[NS S provisional title:]Leveraging Additional Resources for Frame-Semantic Role Labeling

Abstract

The high cost of semantic structure annotation is a major obstacle to automating semantic analysis with broad coverage. The fully annotated datasets that exist are often small, hindering the robustness of models trained on them. However, low-resource tasks may benefit from exploiting *partially* annotated data, as well as data with *different* (but related) forms of annotation, for additional training data or features. This paper considers the argument identification and classification subtask of frame-semantic parsing, which to date has relied exclusively upon full-text annotations in the FrameNet resource.[NS S is this true? I think Dipanjan used exemplars for the latent variable frame ID model, but he hasn't used them at all for arg ID, right?] We augment supervised learning with additional "indirect" training data and features so as to leverage additional resources internal and external to FrameNet (e.g., PropBank). [NS S result]

1 Introduction

[NS S sparseness is a challenge for many computational semantics tasks]

Frame-semantic parsing (Das et al., 2014) is a case in point. This is the task of automating the rich linguistic structure analyses of the FrameNet lexicon and corpus (Baker et al., 1998).[1] FrameNet represents kinds of events, scenarios, and relationships with an inventory of **frames** ([NS S examples]). Each frame is associated with lexical **predicates** (verbs, nouns, adjectives, and adverbs) capable of evoking the scenario, and a set of **roles** (or **frame elements**) called to mind in order to understand the scenario. These roles may be implicit, but are frequently realized linguistically in the same sentence as the predicate. Given a sentence, frame-semantic parsing is the task of mapping tokens in the sentences to evoked frames, and for each evoked frame, finding and labeling its **argument** phrases with roles. An example appears in figure 1; it will be explained in detail in §2.2.

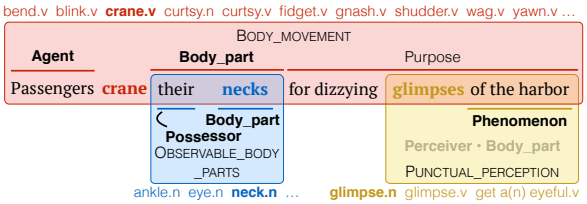

Figure 1: Example sentence from FrameNet full-text annotation. 3 frames and their arguments are shown: BODY_MOVEMENT is evoked by *crane*, OBSERVABLE_BODY_PARTS by *necks*, and PUNCTUAL_PERCEPTION by *glimpse*. (Further, *harbor* is annotated as evoking the LOCALE_BY_USE frame and doubles as its sole argument.) Horizontal lines representing argument spans are labeled with role names.

FrameNet 1.5 defines a structured hierarchy of over 1,000 frames associated with [NS S #] English lexical predicates, and also provides annotations for [NS S # targets annotated total] attestations of these frames/predicates in corpora, annotated in context with their arguments. In FrameNet 1.5, a rather small number of sentences—[NS S #], comprising [NS S #] words—are provided with **full-text** annotations, i.e. the sentence has been analyzed for all available frames. But a full [NS S #]% of sentences in FrameNet—the lexicographic **exemplars**—are annotated for only one frame per sentence, and have thus far not been exploited successfully for frame-semantic parsing. Here, we seek to leverage these exemplar sentences as well as the (type-level) hierarchical structure of the FrameNet lexicon.

In this paper, we address the **argument identification** subtask of finding and labeling arguments given a predicate in context and the frame

[1] http://framenet.icsi.berkeley.edu/

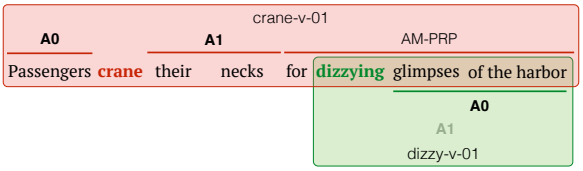

Figure 2: Ideal PropBank annotations for verbs. Though PB uses lexical frames rather than deep frames, there are clear similarities to the FrameNet annotations in figure 1.

it evokes. This is a form of semantic role labeling (SRL). [$^{S}_{T}$ cite Gildea, Roth, Toutanova, etc.] Notably, another resource, **PropBank** (Kingsbury and Palmer, 2002), has been widely used for SRL (Palmer et al., 2010). PropBank annotations capture shallower lexical frames and arguments; additionally, PropBank provides [$^{NS}_{S}$ millions?] of words of fully annotated English sentences (annotation is much less expensive, but also potentially less valuable, because of the shallower representation). Despite a number of differences in the representations and annotation conventions, for many predicates FrameNet and PropBank are really quite similar: figure 1 and figure 2 show this for the verb *crane*. To get the best of both worlds, we aim to tap into PropBank's vast resources as indirect token-level supervision for FrameNet-style analysis. We hypothesize that PropBank analyses can serve as a weak signal for the FrameNet SRL task, either by heuristically transforming PropBank annotations into FrameNet annotations to augment the training data, or by preprocessing sentences with a PropBank SRL system to obtain new features for FrameNet argument identification.

Our experiments expand the *training data* and/or the *feature space* of supervised argument identification in order to integrate evidence from all of these sources into SEMAFOR (Das et al., 2014), the leading open-source frame-semantic parser for English.[2] The results show that some of these sources of evidence succeed at boosting argument identification performance.[$^{NS}_{S}$ SOTA (without constraints)?]

2 Resources

[$^{NS}_{S}$ incl. data analysis of differences from full-text]

[$^{NS}_{S}$ w/in each mention: genre/overall vocabulary; coverage and distributions of predicates, FE labels; oracle coverage of FT test]

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

2.1 The FrameNet Lexicon

Each frame in the Berkeley FrameNet lexicon is intended to represent a gestalt scene. The frame definition includes: a descriptive name; a set of **core roles** representing participants and props that are crucial to understanding the scene; a set of **non-core roles** such as circumstantial information (time, place, manner, purpose, etc.); an English textual description of the scene and how its roles relate to one another; and a set of English predicates that can evoke the scene. For example, the BODY_MOVEMENT frame has **Agent** and **Body_part** as its core roles; the frame description states, "This frame contains words for motions or actions an **Agent** performs using some part of his/her body." Lexical entries in this frame include verbs such as bend, blink, crane, and curtsy, plus the noun use of curtsy.

[$^{NS}_{S}$ hierarchy. which relations do we care about—anything besides Inheritance?]

2.2 Full-text Annotations

Contemporary frame-semantic parsers are trained and evaluated on the **full-text (FT)**[3] portion of the FrameNet corpus. This consists of documents for which annotators made an effort to assign frames and arguments to as many words as possible. Figure 1 gives an example sentence from the FT portion of the corpus. It has 4 frame annotations. BODY_MOVEMENT, as described in the previous section, is evoked by *crane*, and 3 of its roles are filled by overt arguments: the 2 core roles (**Agent**, **Body_part**) happen to be filled by noun phrases (*passengers*, *their necks*), while the non-core role **Purpose** is filled by a prepositional phrase adjunct (*for dizzying glimpses of the harbor*). The frame defines 19 additional non-core roles, none of which have an argument in the example. In frame semantics, non-core roles are considered to be *conceptually* optional; core roles may or may not be *syntactically* optional, but if not locally specified they are expected to be available from context, or else implicit. For example, PUNCTUAL_PERCEPTION—evoked in this case by *glimpses*—is annotated as missing 2 of its core roles. A human listener would resolve the identity of the **Perceiver** from the wider context and the **Body_part** from world knowledge.

In some cases, FT annotation involved creating a new frame or adding a new lexical unit to an ex-

[3] Though these were *annotated* and the document level, and train/dev/test splits are by document, the frame-semantic parsing is currently restricted to the sentence level.

isting frame. In other cases, words that in principle should be considered to evoke a frame were left unannotated because they did not match any existing lexical units. This was the case for *passengers* and *dizzying* in figure 1.

Genres, sizes

Annotation density: (proportion of tokens evoking a frame. breakdown by POS?)

2.3 Exemplars

[$^{NS}_{S}$ how different from FT]

2.4 PropBank

[$^{NS}_{S}$ how different from FN]

2.5 SemLink

[$^{NS}_{S}$ limitations!]

2.6 Illinois SRL system

3 Learning from multiple domains and representations

We consider the argument identification task as statistical classification for each role of the frame in context. Let the classification input be a dependency-parsed sentence $\mathbf{x}$, the token(s) p constituting the predicate in question, and the frame f evoked by p (as determined by frame identification). Suppose we have a heuristic procedure for extracting candidate argument spans for the predicate: call this $spans(\mathbf{x}, p, f)$. We define the **local classifier** as:

$$arg(\mathbf{x}, p, f, r) = \operatorname*{arg\,max}_{a \in spans(\mathbf{x},p,f) \cup \{\varnothing\}} \text{score}_{\mathbf{w}}(a \mid \mathbf{x}, p, f, r) \quad (1)$$

This is used to choose an argument for every role r of frame f: note that in many cases the role will be empty (non-overt), denoted $\varnothing$. The classifier chooses the best value of $\text{score}_{\mathbf{w}}$, a linear function scoring the compatibility of a candidate role–argument pair, whose parameters (feature weights) $\mathbf{w}$ are learned from data. [$^{NS}_{S}$ mention global inference step]

[$^{M}_{K}$ introduce this as a supervised domain adaptation problem? What is the source and what's the target?] [$^{M}_{K}$ Let D_{ft} represent the FT data and D_{ex} the exemplars?]

We experiment with several techniques for modifying the model-fitting portion of the argument identification model's local learning objective (training data, features). All of our experiments use the same form of regularization, condition on the same frame predictions[$^{NS}_{S}$ not oracle, right? these are SEMAFOR's current frame ID model, so not state of the art?] and syntactic preprocessing[$^{NS}_{S}$ does this match Dipanjan's latest experiments?], and use beam search with [$^{NS}_{S}$ hyperparam value] for joint decoding of arguments.[4]

[$^{NS}_{S}$ Domain adaptation/multitask learning techniques]

3.1 Augmenting the Training Data

3.2 Frustratingly Easy

Daumé III (2009) proposed a simple feature augmentation approach that was shown to work well in supervised domain adaptation scenarios, such as ours. We apply their method to incorporate the exemplars data by introducing a feature mapping $\Phi^{ft}(x) = (x, x)$ for the FT annotations $x \in D_{ft}$, and $\Phi^{ex}(x) = (x, 0)$ for the exemplars $x \in D_{ex}$. The resulting transformed data from the two domains is then used to learn a single model. The intuition here is that by replicating the features, we allow for each feature to contribute both "general" and domain-specific weights to the model depending on whether the feature behaves similarly in both domains or not.

3.3 Guide Features

Another approach to introduce supervision for domain adaptation, which does not combine the labeled data from the two domains, to use a supervised model built on the source domain to augment features in the target domain. When training and testing in the target domain, the source domain model is effectively just a form of preprocessing to provide additional features, known as **guide features** (**??**).[5] Formally: let M_s be the model built on the source domain (for instance, the PropBank data). For every target instance x, we introduce "guide" features which use the output $M_s(x)$ obtained by applying M_s on x, which consists of the role labels assigned to various text spans in x. Two types of guide features were used, one to indicate that a span s_x was assigned a role and the other encoding the role label r_g itself. In the case where M_s produces labels that belong to the same schema as the target domain (for instance, the exemplars

[4] [$^{NS}_{S}$ recent work has improved upon global decoding techniques; we expect such improvements to be complementary to the gains due to the local model reported here]

[5] This is related to the technique of model stacking, where successively richer models are trained by cross-validation on the same dataset **???**.

use the same schema as the FT annotations), we use an additional feature $f_{match}(r_t, r_g)$ to indicate that the 'guide' role label r_g of the span s_x is the same as it's true label r_t.

3.4 Type-level hierarchy features

[M K NSS: notation for the frame/role/features etc?] [M K How many total types of relations? Cleanup writeup based on NSS's notation.]

Frames in FrameNet are connected to each other by relations such as inheritance, temporal ordering, causality. For instance, the frame ROBBERY inherits from the more abstract frame COMMITTING_CRIME, the frame FALL_ASLEEP is preceeded by the frame BEING_AWAKE. The roles of related frames have also been mapped to indicate the correspondence between them: ROBBERY.**Perpetrator** is mapped to COMMITTING_CRIME.**Perpetrator**, which in turn maps to MISDEED.**Wrongdoer**. Frames and roles that are far apart in this hierarchy are less related than say neighbours. This hierarchy can be exploited to share information across related roles, thereby benefiting the roles that have few annotations [M K say something about a greater variety of contexts is available for each role]. A simple mechanism to share information is via shared model parameters between related roles. Towards this, we define two types of hierarchical features: (1) 'siblings', where roles that have a common parent share features. For every feature f_i that fires for an argument a, we add a feature which is the conjunction: $(f_i \wedge \text{parent.frame} \wedge \text{parent.role} \wedge I_{hier})$ where parent.frame = parent(frame(a)). I_{hier} is an indicator to distinguish this feature from the regular conjunction features in Semafor that use frame names and roles. (2) 'parents+siblings', where roles share features with their parents and siblings. In addition to the 'siblings' feature, we fire the following conjunction: $(f_i \wedge \text{frame} \wedge \text{frame.role} \wedge I_{hier})$. We use only the 'Inheritance' and 'Subframe' relations between the roles, of which there are 4138 and 589 respectively.

We found that using more than one level of the hierarchy, does not produce any improvements in the performance at the cost of increased computation due to the greater number of features.

[M K Write about the number of features in all models somewhere.. Baseline: 2709128, Combined training: 12972404, Frust-easy: 15681531, 'Parents+siblings': 34282958]

4 Experiments

4.1 Baseline

Our baseline system is SEMAFOR (Das et al., 2014). SEMAFOR treats the argument identification task as a structured prediction problem. It uses a linearly parametrized model that scores each candidate span given a frame element[S T this should probably be talked about earlier?]:

$$\text{score}_{\mathbf{w}}(y \mid x) = \mathbf{w}^{\top}\mathbf{f}(x, y) \tag{2}$$

Automatic syntactic dependency parses from MST-ParserStacked (**?**) are used to narrow the set of candidate spans considered, and as input to feature extraction.

During training, each frame element is treated as an independent log-linear classification instance, with the set of candidate spans (including the NULL span) as its output space. [S T something about L2 regularization] But at test time, it chooses a joint assignment of all arguments that maximizes probability under the model, while satisfying the following constraints:

1. an argument may be assigned to at most one span [S T theta criterion, Chomsky], and
2. spans of realized arguments must not overlap.

Beam search, with a beam size of 100, is used to choose the maximum joint assignment with no overlapping arguments. [NS S does beam search require normalizing to probabilities?]

We have made several modifications to SEMAFOR's training in order to speed up experiments:

- We use squared structured hinge loss (defined below) instead of multiclass logistic regression. Using hinge loss, there is no longer a need to calculate a partition function. Gradients, and hence parameters, are sparser than in logistic regression, allowing us to use a sparse vector implementation.
- We use the online optimization method AdaDelta (Zeiler, 2012) with minibatches[NS S minibatch size?], instead of the batch method L-BFGS (Liu and Nocedal, 1989).

We use these changes for all systems, including the baseline. While the impact on full-text performance is negligible, these changes enabled us

to run more experiments with the larger exemplar dataset and expanded feature space.[6]

The details of squared hinge loss are as follows. Let $(x^{(i)}, y^{(i)})$ be the i^{th} training example. Then the structured hinge loss on the i^{th} example is given by:

$$Hinge_{\mathbf{w}}(i) = \max_{y} \{\mathbf{w}^\top \mathbf{f}(x^{(i)}, y) + \text{cost}(y, y^{(i)})\} - \mathbf{w}^\top \mathbf{f}(x^{(i)}, y^{(i)})$$

and squared hinge loss is:

$$SquaredHinge_{\mathbf{w}}(i) = Hinge_{\mathbf{w}}(i)^2. \quad (3)$$

In our experiments, we use $\text{cost}(y, y^{(i)}) = \mathbf{1}\{y \neq y^{(i)}\}$ [S_T define][7].

[NS_S same FN 1.5 splits as Dipanjan]

[NS_S preprocessing issues: removing duplicate sentences, merging adjacent split args in exemplars, OntoNotes PropBank preprocessing (NLTK), token-level SemLink details (such as filtering out sentences without mappable annotations; copy from WS paper)]

[NS_S tuning regularizer for all experiments] We tune the ℓ_2 regularization parameter λ on the held-out set. We searched over the following values: 10^{-5}, 10^{-7}, 10^{-9}, 10^{-12} (note that our loss is normalized). We also use the performance on the held-out set to determine the stopping criterion for the stochastic optimization.

4.2 Evaluation

[NS_S main eval: FT test; new eval: exemplars] The methods are evaluated on two criteria based on their performance measured using F-score: FT test data and Exemplars test.

4.3 Results

[NS_S results table without hierarchy features]

[NS_S hierarchy features: which ones work best (decided on baseline), how do they improve best result so far]

[NS_S Sam's curves on per-FE F_1]

[NS_S comparison to prior work (baseline, best result). args+frames score vs. args only]

[NS_S discussion throughout]

[6] With SEMAFOR's original features and training data, the result of the above changes is that full-text F_1 decreases from 59.3% to 59.1%, while training time (running optimization to convergence) decreases from 729 minutes to 82 minutes[NS_S say something about hardware this was tested on?].

[7] We experimented with recall-oriented training, where errors of omission are assigned a higher cost, but found that while recall increased, overall F_1 went down[NS_S or: failed to improve?].

	Full-Text	Exemplars
Sentences in training	2,780	145,838
Roles in train data	15,019	137,511
Roles (types) in training	2,644	4,821
Sentences in test data	2,420	4,132
Roles in test data	4,458	4,132
Roles (types) in test data	1,420	1,224
Unseen roles (types) in test data	219	38

Table 1: Characteristics of the training and test data

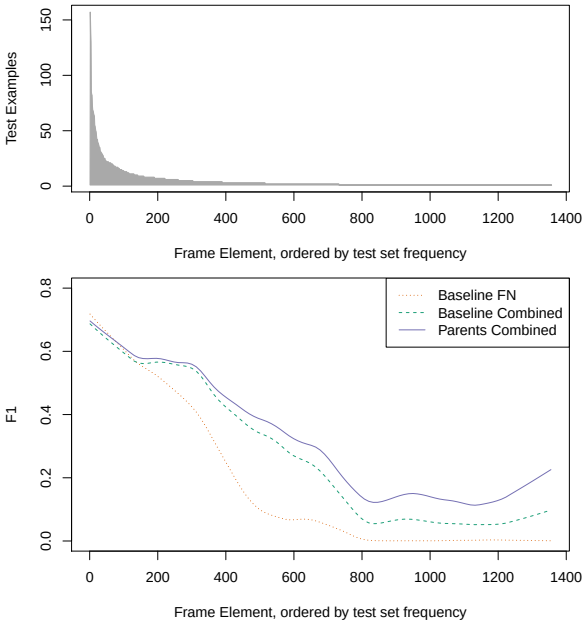

Figure 3: Count and F_1 for each frame element appearing in the test set. F_1 values have been smoothed with `loess`, with a smoothing parameter of 0.2.

5 Related Work

[NS_S Dipanjan's other papers; mention other PB SRL work?; anything using SemLink or combining resources for SRL?]

[NS_S multitask learning?]

6 Conclusion

[NS_S overall findings]

[NS_S future work: testing ground for improvements to PB and SemLink; automatic mappings between resources]

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proc. of COLING-ACL*, pages 86–90. Montreal, Quebec, Canada.

Dipanjan Das, Desai Chen, André F. T. Martins, Nathan Schneider, and Noah A. Smith. 2014.

Additional Resource	Training Configuration (Features)	Full-Text			Exemplars		
		P	R	F_1	P	R	F_1
(Baseline)	FT (Basic)	66.03	53.79	59.29	64.90	33.60	44.27
FN Hierarchy	FT (siblings)	67.24	54.76	60.36	64.81	39.09	48.77
	FT (siblings+parents)	67.67	52.79	59.31	65.25	38.18	48.18
Exemplars	Exemplars $\xrightarrow{\text{guide}}$ FT	65.24	55.96	60.24	67.71	48.08	56.23
	FT+Exemplars (Basic)	66.06	58.23	61.9	75.44	65.11	69.89
	FT+Exemplars (EasyAdapt)	65.7	59.04	62.19	73.88	61.4	67.06
SemLink	SemLink $\xrightarrow{\text{guide}}$ FT	64.67	54.53	59.17	60.95	38.92	47.5
	FT+SemLink	65.5	37.8	47.9	57.15	20.8	30.5
PB-SRL	FT (PB-SRL)						

Table 2: Results on two test sets: Baseline vs. individual other resources. Precision, recall, and F_1 are given as percentages.

FT+Exemplars (Hier: siblings+parents)
FT+Exemplars (PB-SRL)
FT+Exemplars (PB-SRL, Hier: siblings+parents)

Table 3: Combining best techniques across resources [$^{\text{NS}}_{\text{S}}$ TODO]

	Test on FN			Test on Exemplars		
	P	R	F_1	P	R	F_1
Semafor baseline (Ddas' model)	0.6603	0.5379	0.5929	0.64933	0.33582	0.44269
Semafor baseline (Sam's code)	0.65569	0.53820	0.59116	0.6263	0.3765	0.4703
baseline trained on combined	0.66061	0.58234	0.61901	0.75443	0.65107	0.69895
trained only on exemplars	0.61084	0.49049	0.54409	0.77010	0.65958	0.71057
trained on FN + exemplars guide features	0.65241	0.55960	0.60245	0.67709	0.48076	0.56228
frust† on combined	0.65702	0.59043	0.62195	0.73876	0.61397	0.67061
siblings‡ on FN	0.67244	0.54763	0.60365	0.64815	0.39088	0.48766
siblings, trained on combined	0.65991	0.60406	0.63075	0.76140	0.67713	0.71679
parents* on FN	0.67672	0.52790	0.59312	0.65250	0.38184	0.48176
parents, trained on combined	0.65920	0.60382	0.63029	0.76143	0.68317	0.72018
trained on semlink	0.44433	0.10533	0.17029	0.48119	0.12498	0.19842
trained on FN + semlink guide features	0.64671	0.54533	0.59171	0.60951	0.38922	0.47507
trained on FN+SemLink	0.655	0.3776	0.4791	0.57148	0.20780	0.30478
with SRL augmented spans and features (loglinear)	0.70550	0.53178	0.60644	0.65996	0.33351	0.44310
with SRL augmented spans and features (hinge)						

combined: FN + Exemplars training data

†: feature augmentation from the frustratingly easy DA paper

‡: for every feature f_i that fires for an argument a, fire an additional feature which is the conjunction: $(f_i \wedge$ parent.frame $\wedge$ parent.role $\wedge I_{hier})$ where parent.frame = parent(frame(a)). The parent's frame and role are obtained from the FN hierarchy. I_{hier} is an indicator to distinguish this feature from the regular conjunction features that use frame names and roles.

$\star$: fire the siblings feature‡ and an additional feature: $(f_i \wedge \text{frame} \wedge \text{frame.role} \wedge I_{hier})$

Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56.

Hal Daumé III. 2009. Frustratingly easy domain adaptation. *arXiv preprint arXiv:0907.1815*.

Paul Kingsbury and Martha Palmer. 2002. From TreeBank to PropBank. In *Proc. of LREC*, pages 1989–1993. Las Palmas, Canary Islands.

Dong C. Liu and Jorge Nocedal. 1989. On the Limited Memory BFGS Method for Large Scale Optimization. *Math. Program.*, 45(3):503–528.

Martha Palmer, Daniel Gildea, and Nianwen Xue. 2010. *Semantic Role Labeling*. Number 6 in Synthesis Lectures on Human Language Technologies. Morgan & Claypool, San Rafael, CA.

Matthew Zeiler. 2012. ADADELTA: An Adaptive Learning Rate Method. *CoRR*, abs/1212.5701. URL `http://dblp.uni-trier.de/rec/bibtex/journals/corr/abs-1212-5701`.