, Robert Belvin, Sameer Pradhan, Lance Ramshaw, and Nianwen Xue. 2011. OntoNotes: A large training corpus for enhanced processing. In J. Olive, C. Christianson, and J. McCary, editors, *Handbook of Natural Language Processing and Machine Translation*. Springer.

Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. WSABIE: Scaling up to large vocabulary image annotation. In *Proceedings of IJCAI*.

Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of EMNLP*.

Hao Zhang and Ryan McDonald. 2014. Enforcing structural diversity in cube-pruned dependency parsing. In *Proceedings of ACL*.