Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Frame-Semantic Parsing

Dipanjan Das*
Google Inc.

Desai Chen**
Massachusetts Institute of Technology

André F. T. Martins†
Priberam Labs

Nathan Schneider‡
Carnegie Mellon University

Noah A. Smith§
Carnegie Mellon University

Frame semantics (Fillmore 1982) is a linguistic theory that has been instantiated for English in the FrameNet lexicon (Fillmore, Johnson, and Petruck 2003). We solve the problem of frame-semantic ***parsing*** *using a two-stage statistical model that takes lexical targets (i.e., content words and phrases) in their sentential contexts and predicts frame-semantic structures. Given a target in context, the first stage disambiguates it to a semantic frame. This model employs latent variables and semi-supervised learning to improve frame disambiguation for targets unseen at training time. The second stage finds the target's locally expressed semantic arguments. At inference time, a fast exact dual decomposition algorithm collectively predicts all the arguments of a frame at once in order to respect declaratively stated linguistic constraints, resulting in qualitatively better structures than naïve local predictors. Both components are feature-based and discriminatively trained on a small set of annotated frame-semantic parses. On the SemEval 2007 benchmark dataset, the approach, along with a heuristic identifier of frame-evoking targets, outperforms the prior state of the art by significant margins. Additionally, we present experiments on the much larger FrameNet 1.5 dataset. We have released our frame-semantic parser as open-source software.*

1. Introduction

FrameNet (Fillmore, Johnson, and Petruck 2003) is a linguistic resource storing considerable information about lexical and predicate-argument semantics in English. Grounded

* Google Inc., New York, NY 10011. E-mail: dipanjand@google.com.
** Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139. Email: desaic@csail.mit.edu.
† Alameda D. Afonso Henriques, 41 - 2.° Andar, 1000-123, Lisboa, Portugal. Email: afm@cs.cmu.edu.
‡ School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213. Email: nschneid@cs.cmu.edu.
§ School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213. Email: nasmith@cs.cmu.edu.

Submission received: 4 May, 2012; revised submission received: 10 November, 2012; accepted for publication: 22 December, 2012.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

in the theory of frame semantics (Fillmore 1982), it suggests—but does not formally define—a semantic representation that blends representations familiar from word-sense disambiguation (Ide and Véronis 1998) and semantic role labeling (Gildea and Jurafsky 2002). Given the limited size of available resources, accurately producing richly structured frame-semantic structures with high coverage will require data-driven techniques beyond simple supervised classification, such as latent variable modeling, semi-supervised learning, and joint inference.

In this article, we present a computational and statistical model for frame-semantic parsing, the problem of extracting from text semantic predicate-argument structures such as those shown in Figure 1. We aim to predict a frame-semantic representation with two statistical models rather than a collection of local classifiers, unlike earlier approaches (Baker, Ellsworth, and Erk 2007). We use a probabilistic framework that cleanly integrates the FrameNet lexicon and limited available training data. The probabilistic framework we adopt is highly amenable to future extension through new features, more relaxed independence assumptions, and additional semi-supervised models.

Carefully constructed lexical resources and annotated datasets from FrameNet, detailed in Section 3, form the basis of the frame structure prediction task. We decompose this task into three subproblems: *target identification* (Section 4), in which frame-evoking predicates are marked in the sentence; *frame identification* (Section 5), in which the evoked frame is selected for each predicate; and *argument identification* (Section 6), in which arguments to each frame are identified and labeled with a role from that frame. Experiments demonstrating favorable performance to the previous state of the art on SemEval 2007 and FrameNet datasets are described in each section. Some novel aspects of our approach include a latent-variable model (Section 5.2) and a semi-supervised extension of the predicate lexicon (Section 5.5) to facilitate disambiguation of words not

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

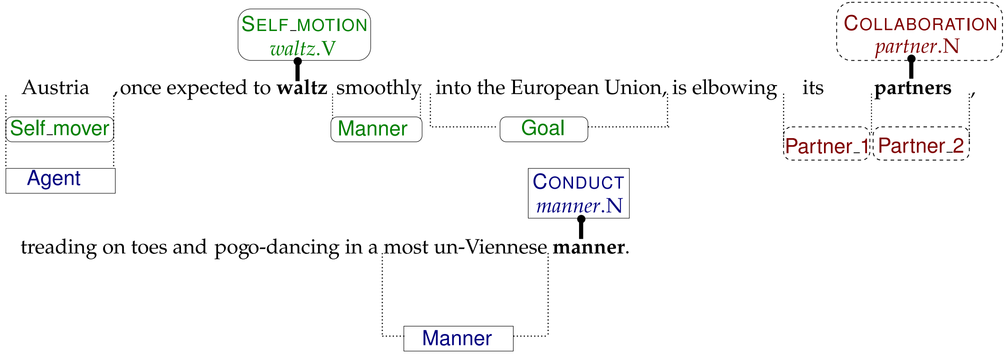

SELF_MOTION
waltz.V
COLLABORATION
partner.N
Austria
, once expected to **waltz** smoothly into the European Union, is elbowing its **partners** ,
Self_mover
Manner
Goal
Partner_1
Partner_2
Agent
CONDUCT
manner.N
treading on toes and pogo-dancing in a most un-Viennese **manner**.
Manner

Figure 1: An example sentence from the annotations released as part of FrameNet 1.5 with three targets marked in bold. Note that this annotation is partial because not all potential targets have been annotated with predicate-argument structures. Each target has its evoked semantic frame marked above it, enclosed in a distinct shape or border style. For each frame, its semantic roles are shown enclosed within the same shape or border style, and the spans fulfilling the roles are connected to the latter using dotted lines. For example, **manner** evokes the CONDUCT frame, and has the Agent and Manner roles fulfilled by “Austria” and “most un-Viennese,” respectively.

in the FrameNet lexicon; a unified model for finding and labeling arguments (Section 6) that diverges from prior work in semantic role labeling; and an exact dual decomposition algorithm (Section 7) that collectively predicts all the arguments of a frame together, thereby incorporating linguistic constraints in a principled fashion.

Our open-source parser, named SEMAFOR (Semantic Analyzer of Frame Representations)[1] achieves the best published results to date on the SemEval 2007 frame-semantic structure extraction task (Baker, Ellsworth, and Erk 2007). Herein, we also present results on newly released data with FrameNet 1.5, the latest edition of the lexicon. Some of the material presented in this article has appeared in previously published conference papers: Das et al. (2010) presented the basic model, Das and Smith (2011) described semi-supervised lexicon expansion, Das and Smith (2012) demonstrated a

[1] See http://www.ark.cs.cmu.edu/SEMAFOR.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

sparse variant of lexicon expansion, and Das, Martins, and Smith (2012) presented the dual decomposition algorithm for constrained joint argument identification. We present here a synthesis of those results and several additional details:

1. The set of features used in the two statistical models for frame identification and argument identification.
2. Details of a greedy beam search algorithm for argument identification that avoids illegal argument overlap.
3. Error analysis pertaining to the dual decomposition argument identification algorithm, in contrast with the beam search algorithm.
4. Results on full frame-semantic parsing using graph-based semi-supervised learning with sparsity-inducing penalties; this expands the small FrameNet predicate lexicon, enabling us to handle unknown predicates.

Our primary contributions are the use of efficient structured prediction techniques suited to shallow semantic parsing problems, novel methods in semi-supervised learning that improve the lexical coverage of our parser and making frame-semantic structures a viable computational semantic representation usable in other language technologies. To set the stage, we next consider related work in the automatic prediction of predicate-argument semantic structures.

2. Related Work

In this section, we will focus on previous scientific work relevant to the problem of frame-semantic parsing. First, we will briefly discuss work done on PropBank-style semantic role labeling, following which we will concentrate on the more relevant problem of frame-semantic structure extraction. Next, we review previous work that has used

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

semi-supervised learning for shallow semantic parsing. Finally, we discuss prior work on joint structure prediction relevant to frame-semantic parsing.

2.1 Semantic Role Labeling

Since Gildea and Jurafsky (2002) pioneered statistical semantic role labeling, there has been a great deal of computational work using predicate-argument structures for semantics. The development of PropBank (Kingsbury and Palmer 2002), followed by CoNLL shared tasks on semantic role labeling (Carreras and Màrquez 2004, 2005) boosted research in this area. Figure 2(a) shows an annotation from PropBank. PropBank annotations are closely tied to syntax, because the dataset consists of the phrase-structure syntax trees from the *Wall Street Journal* section of the Penn Treebank (Marcus, Marcinkiewicz, and Santorini 1993) annotated with predicate-argument structures for verbs. In Figure 2(a), the syntax tree for the sentence is marked with various semantic roles. The two main verbs in the sentence, "created" and "pushed," are the predicates. For the former, the constituent "more than 1.2 million jobs" serves as the semantic role ARG1 and the constituent "In that time" serves as the role ARGM-TMP. Similarly for the latter verb, roles ARG1, ARG2, ARGM-DIR and ARGM-TMP are shown in the figure. PropBank defines *core roles* ARG0 through ARG5, which receive different interpretations for different predicates. Additional *modifier roles* ARGM-* include ARGM-TMP (temporal) and ARGM-DIR (directional), as shown in Figure 2(a). The PropBank representation therefore has a small number of roles, and the training dataset comprises some 40,000 sentences, thus making the semantic role labeling task an attractive one from the perspective of machine learning.

There are many instances of influential work on semantic role labeling using PropBank conventions. Pradhan et al. (2004) present a system that uses SVMs to identify the arguments in a syntax tree that can serve as semantic roles, followed by classification

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163
© Association for Computational Linguistics

of the identified arguments to role names via a collection of binary SVMs. Punyakanok et al. (2004) describe a semantic role labeler that uses integer linear programming for inference and uses several global constraints to find the best suited predicate-argument structures. Joint modeling for semantic role labeling with discriminative log-linear models is presented by Toutanova, Haghighi, and Manning (2005), where global features looking at all arguments of a particular verb together are incorporated into in a dynamic programming and reranking framework. The *Computational Linguistics* special issue on semantic role labeling (Màrquez et al. 2008) includes other interesting papers on the topic, leveraging the PropBank conventions for labeling shallow semantic structures. Recently, there have been initiatives to predict syntactic dependencies as well as PropBank-style predicate-argument structures together using one joint model (Surdeanu et al. 2008; Hajič et al. 2009).

Here, we focus on the related problem of frame-semantic parsing. Note from the annotated semantic roles for the two verbs in the sentence of Figure 2(a) that it is unclear what the core roles ARG1 or ARG2 represent linguistically. To better understand the roles' meaning for a given verb, one has to refer to a verb-specific file provided along with the PropBank corpus. Although collapsing these verb-specific core roles into tags ARG0-ARG5 leads to a small set of classes to be learned from a reasonable sized corpus, analysis shows that the roles ARG2–ARG5 serve many different purposes for different verbs. Yi, Loper, and Palmer (2007) point out that these four roles are highly overloaded and inconsistent, and they mapped them to VerbNet (Schuler 2005) thematic roles to get improvements on the SRL task. Recently, Bauer and Rambow (2011) presented a method to improve the syntactic subcategorization patterns for FrameNet lexical units using VerbNet. Instead of working with PropBank, we focus on shallow semantic parsing of sentences in the paradigm of frame semantics (Fillmore 1982), to which we turn next.

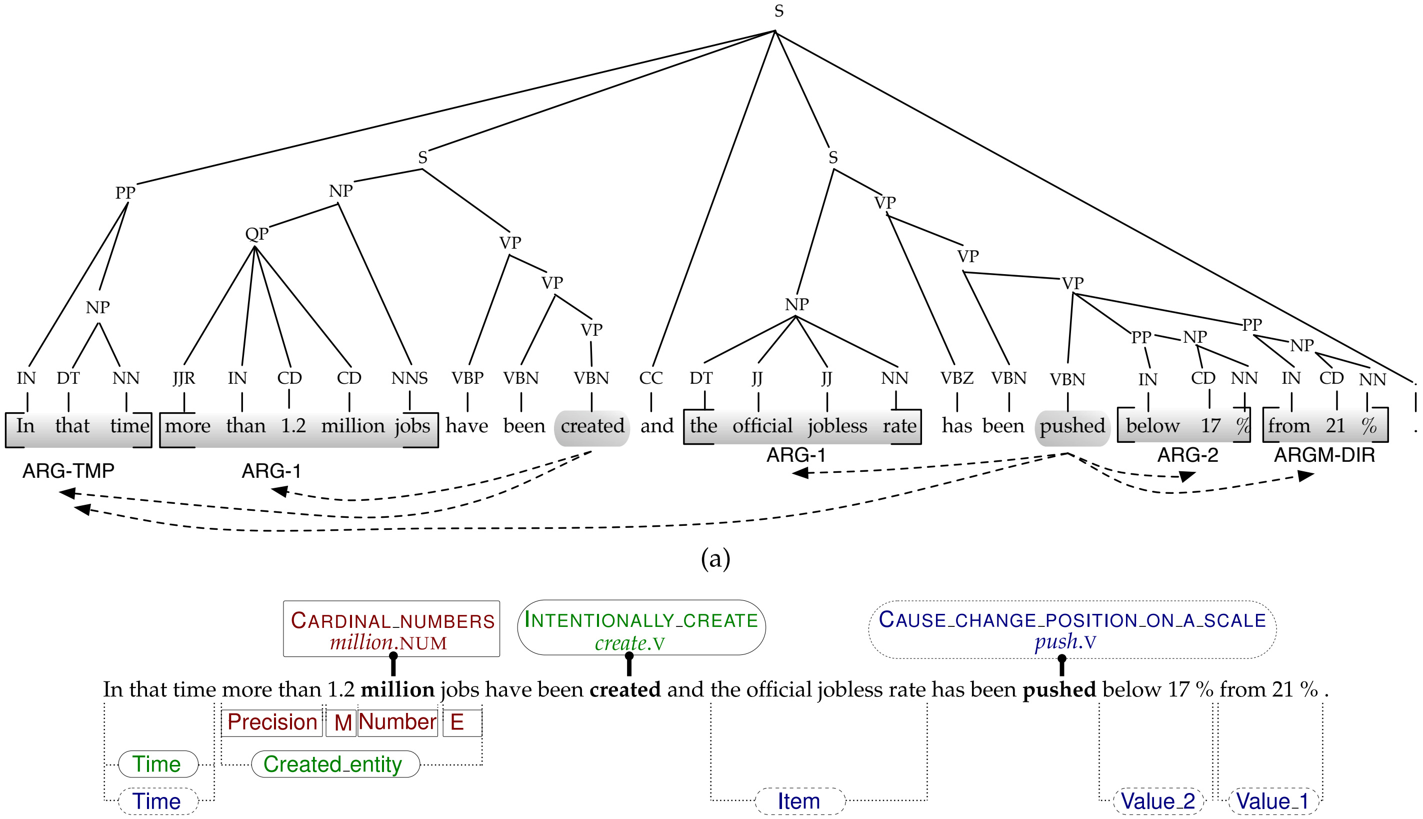
Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

2.2 Frame-Semantic Parsing

The FrameNet lexicon (Fillmore, Johnson, and Petruck 2003) contains rich linguistic information about lexical items and predicate-argument structures. A semantic frame present in this lexicon includes a list of *lexical units*, which are associated words and phrases that can potentially evoke it in a natural language utterance. Each frame in the lexicon also enumerates several *roles* corresponding to facets of the scenario represented by the frame. In a frame-analyzed sentence, predicates evoking frames are known as *targets*, and a word or phrase filling a role is known as an *argument*. Figure 2(b) shows frame-semantic annotations for the same sentence as in Figure 2(a). (In the figure, for example, the CARDINAL_NUMBERS frame, "M" denotes the role Multiplier and "E" denotes the role Entity.) Note that the verbs "created" and "pushed" evoke the frames INTENTIONALLY_CREATE and CAUSE_CHANGE_POSITION_ON_A_SCALE, respectively. The corresponding lexical units[2] from the FrameNet lexicon, *create*.V and *push*.V, are also shown. The PropBank analysis in Figure 2(a) also has annotations for these two verbs. While PropBank labels the roles of these verbs with its limited set of tags, the frame-semantic parse labels each frame's arguments with frame-specific roles shown in the figure, making it immediately clear what those arguments mean. For example, for the INTENTIONALLY_CREATE frame, "more than 1.2 million jobs" is the Created_entity, and "In that time" is the Time when the jobs were created. FrameNet also allows *non-verbal* words and phrases to evoke semantic frames: in this sentence, "million" evokes the frame CARDINAL_NUMBERS and doubles as its Number argument, with "1.2" as Multiplier, "jobs" as the Entity being quantified, and "more than" as the Precision of the quantity expression.

[2] See Section 5.1 for a detailed description of lexical units.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163
© Association for Computational Linguistics


S
PP
NP
QP
VP
IN DT NN JJR IN CD CD NNS VBP VBN VBN CC DT JJ JJ NN VBZ VBN VBN IN CD NN IN CD NN .
In that time more than 1.2 million jobs have been created and the official jobless rate has been pushed below 17 % from 21 % .
ARG-TMP
ARG-1
ARG-1
ARG-2
ARGM-DIR
CARDINAL_NUMBERS
million.NUM
INTENTIONALLY_CREATE
create.V
CAUSE_CHANGE_POSITION_ON_A_SCALE
push.V
Precision
M
Number
E
Time
Created_entity
Time
Item
Value_2
Value_1

(a)

(b)

Figure 2: (a) A phrase-structure tree taken from the Penn Treebank and annotated with PropBank predicate-argument structures. The verbs "created" and "pushed" serve as predicates in this sentence. Dotted arrows connect each predicate to its semantic arguments (bracketed phrases). (b) A partial depiction of frame-semantic structures for the same sentence. The words in bold are *targets*, which instantiate a (lemmatized and part-of-speech–tagged) *lexical unit* and evoke a semantic frame. Every frame annotation is shown enclosed in a distint shape or border style, and its argument labels are shown together on the same vertical tier below the sentence. See text for explanation of abbreviatons.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

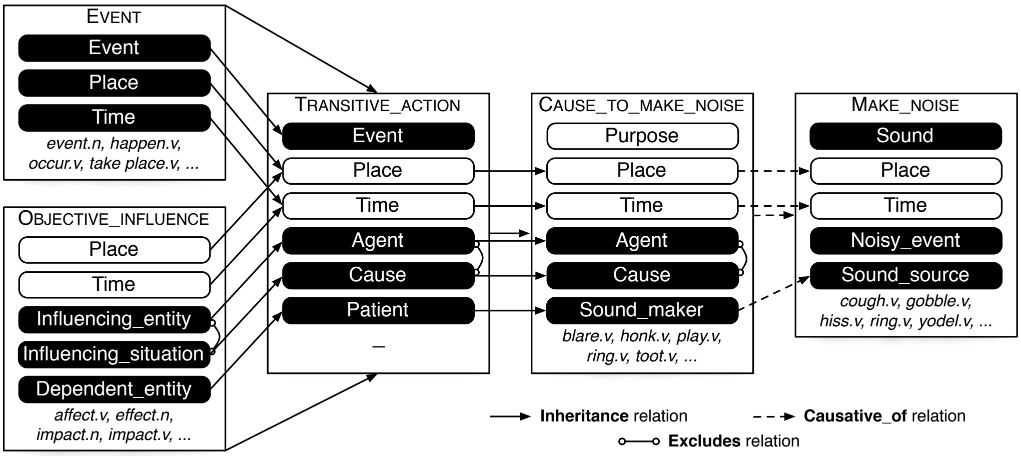


Figure 3: Partial illustration of frames, roles, and lexical units related to the CAUSE_TO_MAKE_NOISE frame, from the FrameNet lexicon. "Core" roles are filled bars. Non-core roles (such as Place and Time) as unfilled bars. No particular significance is ascribed to the ordering of a frame's roles in its lexicon entry (the selection and ordering of roles above is for illustrative convenience). CAUSE_TO_MAKE_NOISE defines a total of 14 roles, many of them not shown here.

Whereas PropBank contains verbal predicates and NomBank (Meyers et al. 2004) contains nominal predicates, FrameNet counts these as well as allowing adjectives, adverbs and prepositions among its lexical units. Finally, FrameNet frames organize predicates according to semantic principles, both by allowing related terms to evoke a common frame (e.g. *push*.V, *raise*.V and *growth*.N for CAUSE_CHANGE_POSITION_ON_A_SCALE) and by defining frames and their roles within a hierarchy (see Figure 3). PropBank does not explicitly encode relationships among predicates.

Most early work on frame-semantic parsing has made use of the *exemplar* sentences in the FrameNet corpus (see Section 3.1), each of which is annotated for a single frame and its arguments. Gildea and Jurafsky (2002) presented a discriminative model for arguments given the frame; Thompson, Levy, and Manning (2003) used a generative model for both the frame and its arguments. Fleischman, Kwon, and Hovy (2003) first used maximum entropy models to find and label arguments given the frame. Shi and

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Mihalcea (2004) developed a rule-based system to predict frames and their arguments in text, and Erk and Padó (2006) introduced the Shalmaneser tool, which employs Naïve Bayes classifiers to do the same. Other FrameNet SRL systems (Giuglea and Moschitti 2006, for instance) have used SVMs. Most of this work was done on an older, smaller version of FrameNet, containing around 300 frames and fewer than 500 unique semantic roles. Unlike this body of work, we experimented with the larger SemEval 2007 shared task dataset, and also the newer FrameNet 1.5,[3] which lists 877 frames and 1068 role types—thus handling many more labels, and resulting in richer frame-semantic parses.

Recent work in frame-semantic parsing—in which sentences may contain multiple frames which need to be recognized along with their arguments—was undertaken as the SemEval 2007 task 19 of frame-semantic structure extraction (Baker, Ellsworth, and Erk 2007). This task leveraged FrameNet 1.3, and also released a small corpus containing a little more than 2000 sentences with full text annotations. The LTH system of Johansson and Nugues (2007), which we use as our baseline (Section 3.4), had the best performance in the SemEval 2007 task in terms of full frame-semantic parsing. Johansson and Nugues (2007) broke down the task as identifying targets that could evoke frames in a sentence, identifying the correct semantic frame for a target, and finally determining the arguments that fill the semantic roles of a frame. They used a series of SVMs to classify the frames for a given target, associating unseen lexical items to frames and identifying and classifying token spans as various semantic roles. Both the full text annotation corpus as well as the FrameNet exemplar sentences were used to train their models. Unlike Johansson and Nugues, we use *only* the full text annotated sentences as training data, model the whole problem with only two statistical models, and obtain significantly better overall parsing scores. We also model the

[3] Available at `http://framenet.icsi.berkeley.edu` as of January 19, 2013.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

argument identification problem using a joint structure prediction model and use semi-supervised learning to improve predicate coverage. We also present experiments on recently released FrameNet 1.5 data.

In other work based on FrameNet, Matsubayashi, Okazaki, and Tsujii (2009) investigated various uses of FrameNet's taxonomic relations for learning generalizations over roles; they trained a log-linear model on the SemEval 2007 data to evaluate features for the subtask of argument identification. Another line of work has sought to extend the coverage of FrameNet by exploiting VerbNet and WordNet (Shi and Mihalcea 2005; Giuglea and Moschitti 2006; Pennacchiotti et al. 2008) and by projecting entries and annotations within and across languages (Boas 2002; Fung and Chen 2004; Pado and Lapata 2005; Fürstenau and Lapata 2009b). Others have explored the application of frame-semantic structures to tasks such as information extraction (Moschitti, Morarescu, and Harabagiu 2003; Surdeanu et al. 2003), textual entailment (Burchardt and Frank 2006; Burchardt et al. 2009), question answering (Narayanan and Harabagiu 2004; Shen and Lapata 2007), and paraphrase recognition (Padó and Erk 2005).

2.3 Semi-Supervised Methods

Although there has been a significant amount of work in supervised shallow semantic parsing using both PropBank- and FrameNet-style representations, a few improvements over vanilla supervised methods using unlabeled data are notable. Fürstenau and Lapata (2009b) present a method of projecting predicate-argument structures from some seed examples to unlabeled sentences, and use a linear program formulation to find the best alignment explaining the projection. Next, the projected information as well as the seeds are used to train statistical model(s) for SRL. The authors ran experiments using a set of randomly chosen verbs from the exemplar sentences of FrameNet and found improvements over supervised methods. In an extension to this work, Fürstenau

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

	SemEval 2007 Data	**FrameNet 1.5 Release**
	count	*count*
Exemplar sentences	139,439	154,607
Frame labels (types)	665	877
Role labels (types)	720	1,068
Sentences in training data	2,198	3,256
Targets in training data	11,195	19,582
Sentences in test data	120	2,420
Targets in test data	1,059	4,458
Unseen targets in test data	210	144

Table 1: Salient statistics of the datasets used in our experiments. There is a significant overlap between the two datasets.

and Lapata (2009a) present a method for finding examples for unseen verbs using a graph alignment method; this method represents sentences and their syntactic analysis as graphs and graph alignment is used to project annotations from seed examples to unlabeled sentences. This alignment problem is again modeled as a linear program. Fürstenau and Lapata (2012) present an detailed expansion of the aforementioned papers. Although this line of work presents a novel direction in the area of SRL, the published approach does not yet deal with non-verbal predicates and does not evaluate the presented methods on the full text annotations of the FrameNet releases.

Deschacht and Moens (2009) present a technique of incorporating additional information from unlabeled data by using a latent words language model. Latent variables are used to model the underlying representation of words, and parameters of this model are estimated using standard unsupervised methods. Next, the latent information is used as features for an SRL model. Improvements over supervised SRL techniques are observed with the augmentation of these extra features. The authors also compare their method with the aforementioned two methods of Fürstenau and Lapata (2009a, 2009b) and show relative improvements. Experiments are performed on the CoNLL 2008 shared task dataset (Surdeanu et al. 2008), which follows the PropBank conventions

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

and only labels verbal and nominal predicates—in contrast to our work, which includes most lexicosyntactic categories. A similar approach is presented by Weston, Ratle, and Collobert (2008) who use neural embeddings of words, which are eventually used for SRL; improvements over state-of-the-art PropBank-style SRL systems are observed.

Recently, there has been related work in unsupervised semantic role labeling (Lang and Lapata 2010, 2011; Titov and Klementiev 2012) that attempts to induce semantic roles automatically from unannotated data. This line of work may be useful in discovering new semantic frames and roles, but here we stick to the concrete representation provided in FrameNet, without seeking to expand its inventory of semantic types. We present a new semi-supervised technique to expand the set of lexical items with the potential semantic frames that they could evoke; we use a graph-based semi-supervised learning framework to achieve this goal (Section 5.5).

2.4 Joint Inference and Shallow Semantic Parsing

Most high-performance SRL systems that use conventions from PropBank (Kingsbury and Palmer 2002) and NomBank (Meyers et al. 2004) employ joint inference for semantic role labeling (Màrquez et al. 2008). To our knowledge, the separate line of work investigating frame-semantic parsing has not previously dealt with joint inference. A common trait in prior work, both in PropBank and FrameNet conventions, has been the use of a two-stage model that identifies arguments first, then labels them, often using dynamic programming or integer linear programs (ILPs); we treat both problems together here.[4]

Recent work in NLP problems has focused on ILP formulations for complex structure prediction tasks like dependency parsing (Riedel and Clarke 2006; Martins, Smith,

[4] In prior work, there are exceptions where identification and classification of arguments have been treated in one step; for more details, please refer to the systems participating in the CoNLL-2004 shared task on semantic role labeling (Carreras and Màrquez 2004).

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

and Xing 2009; Martins et al. 2010), sequence tagging (Roth and Yih 2004) as well as PropBank SRL (Punyakanok et al. 2004). While early work in this area focused on declarative formulations tackled with off-the-shelf solvers, Rush et al. (2010) proposed subgradient-based dual decomposition (also called Lagrangian relaxation) as a way of exploiting the structure of the problem and existing combinatorial algorithms. The method allows the combination of models which are individually tractable, but not jointly tractable, by solving a relaxation of the original problem. Since then, dual decomposition has been used to build more accurate models for dependency parsing (Koo et al. 2010), CCG supertagging and parsing (Auli and Lopez 2011) and machine translation (DeNero and Macherey 2011; Rush and Collins 2011; Chang and Collins 2011).

Recently, Martins et al. (2011b) showed that the success of subgradient-based dual decomposition strongly relies on breaking down the original problem into a "good" decomposition, *i.e.*, one with few overlapping components. This leaves out many declarative constrained problems, for which such a good decomposition is not readily available. For those, Martins et al. (2011b) proposed the *Alternating Directions Dual Decomposition* (AD^3) algorithm, which retains the modularity of previous methods, but can handle thousands of small overlapping components. We adopt that algorithm as it perfectly suits the problem of argument identification, as we observe in Section 7.[5] We also contribute an exact branch-and-bound technique wrapped around AD^3.

Before delving into the details of our modeling framework, we describe in detail the structure of the FrameNet lexicon and the datasets used to train our models.

[5] AD^3 was previously referred to as "DD-ADMM," in reference to the use of dual decomposition with the alternating directions method of multipliers.

	targets			**arguments**	
	count	%		*count*	%
Noun	5,155	52	Noun	9,439	55
Verb	2,785	28	Preposition or		
Adjective	1,411	14	complementizer	2,553	15
Preposition	296	3	Adjective	1,744	10
Adverb	103	1	Verb	1,156	7
Number	63	1	Pronoun	736	4
Conjunction	8		Adverb	373	2
Article	3		Other	1047	6
	9,824			17,048	

Table 2: Breakdown of targets and arguments in the SemEval 2007 training set in terms of part of speech. The target POS is based on the LU annotation for the frame instance. For arguments, this reflects the part of speech of the head word (estimated from an automatic dependency parse); the percentage is out of all overt arguments.

3. Resources and Task

We consider frame-semantic parsing resources consisting of a lexicon and annotated sentences with frame-semantic structures, evaluation strategies and previous baselines.

3.1 FrameNet Lexicon

The FrameNet lexicon is a taxonomy of manually identified general-purpose semantic **frames** for English.[6] Listed in the lexicon with each frame are a set of lemmas (with parts of speech) that can denote the frame or some aspect of it—these are called **lexical units** (LUs). In a sentence, word or phrase tokens that evoke a frame are known as **targets**. The set of LUs listed for a frame in FrameNet may not be exhaustive; we may see a target in new data that does not correspond to an LU for the frame it evokes. Each frame definition also includes a set of frame elements, or **roles**, corresponding to different aspects of the concept represented by the frame, such as participants, props, and attributes. We use the term **argument** to refer to a sequence of word tokens annotated as filling a frame

[6] Like the SemEval 2007 participants, we used FrameNet 1.3 and also the newer version of the lexicon, FrameNet 1.5 (`http://framenet.icsi.berkeley.edu`).

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

role. Figure 1 shows an example sentence from the training data with annotated targets, LUs, frames, and role-argument pairs. The FrameNet lexicon also provides information about relations between frames and between roles (e.g., INHERITANCE). Figure 3 shows a subset of the relations between five frames and their roles.

Accompanying most frame definitions in the FrameNet lexicon is a set of lexicographic **exemplar sentences** (primarily from the British National Corpus) annotated for that frame. Typically chosen to illustrate variation in argument realization patterns for the frame in question, these sentences only contain annotations for a single frame.

In preliminary experiments, we found that using exemplar sentences directly to train our models hurt performance as evaluated on SemEval 2007 data, which formed a benchmark for comparison with previous state of the art. This was a noteworthy observation, given that the number of exemplar sentences is an order of magnitude larger than the number of sentences in training data that we consider in our experiments (Section 3.2). This is presumably because the exemplars are not representative as a sample, do not have complete annotations, and are not from a domain similar to the test data. Instead, we make use of these exemplars in the construction of features (Section 5.2).

3.2 Data

In our experiments on frame-semantic parsing, we use two sets of data:

1. **SemEval 2007 data:** In benchmark experiments for comparison with previous state of the art, we use a dataset that was released as part of the **SemEval 2007 shared task** on frame-semantic structure extraction (Baker, Ellsworth, and Erk 2007). Full text annotations in this dataset consisted of a few thousand sentences containing multiple targets, each annotated with a frame and its arguments. The

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

then-current version of the lexicon (**FrameNet 1.3**) was employed for the shared task as the inventory of frames, roles, and lexical units (Figure 3 illustrates a small portion of the lexicon). In addition to the frame hierarchy, FrameNet 1.3 also contained 139,439 exemplar sentences containing one target each. Statistics of the data used for the SemEval 2007 shared task are given in the first column of Table 1. 665 frame types and 720 role types appear in the exemplars and the training portion of the data. We adopted the same training and test split as the SemEval 2007 shared task; however, we removed four documents from the training set[7] for development. Table 2 shows some additional information about the SemEval dataset; the variety of lexicosyntactic categories of targets stands in marked contrast with the PropBank-style SRL data and task.

2. **FrameNet 1.5 release:** A more recent version of the FrameNet lexicon was released in 2010.[8] We also test our statistical models (only frame identification and argument identification) on this dataset to get an estimate of how much improvement additional data can provide. Details of this dataset are shown in the second column of Table 1. Of the 78 documents in this release with full text annotations, we selected 55 (19,582 targets) for training and held out the remaining 23 (4,458 targets) for testing. There are fewer target annotations per sentence in the test set than the training set.[9] Das and Smith (2011, supplementary material) give the names of the test documents for fair replication of our work. We also randomly

[7] These were: StephanopoulousCrimes, Iran_Biological, NorthKorea_Introduction, and WMDNews_042106.

[8] Released on September 15, 2010, and downloadable from `http://framenet.icsi.berkeley.edu` as of February 13, 2013. In our experiments, we used a version downloaded on September 22, 2010.

[9] For creating the splits, we first included the documents that had incomplete annotations as mentioned in the initial FrameNet 1.5 data release in the test set; since we do not evaluate target identification for this version of data, the small number of targets per sentence does not matter. After these documents were put into the test set, we randomly selected 55 remaining documents for training, and picked the rest for additional testing. The final test set contains a total of 23 documents. When these documents are annotated in their entirety, the test set will become larger and the training set will be unaltered.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

selected 4,462 targets from the training data for development of the argument identification model (Section 6.1).

Preprocessing. We preprocessed sentences in our dataset with a standard set of annotations: POS tags from MXPOST (Ratnaparkhi 1996) and dependency parses from the MST parser (McDonald, Crammer, and Pereira 2005); manual syntactic parses are not available for most of the FrameNet-annotated documents. We used WordNet (Fellbaum 1998) for lemmatization. Our models treat these pieces of information as observations. We also labeled each verb in the data as having ACTIVE or PASSIVE voice, using code from the SRL system described by Johansson and Nugues (2008).

3.3 Task and Evaluation Methodology

Automatic annotations of frame-semantic structure can be broken into three parts: (1) *targets*, the words or phrases that evoke frames; (2) the *frame type*, defined in the lexicon, evoked by each target; and (3) the *arguments*, or spans of words that serve to fill roles defined by each evoked frame. These correspond to the three subtasks in our parser, each described and evaluated in turn: target identification (Section 4), frame identification (Section 5, not unlike word-sense disambiguation), and argument identification (Section 6, essentially the same as semantic role labeling).

The standard evaluation script from the SemEval 2007 shared task calculates precision, recall, and F_1-measure for frames and arguments; it also provides a score that gives partial credit for hypothesizing a frame related to the correct one. We present precision, recall, and F_1-measure microaveraged across the test documents, report *labels-only* matching scores (spans must match exactly), and do not use named entity labels.[10]

[10] For microaveraging, we concatenated all sentences of the test documents and measured precision and recall over the concatenation. Macroaveraging, on the other hand, would mean calculating these metrics for each document, then averaging them. Microaveraging treats every frame or argument as a unit, regardless of the length of the document it occurs in.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

More details can be found in the task description paper from SemEval 2007 (Baker, Ellsworth, and Erk 2007). For our experiments, statistical significance is measured using a reimplementation of Dan Bikel's randomized parsing evaluation comparator, a stratified shuffling test whose original implementation[11] is accompanied by the following description (quoted verbatim, with explanations of our use of the test given in square brackets):

> The null hypothesis is that the two models that produced the observed results are the same, such that for each test instance [here, a set of predicate-argument structures for a sentence], the two observed scores are equally likely. This null hypothesis is tested by randomly shuffling individual sentences' scores between the two models and then re-computing the evaluation metrics [precision, recall or F_1 score in our case]. If the difference in a particular metric after a shuffling is equal to or greater than the original observed difference in that metric, then a counter for that metric is incremented. Ideally, one would perform all 2^n shuffles, where n is the number of test cases (sentences), but given that this is often prohibitively expensive, the default number of iterations is 10,000 [we use independently sampled 10,000 shuffles]. After all iterations, the likelihood of incorrectly rejecting the null [hypothesis, i.e., the p-value] is simply $(nc + 1)/(nt + 1)$, where nc is the number of random differences greater than the original observed difference, and nt is the total number of iterations.

3.4 Baseline

A strong baseline for frame-semantic parsing is the system presented by Johansson and Nugues (2007, hereafter J&N'07), the best system in the SemEval 2007 shared task. That system is based on a collection of SVMs. They used a set of rules for target identification which we describe in Appendix A. For frame identification, they used an SVM classifier to disambiguate frames for known frame-evoking words. They used WordNet synsets to extend the vocabulary of frame-evoking words to cover unknown words, and then used a collection of separate SVM classifiers—one for each frame—to predict a single evoked frame for each occurrence of a word in the extended set.

J&N'07 followed Xue and Palmer (2004) in dividing the argument identification problem into two subtasks: first, they classified candidate spans as to whether they were

[11] See `http://www.cis.upenn.edu/~dbikel/software.html#comparator`.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

arguments or not; then they assigned roles to those that were identified as arguments. Both phases used SVMs. Thus, their formulation of the problem involves a multitude of independently trained classifiers that share no information—whereas ours uses two log-linear models, each with a single set of parameters shared across all contexts, to find a full frame-semantic parse.

We compare our models with J&N'07 using the benchmark dataset from SemEval 2007. However, since we are not aware of any other work using the FrameNet 1.5 full text annotations, we report our results on that dataset without comparison to any other system.

4. Target Identification

Target identification is the problem of deciding which word tokens (or word token sequences) evoke frames in a given sentence. In other semantic role labeling schemes (e.g., PropBank), simple part-of-speech criteria typically distinguish targets from non-targets. But in frame semantics, verbs, nouns, adjectives, and even prepositions can evoke frames under certain conditions. One complication is that semantically-impoverished **support predicates** (such as *make* in *make a request*) do not evoke frames in the context of a frame-evoking, syntactically dependent noun (*request*). Furthermore, only temporal, locative, and directional senses of prepositions evoke frames.[12]

Preliminary experiments using a statistical method for target identification gave unsatisfactory results; instead, we followed J&N'07 in using a small set of rules to identify targets. First, we created a master list of all the morphological variants of targets that appear in the exemplar sentences and a given training set. For a sentence in

[12] Note that there have been dedicated shared tasks to determine relationships between nominals (Girju et al. 2007) and word-sense disambiguation of prepositions (Litkowski and Hargraves 2007), but we do not build specific models for predicates of these categories.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

TARGET IDENTIFICATION	P	R	F_1
Our technique (§4)	**89.92**	**70.79**	**79.21**
Baseline: J&N'07	*87.87*	*67.11*	*76.10*

Table 3: Target identification results for our system and the baseline on the **SemEval'07 dataset**. Scores in bold denote significant improvements over the baseline ($p < 0.05$).

new data, we considered as candidate targets only those substrings that appear in this master list. We also did not attempt to capture discontinuous frame targets: e.g. we treat *there would have been* as a single span even though the corresponding LU is *there be*.V.[13]

Next, we pruned the candidate target set by applying a series of rules identical to the ones described by Johansson and Nugues (2007, see Appendix A), with two exceptions. First, they identified locative, temporal, and directional prepositions using a dependency parser so as to retain them as valid LUs. In contrast, we pruned all types of prepositions because we found them to hurt our performance on the development set due to errors in syntactic parsing. In a second departure from their target extraction rules, we did not remove the candidate targets that had been tagged as support verbs for some other target. Note that we used a conservative white list which filters out targets whose morphological variants were not seen either in the lexicon or the training data.[14] Therefore, when this conservative process of automatic target identification is used, our system loses the capability to predict frames for completely unseen LUs, despite the fact that our our powerful frame identification model (Section 5) can accurately label frames for new LUs.[15]

[13] There are 629 multiword LUs in the lexicon, and they correspond to 4.8% of the targets in the training set; among them are *screw up*.V, *shoot the breeze*.V, and *weapon of mass destruction*.N. In the SemEval 2007 training data, there are just 99 discontinuous multiword targets (1% of all targets).

[14] This conservative approach violates theoretical linguistic assumptions about frame-evoking targets as governed by frame semantics. It also goes against the spirit of using linguistic constraints to improve the separate subtask of argument identification (see Section 7); however, due to varying distributions of target annotations, high empirical error in identifying locative, temporal and directional prepositions and support verbs, we resorted to this aggressive filtering heuristic to avoid making too many target identification mistakes.

[15] To predict frames and roles for new and unseen LUs, SEMAFOR provides the user with an option to mark those LUs in the input.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Results. Table 3 shows results on target identification tested on the SemEval 2007 test set; our system gains 3 F_1 points over the baseline. This is statistically significant with $p < 0.01$. Our results are also significant in terms of precision ($p < 0.05$) and recall ($p < 0.01$). There are 85 distinct LUs for which the baseline fails to identify the correct target while our system succeeds. A considerable proportion of these units have more than one token (e.g. *chemical and biological weapon*.N, *ballistic missile*.N, etc.), which J&N'07 do not model. The baseline also does not label variants of *there be*.V, e.g. *there are* and *there has been*, which we correctly label as targets. Some examples of other single token LUs that the baseline fails to identify are names of months, LUs that belong to the ORIGIN frame (e.g. *iranian*.A) and directions, e.g., *north*.A or *north-south*.A.[16]

5. Frame Identification

Given targets, our parser next identifies their frames, using a statistical model.

5.1 Lexical units

FrameNet specifies a great deal of structural information both within and among frames. For frame identification we make use of frame-evoking **lexical units**, the (lemmatized and POS-tagged) words and phrases listed in the lexicon as referring to specific frames. For example, listed with the BRAGGING frame are 10 LUs, including *boast*.N, *boast*.V, *boastful*.A, *brag*.V, and *braggart*.N. Of course, due to polysemy and homonymy, the same LU may be associated with multiple frames; for example, *gobble*.V is listed under both the INGESTION and MAKE_NOISE frames. We thus term *gobble*.V an **ambiguous**

[16] We do not evaluate the target identification module on the FrameNet 1.5 dataset; we instead ran controlled experiments on those data to measure performance of the statistical frame identification and argument identification subtasks, assuming that the correct targets were given. Moreover, as discussed in Section 3.2, the target annotations on the FrameNet 1.5 test set were fewer in number in comparison to the training set, resulting in a mismatch of target distributions between train and test settings.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

LU. All targets in the exemplar sentences, our training data, and most in our test data, correspond to known LUs. (See Section 5.4 for statistics of unknown LUs in the test sets.)

To incorporate frame-evoking expressions found in the training data but not the lexicon—and to avoid the possibility of lemmatization errors—our frame identification model will incorporate, via a latent variable, features based directly on exemplar and training **targets** rather than LUs. Let $\mathcal{L}$ be the set of (unlemmatized and automatically POS-tagged) targets found in the exemplar sentences of the lexicon and/or the sentences in our training set. Let $\mathcal{L}_f \subseteq \mathcal{L}$ be the subset of these targets annotated as evoking a particular frame f.[17] Let $\mathcal{L}^l$ and $\mathcal{L}^l_f$ denote the lemmatized versions of $\mathcal{L}$ and $\mathcal{L}_f$, respectively. Then, we write *boasted*.VBD $\in \mathcal{L}_{\textsf{BRAGGING}}$ and *boast*.VBD $\in \mathcal{L}^l_{\textsf{BRAGGING}}$to indicate that this inflected verb *boasted* and its lemma *boast* have been seen to evoke the BRAGGING frame. Significantly, however, another target, such as *toot your own horn*, might be used elsewhere to evoke this frame. We thus face the additional hurdle of predicting frames for unknown words.

In producing full text annotations for the SemEval 2007 dataset, annotators created several domain-critical frames that were not already present in version 1.3 of the lexicon. For our experiments we omit frames attested in neither the training data nor the exemplar sentences from the lexicon.[18] This leaves a total of 665 frames for the SemEval 2007 dataset and a total of 877 frames for the FrameNet 1.5 dataset.

[17] For example, on average, there are 34 targets per frame in the SemEval 2007 dataset; the average frame ambiguity of each target in $\mathcal{L}$ is 1.17.

[18] Automatically predicting new frames is a challenge not yet attempted to our knowledge (including here). Note that the scoring metric (Section 3.3) gives partial credit for *related* frames (e.g., a more general frame from the lexicon).

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

- the POS of the parent of the head word of t_i
- * the set of syntactic dependencies of the head word[20] of t_i
- * if the head word of t_i is a verb, then the set of dependency labels of its children
- the dependency label on the edge connecting the head of t_i and its parent
- the sequence of words in the prototype, $\mathbf{w}_\ell$
- the lemmatized sequence of words in the prototype
- the lemmatized sequence of words in the prototype and their part-of-speech tags $\boldsymbol{\pi}_\ell$
- WordNet relation[21] ρ holds between ℓ and t_i
- WordNet relation[21] ρ holds between ℓ and t_i, and the prototype is ℓ
- WordNet relation[21] ρ holds between ℓ and t_i, the POS tag sequence of ℓ is $\boldsymbol{\pi}_\ell$, and the POS tag sequence of t_i is $\boldsymbol{\pi}_t$

Table 4: Features used for frame identification (Equation 2). All also incorporate f, the frame being scored. $\ell = \langle \mathbf{w}_\ell, \boldsymbol{\pi}_\ell \rangle$ consists of the words and POS tags[22] of a target seen in an exemplar or training sentence as evoking f. The features with starred bullets were also used by Johansson and Nugues (2007).

5.2 Model

For a given sentence $\mathbf{x}$ with frame-evoking targets $\mathbf{t}$, let t_i denote the ith target (a word sequence).[19] Let t_i^l denote its lemma. We seek a list $\mathbf{f} = \langle f_1, \ldots, f_m \rangle$ of frames, one per target. In our model, the set of candidate frames for t_i is defined to include every frame f such that $t_i^l \in \mathcal{L}_f^l$—or if $t_i^l \notin \mathcal{L}^l$, then every known frame (the latter condition applies for 4.7% of the annotated targets in the SemEval 2007 development set). In both cases, we let $\mathcal{F}_i$ be the set of candidate frames for the ith target in $\mathbf{x}$. We denote the entire set of frames in the lexicon as $\mathcal{F}$.

To allow frame identification for targets whose lemmas were seen in neither the exemplars nor the training data, our model includes an additional variable, ℓ_i. This

[19] Here each t_i is a word sequence $\langle w_u, \ldots, w_v \rangle$, $1 \leq u \leq v \leq n$, though in principle targets can be noncontiguous.

[20] If the target is not a subtree in the parse, we consider the words that have parents outside the span, and apply three heuristic rules to select the head: 1) choose the first word if it is a verb; 2) choose the last word if the first word is an adjective; 3) if the target contains the word *of*, and the first word is a noun, we choose it. If none of these hold, choose the last word with an external parent to be the head.

[21] These are: IDENTICAL-WORD, SYNONYM, ANTONYM (including extended and indirect antonyms), HYPERNYM, HYPONYM, DERIVED FORM, MORPHOLOGICAL VARIANT (e.g., plural form), VERB GROUP, ENTAILMENT, ENTAILED-BY, SEE-ALSO, CAUSAL RELATION, and NO RELATION.

[22] POS tags are found automatically during preprocessing.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

variable ranges over the seen targets in $\mathcal{L}_{f_i}$, which can be thought of as **prototypes** for the expression of the frame. Importantly, frames are *predicted*, but prototypes are summed over via the latent variable. The prediction rule requires a probabilistic model over frames for a target:

$$f_i \leftarrow \underset{f \in \mathcal{F}_i}{\operatorname{argmax}} \sum_{\ell \in \mathcal{L}_f} p_{\boldsymbol{\theta}}(f, \ell \mid t_i, \mathbf{x}) \tag{1}$$

We model the probability of a frame f and the prototype unit ℓ, given the target and the sentence $\mathbf{x}$ as:

$$p_{\boldsymbol{\theta}}(f, \ell \mid t_i, \mathbf{x}) = \frac{\exp \boldsymbol{\theta}^\top \mathbf{g}(f, \ell, t_i, \mathbf{x})}{\sum_{f' \in \mathcal{F}} \sum_{\ell' \in \mathcal{L}_{f'}} \exp \boldsymbol{\theta}^\top \mathbf{g}(f', \ell', t_i, \mathbf{x})} \tag{2}$$

The above is a conditional log-linear model: for $f \in \mathcal{F}_i$ and $\ell \in \mathcal{L}_f$, where $\boldsymbol{\theta}$ are the model weights, and $\mathbf{g}$ is a vector-valued feature function. This discriminative formulation is very flexible, allowing for a variety of (possibly overlapping) features; e.g., a feature might relate a frame type to a prototype, represent a lexical-semantic relationship between a prototype and a target, or encode part of the syntax of the sentence.

Previous work has exploited WordNet for better coverage during frame identification (Johansson and Nugues 2007; Burchardt, Erk, and Frank 2005, e.g., by expanding the set of targets using synsets), and others have sought to extend the lexicon itself. We differ in our use of a latent variable to incorporate lexical-semantic *features* in a discriminative model, relating known lexical units to unknown words that may evoke frames. Here we are able to take advantage of the large inventory of partially-annotated exemplar sentences. Note that this model makes an independence assumption: each frame is predicted independently of all others in the document. In this way the model is similar to J&N'07. However, ours is a single conditional model that shares features and weights across all targets, frames, and prototypes, whereas the approach of J&N'07

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163
© Association for Computational Linguistics

consists of many separately trained models. Moreover, our model is unique in that it uses a latent variable to smooth over frames for unknown or ambiguous LUs.

Frame identification features depend on the preprocessed sentence $\mathbf{x}$, the prototype ℓ and its WordNet lexical-semantic relationship with the target t_i, and of course the frame f. Our model uses binary features, which are detailed in Table 4.

5.3 Parameter Estimation

Given a training dataset (either SemEval 2007 dataset or the FrameNet 1.5 full text annotations), which is of the form $\langle\langle \mathbf{x}^{(j)}, \mathbf{t}^{(j)}, \mathbf{f}^{(j)}, \mathcal{A}^{(j)} \rangle\rangle_{j=1}^{N}$, we discriminatively train the frame identification model by maximizing the training data log-likelihood:[23]

$$\max_{\boldsymbol{\theta}} \sum_{j=1}^{N} \sum_{i=1}^{m_j} \log \sum_{\ell \in \mathcal{L}_{f_i^{(j)}}} p_{\boldsymbol{\theta}}(f_i^{(j)}, \ell \mid t_i^{(j)}, \mathbf{x}^{(j)}) \qquad (3)$$

Above, m_j denotes the number of frames in a sentence indexed by j. Note that the training problem is non-convex because of the summed-out prototype latent variable ℓ for each frame. To calculate the objective function, we need to cope with a sum over frames and prototypes for each target (see Equation 2), often an expensive operation. We locally optimize the function using a distributed implementation of L-BFGS.[24] This is the most expensive model that we train: with 100 parallelized CPUs using MapReduce (Dean and Ghemawat 2008), training takes several hours.[25] Decoding takes only a few minutes on one CPU for the test set.

[23] We found no benefit on either development dataset from using an L_2 regularizer (zero-mean Gaussian prior).

[24] We do not experiment with the initialization of model parameters during this non-convex optimization process; all parameters are initialized to 0.0 before running the optimizer. However, in future work, experiments can be conducted with different random initialization points to seek non-local optima.

[25] In later experiments, we used another implementation with 128 parallel cores in a multi-core MPI setup (Gropp, Lusk, and Skjellum 1994), where training took several hours.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Frame Identification (§5.2)		exact matching			partial matching		
		P	R	F_1	P	R	F_1
SemEval 2007 Data	gold targets	60.21	60.21	60.21	74.21	74.21	74.21
	automatic targets (§4)	**69.75**	**54.91**	**61.44**	**77.51**	**61.03**	**68.29**
	J&N'07 targets	65.34	49.91	56.59	74.30	56.74	64.34
	Baseline: J&N'07	*66.22*	*50.57*	*57.34*	*73.86*	*56.41*	*63.97*
FrameNet 1.5 Release	gold targets	82.97	82.97	82.97	90.51	90.51	90.51
	– unsupported features	80.30	80.30	80.30	88.91	88.91	88.91
	& – latent variable	75.54	75.54	75.54	85.92	85.92	85.92

Table 5: Frame identification results on both the SemEval 2007 dataset and the FrameNet 1.5 release. Precision, recall, and F_1 were evaluated under exact and partial frame matching; see §3.3. Bold indicates best results on the SemEval 2007 data, which are also statistically significant with respect to the baseline ($p < 0.05$).

5.4 Supervised Results

SemEval 2007 Data. On the SemEval 2007 dataset, we evaluate the performance of our frame identification model given gold-standard targets and automatically identified targets (Section 4); see Table 5. Together, our target and frame identification outperform the baseline by 4 F_1 points. To compare the frame identification stage in isolation with that of J&N'07, we ran our frame identification model with the targets identified by their system as input. With partial matching, our model achieves a relative improvement of 0.6% F_1 over J&N'07, as shown in the third row of Table 5 (though this is not significant). Note that for exact maching, the F_1 score of the automatic targets setting is better than the gold target setting. This is due to the fact that there are many unseen predicates in the test set on which the frame identification model performs poorly; however, for the automatic targets which are mostly seen in the lexicon and training data, the model gets high precision, resulting in better overall F_1 score.

Our frame identification model thus performs on par with the previous state of the art for this task, and offers several advantages over J&N's formulation of the problem: it requires only a single model, learns lexical-semantic features as part of that model

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

rather than requiring a preprocessing step to expand the vocabulary of frame-evoking words, and is probabilistic, which can facilitate global reasoning.

In the SemEval 2007 dataset, for gold-standard targets, 210 out of 1,059 lemmas were not present in the white list that we used for target identification (see Section 4). Our model correctly identifies the frames for 4 of these 210 lemmas. For 44 of these lemmas, the evaluation script assigns a score of 0.5 or more, suggesting that our model predicts a closely related frame. Finally, for 190 of the 210 lemmas, a positive score is assigned by the evaluation script. This suggests that the hidden variable model helps in identifying related (but rarely exact) frames for unseen targets, and explains why under exact—but not partial—frame matching, the F_1 score using automatic targets is commensurate with the score for oracle targets.[26]

For automatically identified targets, the F_1 score falls because the model fails to predict frames for unseen lemmas. However, our model outperforms J&N'07 by 4 F_1 points. The partial frame matching F_1 score of our model represents a significant improvement over the baseline ($p < 0.01$). The precision and recall measures are significant as well ($p < 0.05$ and $p < 0.01$, respectively). However, because targets identified by J&N'07 and frames classified by our frame identification model resulted in scores on par with the baseline, we note that the significant results follow due to better target identification. Note from the results that the automatic target identification model shows an increase in precision, at the expense of recall. This is because the white list for target identification restricts the model to predict frames only for known LUs. If we label the subset of test set with already seen LUs (seen only in the training set, excluding the exemplars) with their corresponding most frequent frame, we achieve

[26] J&N'07 did not report frame identification results for oracle targets; thus directly comparing the frame identification models is difficult.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163
© Association for Computational Linguistics

an exact match accuracy between 52.9% and 91.2%, depending on the accuracy of the unseen LUs (these bounds assume, respectively, that they are all incorrectly labeled or all correctly labeled).

FrameNet 1.5 Release. The bottom three rows of Table 5 show results on the full text annotation test set of the FrameNet 1.5 release. Since the number of annotations nearly doubled, we see large improvements in frame identification accuracy. Note that we only evaluate with gold targets as input to frame identification. (As mentioned in Section 3.2, some documents in the test set have not been annotated for all targets, so evaluating automatic target identification would not be informative.) We found that 50.1% of the targets in the test set were ambiguous, i.e. they associated with more than one frame either in FrameNet or our training data. On these targets, the exact frame identification accuracy is 73.10% and the partial accuracy is 85.77%, which indicates that the frame identification model is robust to target ambiguity. On this dataset, the most frequent frame baseline achieves an exact match accuracy between 74.0% and 88.1%, depending on the accuracy of the unseen LUs.

We conducted further experiments with ablation of the latent variable in our frame identification model. Recall that the decoding objective used to choose a frame by marginalizing over a latent variable ℓ, whose values range over targets known to associate with the frame f being considered (see Equations 1–2) in training. How much do the prototypes, captured by the latent variable, contribute to performance? Instead of treating ℓ as a marginalized latent variable, we can fix its value to the observed target.

An immediate effect of this choice is a blow-up in the number of features; we must instantiate features (see Table 4) for all 4,194 unique targets observed in training. Because each of these features needs to be associated with all 877 frames in the partition

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

function of Equation 2, the result is an eighty-fold blowup of the feature space (the latent variable model had 465,317 features). Such a model is not computationally feasible in our engineering framework, so we considered a model using only features observed to fire at some point in the training data (called "supported" features),[27] resulting in only 72,058 supported features. In Table 5, we see a significant performance drop (on both exact and partial matching accuracy) with this latent variable–free model, compared both to our latent variable model with all features and with only supported features (of which there are 165,200). This establishes that the latent variable in our frame identification model helps in terms of accuracy, and lets us use a moderately sized feature set incorporating helpful unsupported features.

Finally, in our test set, we found that 144 out of the 4,458 annotated targets were unseen, and our full frame identification model only labeled 23.1% of the frames correctly for those unseen targets; in terms of partial match accuracy, the model achieved a score of 46.6%. This, along with the results on the SemEval 2007 unseen targets, shows that there is substantial opportunity for improvement when unseen targets are presented to the system. We address this issue next.

5.5 Semi-Supervised Lexicon Expansion

We next address the poor performance of our frame identification model on targets that were unseen as LUs in FrameNet or as instances in training data, and briefly describe a technique for expanding the set of lexical units with potential semantic frames that they can associate with. These experiments were carried out on the FrameNet 1.5 data only. We employ a semi-supervised learning (SSL) technique that uses a graph constructed

[27] The use of *unsupported* features, i.e., those that can fire for an analysis in the partition function but not observed to fire in the training data, has been observed to give performance improvements in NLP problems; see, e.g., Sha and Pereira (2003) and Martins et al. (2010).

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

from labeled and unlabeled data. The widely-used **graph-based SSL** framework—see Bengio, Delalleau, and Le Roux (2006) and Zhu (2008) for introductory material on this topic—has been shown to perform better than several other semi-supervised algorithms on benchmark datasets (Chapelle, Schölkopf, and Zien 2006, ch. 21). The method constructs a graph where a small portion of vertices correspond to labeled instances, and the rest are unlabeled. Pairs of vertices are connected by weighted edges denoting the similarity between the pair. Traditionally, Markov random walks (Szummer and Jaakkola 2001; Baluja et al. 2008) or optimization of a loss function based on smoothness properties of the graph (Corduneanu and Jaakkola 2003; Zhu, Ghahramani, and Lafferty 2003; Subramanya and Bilmes 2008, *inter alia*) are performed to propagate labels from the labeled vertices to the unlabeled ones. In our work, we are interested in multi-class generalizations of graph-propagation algorithms suitable for NLP applications, where each graph vertex can assume one *or more* out of many possible labels (Talukdar and Crammer 2009; Subramanya and Bilmes 2008, 2009). For us, graph vertices correspond to natural language *types* (not tokens) and undirected edges between them are weighted using a similarity metric. Recently, this setup has been used to learn soft labels on natural language types (say, word *n*-grams or in our case, syntactically disambiguated predicates) from seed data, resulting in large but noisy *lexicons*, which are used to constrain structured prediction models. Applications have ranged from domain adaptation of sequence models (Subramanya, Petrov, and Pereira 2010) to unsupervised learning of POS taggers by using bilingual graph-based projections (Das and Petrov 2011).

We describe our approach to graph construction, propagation for lexicon expansion, and the use of the result to impose constraints on frame identification.

5.5.1 Graph Construction. We construct a graph with lexical units as vertices. Thus, each vertex corresponds to a lemmatized word or phrase appended with a coarse POS

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

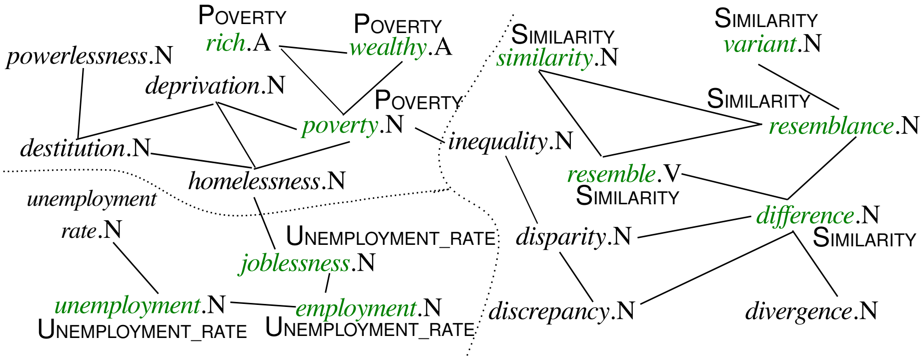


Figure 4: Excerpt from our constructed graph over LUs. Green LUs are observed in the FrameNet 1.5 data. Above/below them are shown the most frequently observed frame that these LUs associate with. The black LUs are unobserved and graph propagation produces a distribution over most likely frames that they could evoke as target instances.

tag. We use two resources for graph construction. First, we take all the words and phrases present in a dependency-based thesaurus constructed using syntactic cooccurrence statistics (Lin 1998), and aggregate words and phrases that share the same lemma and coarse POS tag. To construct this resource, Lin used a corpus containing 64 million words that was parsed with a fast dependency parser (Lin 1993, 1994), and syntactic contexts were used to find similar lexical items for a given word or phrase. Lin separately treated nouns, verbs and adjectives/adverbs, so these form the three parts of the thesaurus. This resource gave us a list of possible LUs, much larger in size than the LUs present in FrameNet data.

The second component of graph construction comes from FrameNet itself. We scanned the exemplar sentences in FrameNet 1.5 and the training section of the full text annotations and gathered a distribution over frames for each LU appearing in FrameNet data. For a pair of LUs, we measured the Euclidean distance between their frame distributions. This distance was next converted to a similarity score and interpolated with the similarity score from Lin's dependency thesaurus. We omit further details about the interpolation and refer the reader to full details given in Das and Smith (2011).

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163
© Association for Computational Linguistics

For each LU, we create a vertex and link it to the K nearest neighbor LUs under the interpolated similarity metric. The resulting graph has 64,480 vertices, 9,263 of which are labeled seeds from FrameNet 1.5 and 55,217 of which are unlabeled. Each vertex has a possible set of labels corresponding to the 877 frames defined in the lexicon. Figure 4 shows an excerpt from the constructed graph.

5.5.2 Propagation by Optimization. Once the graph is constructed, the 9,263 seed vertices with supervised frame distributions are used to propagate the semantic frame information via their nearest neighbors to all vertices. Here we discuss two graph-based SSL objective functions. Das and Smith (2012) compare several other graph-based SSL algorithms for this problem; we refer the interested reader to that paper. Let V denote the set of all vertices in our graph, $\hat{V} \subset V$ be the set of seed vertices and $\mathcal{F}$ denote the set of all frames. Let $\mathcal{N}(v)$ denote the set of neighbors of vertex $v \in V$. Let $\boldsymbol{q} = \{q_1, q_2, \ldots, q_{|V|}\}$ be the set of frame distributions, one per vertex. For each seed vertex $v \in \hat{V}$, we have a supervised frame distribution $\hat{q}_v$. All edges in the graph are weighted according to the aforementioned interpolated similarity score, denoted w_{uv} for the edge adjacent to vertices u and v. We find $\boldsymbol{q}$ by solving:

$$\textbf{NGF-}\ell_2: \underset{\substack{\boldsymbol{q},\text{ s.t. }\boldsymbol{q}\geq\mathbf{0},\\ \forall v\in V, \|q_v\|_1=1}}{\arg\min} \sum_{v\in\hat{V}} \|\hat{q}_v - q_v\|_2^2 + \mu \sum_{v\in V, u\in\mathcal{N}(v)} w_{uv}\|q_v - q_u\|_2^2 + \nu \sum_{v\in V} \|q_v - \tfrac{1}{|\mathcal{F}|}\|_2^2 \tag{4}$$

We call the objective in Equation 4 **NGF-**ℓ_2 because it uses *normalized* probability distributions at each vertex and is a *Gaussian field*; it also employs a uniform ℓ_2 penalty—the third term in the objective function. This is a multiclass generalization of the quadratic cost criterion (Bengio, Delalleau, and Le Roux 2006), also used by Subramanya, Petrov, and Pereira (2010) and Das and Petrov (2011). Our second graph objective function is as

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

follows:

$$\textbf{UJSF-}\ell_{1,2} : \underset{\boldsymbol{q},\text{ s.t. }\boldsymbol{q}\geq\mathbf{0}}{\arg\min} \sum_{v\in\hat{V}} D_{JS}(\hat{q}_v \| q_v) + \mu \sum_{v\in V, u\in\mathcal{N}(v)} w_{uv} D_{JS}(q_v \| q_u) + \nu \sum_{v\in V} \|q_v\|_1^2 \quad (5)$$

We call it **UJSF**-$\ell_{1,2}$ since it uses *unnormalized* probability measures at each vertex and is a *Jensen-Shannon field*, employing pairwise Jensen-Shannon divergences (Burbea and Rao 2006; Lin 1991) and a sparse $\ell_{1,2}$ penalty (Kowalski and Torrésani 2009) as the third term. Das and Smith (2012) proposed the objective function in Equation 5. It seeks at each graph vertex a sparse measure, as we expect in a lexicon (i.e., few frames have nonzero probability for a given target). The above two graph objectives can be optimized by iterative updates, whose details we omit in this article; more information about the motivation behind using the $\ell_{1,2}$ penalty in the **UJSF**-$\ell_{1,2}$ objective, the optimization procedure, and an empirical comparison of these and other objectives on another NLP task can be found in Das and Smith (2012).

5.5.3 Constraints for Frame Identification. Once a graph-based SSL objective function is minimized, we arrive at the optimal set of frame distributions $\boldsymbol{q}^*$, which we use to constrain our frame identification inference rule, expressed in Equation 1. In that rule, t_i is the ith target in a sentence $\mathbf{x}$, and f_i is the corresponding evoked frame. We now add a constraint to that rule. Recall from Section 5.2 that for targets with known lemmatized forms, $\mathcal{F}_i$ was defined to be the set of frames that associate with lemma t_i^l in the supervised data. For unknown lemmas, $\mathcal{F}_i$ was defined to be all the frames in the lexicon. If the LU corresponding to t_i is present in the graph, let it be the vertex v_i. For such targets t_i covered by the graph, we redefine $\mathcal{F}_i$ as:

$$\mathcal{F}_i = \{f : f \in M\text{-best frames under } q^*_{v_i}\} \quad (6)$$

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

	UNKNOWN TARGETS		ALL TARGETS		Graph
	exact	**partial**	**exact**	**partial**	Lexicon
	frame matching	**frame matching**	**frame matching**	**frame matching**	Size
Supervised	23.08	46.62	82.97	90.51	–
Self-training	18.88	42.67	82.27	90.02	–
NGF-ℓ_2	39.86	62.35	83.51	91.02	128,960
UJSF-$\ell_{1,2}$	**42.67**	**65.29**	**83.60**	**91.12**	**45,544**

Table 6: Exact and partial frame identification accuracy on the FrameNet 1.5 dataset with the size of lexicon (in terms of non-zero frame components in the truncated frame distributions) used for frame identification, given *gold* targets. The supervised model is compared to alternatives in Table 5. Bold indicates best results. **UJSF**-$\ell_{1,2}$ produces statistically significant results ($p < 0.001$) for all metrics with respect to the supervised baseline for both the unseen LUs as well as the whole test set. Although the **NGF**-ℓ_2 and **UJSF**-$\ell_{1,2}$ models are statistically indistinguishable, it is noteworthy that the **UJSF**-$\ell_{1,2}$ objective produces a much smaller lexicon.

For targets t_i in test data whose LUs are not present in the graph (and hence in supervised data), $\mathcal{F}_i$ is the set of all frames. Note that in this semi-supervised extension of our frame identification inference procedure, we introduced several hyperparameters, namely μ, ν, K (the number of nearest neighbors for each vertex included in the graph) and M (the number of highest scoring frames per vertex according to the induced frame distribution). We choose these hyperparameters using cross-validation by tuning the frame identification accuracy on unseen targets. (Different values of the first three hyperparameters were chosen for the different graph objectives and we omit their values here for brevity; M turned out to be 2 for all models.)

Table 6 shows frame identification accuracy, both using exact match as well as partial match. Performance is shown on the portion of the test set containing unknown LUs, as well as the whole test set. The final column presents lexicon size in terms of the set of truncated frame distributions (filtered according to the top M frames in q_v for a vertex v) for all the LUs in a graph. For comparison with a semi-supervised baseline, we consider a self-trained system. For this system, we used the supervised frame identification system to label 70,000 sentences from the English Gigaword corpus

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

with frame-semantic parses. For finding targets in a raw sentence, we used a relaxed target identification scheme, where we marked as potential frame-evoking units all targets seen in the lexicon and all other words which were not prepositions, particles, proper nouns, foreign words or WH-words. We appended these automatic annotations to the training data, resulting in 711,401 frame annotations, more than 36 times the annotated data. These data were next used to train a frame identification model.[28] This setup is very similar to that of Bejan (2009) who used self-training to improve frame identification. In our setting, however, self-training hurts relative to the fully supervised approach (Table 6).

Note that for the unknown part of the test set the graph-based objectives outperform both the supervised model as well as the self-training baseline by a margin of $\sim$20% absolute. The best model is **UJSF**-$\ell_{1,2}$, and its performance is significantly better than the supervised model ($p < 0.01$). It also produces a smaller lexicon (using the sparsity-inducing penalty) than **NGF**-ℓ_2, requiring less memory during frame identification inference. The small footprint can be attributed to the removal of LUs for which all frame components were zero ($q_i = \mathbf{0}$). The improvements of the graph-based objectives over the supervised and the self-trained models are modest for the whole test set, but the best model still has statistically significant improvements over the supervised model ($p < 0.01$).

6. Argument Identification

Given a sentence $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$, the set of targets $\mathbf{t} = \langle t_1, \ldots, t_m \rangle$, and a list of evoked frames $\mathbf{f} = \langle f_1, \ldots, f_m \rangle$ corresponding to each target, argument identification is the task

[28] We ran self-training with smaller amounts of data, but found no significant difference with the results achieved with 711,401 frame annotations. As we observe in Table 6, in our case, self-training performs worse than the supervised model, and we do not hope to improve with even more data.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

of choosing which of each f_i's roles are filled, and by which parts of $\mathbf{x}$. This task is most similar to the problem of semantic role labeling, but uses a richer set of frame-specific labels than PropBank annotations.

6.1 Model

Let $\mathcal{R}_{f_i} = \{r_1, \ldots, r_{|\mathcal{R}_{f_i}|}\}$ denote frame f_i's **roles** (named frame element types) observed in an exemplar sentence and/or our training set. A subset of each frame's roles are marked as **core** roles; these roles are conceptually and/or syntactically necessary for any given use of the frame, though they need not be overt in every sentence involving the frame. These are roughly analogous to the core arguments ARG0–ARG5 in PropBank. Non-core roles—analogous to the various ARGM-* in PropBank—loosely correspond to syntactic adjuncts, and carry broadly applicable information such as the time, place, or purpose of an event. The lexicon imposes some additional structure on roles, including relations to other roles in the same or related frames, and semantic types with respect to a small ontology (marking, for instance, that the entity filling the protagonist role must be sentient for frames of cognition). Figure 3 illustrates some of the structural elements comprising the frame lexicon by considering the CAUSE_TO_MAKE_NOISE frame.

We identify a set $\mathcal{S}$ of spans that are candidates for filling any role $r \in \mathcal{R}_{f_i}$. In principle, $\mathcal{S}$ could contain any subsequence of $\mathbf{x}$, but in this work we only consider the set of contiguous spans that (a) contain a single word or (b) comprise a valid subtree of a word and all its descendants in the dependency parse produced by the MST parser. This covers approximately 80% of arguments in the development data for both datasets.

The empty span, denoted $\emptyset$, is also included in $\mathcal{S}$, since some roles are not explicitly filled; in the SemEval 2007 development data, the average number of roles an evoked frame defines is 6.7, but the average number of overt arguments is only 1.7.[29] In training, if a labeled argument is not a subtree of the dependency parse, we add its span to $\mathcal{S}$.[30]

Let $\mathcal{A}_i$ denote the mapping of roles in $\mathcal{R}_{f_i}$ to spans in $\mathcal{S}$. Our model makes a prediction for each $\mathcal{A}_i(r_k)$ (for all roles $r_k \in \mathcal{R}_{f_i}$) using:

$$\mathcal{A}_i(r_k) \leftarrow \operatorname*{argmax}_{s \in \mathcal{S}} p_{\boldsymbol{\psi}}(s \mid r_k, f_i, t_i, \mathbf{x}) \tag{7}$$

We use a conditional log-linear model over spans for each role of each evoked frame:

$$p_{\boldsymbol{\psi}}(\mathcal{A}_i(r_k) = s \mid f_i, t_i, \mathbf{x}) = \frac{\exp \boldsymbol{\psi}^\top \mathbf{h}(s, r_k, f_i, t_i, \mathbf{x})}{\sum_{s' \in \mathcal{S}} \exp \boldsymbol{\psi}^\top \mathbf{h}(s', r_k, f_i, t_i, \mathbf{x})} \tag{8}$$

Note that our model chooses the span for each role separately from the other roles and ignores all frames except the frame the role belongs to. Our model departs from the traditional SRL literature by modeling the argument identification problem in a single stage, rather than first classifying token spans as arguments and then labeling them. A constraint implicit in our formulation restricts each role to have at most one overt argument, which is consistent with 96.5% of the role instances in the SemEval 2007 training data and 96.4% of the role instances in the FrameNet 1.5 full text annotations.

Out of the overt argument spans in the training data, 12% are duplicates, having been used by some previous frame in the sentence (supposing some arbitrary ordering of frames). Our role-filling model, unlike a sentence-global argument detection-and-

[29] In the annotated data, each core role is filled with one of three types of *null instantiations* indicating how the role is conveyed implicitly. For instance, the imperative construction implicitly designates a role as filled by the addressee, and the corresponding filler is thus CNI (constructional null instantiation). In this work we do not distinguish different types of null instantiation. The interested reader may refer to Chen et al. (2010), who handle the different types of null instantions during argument identification.

[30] Here is an example in the FrameNet 1.5 training data where this occurs. In the sentence: *As capital of Europe 's most explosive economy, Dublin seems to be changing before your very eyes.*, the word *economy* evokes the ECONOMY frame with the phrase *most explosive* fulfilling the Descriptor role. However, in the dependency parse for the sentence the phrase is not a subtree since both words in the frame attach to the word *economy*. Future work may consider better heuristics to select potential arguments from the dependency parses to recover more gold arguments than what the current work achieves.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

classification approach,[31] permits this sort of argument sharing among frames. Word tokens belong to an average of 1.6 argument spans, including the quarter of words that do not belong to any argument.

Appending together the local inference decisions from Equation 7 gives us the best mapping $\hat{\mathcal{A}}_t$ for target t. Features for our log-linear model (Equation 8) depend on the preprocessed sentence $\mathbf{x}$; the target t; a role r of frame f; and a candidate argument span $s \in \mathcal{S}$.[32] For features using the head word of the target t or a candidate argument span s, we use the heuristic described in footnote 20 for selecting the head of non-subtree spans.

Table 7 lists the feature templates used in our model. Every feature template has a version which does not take into account the role being filled (so as to incorporate overall biases). The ◐ symbol indicates that the feature template also has a variant which is conjoined with r, the name of the role being filled; and ● indicates that the feature template additionally has a variant which is conjoined with both r and f, the name of the frame.[33] The role-name-only variants provide for smoothing over frames for common types of roles such as Time and Place; see Matsubayashi, Okazaki, and Tsujii (2009) for a detailed analysis of the effects of using role features at varying levels of granularity. Certain features in our model rely on closed-class POS tags, which are defined to be all Penn Treebank tags except for CD and tags that start with V, N, J, or R. Finally, the features that encode a count or a number are binned into groups: $(-\infty, -20]$, $[-19, -10]$, $[-9, -5]$, -4, -3, -2, -1, 0, 1, 2, 3, 4, $[5, 9]$, $[10, 19]$, $[20, \infty)$.

[31] J&N'07, like us, identify arguments for each target.

[32] In this section we use t, f, and r without subscripts since the features only consider a single role of a single target's frame.

[33] I.e., the ● symbol subsumes ◐, which in turn subsumes ○.

6.2 Parameter Estimation

We train the argument identification model by:

$$\max_{\boldsymbol{\psi}} \sum_{j=1}^{N} \sum_{i=1}^{m_j} \sum_{k=1}^{|\mathcal{R}_{f_i^{(j)}}|} \log p_{\boldsymbol{\psi}}(\mathcal{A}_i^{(j)}(r_k) \mid f_i^{(j)}, t_i^{(j)}, \mathbf{x}^{(j)}) - C \,\|\boldsymbol{\psi}\|_2^2 \tag{9}$$

Here N is the number of data points (sentences) in the training set, and m is the number of frame annotations per sentence. The above objective function is concave. For experiments with the SemEval 2007 data, we trained the model using stochastic gradient ascent (Bottou 2004) with no Gaussian regularization ($C = 0$).[34] Early stopping was done by tuning on the development set, and the best results were obtained with a batch size of 2 and 23 passes through the data.

On the FrameNet 1.5 release, we trained this model using L-BFGS (Liu and Nocedal 1989) and ran it for 1000 iterations. C was tuned on the development data, and we obtained best results for $C = 1.0$. We did not use stochastic gradient descent for this dataset as the number of training instances increased and parallelization of L-BFGS on a multicore setup implementing MPI (Gropp, Lusk, and Skjellum 1994) gave faster training speeds.

6.3 Decoding with Beam Search

Naïve prediction of roles using Equation 7 may result in overlap among arguments filling different roles of a frame, since the argument identification model fills each role independently of the others. We want to enforce the constraint that two roles of a single frame cannot be filled by overlapping spans.[35] Toutanova, Haghighi, and Manning

[34] This was the setting used by Das et al. (2010) and we kept it unchanged.

[35] On rare occasions a frame annotation may include a *secondary frame element layer*, allowing arguments to be shared among multiple roles in the frame; see Ruppenhofer et al. (2006) for details. The evaluation for this task only considers the primary layer, which is guaranteed to have disjoint arguments.

Algorithm 1 Joint decoding of frame f_i's arguments via beam search. $\mathsf{top}_\mathsf{k}(\mathcal{S}, p_\psi, r_j)$ extracts the k most probable spans from $\mathcal{S}$, under p_ψ, for role r_j. $\mathsf{extend}(D^{0:(j-1)}, \mathcal{S}')$ extends each span vector in $D^{0:(j-1)}$ with the most probable non-overlapping span from $\mathcal{S}'$, resulting in k best extensions overall.

```
Input: $\mathsf{k} > 0$, $\mathcal{R}_{f_i}$, $\mathcal{S}$, the distribution $p_\psi$ from Equation 8 for each role $r_j \in \mathcal{R}_{f_i}$
Output: $\hat{\mathcal{A}}_i$, a high-scoring mapping of roles of $f_i$ to spans with no token overlap among the spans
Calculate $\mathcal{A}_i$ according to Equation 7
$\forall r \in \mathcal{R}_{f_i}$ such that $\mathcal{A}_i(r) = \emptyset$, let $\hat{\mathcal{A}}_i(r) \leftarrow \emptyset$
$\mathcal{R}^+_{f_i} \leftarrow \{r : r \in \mathcal{R}_{f_i}, \mathcal{A}_i(r) \neq \emptyset\}$
$\mathsf{n} \leftarrow |\mathcal{R}^+_{f_i}|$
Arbitrarily order $\mathcal{R}^+_{f_i}$ as $\{r_1, r_2, \ldots r_\mathsf{n}\}$
Let $D^{0:j} = \langle D^{0:j}_1, \ldots, D^{0:j}_\mathsf{k} \rangle$ refer to the k-best list of vectors of compatible filler spans for roles $r_1$ through $r_j$
Initialize $D^{0:0}$ to be empty
for $j = 1$ to $\mathsf{n}$ do
    $D^{0:j} \leftarrow \mathsf{extend}(D^{0:(j-1)}, \mathsf{top}_\mathsf{k}(\mathcal{S}, p_\psi, r_j))$
end for
$\forall j \in \{1, \ldots, \mathsf{n}\}, \hat{\mathcal{A}}_i(r_j) \leftarrow D^{0:\mathsf{n}}_1[j]$
return $\hat{\mathcal{A}}_i$
```

(2005) presented a dynamic programming algorithm to prevent overlapping arguments for SRL; however, their approach used an orthogonal view to the argument identification stage, wherein they labeled phrase-structure tree constituents with semantic roles. That formulation admitted a dynamic programming approach; our formulation of finding the best argument span for each role does not.

To eliminate illegal overlap, we adopt the beam search technique detailed in Algorithm 1. The algorithm produces a set of k-best hypotheses for a frame instance's full set of role-span pairs, but uses an approximation in order to avoid scoring an exponential number of hypotheses. After determining which roles are most likely not explicitly filled, it considers each of the other roles in turn: in each iteration, hypotheses

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

incorporating a subset of roles are extended with high-scoring spans for the next role, always maintaining k alternatives. We set k=10,000 as the beam width.[36]

6.4 Results

Performance of the argument identification model is presented in Table 8 for both datasets in consideration. We analyze them below.

SemEval 2007 Data: For the SemEval dataset, the table shows how performance varies given different types of input: correct targets and correct frames, correct targets but automatically identified frames, and ultimately, no oracle input (the full frame parsing scenario). Rows 1–2 isolate the argument identification task from the frame identification task. Given gold targets and frames, our argument identification model (without beam search) gets an F_1 score of 68.09%; when beam search is applied, this increases to 68.46%, with a noticeable increase in precision. Note that an estimated 19% of correct arguments are excluded because they are neither single words nor complete subtrees (see Section 6.1) of the automatic dependency parses.[37]

Qualitatively, the problem of candidate span recall seems to be largely due to syntactic parse errors.[38] Although our performance is limited by errors when using the syntactic parse to determine candidate spans, it could still improve; this suggests that the model has trouble discriminating between good and bad arguments, and that additional feature engineering or jointly decoding arguments of a sentence's frames may be beneficial.

[36] We show the effect of varying beam widths in Table 9, where we present performance of an *exact* algorithm for argument identification.

[37] We found that using all constituents from the 10-best syntactic parses would improve oracle recall of spans in the development set by just a couple of percentage points, at the computational cost of a larger pool of candidate arguments per role.

[38] Note that, because of our labels-only evaluation scheme (Section 3.3), arguments missing a word or containing an extra word receive no credit. In fact, of the frame roles correctly predicted as having an overt span, the correct span was predicted 66% of the time, while 10% of the time the predicted starting and ending boundaries of the span were off by a total of 1 or 2 words.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Features with both null and non-null variants: These features come in two flavors: if the argument is null, then one version fires; if it is overt (non-null), then another version fires.

- ● some word in t has lemma λ
- ◐ some word in t has lemma λ, and the sentence uses PASSIVE voice
- ◐ the head of t has subcategorization sequence $\boldsymbol{\tau} = \langle \tau_1, \tau_2, \ldots \rangle$
- ● the head of t has c syntactic dependents
- ● some word in t has POS π
- ◐ some word in t has lemma λ, and the sentence uses ACTIVE voice
- ◐ some syntactic dependent of the head of t has dependency type τ
- ● bias feature (always fires)

Span content features: apply to overt argument candidates.

- ○ POS tag π occurs for some word in s
- ○ the first word of s has POS π
- ○ the last word of s has POS π
- ○ the head word of s has syntactic dependency type τ
- ● w_{s_2} and its closed-class POS tag π_{s_2}, provided that $|s| \geq 2$
- ○ the head word of s has lemma λ
- ○ the last word of s: $w_{s_{|s|}}$ has lemma λ
- ● $w_{s_{|s|}}$, and its closed-class POS tag $\pi_{s_{|s|}}$, provided that $|s| \geq 3$
- ◐ lemma λ is realized in some word in s, the voice denoted in the span (ACTIVE or PASSIVE)
- ○ the head word of s has POS π
- ● $|s|$, the number of words in the span
- ○ the first word of s has lemma λ
- ● the first word of s: w_{s_1}, and its POS tag π_{s_1}, if π_{s_1} is a closed-class POS
- ● the syntactic dependency type τ_{s_1} of the first word with respect to its head
- ● τ_{s_2}, provided that $|s| \geq 2$
- ● $\tau_{s_{|s|}}$, provided that $|s| \geq 3$
- ◐ lemma λ is realized in some word in s
- ◐ lemma λ is realized in some word in s, the voice denoted in the span, s's position with respect to t (BEFORE, AFTER, or OVERLAPPING)

Syntactic features: apply to overt argument candidates.

- ○ dependency path: sequence of labeled, directed edges from the head word of s to the head word of t
- ○ length of the dependency path

Span context POS features: for overt candidates, up to 6 of these features will be active.

- ○ a word with POS π occurs up to 3 words before the first word of s
- ○ a word with POS π occurs up to 3 words after the last word of s

Ordering features: apply to overt argument candidates.

- ● the position of s with respect to to the span of t: BEFORE, AFTER, or OVERLAPPING (i.e. there is at least one word shared by s and t)
- ○ linear word distance between the nearest word of s and the nearest word of t, provided s and t do not overlap
- ○ target-argument crossing: there is at least one word shared by s and t, at least one word in s that is not in t, and at least one word in t that is not in s
- ○ linear word distance between the middle word of s and the middle word of t, provided s and t do not overlap

Table 7: Features used for argument identification. Section 6.1 describes the meanings of the different circles attached to each feature.

ARGUMENT IDENTIFICATION					exact matching			partial matching			
		targets	*frames*	*decoding*	*P*	*R*	F_1	*P*	*R*	F_1	
SemEval'07 Data	Argument identification (full)	gold	gold	naïve	77.43	60.76	68.09				1
		gold	gold	beam	78.71	60.57	68.46				2
	Parsing (oracle targets)	gold	supervised (§5.2)	beam	49.68	42.82	46.00	57.85	49.86	53.56	3
	Parsing (full)	auto	supervised (§5.2)	beam	**58.08**	**38.76**	**46.49**	**62.76**	**41.89**	**50.24**	4
	Parsing (J&N'07 targets and frames)	auto	supervised (§3.4)	beam	56.26	36.63	44.37	60.98	39.70	48.09	5
	Baseline: J&N'07	*auto*	*supervised* (§3.4)	*N/A*	*51.59*	*35.44*	*42.01*	*56.01*	*38.48*	*45.62*	6
FrameNet 1.5 Release	Argument identification (full)	gold	gold	naïve	82.00	76.36	79.08				7
		gold	gold	beam	83.83	76.28	79.88				8
	Parsing (oracle targets)	gold	supervised (§5.2)	beam	67.81	60.68	64.05	72.47	64.85	68.45	9
		gold	SSL (**NGF**-ℓ_2, §5.5)	beam	68.22	61.04	64.43	72.87	65.20	68.82	10
		gold	SSL (**UJSF**-$\ell_{1,2}$, §5.5)	beam	**68.33**	**61.14**	**64.54**	**72.98**	**65.30**	**68.93**	11

Table 8: Argument identification results on both the SemEval'07 data as well as the full text annotations of FrameNet 1.5. For decoding, "beam" and "naïve" indicate whether the approximate joint decoding algorithm has been used or local independent decisions have been made for argument identification, respectively. On the SemEval 2007 data, for full parsing (automatic target, frame and argument identification), bold scores indicate best results, which are also significant improvements relative to the baseline ($p < 0.05$). On the FrameNet 1.5 dataset, bold scores indicate best results on automatic frame and argument identification—this is achieved by the frame identification model that uses the **UJSF**-$\ell_{1,2}$ graph-objective and automatic argument identification using beam search. This result is statistically significant over the supervised results shown in row 9 ($p < 0.001$). In terms of precision and F_1 score measured with partial frame matching, the results with the **UJSF**-$\ell_{1,2}$ model is statistically significant over the **NGF**-ℓ_2 model ($p < 0.05$). For recall with partial frame matching, and for all the three metrics with exact frame matching, the results with the two graph objectives are statistically indistinguishable. Note that certain partial match results are missing because in those settings, gold frames have been used for argument identification.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Rows 3–4 show the effect of automatic supervised frame identification on overall frame parsing performance. There is a 22% absolute decrease in F_1 (18% when partial credit is given for related frames), suggesting that improved frame identification or joint prediction of frames and arguments is likely to have a sizeable impact on overall performance. Rows 4–6 compare our full model (target, frame, and argument identification) with the baseline, showing significant improvement of more than 4.4 F_1 points for both exact and partial frame matching. As with frame identification, we compared the argument identification stage with that of J&N'07 in isolation, using the automatically identified targets and frames from the latter as input to our model. As shown in row 5, with partial frame matching, this gave us an F_1 score of 48.1% on the test set—significantly better ($p < 0.05$) than 45.6%, the full parsing result from J&N'07 (row 6 in Table 8). This indicates that our argument identification model—which uses a single discriminative model with a large number of features for role filling (rather than argument labeling)—is more accurate than the previous state of the art.

FrameNet 1.5 Release: Rows 7–12 show results on the newer dataset, which is part of the FrameNet 1.5 release. As in the frame identification results of Table 5, we do not show results using predicted targets, as we only test the performance of the statistical models. First, we observe that for results with gold frames, the F_1 score is 79.08% with naïve decoding, which is significantly higher than the SemEval counterpart. This indicates that increased training data greatly improves performance on the task. We also observe that beam search improves precision by nearly 2%, while getting rid of overlapping arguments. When both model frames and model arguments are used, we get an F_1 score of 68.45%, which is encouraging in comparison to the best results we achieved on the SemEval 2007 dataset. Semi-supervised lexicon expansion for frame identification

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

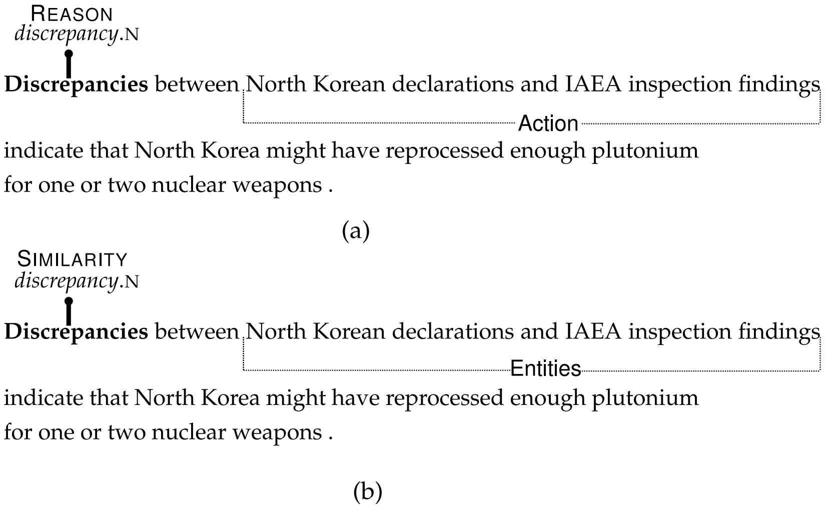

REASON
discrepancy.N
Discrepancies between North Korean declarations and IAEA inspection findings
Action
indicate that North Korea might have reprocessed enough plutonium
for one or two nuclear weapons .
(a)
SIMILARITY
discrepancy.N
Discrepancies between North Korean declarations and IAEA inspection findings
Entities
indicate that North Korea might have reprocessed enough plutonium
for one or two nuclear weapons .
(b)

Figure 5: (a) Output of the supervised frame-semantic parsing model, with beam search for argument identification, given the target **discrepancies**. The output is incorrect. (b) Output using the constrained frame identification model that takes into account the graph-based frame distributions over unknown predicates. In this particular example, the **UJSF**-$\ell_{1,2}$ graph objective is used. This output matches the gold annotation. The LU *discrepancy*.N is unseen in supervised FrameNet data.

further improves parsing performance. We observe the best results when the **UJSF**-$\ell_{1,2}$ graph objective is used for frame identification, significantly outperforming the fully supervised model on parsing ($p < 0.001$) for all evaluation metrics. The improvements with SSL can be explained by noting that frame identification performance goes up when the graph objectives are used, which carries over to argument identification. Figure 5 shows an example where the graph-based model **UJSF**-$\ell_{1,2}$ corrects an error made by the fully supervised model for the unseen LU *discrepancy*.N, both for frame identification and full frame-semantic parsing.

7. Collective Argument Identification with Constraints

The argument identification strategy described in the previous section does not capture some facets of semantic knowledge represented declaratively in FrameNet. In this section, we present an approach that exploits such knowledge in a principled, unified, and intuitive way. In prior NLP research using FrameNet, these interactions have been

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

largely ignored, though they have the potential to improve the quality and consistency of semantic analysis. The beam search technique (Algorithm 1) handles one kind of constraint: avoiding argument overlaps. It is, however, approximate and cannot handle other forms of constraints.

Here, we present an algorithm that exactly identifies the best full collection of arguments of a target given its semantic frame. Although we work within the conventions of FrameNet, our approach is generalizable to other semantic role labeling (SRL) frameworks. We model argument identification as constrained optimization, where the constraints come from expert knowledge encoded in FrameNet. Following prior work on PropBank-style SRL that dealt with similar constrained problems (Punyakanok et al. 2004; Punyakanok, Roth, and Yih 2008, *inter alia*), we incorporate this declarative knowledge in an integer linear program.

Because general-purpose ILP solvers are proprietary and do not fully exploit the structure of the problem, we turn to a class of optimization techniques called **dual decomposition** (Komodakis, Paragios, and Tziritas 2007; Rush et al. 2010; Martins et al. 2011a). We derive a modular, extensible, parallelizable approach in which semantic constraints map not just to declarative components in the algorithm, but also to procedural ones, in the form of "workers." While dual decomposition algorithms only solve a relaxation of the original problem, we make our approach *exact* by wrapping the algorithm in a branch-and-bound search procedure.[39]

We experimentally find that our algorithm achieves accuracy comparable to the results presented in Table 8, while respecting all imposed linguistic constraints. In comparison to beam search that violates many of these constraints, the presented ex-

[39] Open-source code in C++ implementing the AD^3 algorithm can be found at `http://www.ark.cs.cmu.edu/AD3`.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

act decoder is slower, but it decodes nine times faster than CPLEX, a state-of-the-art, proprietary, general-purpose exact ILP solver.[40]

7.1 Joint Inference

Here, we take a declarative approach to modeling argument identification using an ILP and relate our formulation to prior work in shallow semantic parsing. We show how knowledge specified in a linguistic resource (FrameNet in our case) can be used to derive the constraints in our ILP. Finally, we draw connections of our specification to graphical models, a popular formalism in artificial intelligence and machine learning, and describe how the constraints can be treated as factors in a factor graph.

7.1.1 Declarative Specification. Let us simplify notation by considering a given target t and not consider its index in a sentence $\mathbf{x}$; let the semantic frame it evokes be f. To solely evaluate argument identification, we assume that the semantic frame f is given, which is traditionally the case in controlled experiments used to evaluate SRL systems (Màrquez et al. 2008). Let the set of roles associated with the frame f be $\mathcal{R}_f$. In sentence $\mathbf{x}$, the set of candidate spans of words that might fill each role is enumerated, usually following an overgenerating heuristic, which is described in Section 6.1; as before, we call this set of spans $\mathcal{S}$. As before, this set also includes the null span $\emptyset$; connecting it to a role $r \in \mathcal{R}_f$ denotes that the role is not overt. Our approach assumes a scoring function that gives a strength of association between roles and candidate spans. For each role $r \in \mathcal{R}_f$ and span $s \in \mathcal{S}$, this score is parameterized as:

$$c(r, s) = \boldsymbol{\psi}^\top \mathbf{h}(s, r, f, t, \mathbf{x}), \tag{10}$$

[40] See http://www-01.ibm.com/software/integration/optimization/cplex-optimizer.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

where ψ are model weights and $\mathbf{h}$ is a feature function that looks at the target t, the evoked frame f, sentence $\mathbf{x}$, and its syntactic analysis, along with r and s. This scoring function is identical in form to the numerator's exponent in the log-linear model described in Equation 8. The SRL literature provides many feature functions of this form and many ways to use machine learning to acquire ψ. Our presented method does not make any assumptions about the score except that it has the form in Equation 10.

We define a vector $\mathbf{z}$ of binary variables $z_{r,s} \in \{0, 1\}$ for every role and span pair. We have that: $\mathbf{z} \in \{0, 1\}^d$, where $d = |\mathcal{R}_f| \times |\mathcal{S}|$. $z_{r,s} = 1$ means that role r is filled by span s. Given the binary $\mathbf{z}$ vector, it is straightforward to recover the collection of arguments by checking which components $z_{r,s}$ have an assignment of 1; we use this strategy to find arguments, as described in Section 7.3 (strategies 4 and 6). The joint argument identification task can be represented as a constrained optimization problem:

$$\begin{aligned} \text{maximize} \quad & \sum_{r \in \mathcal{R}_f} \sum_{s \in \mathcal{S}} c(r, s) \times z_{r,s} \\ \text{with respect to} \quad & \mathbf{z} \in \{0, 1\}^d \\ \text{such that} \quad & \mathbf{A}\mathbf{z} \leq \mathbf{b}. \end{aligned} \tag{11}$$

In the last line, $\mathbf{A}$ is a k-by-d matrix and $\mathbf{b}$ is a vector of length k. Thus, $\mathbf{A}\mathbf{z} \leq \mathbf{b}$ is a set of k inequalities representing constraints that are imposed on the mapping between roles and spans; these are motivated on linguistic grounds and are described next.[41]

[41] Note that equality constraints $\mathbf{a} \cdot \mathbf{z} = b$ can be transformed into double-side inequalities $\mathbf{a} \cdot \mathbf{z} \leq b$ and $-\mathbf{a} \cdot \mathbf{z} \leq -b$.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Uniqueness. Each role r is filled by at most one span in $\mathcal{S}$. This constraint can be expressed by:

$$\forall r \in \mathcal{R}_f, \sum_{s \in \mathcal{S}} z_{r,s} = 1. \tag{12}$$

There are $O(|\mathcal{R}_f|)$ such constraints. Note that since $\mathcal{S}$ contains the null span $\emptyset$, non-overt roles are also captured using the above constraints. Such a constraint is used extensively in prior literature (Punyakanok, Roth, and Yih 2008, Section 3.4.1).

Overlap. SRL systems commonly constrain roles to be filled by non-overlapping spans. For example, Toutanova, Haghighi, and Manning (2005) used dynamic programming over a phrase structure tree to prevent overlaps between arguments, and Punyakanok, Roth, and Yih (2008) used constraints in an ILP to respect this requirement. Inspired by the latter, we require that each input sentence position of $\mathbf{x}$ be covered by at most one argument of t. We define:

$$\mathcal{G}(i) = \{s \mid s \in \mathcal{S}, s \text{ covers position } i \text{ in } \mathbf{x}\}. \tag{13}$$

We can define our overlap constraints in terms of $\mathcal{G}$ as follows, for every sentence position i:

$$\forall i \in \{1, \ldots, |\mathbf{x}|\}, \quad \sum_{r \in \mathcal{R}_f} \sum_{s \in \mathcal{G}(i)} z_{r,s} \leq 1, \tag{14}$$

This gives us $O(|\mathbf{x}|)$ constraints. It is worth noting that this constraint aims to achieve the same effect as beam search, as described in Section 6.3, which tries to avoid argument overlap greedily.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Pairwise "Exclusions". For many target classes, there are pairs of roles forbidden to appear together in the analysis of a single target token. Consider the following two sentences:

(1) $\underbrace{\text{A blackberry}}_{\text{Entity\_1}}$ **resembles** $\underbrace{\text{a loganberry}}_{\text{Entity\_2}}$.

(2) $\underbrace{\text{Most berries}}_{\text{Entities}}$ **resemble** each other.

Consider the uninflected target **resemble** in both sentences, evoking the same meaning. In example 1 above, two roles—which we call Entity_1 and Entity_2—describe two entities that are similar to each other. In the second sentence, a phrase fulfills a third role, called Entities, that collectively denotes some objects that are similar. It is clear that the roles Entity_1 and Entities cannot be overt for the same target at once, because the latter already captures the function of the former; a similar argument holds for the Entity_2 and Entities roles. We call this phenomenon the "excludes" relationship. Let us define a set of pairs from $\mathcal{R}_f$ that have this relationship:

$$\mathit{Excl}_f = \{(r_i, r_j) \mid r_i \text{ and } r_j \textit{ exclude} \text{ each other}\}$$

Using the above set, we define the constraint:

$$\forall (r_i, r_j) \in \mathit{Excl}_f,\; z_{r_i,\emptyset} + z_{r_j,\emptyset} \geq 1 \tag{15}$$

If both roles are overt in a parse, this constraint will be violated, contravening the "excludes" relationship specified between the pair of roles. If neither or only one of the roles is overt, the constraint is satisfied. The total number of such constraints is $O(|\mathit{Excl}_f|)$, which is the number of pairwise "excludes" relationships of a given frame.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Pairwise "Requirements". The sentence in example 1 illustrates another kind of constraint. The target **resemble** cannot have only one of Entity_1 and Entity_2 as roles in text. For example,

(3) * $\underbrace{\text{A blackberry}}_{\text{Entity\_1}}$ **resembles**.

Enforcing the overtness of two roles sharing this "requires" relationship is straightforward. We define the following set for a frame f:

$$\mathit{Req}_f = \{(r_i, r_j) \mid r_i \text{ and } r_j \textit{ require} \text{ each other}\}$$

This leads to constraints of the form

$$\forall (r_i, r_j) \in \mathit{Req}_f, z_{r_i,\emptyset} - z_{r_j,\emptyset} = 0 \tag{16}$$

If one role is overt (or absent), the other must be as well. A related constraint has been used previously in the SRL literature, enforcing joint overtness relationships between core arguments and referential arguments (Punyakanok, Roth, and Yih 2008, Section 3.4.1), which are formally similar to the example above.[42]

7.1.2 Integer Linear Program and Relaxation. Plugging the constraints in Equations 12, 14, 15 and 16 into the last line of Equation 11, we have the argument identification problem expressed as an ILP, since the indicator variables $\mathbf{z}$ are binary. Here, apart from the ILP formulation, we will consider the following *relaxation* of Equation 11,

[42] We noticed that, in the annotated data, in some cases, the "requires" constraint is violated by the FrameNet annotators. This happens mostly when one of the required roles is absent in the sentence containing the target, but is rather instantiated in an earlier sentence (Gerber and Chai 2010). We apply the hard constraint in Equation 16, though extending our algorithm to seek arguments outside the sentence is straightforward. For preliminary work extending SEMAFOR this way, see Chen et al. (2010).

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

which replaces the binary constraint $\mathbf{z} \in \{0, 1\}^d$ by a unit interval constraint $\mathbf{z} \in [0, 1]^d$, yielding a *linear* program:

$$\begin{aligned} \text{maximize} \quad & \sum_{r \in \mathcal{R}_f} \sum_{s \in \mathcal{S}} c(r, s) \times z_{r,s} \\ \text{with respect to} \quad & \mathbf{z} \in [0, 1]^d \\ \text{such that} \quad & \mathbf{A}\mathbf{z} \leq \mathbf{b}. \end{aligned} \tag{17}$$

There are several LP and ILP solvers available, and a great deal of effort has been spent by the optimization community to devise efficient generic solvers. An example is CPLEX, a state-of-the-art solver for mixed integer programming that we employ as a baseline to solve the ILP in Equation 11 as well as its LP relaxation in Equation 17. Like many of the best implementations, CPLEX is proprietary.

7.1.3 Linguistic Constraints from FrameNet. Although enforcing the four different sets of constraints above is intuitive from a general linguistic perspective, we ground their use in definitive linguistic information present in the FrameNet lexicon. From the annotated data in the FrameNet 1.5 release, we gathered that only 3.6% of the time is a role instantiated multiple times by different spans in a sentence. This justifies the uniqueness constraint enforced by Equation 12. Use of such a constraint is also consistent with prior work in frame-semantic parsing (Johansson and Nugues 2007). Similarly, we found that in the annotations, no arguments overlapped with each other for a given target. Hence, the overlap constraints in Equation 14 are also justified.

Our third and fourth sets of constraints, presented in Equations 15 and 16, come from FrameNet, too. Examples 1–2 are instances where the target **resemble** evokes the SIMILARITY frame, which is defined in FrameNet as:

> Two or more distinct entities, which may be concrete or abstract objects or types, are characterized as being similar to each other. Depending on figure/ground relations, the

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

> entities may be expressed in two distinct frame elements and constituents, Entity_1 and Entity_2, or jointly as a single frame element and constituent, Entities.

For this frame, the lexicon lists several roles other than the three we have already observed, such as Dimension (the dimension along which the entities are similar), Differentiating_fact (a fact that reveals how the concerned entities are similar or different), and so forth. Along with the roles, FrameNet also declares the "excludes" and "requires" relationships noted in our discussion in Section 7.1.1. The case of the SIMILARITY frame is not unique; in Figure 1, the frame COLLABORATION, evoked by the target **partners**, also has two roles Partner_1 and Partner_2 that share the "requires" relationship. In fact, out of 877 frames in FrameNet 1.5, 204 frames have at least a pair of roles for which the "excludes" relationship holds, and 54 list at least a pair of roles that share the "requires" relationship.

7.1.4 Constraints as Factors in a Graphical Model. The LP in Equation 17 can be represented as a maximum *a posteriori* (MAP) inference problem in an undirected graphical model. In the factor graph, each component ($z_{r,s}$) of the vector **z** corresponds to a binary variable, and each instantiation of a constraint in Equations 12, 14, 15 and 16 corresponds to a factor. Smith and Eisner (2008) and Martins et al. (2010) used such a representation to impose constraints in a dependency parsing problem; the latter discussed the equivalence of linear programs and factor graphs for representing discrete optimization problems. All of our constraints take standard factor forms we can describe using the terminology of Smith and Eisner (2008) and Martins et al. (2010). The uniqueness constraint in Equation 12 corresponds to an XOR factor, while the overlap constraint in Equation 14 corresponds to an ATMOSTONE factor. The constraints in Equation 15 enforcing the "excludes" relationship can be represented with an OR factor.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Finally, each "requires" constraints in Equation 16 is equivalent to an XORWITHOUTPUT factor.

In the following section, we describe how we arrive at solutions for the LP in Equation 17 using dual decomposition, and how we adapt it to efficiently recover the *exact* solution of the ILP (Equation 11), without the need of an off-the-shelf ILP solver.

7.2 "Augmented" Dual Decomposition

Dual decomposition methods address complex optimization problems in the dual, by dividing them into simple worker problems (*subproblems*), which are repeatedly solved until a consensus is reached. The simplest technique relies on the subgradient algorithm (Komodakis, Paragios, and Tziritas 2007; Rush et al. 2010); as an alternative, Martins et al. (2011a, 2011b) proposed an augmented Lagrangian technique, which is more suitable when there are many small components—commonly the case in declarative constrained problems, like the one at hand. Here, we present a brief overview of the latter, which is called *Alternating Directions Dual Decomposition* (AD^3).

Let us start by establishing some notation. Let $m \in \{1, \ldots, M\}$ index a factor, and denote by $\mathbf{i}(m)$ the vector of indices of variables linked to that factor. (Recall that each factor represents the instantiation of a constraint.) We introduce a new set of variables, $\mathbf{u} \in \mathbb{R}^d$, called the "witness" vector. We split the vector $\mathbf{z}$ into M overlapping pieces $\mathbf{z}_1, \ldots, \mathbf{z}_M$, where each $\mathbf{z}_m \in [0,1]^{|\mathbf{i}(m)|}$, and add M constraints $\mathbf{z}_m = \mathbf{u}_{\mathbf{i}(m)}$ to impose that all the pieces must agree with the witness (and therefore with each other). Each of the M constraints described in Section 7.1 can be encoded with its own matrix $\mathbf{A}_m$ and vector $\mathbf{b}_m$ (which jointly define $\mathbf{A}$ and $\mathbf{b}$ in Equation 17). For convenience, we denote by $\mathbf{c} \in \mathbb{R}^d$ the score vector, whose components are $c(r, s)$, for each $r \in \mathcal{R}_f$ and $s \in \mathcal{S}$

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

(Equation 10), and define the following scores for the mth subproblem:

$$c_m(r,s) = \delta(r,s)^{-1} c(r,s), \; \forall (r,s) \in \mathbf{i}(m)$$

where $\delta(r,s)$ is the number of constraints that involve role r and span s. Note that according to this definition, $\mathbf{c} \cdot \mathbf{z} = \sum_{m=1}^{M} \mathbf{c}_m \cdot \mathbf{z}_m$. We can rewrite the LP in Equation 17 in the following equivalent form:

$$\begin{aligned} \text{maximize} \quad & \sum_{m=1}^{M} \mathbf{c}_m \cdot \mathbf{z}_m \\ \text{with respect to } \mathbf{u} \in \mathbb{R}^d, \; & \mathbf{z}_m \in [0,1]^{\mathbf{i}(m)}, \quad \forall m \\ \text{such that} \quad & \mathbf{A}_m \mathbf{z}_m \le \mathbf{b}_m, \quad \forall m \\ & \mathbf{z}_m = \mathbf{u}_{\mathbf{i}(m)}, \quad \forall m. \end{aligned} \tag{18}$$

We introduce Lagrange multipliers $\boldsymbol{\lambda}_m$ for the equality constraints in the last line. The AD^3 algorithm is depicted as Algorithm 2. Like dual decomposition approaches, it repeatedly performs a *broadcast* operation (the $\mathbf{z}_m$-updates, which can be done in parallel, one constraint per "worker") and a *gather* operation (the $\mathbf{u}$- and $\boldsymbol{\lambda}$-updates). Each $\mathbf{u}$-operation can be seen as an averaged voting which takes into consideration each worker's results.

Like in the subgradient method, the $\boldsymbol{\lambda}$-updates can be regarded as price adjustments, which will affect the next round of $\mathbf{z}_m$-updates. The only difference with respect to the subgradient method (Rush et al. 2010) is that each subproblem involved in a $\mathbf{z}_m$-update also has a quadratic penalty that penalizes deviations from the previous average voting; it is this term that accelerates consensus and therefore convergence. Martins et al. (2011b) also provide stopping criteria for the iterative updates using primal and dual residuals that measure convergence; we refer the reader to that paper for details.

Algorithm 2 AD^3 for Argument Identification

```
Input: role-span matching scores $\mathbf{c} := \langle c(r,s) \rangle_{r,s}$, structural constraints $\langle \mathbf{A}_m, \mathbf{b}_m \rangle_{m=1}^{M}$,
    penalty $\rho > 0$
initialize $t \leftarrow 1$
initialize $\mathbf{u}^1$ uniformly (i.e., $u(r,s) = 0.5,\ \forall r, s$)
initialize each $\boldsymbol{\lambda}_m^1 = \mathbf{0},\ \forall m \in \{1, \ldots, M\}$
repeat
    for each $m = 1, \ldots, M$ do
        make a $\mathbf{z}_m$-update by finding the best scoring analysis for the $m$th constraint,
        with penalties for deviating from the consensus $\mathbf{u}$:
            $$\mathbf{z}_m^{(t+1)} \leftarrow \underset{\mathbf{A}_m \mathbf{z}_m^t \leq \mathbf{b}_m}{\operatorname{argmax}} \ (\mathbf{c}_m + \boldsymbol{\lambda}_m^t) \cdot \mathbf{z}_m - \frac{\rho}{2} \|\mathbf{z}_m - \mathbf{u}_{\mathbf{i}(m)}^t\|^2 \qquad (19)$$
    end for
    make a $\mathbf{u}$-update by updating the consensus solution, averaging $\mathbf{z}_1, \ldots, \mathbf{z}_m$:
        $$u^{(t+1)}(r,s) \leftarrow \frac{1}{\delta(r,s)} \sum_{m:(r,s) \in \mathbf{i}(m)} z_m^{(t+1)}(r,s)$$
    make a $\boldsymbol{\lambda}$-update:
        $$\boldsymbol{\lambda}_m^{(t+1)} \leftarrow \boldsymbol{\lambda}_m^t - \rho(\mathbf{z}_m^{(t+1)} - \mathbf{u}_{\mathbf{i}(m)}^{(t+1)}), \quad \forall m$$
    $t \leftarrow t + 1$
until convergence.
Output: relaxed primal solution $\mathbf{u}^*$ and dual solution $\boldsymbol{\lambda}^*$. If $\mathbf{u}^*$ is integer, it will encode
    an assignment of spans to roles. Otherwise, it will provide an upper bound of the
    true optimum.
```

A key attraction of this algorithm is that all the components of the declarative specification remain intact in the procedural form. Each worker corresponds exactly to one constraint in the ILP, which corresponds to one linguistic constraint. There is no need to work out *when*, during the procedure, each constraint might have an effect, as in beam search.

7.2.1 Solving the subproblems. In a different application, Martins et al. (2011b, §4) showed how to solve each $\mathbf{z}_m$-subproblem associated with the XOR, XORWITHOUTPUT and OR factors in runtime $O(|\mathbf{i}(m)| \log |\mathbf{i}(m)|)$. The only subproblem that remains is that of the ATMOSTONE factor; a solution with the same runtime is given in Appendix B.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

7.2.2 Exact decoding. It is worth recalling that AD^3, like other dual decomposition algorithms, solves a *relaxation* of the actual problem. Although we have observed that the relaxation is often tight—cf. Section 7.3—this is not always the case. Specifically, a fractional solution may be obtained, which is not interpretable as an argument, and therefore it is desirable to have a strategy to recover the exact solution. Two observations are noteworthy. First, the optimal value of the relaxed problem (Equation 17) provides an upper bound to the original problem (Equation 11). This is because Equation 11 has the additional integer constraint on the variables. In particular, any feasible dual point provides an upper bound to the original problem's optimal value. Second, during execution of the AD^3 algorithm, we always keep track of a sequence of feasible dual points. Therefore, each iteration constructs tighter and tighter upper bounds. With this machinery, we have all that is necessary for implementing a branch-and-bound search that finds the exact solution of the ILP. The procedure works recursively as follows:

1. Initialize $L = -\infty$ (our best value so far).
2. Run Algorithm 2. If the solution $\mathbf{u}^*$ is integer, return $\mathbf{u}^*$ and set L to the objective value. If along the execution we obtain an upper bound less than L, then Algorithm 2 can be safely stopped and return "infeasible"—this is the *bound* part. Otherwise (if $\mathbf{u}^*$ is fractional) go to step 3.
3. Find the "most fractional" component of $\mathbf{u}^*$ (call it u_j^*) and *branch*: constrain $u_j = 0$ and go to step 2, eventually obtaining an integer solution $\mathbf{u}_0^*$ or infeasibility; and then constrain $u_j = 1$ and do the same, obtaining $\mathbf{u}_1^*$. Return the $\mathbf{u}^* \in \{\mathbf{u}_0^*, \mathbf{u}_1^*\}$ that yields the largest objective value.

Although this procedure may have worst-case exponential runtime, we found it empirically to rapidly obtain the exact solution in all test cases.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

7.3 Results with Collective Argument Identification

We present experiments only on argument identification in this section, as our goal is to exhibit the importance of incorporating the various linguistic constraints during our inference procedure. We present results on the full text annotations of FrameNet 1.5, and do not experiment on the SemEval 2007 benchmark, as we have already established our constraint-agnostic models as state-of-the-art. The model weights ψ used in the scoring function c were learned as in Section 6.1, i.e. by training a logistic regression model to maximize conditional log-likelihood. The AD^3 parameter ρ was initialized to 0.1, and we followed Martins et al. (2011b) in dynamically adjusting it to keep a balance between the primal and dual residuals.

We compare the following algorithms to demonstrate the efficacy of our collective argument identification approach:[43]

1. **Naïve**: This strategy selects the best span for each role r according to the score function $c(r, s)$, independently of all other roles—the decoding rule formalized in Equation 7 of Section 6.1. It ignores all constraints except "uniqueness."
2. **Beam**: This strategy employs greedy beam search to eliminate overlaps between predicted arguments, as described in Algorithm 1. Note that it does not try to respect the "excludes" and "requires" constraints between pairs of roles. The default size of the beam in Section 1 was a safe 10,000; this resulted in extremely slow decoding times. For time comparison, we tried beam sizes of 100 and 2 (the latter being the smallest size that achieves the same F_1 score on the FrameNet 1.5 dev set).

[43] The first two strategies correspond to rows 7 and 9, respectively, of Table 8.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

3. **CPLEX**, *LP*: This uses CPLEX to solve the relaxed LP in Equation 17. To handle fractional $\mathbf{z}$, for each role r, we choose the best span s^* such that $s^* = \mathrm{argmax}_{s \in \mathcal{S}_r} z_{r,s}$, breaking ties arbitrarily.
4. **CPLEX**, *exact*: This tackles the actual ILP (Equation 11) with CPLEX.
5. **AD**^{3}, *LP*: The relaxed problem is solved using AD^3. We choose a span for each role as in strategy 3.
6. **AD**^{3}, *exact*: This couples AD^3 with branch-and-bound search to get the exact integer solution.

Table 9 shows performance of these decoding strategies on the test set. We report precision, recall, and F_1 scores. As with experiments in previous sections, we use the evaluation script from SemEval 2007 shared task. Since these scores do not penalize constraint violations, we also report the number of overlap, “excludes” and “requires” constraints that were violated in the test set. Finally, we tabulate each setting’s decoding time in seconds on the whole test set averaged over 5 runs.[44] The naïve model is very fast but suffers degradation in precision and violates one constraint roughly per nine targets. The decoding strategy of Section 6.1 used a default beam size of 10,000, which is extremely slow; a faster version of beam size 100 results in the same precision and recall values, but is 15 times faster on our test set. Beam size 2 results in slightly worse precision and recall values, but is even faster. All of these, however, result in many constraint violations. Strategies involving CPLEX and AD^3 perform similarly to each other and to beam search on precision and recall, but eliminate most or all of the constraint violations. With respect to precision and recall, exact AD^3 and beam search with a width of 10,000 were found to be statistically indistinguishable ($p > 0.01$). The

[44] Experiments were conducted on a 64-bit machine with 2 2.6GHz dual-core CPUs (i.e., 4 processors in all) and a total of 8GB of RAM. The workers in AD^3 were not parallelized, while CPLEX automatically parallelized execution.

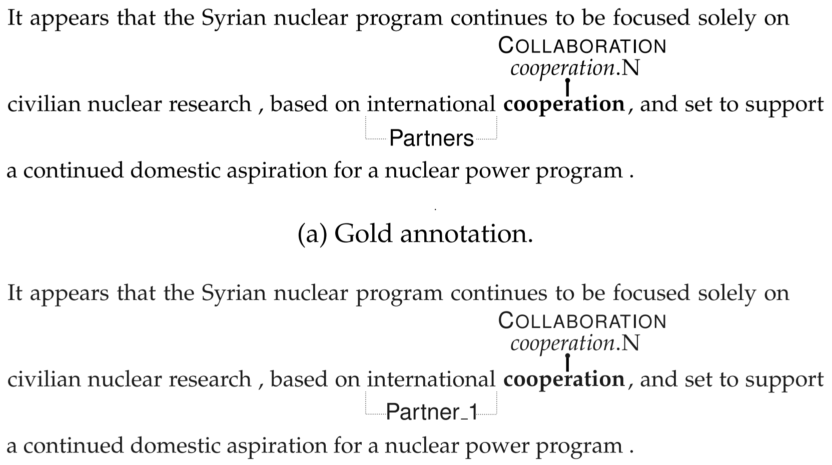
ARGUMENT IDENTIFICATION

Method	P	R	F_1	**Violations**			**Time (s)**	
naïve	82.00	76.36	79.08	441	45	15	1.26	± 0.01
beam = 2	83.68	76.22	79.78	0	49	0	2.74	± 0.10
beam = 100	83.83	76.28	79.88	0	50	1	29.00	± 0.25
beam = 10000	83.83	76.28	79.88	0	50	1	440.67	± 5.53
CPLEX, *LP*	83.80	76.16	79.80	0	1	0	32.67	± 1.29
CPLEX, *exact*	83.78	76.17	79.79	0	0	0	43.12	± 1.26
AD^3, *LP*	83.77	76.17	79.79	2	2	0	4.17	± 0.01
AD^3, *exact*	83.78	76.17	79.79	0	0	0	4.78	± 0.04

Table 9: Comparison of decoding strategies in Section 7.3 on the dataset released with the **FrameNet 1.5 Release**, given *gold* frames. We evaluate in terms of precision, recall and F_1 score on our test set containing 4,458 targets. We also compute the number of constraint violations each model makes: the three values are the numbers of overlapping arguments and violations of the "requires" and "excludes" constraints of Section 7.1. Finally, decoding time (without feature computation steps) on the *whole* test set is shown in the last column averaged over 5 runs.

decoding strategy with beam size 2 is 11–16 times faster than the CPLEX strategies, but is only twice as fast as AD^3, and results in significantly more constraint violations. The exact algorithms are slower than the LP versions, but compared to CPLEX, AD^3 is significantly faster and has a narrower gap between its exact and LP versions. We found that relaxation was tight 99.8% of the time on the test examples.

The example in Figure 1 is taken from our test set, and shows an instance where two roles, Partner_1 and Partner_2, share the "requires" relationship; for this example, the beam search decoder misses the Partner_2 role, which is a violation, while our AD^3 decoder identifies both arguments correctly. Note that beam search makes plenty of linguistic violations. We found that beam search, when violating many "requires" constraints, often finds one role in the pair, which increases its recall. AD^3 is sometimes more conservative in such cases, predicting neither role. Figure 6 shows such an example where beam search finds one role (Partner_1) while AD^3 is more conservative and predicts no roles. Figure 7 shows another example contrasting the output of beam search and AD^3 where the former predicts two roles sharing an "excludes" relationship;


It appears that the Syrian nuclear program continues to be focused solely on
COLLABORATION
cooperation.N
civilian nuclear research , based on international **cooperation**, and set to support
Partners
a continued domestic aspiration for a nuclear power program .

(a) Gold annotation.

It appears that the Syrian nuclear program continues to be focused solely on
COLLABORATION
cooperation.N
civilian nuclear research , based on international **cooperation**, and set to support
Partner_1
a continued domestic aspiration for a nuclear power program .

(b) Beam search output.

Figure 6: An example from the test set where (a) exhibits the gold annotation for a target that evokes the COLLABORATION frame, with the Partners role filled by the span "international." (b) shows the prediction made by the beam search decoding scheme (beam = 10,000), where it marks "international" with the Partner_1 role, violating the "requires" constraint; FrameNet notes that this role should be present with the Partner_2 role. AD^3 is conservative and predicts no role—it is penalized by the evaluation script, but does not produce output that violates linguistic constraints.

AD^3 does not violate this constraint and tries to predict a more consistent argument set. Overall, we found it interesting that imposing the constraints did not have much effect on standard measures of accuracy.

Table 9 only shows results with gold frames. We ran the exact version of AD^3 with automatic frames as well. When the semi-supervised graph objective **UJSF-**$\ell_{1,2}$ is used for frame identification, the performance with AD^3 is only a bit worse in comparison to beam search (row 11 in Table 8) when frame and argument identification are evaluated together. We get a precision of 72.92, a recall of 65.22 and F_1 score of 68.86 (partial frame matching). Again, all linguistic constraints are respected, unlike beam search.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

8. Conclusion

We have presented an approach to rich frame-semantic parsing, based on a combination of knowledge from FrameNet, two probabilistic models trained on full text annotations released along with the FrameNet lexicon, and expedient heuristics. The frame identification model employs latent variables in order to generalize to predicates unseen in either the FrameNet lexicon or training data, and our results show that, quite often, this model chooses a frame closely related to the gold-standard annotation. We also presented an extension of this model that uses graph-based semi-supervised learning to better generalize to new predicates; this achieves significant improvements over the fully supervised approach. Our argument identification model, trained using maximum conditional log-likelihood, unifies the traditionally separate steps of detecting and labeling arguments. Our system achieves improvements over the previous state of the art on the SemEval 2007 benchmark dataset at each stage of processing and collectively. We also report stronger results on the more recent, larger FrameNet 1.5 release.

We applied the AD^3 algorithm to collective prediction of a target's arguments, incorporating declarative linguistic knowledge as constraints. It outperforms the naïve local decoding scheme that is oblivious to the constraints. Furthermore, it is significantly faster than a decoder employing a state-of-the-art proprietary solver; it is only twice as slow as beam search (our chosen decoding method for comparison with the state of the art), which is inexact and does not respect all linguistic constraints. This method is easily amenable to the inclusion of additional constraints.

From our results, we observed that in comparison to the SemEval 2007 dataset, frame-semantic parsing performance significantly increases when we use the FrameNet 1.5 release; this suggests that the increase in the number of full text annotations and the size of the FrameNet lexicon is beneficial. We believe that with more annotations in the

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

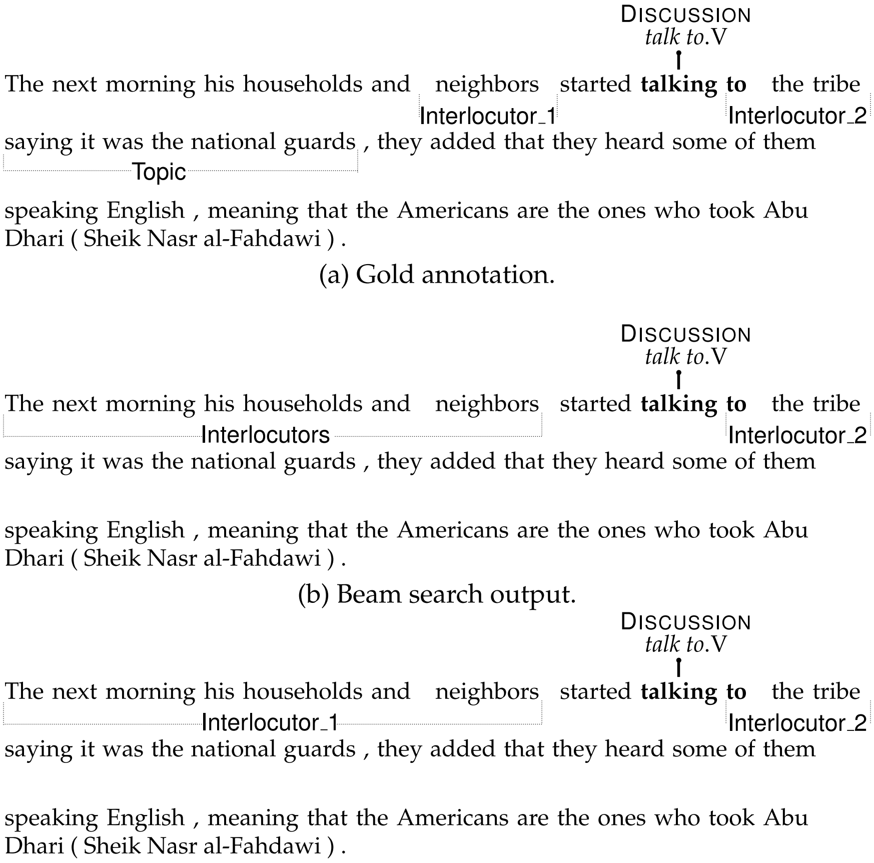
future (say, in the range of the number of PropBank annotations), our frame-semantic parser can reach even better accuracy, making it more useful for NLP applications that require semantic analysis.

There are several open problems to be addressed. Firstly, we could further improve the coverage of the frame-semantic parser by improving our semi-supervised learning approach; two possibilities are custom metric learning approaches (Dhillon, Talukdar, and Crammer 2010) that suit the frame identification problem in graph-based SSL, and sparse word representations (Turian, Ratinov, and Bengio 2010) as features in frame identification. The argument identification model might also benefit from semi-supervised learning. Further feature engineering and improved preprocessing, including tokenization into lexical units, improved syntactic parsing, and the use of external knowledge bases, is expected to improve the system's accuracy. Finally, the FrameNet lexicon does not contain exhaustive semantic knowledge. Automatic frame and role induction is an exciting direction of future research that could further enhance our methods of automatic frame-semantic parsing. The parser described in this article is available for download at `http://www.ark.cs.cmu.edu/SEMAFOR`.

Acknowledgments

We thank Collin Baker, Katrin Erk, Richard Johansson, and Nils Reiter for software, data, evaluation scripts, and methodological details. We thank the reviewers of this and the earlier papers, Alan Black, Ric Crabbe, Michael Ellsworth, Rebecca Hwa, Dan Klein, Russell Lee-Goldman, Slav Petrov, Dan Roth, Josef Ruppenhofer, Amarnag Subramanya, Partha Talukdar, and members of the ARK group for helpful comments. This work was supported by DARPA grant NBCH-1080004, NSF grants IIS-0836431 and IIS-0915187, Qatar National Research Foundation grant NPRP 08-485-1-083, Google's support of the Worldly Knowledge Project at CMU, computational resources provided

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163


DISCUSSION
talk to.V
The next morning his households and neighbors started **talking to** the tribe saying it was the national guards , they added that they heard some of them speaking English , meaning that the Americans are the ones who took Abu Dhari (Sheik Nasr al-Fahdawi) .
Interlocutor_1
Interlocutor_2
Topic

(a) Gold annotation.

DISCUSSION
talk to.V
The next morning his households and neighbors started **talking to** the tribe saying it was the national guards , they added that they heard some of them speaking English , meaning that the Americans are the ones who took Abu Dhari (Sheik Nasr al-Fahdawi) .
Interlocutors
Interlocutor_2

(b) Beam search output.

DISCUSSION
talk to.V
The next morning his households and neighbors started **talking to** the tribe saying it was the national guards , they added that they heard some of them speaking English , meaning that the Americans are the ones who took Abu Dhari (Sheik Nasr al-Fahdawi) .
Interlocutor_1
Interlocutor_2

(c) AD^3 output.

Figure 7: An example from the test set where (a) exhibits the gold annotation for a target that evokes the DISCUSSION frame, with the Interlocutor_1 role filled by the span "neighbors." (b) shows the prediction made by the beam search decoding scheme (beam = 10,000), where it marks "The next morning his households and neighbors" with the Interlocutors role, which violates the "excludes" constraint with respect to the Interlocutor_2 role. In (c), AD^3 marks the wrong span as the Interlocutor_1 role, but it does not violate the constraint. Both beam and AD^3 inference miss the Topic role.

by Yahoo, and TeraGrid resources provided by the Pittsburgh Supercomputing Center under grant TG-DBS110003.

References

Auli, Michael and Adam Lopez. 2011. A comparison of loopy belief propagation and dual decomposition for integrated CCG supertagging and parsing. In *Proceedings of the 49th Annual*

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 470–480, Portland, Oregon, USA, June. Association for Computational Linguistics.

Baker, Collin, Michael Ellsworth, and Katrin Erk. 2007. SemEval-2007 task 19: Frame semantic structure extraction. In *Proceedings of the Fourth International Workshop on Semantic Evaluations*, pages 99–104, Prague, Czech Republic, June. Association for Computational Linguistics.

Baluja, Shumeet, Rohan Seth, D. Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. 2008. Video suggestion and discovery for YouTube: taking random walks through the view graph. In *Proceedings of the 17th international conference on World Wide Web*, pages 895–904, Beijing, China. ACM.

Bauer, Daniel and Owen Rambow. 2011. Increasing coverage of syntactic subcategorization patterns in FrameNet using VerbNet. In *Proceedings of the 2011 IEEE Fifth International Conference on Semantic Computing*, pages 181–184, Washington, DC, USA. IEEE Computer Society.

Bejan, Cosmin A. 2009. *Learning Event Structures From Text*. Ph.D. thesis, The University of Texas at Dallas.

Bengio, Yoshua, Olivier Delalleau, and Nicolas Le Roux. 2006. Label propagation and quadratic criterion. In Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien, editors, *Semi-Supervised Learning*. MIT Press, pages 193–216.

Boas, Hans C. 2002. Bilingual FrameNet dictionaries for machine translation. In *Proceedings of the Third International Conference on Language Resources and Evaluation*, pages 1364–1371, Las Palmas, Canary Islands, Span. ELRA.

Bottou, Léon. 2004. Stochastic learning. In Olivier Bousquet and Ulrike von Luxburg, editors, *Advanced Lectures on Machine Learning*, Lecture Notes in Artificial Intelligence, LNAI 3176. Springer Verlag, Berlin, pages 146–168.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Burbea, Jacob and Calyampudi R. Rao. 2006. On the convexity of some divergence measures based on entropy functions. *IEEE Transactions on Information Theory*, 28(3):489–495, September.

Burchardt, Aljoscha, Katrin Erk, and Anette Frank. 2005. A WordNet detour to FrameNet. In Bernhard Fisseni, Hans-Christian Schmitz, Bernhard Schröder, and Petra Wagner, editors, *Sprachtechnologie, mobile Kommunikation und linguistische Resourcen*, volume 8 of *Computer Studies in Language and Speech*. Peter Lang, Frankfurt am Main, pages 408–421.

Burchardt, Aljoscha and Anette Frank. 2006. Approaching textual entailment with LFG and FrameNet frames. In *Proceedings of the Second PASCAL RTE Challenge Workshop*, pages 92–97, Venice, Italy, April.

Burchardt, Aljoscha, Marco Pennacchiotti, Stefan Thater, and Manfred Pinkal. 2009. Assessing the impact of frame semantics on textual entailment. *Natural Language Engineering*, 15(4):527–550, October.

Carreras, Xavier and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning*, pages 89–97, Boston, Massachusetts, USA, June. Association for Computational Linguistics.

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

Chang, Yin-Wen and Michael Collins. 2011. Exact decoding of phrase-based translation models through lagrangian relaxation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 26–37, Edinburgh, Scotland, UK, July. Association for Computational Linguistics.

Chapelle, Olivier, Bernhard Schölkopf, and Alexander Zien, editors. 2006. *Semi-Supervised Learning*. MIT Press.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Chen, Desai, Nathan Schneider, Dipanjan Das, and Noah A. Smith. 2010. SEMAFOR: Frame argument resolution with log-linear models. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 264–267, Upssala, Sweden. Association for Computational Linguistics.

Corduneanu, Adrian and Tommi Jaakkola. 2003. On information regularization. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 151–158, Acapulco, Mexico. Morgan Kaufmann Publishers Inc.

Das, Dipanjan, André F. T. Martins, and Noah A. Smith. 2012. An exact dual decomposition algorithm for shallow semantic parsing with constraints. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 209–217, Montréal, Canada, 7-8 June. Association for Computational Linguistics.

Das, Dipanjan and Slav Petrov. 2011. Unsupervised part-of-speech tagging with bilingual graph-based projections. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 600–609, Portland, Oregon, USA, June. Association for Computational Linguistics.

Das, Dipanjan, Nathan Schneider, Desai Chen, and Noah A. Smith. 2010. Probabilistic frame-semantic parsing. In *Proceedings of the Human Language Technologies Conference of the North American Chapter of the Association for Computational Linguistics*, pages 948–956, Los Angeles, California, June. Association for Computational Linguistics.

Das, Dipanjan and Noah A. Smith. 2011. Semi-supervised frame-semantic parsing for unknown predicates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1435–1444, Portland, Oregon, USA, June. Association for Computational Linguistics.

Das, Dipanjan and Noah A. Smith. 2012. Graph-based lexicon expansion with sparsity-inducing penalties. In *Proceedings of the Human Language Technologies Conference of the North American Chapter of the Association for Computational Linguistics*, pages 677–687, Montréal, Canada, June. Association for Computational Linguistics.

Dean, Jeffrey and Sanjay Ghemawat. 2008. MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, January.

DeNero, John and Klaus Macherey. 2011. Model-based aligner combination using dual decomposition. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 420–429, Portland, Oregon, USA, June. Association for Computational Linguistics.

Deschacht, Koen and Marie-Francine Moens. 2009. Semi-supervised semantic role labeling using the Latent Words Language Model. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 21–29, Singapore, August. Association for Computational Linguistics.

Dhillon, Paramveer S., Partha Pratim Talukdar, and Koby Crammer. 2010. Learning better data representation using inference-driven metric learning. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 377–381, Uppsala, Sweden, July. Association for Computational Linguistics.

Erk, Katrin and Sebastian Padó. 2006. Shalmaneser - a toolchain for shallow semantic parsing. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, pages 527–532, Genoa, Italy. ELRA.

Fellbaum, Christiane, editor. 1998. *WordNet: an electronic lexical database*. MIT Press.

Fillmore, Charles J. 1982. Frame semantics. In *Linguistics in the Morning Calm*. Hanshin Publishing Co., Seoul, South Korea, pages 111–137.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Fillmore, Charles J., Christopher R. Johnson, and Miriam R.L. Petruck. 2003. Background to FrameNet. *International Journal of Lexicography*, 16.3:235–250.

Fleischman, Michael, Namhee Kwon, and Eduard Hovy. 2003. Maximum entropy models for FrameNet classification. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 49–56, Sapporo, Japan. Association of Computational Linguistics.

Fung, Pascale and Benfeng Chen. 2004. BiFrameNet: bilingual frame semantics resource construction by cross-lingual induction. In *Proceedings of the 20th international conference on Computational Linguistics*, pages 931–937, Geneva, Switzerland. Association for Computational Linguistics.

Fürstenau, Hagen and Mirella Lapata. 2009a. Graph alignment for semi-supervised semantic role labeling. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 11–20, Singapore, August. Association for Computational Linguistics.

Fürstenau, Hagen and Mirella Lapata. 2009b. Semi-supervised semantic role labeling. In *Proceedings of the 12th Conference of the European Chapter of the ACL*, pages 220–228, Athens, Greece, March. Association for Computational Linguistics.

Fürstenau, Hagen and Mirella Lapata. 2012. Semi-supervised semantic role labeling via structural alignment. *Computational Linguistics*, 38(1):135–171, March.

Gerber, Matthew and Joyce Chai. 2010. Beyond NomBank: A study of implicit arguments for nominal predicates. In *Proceedings of ACL*, pages 1583–1592, Uppsala, Sweden, July. Association for Computational Linguistics.

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

Girju, Roxana, Preslav Nakov, Vivi Nastase, Stan Szpakowicz, Peter Turney, and Deniz Yuret. 2007. SemEval-2007 task 04: Classification of semantic relations between nominals. In

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Proceedings of the Fourth International Workshop on Semantic Evaluations, pages 13–18, Prague, Czech Republic, June. Association for Computational Linguistics.

Giuglea, Ana-Maria and Alessandro Moschitti. 2006. Shallow semantic parsing based on FrameNet, VerbNet and PropBank. In *Proceedings of the 17th European Conference on Artificial Intelligence*, pages 563–567, Amsterdam, The Netherlands. IOS Press.

Gropp, W., E. Lusk, and A. Skjellum. 1994. *Using MPI: Portable Parallel Programming with the Message-Passing Interface*. MIT Press.

Hajič, Jan, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 1–18, Boulder, Colorado, June. Association for Computational Linguistics.

Ide, Nancy and Jean Véronis. 1998. Introduction to the special issue on word sense disambiguation: the state of the art. *Computational Linguistics*, 24(1):2–40, March.

Johansson, Richard and Pierre Nugues. 2007. LTH: semantic structure extraction using nonprojective dependency trees. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 227–230, Prague, Czech Republic. Association for Computational Linguistics.

Johansson, Richard and Pierre Nugues. 2008. Dependency-based semantic role labeling of PropBank. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 69–78, Honolulu, Hawaii, October. Association for Computational Linguistics.

Kingsbury, Paul and Martha Palmer. 2002. From TreeBank to PropBank. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, pages 1989–1993, Las Palmas, Spain, May. ELRA.

Komodakis, Nikos, Nikos Paragios, and Georgios Tziritas. 2007. MRF optimization via dual decomposition: Message-passing revisited. In *Eleventh International Conference on Computer Vision*, pages 1–8, Rio de Janeiro, Brazil, October. IEEE Press.

Koo, Terry, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. 2010. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1288–1298, Cambridge, MA, October. Association for Computational Linguistics.

Kowalski, Matthieu and Bruno Torrésani. 2009. Sparsity and persistence: mixed norms provide simple signal models with dependent coefficients. *Signal, Image and Video Processing*, 3:251–264.

Lang, Joel and Mirella Lapata. 2010. Unsupervised induction of semantic roles. In *Proceedings of the Human Language Technologies Conference of the North American Chapter of the Association for Computational Linguistics*, pages 939–947, Los Angeles, California, June. Association for Computational Linguistics.

Lang, Joel and Mirella Lapata. 2011. Unsupervised semantic role induction with graph partitioning. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1320–1331, Edinburgh, Scotland, UK, July. Association for Computational Linguistics.

Lin, Dekang. 1993. Principle-based parsing without overgeneration. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 112–120, Columbus, Ohio. Association for Computational Linguistics.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Lin, Dekang. 1994. Principar–an efficient, broad-coverage, principle-based parser. In *Proceedings of the 15th conference on Computational linguistics*, pages 482–488, Kyoto, Japan. Association for Computational Linguistics.

Lin, Dekang. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 768–774, Montreal, Quebec, Canada. Association for Computational Linguistics.

Lin, Jianhua. 1991. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151.

Litkowski, Kenneth C. and Orin Hargraves. 2007. SemEval-2007 task 06: Word-sense disambiguation of prepositions. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 24–29, Prague, Czech Republic, June. Association for Computational Linguistics.

Liu, Dong C. and Jorge Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(3).

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

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

Martins, André F. T., Mario A. T. Figueiredo, Pedro M. Q. Aguiar, Noah A. Smith, and Eric P. Xing. 2011a. An augmented Lagrangian approach to constrained MAP inference. In *Proceedings of the 28th International Conference on Machine Learning*, pages 169–176, Bellevue, Washington, USA, June. ACM.

Martins, André F. T., Noah A. Smith, Pedro M. Q. Aguiar, and Mario A. T. Figueiredo. 2011b. Dual decomposition with many overlapping components. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 238–249, Edinburgh, Scotland, UK, July. Association for Computational Linguistics.

Martins, André F. T., Noah A. Smith, and Eric P. Xing. 2009. Concise integer linear programming formulations for dependency parsing. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association of Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 342–350, Suntec, Singapore, August. Association for Computational Linguistics.

Martins, André F. T., Noah A. Smith, Eric P. Xing, Mario A. T. Figueiredo, and Pedro M. Q. Aguiar. 2010. Turbo parsers: Dependency parsing by approximate variational inference. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 34–44, Cambridge, MA, October. Association for Computational Linguistics.

Matsubayashi, Yuichiroh, Naoaki Okazaki, and Jun'ichi Tsujii. 2009. A comparative study on generalization of semantic roles in FrameNet. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association of Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 19–27, Suntec, Singapore, August. Association for Computational Linguistics.

McDonald, Ryan, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 91–98, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Meyers, Adam, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank project: An interim report. In *Proceedings of*

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

the NAACL-HLT Workshop on Frontiers in Corpus Annotation, pages 24–31, Boston, Massachusetts, USA, May. Association for Computational Linguistics.

Moschitti, Alessandro, Paul Morarescu, and Sanda M. Harabagiu. 2003. Open-domain information extraction via automatic semantic labeling. In Ingrid Russell and Susan M. Haller, editors, *Proceedings of the Sixteenth International Florida Artificial Intelligence Research Society Conference*, pages 397–401, St. Augustine, Florida, USA, May. AAAI Press.

Narayanan, Srini and Sanda Harabagiu. 2004. Question answering based on semantic structures. In *Proceedings of the 20th international conference on Computational Linguistics*, Geneva, Switzerland. Association for Computational Linguistics.

Padó, Sebastian and Katrin Erk. 2005. To cause or not to cause: cross-lingual semantic matching for paraphrase modelling. In *Proceedings of the Cross-Language Knowledge Induction Workshop*, Cluj-Napoca, Romania, July.

Pado, Sebastian and Mirella Lapata. 2005. Cross-linguistic projection of role-semantic information. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 859–866, Vancouver, British Columbia, Canada, October. Association for Computational Linguistics.

Pennacchiotti, Marco, Diego De Cao, Roberto Basili, Danilo Croce, and Michael Roth. 2008. Automatic induction of FrameNet lexical units. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 457–465, Honolulu, Hawaii, October. Association for Computational Linguistics.

Pradhan, Sameer S., Wayne H. Ward, Kadri Hacioglu, James H. Martin, and Dan Jurafsky. 2004. Shallow semantic parsing using support vector machines. In *Proceedings of the Human Language Technologies Conference of the North American Chapter of the Association for Computational Linguistics,*, pages 233–240, Boston, Massachusetts, USA, May. Association for Computational

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Linguistics.

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

Punyakanok, Vasin, Dan Roth, Wen-tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proceedings of the 20th international conference on Computational Linguistics*, pages 1346–1352, Geneva, Switzerland. Association for Computational Linguistics.

Ratnaparkhi, Adwait. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the 1996 Empirical Methods in Natural Language Processing*, pages 133–142, Copenhagen, Denmark, August. Association for Computational Linguistics.

Riedel, Sebastian and James Clarke. 2006. Incremental integer linear programming for non-projective dependency parsing. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 129–137, Sydney, Australia, July.

Roth, Dan and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In Hwee Tou Ng and Ellen Riloff, editors, *Proceedings of the Eighth Conference on Computational Natural Language Learning*, pages 1–8, Boston, Massachusetts, USA, May 6 - May 7. Association for Computational Linguistics.

Ruppenhofer, Josef, Michael Ellsworth, Miriam R. L. Petruck, Christopher R. Johnson, and Jan Scheffczyk. 2006. FrameNet II: extended theory and practice.

Rush, Alexander M. and Michael Collins. 2011. Exact decoding of syntactic translation models through Lagrangian relaxation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 72–82, Portland, Oregon, USA, June. Association for Computational Linguistics.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Rush, Alexander M, David Sontag, Michael Collins, and Tommi Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1–11, Cambridge, MA, October. Association for Computational Linguistics.

Schuler, Karin K. 2005. *VerbNet: a broad-coverage, comprehensive verb lexicon*. Ph.D. thesis, University of Pennsylvania.

Sha, Fei and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 134–141, Edmonton, Alberta, Canada, May–June.

Shen, Dan and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 12–21, Prague, Czech Republic, June. Association for Computational Linguistics.

Shi, Lei and Rada Mihalcea. 2004. An algorithm for open text semantic parsing. In *Proceedings of Workshop on Robust Methods in Analysis of Natural Language Data*, pages 59–67, Geneva, Switzerland. Association for Computational Linguistics.

Shi, Lei and Rada Mihalcea. 2005. Putting pieces together: Combining FrameNet, VerbNet and WordNet for robust semantic parsing. In *Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 100–111, Mexico City, Mexico, February. Springer.

Smith, David A. and Jason Eisner. 2008. Dependency parsing by belief propagation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 145–156, Honolulu, Hawaii, October. Association for Computational Linguistics.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Subramanya, Amarnag and Jeff Bilmes. 2008. Soft-supervised learning for text classification. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1090–1099, Honolulu, Hawaii, October. Association for Computational Linguistics.

Subramanya, Amarnag, Slav Petrov, and Fernando Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 167–176, Cambridge, MA, October. Association for Computational Linguistics.

Surdeanu, Mihai, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 8–15, Sapporo, Japan. Association for Computational Linguistics.

Surdeanu, Mihai, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The CoNLL 2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 159–177, Manchester, England, August. Association for Computational Linguistics.

Szummer, Martin and Tommi Jaakkola. 2001. Partially labeled classification with Markov random walks. In *Advances in Neural Information Processing Systems 14*, pages 945–952, Vancouver, British Columbia, Canada, December. MIT Press.

Talukdar, Partha Pratim and Koby Crammer. 2009. New regularized algorithms for transductive learning. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 442–457, Bled, Slovenia. Springer-Verlag.

Thompson, Cynthia A., Roger Levy, and Christopher D. Manning. 2003. A generative model for semantic role labeling. In *Proceedings of the European Conference on Machine Learning*, pages 397–408, Cavtat-Dubrovnik, Croatia, September. Springer.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Titov, Ivan and Alexandre Klementiev. 2012. A Bayesian approach to unsupervised semantic role induction. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 12–22, Avignon, France, April. Association for Computational Linguistics.

Toutanova, Kristina, Aria Haghighi, and Christopher Manning. 2005. Joint learning improves semantic role labeling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 589–596, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Turian, Joseph, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden, July. Association for Computational Linguistics.

Weston, Jason, Frédéric Ratle, and Ronan Collobert. 2008. Deep learning via semi-supervised embedding. In *Proceedings of the 25th international conference on Machine learning*, pages 1168–1175, Helsinki, Finland. ACM.

Xue, Nianwen and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 88–94, Barcelona, Spain, July. Association for Computational Linguistics.

Yi, Szu-ting, Edward Loper, and Martha Palmer. 2007. Can semantic roles generalize across genres? In *Proceedings of the Human Language Technologies Conference of the North American Chapter of the Association for Computational Linguistics*, pages 548–555, Rochester, New York, April. Association for Computational Linguistics.

Zhu, Xiaojin. 2008. Semi-supervised learning literature survey. Online publication, July.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

Zhu, Xiaojin, Zoubin Ghahramani, and John Lafferty. 2003. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the 20th International Conference on Machine Learning*, pages 912–919, Washington, D.C., USA, August. AAAI Press.

A. Target Identification Heuristics from J&N'07

We describe here the filtering rules that Johansson and Nugues (2007) used for identifying frame evoking targets in their SemEval 2007 shared task paper. They built a filtering component based on heuristics that removed words that appear in certain contexts, and kept the remaining ones.[45] These are:

- *have* was retained only if had an object,
- *be* was retained only if it was preceded by *there*,
- *will* was removed in its modal sense,
- *of course* and *in particular* were removed,
- the prepositions *above*, *against*, *at*, *below*, *beside*, *by*, *in*, *on*, *over*, and *under* were removed unless their head was marked as locative,
- *after* and *before* were removed unless their head was marked as temporal,
- *into*, *to* and *through* were removed unless their head was marked as direction,
- *as*, *for*, *so* and *with* were always removed,
- since the only sense of the word *of* was the frame PARTITIVE, it was removed unless it was preceded by *only*, *member*, *one*, *most*, *many*, *some*, *few*,

[45] Although not explicitly mentioned in the paper, we believe that these rules were applied on a white list of potential targets seen in FrameNet and the SemEval 2007 training data.

Computational Linguistics Just Accepted MS.
doi: 10.1162/COLI_a_00163

part, *majority*, *minority*, *proportion*, *half*, *third*, *quarter*, *all*, or *none*, or it was followed by *all*, *group*, *them* or *us*,

- all targets marked as support verbs for some other target were removed.

Note that J&N'07 used a syntactic parser that provided dependency labels corresponding to locative, temporal and directional arguments, which our syntactic parser of choice (the MST parser) does not provide.

B. Solving ATMOSTONE **subproblems in AD^3**

The ATMOSTONE subproblem can be transformed into that of projecting a point $(a_1, \ldots, a_k)$ onto the set

$$\mathcal{S}_m = \left\{ \mathbf{z}_m \in [0,1]^{|\mathbf{i}(m)|} \;\middle|\; \textstyle\sum_{j=1}^{|\mathbf{i}(m)|} z_{m,j} \leq 1 \right\}.$$

This projection can be computed as follows:

1. Clip each a_j into the interval $[0,1]$ (*i.e.*, set $a'_j = \min\{\max\{a_j, 0\}, 1\}$). If the result satisfies $\sum_{j=1}^{k} a'_j \leq 1$, then return $(a'_1, \ldots, a'_k)$.
2. Otherwise project $(a_1, \ldots, a_k)$ onto the probability simplex:

$$\left\{ \mathbf{z}_m \in [0,1]^{|\mathbf{i}(m)|} \;\middle|\; \textstyle\sum_{j=1}^{|\mathbf{i}(m)|} z_{m,j} = 1 \right\}.$$

This is precisely the XOR subproblem and can be solved in time $O(|\mathbf{i}(m)| \log |\mathbf{i}(m)|)$.

The proof of this procedure's correctness follows from the proof in Appendix B of Martins et al. (2011b).