Neural Models for Large-Scale Semantic Role Labelling

Nicholas FitzGerald

A dissertation submitted in partial fulfillment of

the requirements for the degree of

Doctor of Philosophy

University of Washington

2018

Reading Committee:

Luke Zettlemoyer, Chair

Noah Smith

Yejin Choi

Program Authorized to Offer Degree:

Computer Science and Engineering

University of Washington

Abstract

Neural Models for Large-Scale Semantic Role Labeling

Nicholas FitzGerald

Chair of the Supervisory Committee:
Professor Luke Zettlemoyer
Computer Science and Engineering

Recovering predicate-argument structures from natural language sentences is an important task in natural language processing (NLP), where the goal is to identify "who did what to whom" with respect to events described in a sentence. A key challenge in this task is sparsity of labeled data: a given predicate-role instance may only occur a handful of times in the training set. While attempts have been made to collect large, diverse datasets which could help mitigate this sparseness, these effort are hampered by the difficulty inherent in labelling traditional SRL formalisms such as PropBank and FrameNet.

We take a two-pronged approach to solving these issues. First, we develop models which can be used to jointly represent multiple SRL annotation schemes, allowing us to pool annotations between multiple datasets. We present a new method for semantic role labeling in which arguments and semantic roles are jointly embedded in a shared vector space for a given predicate. We further show how the model can learn jointly from PropBank and FrameNet annotations to obtain additional improvements on the smaller FrameNet dataset.

Next, we demonstrate that crowdsourcing techniques can be used to collect a large, high-quality SRL dataset at much lower cost than previous methods, and that this data can be used to learn a high-quality SRL parser. Our corpus, QA-SRL Bank 2.0, consists of over 250,000 question-answer pairs for over 64,000 sentences across 3 domains and was gathered with a new crowd-sourcing scheme that we show has high precision and good recall at modest cost. We also present neural models for two QA-SRL subtasks: detecting argument spans for a predicate and generating questions to label

the semantic relationship.

Finally, we combine these two approaches, investigating whether QA-SRL annotations can be used to improve perfomance on PropBank in a multitask learning setup. We find that using the QA-SRL data improves performance in regimes with small amounts of in-domain PropBank data, but that these improvements are overshadowed by those obtained by using deep contextual word representations trained on large amounts of unlabeled text, raising important questions for future work as to the utility of multitask training relative to these unsupervised approaches.

Acknowledgement

Completing a PhD is impossible without the help and support of many wonderful people.

I owe a deep debt of gratitude to my advisor Luke Zettlemoyer, who has been my greatest guide and advocate throughout this journey – indulging me in my exuberance, coaching me through the disappointments, and helping me pick my way through the difficult terrain that is a PhD. Thank you to the other members of my committee, Noah Smith and Yejin Choi, to Dieter Fox, who provided valuable guidance early in my program, and to the late Ben Taskar.

Thank you to my uncle and aunt, Prof. Robin Allshire, F.R.S. and Prof. Wendy Bickmore, F.R.S., who kindled and encouraged by my nascent interest in research. I would not be where I am today without the early support and encouragement of Prof. Giuseppe Carenini at UBC, who guided and championed me throughout the latter years of my undergraduate degree, and my application to grad schools. Thank you also to Profs. Christina Conati and Mike Tyers.

Thank you to Dipanjan Das, who guided and supported me during two internships at Google, and to the many wonderful people I had the opportunity to work with there, including Oscar Täckström, Kuzman Ganchev, Slav Petrov, Michael Collins, David Weiss, and Emily Pitler.

Thank you to Yoav Artzi, Cynthia Matuszek, and Mark Yatskar, who were the first to welcome me to UW and have remained my steadfast friends. I have had the privilege to work and live alongside many wonderful fellow grad students and post-docs at UW: Matthew Kay, Ricardo Martin, Eunsol Choi, Robert Gens, Tony Fader, Abe Friesen, Tom Kwiatkowski, Mike Lewis, Cyrus Rashtchian, Chloé Kiddon, Swabha Swayamdipta, Maarten Sap, Sam Thompson, Julian Michael,

Kenton Lee, Luheng He, Victoria Lin, Yannis Konstas, Liefeng Bo, Paris Koutris, Omer Levy, Antoine Bosselut, Daniel Perlman, Dallas Card, Yonatan Bisk, Aditya Sankar, Gabriel Schubiner and many more. Thank you also to the wonderful UW CSE staff who have assisted me, including Lindsay Michimoto, Elise DeGoede, Joel Cohn, and Chiemi Yamaoka.

I would not have long endured the rigours of this journey without a long list of friends who have been a constant source of support, needed distraction, and joy. Thank you to Tyson Gratton, Maly Oudommahavanh, Amy Height, Jake Waltier, Simon Hlywa, Cameron Hassal, Benjamin Keller, Eric Campbell, Kyle Smith, Richard Bowen, Barry Nelipowitz, Julija Lazukaite, Jacob Best, Graham Riach, Daniel McNamara, Victor Soto, and many, many more. Thanks to the GBX Crew, the Cafe Mox crew, and the Raygun Lounge regulars for all the games, and Alex Sturbaum and the musicians of Murphy's and Shawn O'Donnell's for all the tunes. Thank you to Laura Vianna for the happiness you have brought to my life the last few years.

Finally, thank you to my parents, Zinda and Robin, for their constant support and encouragement and to my brothers, Graham and William.

DEDICATION

To my parents, Zinda and Robin

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Semantic Role Labelling – the problem of recovering predicate-argument structures from natural language sentences – is an important task in natural language processing (NLP), where the goal is to identify "who did what to whom" with respect to events described in a sentence. For example, given the sentence "John ate the burrito at 2am", we would like to know that "John" is the one doing the eating, "the burrito" is the thing being eaten, and "2am" is the time at which this occurred. Recovering these shallow semantic structures of this sort has been shown to improve performance on important NLP tasks such as Question Answering [Shen and Lapata, 2007] and Machine Translation [Liu and Gildea, 2010], and these structures are also used as a semantic interface for information extraction [Danilevsky et al., 2018; Stanovsky et al., 2018]. Consequently, training wide coverage SRL parsers which can work across multiple domains is an important problem.

For the past decade-and-a-half, research has largely focused around the related formalisms of PropBank [Palmer et al., 2005a] and FrameNet [Baker et al., 1998a], thanks in large part to the creation of large annotated datasets [Pradhan et al., 2013; Meyers et al., 2004], as well as multiple Shared Tasks [Carreras and Màrquez, 2005a; Surdeanu et al., 2008a; Hajič et al., 2009a]. A key challenge in this task is sparsity of labeled data: a given predicate-role instance may only occur a handful of times in the training set. While attempts have been made to large, diverse datasets

which could help mitigate this sparseness Weischedel et al. [2011], these effort are hampered by the complicated nature of traditional SRL formalisms: The PropBank and FrameNet annotation guidelines are 89 and 119 pages long, respectively, and require trained linguists to execute.

In this thesis, I take a two-pronged approach to solving these issues. First, I develop models which can be used to jointly model multiple SRL annotation schemes, allowing us to pool annotations between multiple datasets. Next, I demonstrate that crowdsourcing techniques can be used to collect a large, high-quality SRL dataset at much lower cost than previous methods, and that this data can be used to learn a high-quality SRL parser. Finally, I combine these two approaches, and investigate whether our new data can be used to improve performance on traditional SRL tasks.

Multitask Learning for Multiple SRL Formalisms Most existing SRL systems model each semantic role as an atomic unit of meaning, ignoring finer-grained semantic similarity between roles that can be leveraged to share context between similar labels, both within and across annotation conventions. In chapter 3, we present a new model for SRL that embeds candidate arguments and semantic roles (in context of a predicate frame) in a shared vector space. A feed-forward neural network is learned to capture correlations of the respective embedding dimensions to create argument and role representations. The similarity of these two representations, as measured by their dot product, is used to score possible roles for candidate arguments within a graphical model. This graphical model jointly models the assignment of semantic roles to all arguments of a predicate, subject to structural linguistic constraints. Experiments show that we can leverage the larger CoNLL 2005 PropBank dataset Carreras and Màrquez [2005a] in order to improve performance on the smaller FrameNet Baker et al. [1998a] task, achieving what was, at the time, state-of-the-art performance on this dataset.

Collecting SRL Data Next, we tackle the difficulty of collecting new SRL data. Recent research has explored training non-experts to provide this style of semantic supervision [Abend and Rap-

poport, 2013; Basile et al., 2012; Reisinger et al., 2015; He et al., 2015], but we show for the first time that it is possible to go even further by crowdsourcing a large dataset that can be used to train high quality parsers at modest cost. We adopt the Question-Answer-driven Semantic Role Labeling (QA-SRL) annotation scheme [He et al., 2015]. QA-SRL is appealing because it is intuitive to non-experts, has been shown to closely match the structure of traditional predicate-argument structure annotation schemes [He et al., 2015], and has been used for end tasks such as Open IE Stanovsky and Dagan [2016]. Using a streamlined web interface, we collect QA-SRL Bank 2.0, a dataset with 133,479 verbs from 64,018 sentences across 3 domains, totaling 265,140 question-answer pairs, in just 9 days. Our analysis shows that the data has high precision with good recall.

Using this data, we perform a comparison of several new models for learning a QA-SRL parser. We follow a pipeline approach where the parser does (1) unlabeled *span detection* to determine the arguments of a given verb, and (2) *question generation* to label the relationship between the predicate and each detected span. Our best model uses a span-based representation similar to that introduced by Lee et al. [2016] and a custom LSTM to decode questions from a learned span encoding. Our model does not require syntactic information and can be trained directly from the crowdsourced span labels. Our experiments demonstrate that we can learn a high quality SRL parser from our crowd-sourced data.

Multitask Learning with QA-SRL Putting it all together, we experiment with using our new large-scale QA-SRL dataset and parser in order to improve performance on existing SRL formalisms, by learning a model which embeds QA-SRL labels in the same space.

A related question emerges from recent work in deep, contextualized word representations. ELMo [Peters et al., 2018] is a recent method for learning deep contextualized word representations trained to perform language modeling on an unlabeled text corpus consisting of billions of tokens. Incorporating these representations into existing models has proven to significantly improve the state of the art on a wide range of NLP tasks. These improvements in many cases exceed what has

been achieved by way of multitask training. It is unknown whether the improvements obtained by pretraining a deep word representation to perform language modeling are complementary to those achievable by multitask training with a more similar scaffold task.

We investigate multitask training for PropBank Semantic Role Labelling on the CoNLL 2005 dataset, with the addition of QA-SRL as a scaffold task. We additionally experiment with the inclusion of ELMo. Our results indicate that while multitask training with QA-SRL can provide improvements when there is little in-domain data, these improvements are overshadowed by those obtained by incorporating ELMo embeddings into the model, suggesting that multitask training may simply be learning improved lexical representations which can be more effectively obtained by language modeling on large unlabeled corpora.

Thesis Outline To recap, we make three main contributions:

1. First, in chapter 3, we present a neural network model for semantic role labeling in which arguments and semantic roles are jointly embedded in a shared vector space. This model allows us to train jointly using multiple different annotation schemes (namely PropBank and FrameNet), which allows us to leverage datasets with a lot of annotations in order to improve performance on those with less.

2. Next, in chapter 4, we introduce a large, crowdsourced dataset of QA-SRL annotations, and the first high-quality QA-SRL parser. This dataset rivals the size of the largest PropBank dataset [Weischedel et al., 2011], but is collected at a fraction of the cost by utilizing non-expert crowd workers. Our model demonstrates that a high-quality semantic parser can be learned from this data.

3. Finally, in chapter 5, we investigate multitask training for PropBank SRL, using our large QA-SRL dataset as an auxilliary task. While we find that using the QA-SRL data improves performance in regimes with small amounts of in-domain PropBank data, but that these

improvements are overshadowed by those obtained by using deep contextual word representations trained on large amounts of unlabeled text [Peters et al., 2018]. This experiment raises important questions for future work as to the utility of multitask training relative to these unsupervised approaches.

Chapter 2

Background and Related Work

2.1 Semantic Role Labelling

SRL annotations rely on a frame lexicon containing *frames* that could be evoked by one or more *lexical units*. A lexical unit consists of a word lemma conjoined with its coarse-grained part-of-speech tag.[1] Each frame is further associated with a set of possible *core* and *non-core* semantic roles which are used to label its arguments. This description of a frame lexicon covers both PropBank and FrameNet conventions, but there are some differences outlined below. See Figure 2.1 for example annotations.

PropBank defines frames that are essentially sense distinctions of a given lexical unit. The set of PropBank roles consists of seven generic core roles (labeled A0-A5 and AA) that assume different semantics for different frames, each associating with a subset of the core roles. In addition, there are 21 non-core roles that encapsulate further arguments of a frame, such as temporal (AM-TMP) and locative (AM-LOC) adjuncts. The non-core roles are shared between all frames and assume similar meaning. In contrast, a FrameNet frame often associates with multiple lexical units and the frame

[1] We borrow the term "lexical unit" from the frame semantics literature. The CoNLL 2005 dataset is restricted to verbal lexical units, while the CoNLL 2009 and 2012 datasets contains both verbal and nominal lexical units. FrameNet has lexical units of several coarse syntactic categories.

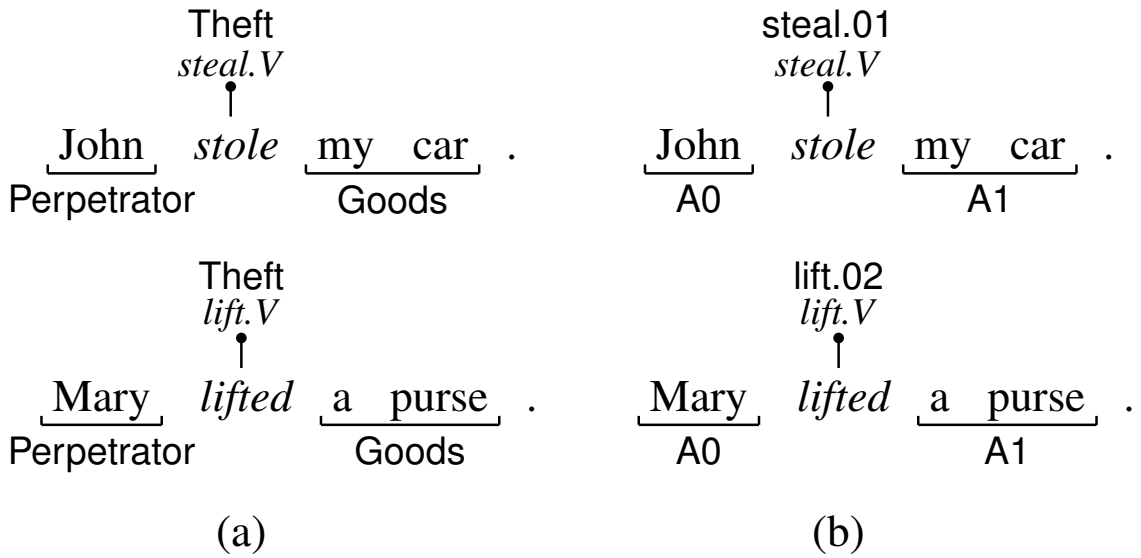


Figure 2.1: FrameNet (a) and PropBank (b) annotations for two sentences.

lexicon contains several hundred core and non-core roles that are shared across frames. For example, the FrameNet frame Theft could be evoked by the verbs *steal*, *pickpocket*, or *lift*, while PropBank has distinct frames for each of them. The Theft frame also contains the core roles Goods and Perpetrator that additionally belong to the Commercial_transaction and Committing_crime frames respectively.

A typical SRL dataset consists of sentence-level annotations that identify (possibly multiple) target predicates in each sentence, a disambiguated frame for each predicate, and the associated argument spans (or single word argument heads) labeled with their respective semantic roles.

SRL using PropBank conventions Palmer et al. [2005b] has been widely studied. There have been two shared tasks at CoNLL 2004-2005 to identify the phrasal arguments of verbal predicates Carreras and Màrquez [2004, 2005b]. The CoNLL 2008-2009 shared tasks introduced a variant where semantic dependencies are annotated rather than phrasal arguments Surdeanu et al. [2008b]; Hajič et al. [2009b]. Similar approaches Das et al. [2014]; Hermann et al. [2014] have been applied to frame-semantic parsing using FrameNet conventions Baker et al. [1998b]. In chapter 3 treat PropBank and FrameNet annotations in a common framework, similar to Hermann et al. [2014].

2.2 Models for SRL

Most prior work on SRL rely on syntactic parses provided as input and use locally estimated classifiers for each span-role pair that are only combined at prediction time.[2] This is done by picking the highest scoring role for each span, subject to a set of structural constraints, such as avoiding overlapping arguments and repeated core roles. Typically, these constraints have been enforced by integer linear programming (ILP), as in Punyakanok et al. [2008]. Täckström et al. [2015] interpreted this as a graphical model with local factors for each span-role pair, and global factors that encode the structural constraints. They derived a dynamic program (DP) that enforces most of the constraints proposed by Punyakanok et al. and showed how the DP can be used to take these constraints into account during learning. In chapter 3, we use an identical graphical model, but extend the model of Täckström et al. by replacing its linear potential functions with a multi-layer neural network. A similar use of non-linear potential functions in a structured model was proposed by Do and Artières [2010] for speech recognition, and by Durrett and Klein [2015] for syntactic phrase-structure parsing.

Feature-based approaches to SRL employ hand-engineered linguistically-motivated feature templates to represent the semantic structure. Some recent work has focused on low-dimensional representations that reduce the need for intensive feature engineering and lead to better generalization in the face of data sparsity. Lei et al. [2015] employ low-rank tensor factorization to induce a compact representation of the full cross-product of atomic features; akin to this work, they represent semantic roles as real-valued vectors, but use a different scoring formulation for modeling potential arguments. Moreover, they restrict their experiments to CoNLL 2009 semantic dependencies. Roth and Woodsend [2014] improve on the state-of-the-art feature-based system of Björkelund et al. [2010] by adding distributional word representations trained on large unlabeled corpora as features.

Collobert and Weston [2007a] use a neural network and do not rely on syntactic parses as

[2]Some recent work have successfully proposed joint models for syntactic parsing and SRL instead of a pipeline approach Lewis et al. [2015].

input. While they use non-standard evaluation, they report accuracy similar to the ASSERT system Pradhan et al. [2005], to which we compare in Table 3.4. Recently, Zhou and Xu [2015] proposed a deep bidirectional LSTM model for SRL that does not rely on syntax trees as input; their approach achieves the best results on CoNLL 2005 and 2012 corpora to date, but unlike this work, they do not report results on FrameNet and CoNLL 2009 dependencies and do not investigate joint learning approaches involving multiple annotation conventions. This approach was refined by He et al. [2017], and forms the basis of our BIO-based argument identification model in chapter 4. Several recent works have explored neural models for SRL tasks [Collobert and Weston, 2007b; Swayamdipta et al., 2017; Yang and Mitchell, 2017]. Recently, span-based models have proven to be useful for question answering [Lee et al., 2016] and coreference resolution [Lee et al., 2017], and PropBank SRL [He et al., 2018].

For FrameNet-style SRL, Kshirsagar et al. [2015] recently proposed the use of PropBank-based features, but their system performance falls short of the state of the art. Roth and Lapata [2015] proposed another approach exploring linguistically motivated features tuned towards the FrameNet lexicon.

2.3 Crowd-sourcing SRL

Resources and formalisms for semantic role labelling often require expert annotation and underlying syntax [Palmer et al., 2005a; Baker et al., 1998a; Meyers et al., 2004; Banarescu et al., 2013]. A number of different approaches have been taken in order to allow SRL annotation to be performed by crowdsource workers, including reducing the decision to selection for a set of candidates [Fossati et al., 2013], or selecting text passages [Feizabadi and Padó, 2014], but the task remains difficult for crowd workers, and annotation quality remains far below that of experts Pavlick et al. [2015].

Crowdsourcing has been shown effective in collecting large-scale datasets, especially for question-answering tasks [Rajpurkar et al., 2016; Richardson et al., 2013; Antol et al., 2015].

Crowdsourcing has also been used for indirectly annotating syntax He et al. [2016]; Duan et al. [2016], and to complement expert annotation of SRL [Wang et al., 2017]. Our crowdsourcing approach in chapter 4 draws heavily on that of Michael et al. [2017], with automatic two-stage validation for the collected question-answer pairs.

2.4 Alternative SRL Formalisms

Recently, several new semantic formalisms have been developed which contain among their design desiderata a requirement for less annotator training. UCCA [Abend and Rappoport, 2013] labels roles with general labels such as *argument* and *scene*, in order to be language independant. These labels identify semantic arguments, but do not distinguish between roles such as *agent* and *patient*. Reisinger et al. [2015] assign semantic proto-roles to gold-standard predicate and argument mentions. The Gronigen Meaning Bank [Basile et al., 2012] project uses crowd workers to correct mistakes made by existing SRL parsers, however they find that correcting predicate-argument structure is still difficult. AMR [Banarescu et al., 2013] goes beyond simple predicate-argument structure, representing additional semantic relations, but at the cost of increased difficulty of annotation. Most of these still require significant training of annotators. In particular, the original QA-SRL He et al. [2015] dataset is annotated by freelancers who received direct training by the authors. [Michael et al., 2017] use Amazon Mechanical Turk to collect free-form question-answer structures, however they find that reducing the constraints of the task in this way leads to lower quality annotations. In contrast, in chapter 4, we develop streamlined crowdsourcing approaches for more scalable, high-quality annotation of QA-SRL.

More recently, models have been developed for these newer semantic resources, such as UCCA [Hershcovich et al., 2017] and Semantic Proto-Roles [Teichert et al., 2017; White et al., 2017]. Our work in chapter 4 is the first high-quality parser for QA-SRL, which has several unique modeling challenges, such as its highly structured nature and the noise in crowdsourcing.

2.5 Multitask Training

Multitask learning – training on two or more related tasks in order to improve performance on a target task – has been widely employed across many machine learning tasks, including NLP [Collobert and Weston, 2008] and computer vision [Girshick, 2015]. Typically, this is done by sharing core parts of a network which are used for both (or all) tasks [Caruana, 1993; Søgaard and Goldberg, 2016].

A number of works have experimented with learning multiple related semantic representations simultaneously. Peng et al. [2018] learn across the three semantic graph representations from the 2015 SemEval shared task on Broad-Coverage Semantic Dependency Parsing [Oepen et al., 2015], experimenting with both parameter sharing, and joint decoding. Peng et al. [2018] learn across entirely disjoint tasks by treating each task as latent on the data for which it is not provided. Swayamdipta et al. [2017] use syntactic parsing (both dependency and constituency) as a scaffold task for frame semantic parsing.

The inspiration behind our approach in chapter 3 stems from work on bilinear models Mnih and Hinton [2007]. There have been several studies representing input observations and output labels with distributed representations, for example, in the WSABIE model for image annotation [Weston et al., 2011], in models for embedding labels in structured graphical models [Srikumar and Manning, 2014], and in techniques to learn joint embeddings of predicate words and their semantic frames in a vector space [Hermann et al., 2014].

Chapter 3

Semantic Role Labeling with Neural Network Factors

3.1 Introduction

[1] Semantic role labeling (SRL) is the task of identifying the semantic arguments of a predicate and labeling them with their semantic roles. A key challenge in this task is sparsity of labeled data: a given predicate-role instance may only occur a handful of times in the training set. Most existing SRL systems model each semantic role as an atomic unit of meaning, ignoring finer-grained semantic similarity between roles that can be leveraged to share context between similar labels, both within and across annotation conventions.

Low-dimensional embedding representations have been shown to be successful in overcoming sparsity and representing label similarity across a wide range of tasks Weston et al. [2011]; Srikumar and Manning [2014]; Hermann et al. [2014]; Lei et al. [2015]. In this chapter, we present a new model for SRL that embeds candidate arguments and semantic roles (in context of a predicate frame) in a shared vector space. A feed-forward neural network is learned to capture correlations of

[1] The work presented in this chapter was done with Oscar Täckström, Kuzman Ganchev, and Dipanjan Das, and published as FitzGerald et al. [2015].

the respective embedding dimensions to create argument and role representations. The similarity of these two representations, as measured by their dot product, is used to score possible roles for candidate arguments within a graphical model. This graphical model jointly models the assignment of semantic roles to all arguments of a predicate, subject to structural linguistic constraints.

Our model has several advantages. Compared to linear multiclass classifiers used in prior work, vector embeddings of the predictions overcome the assumption of modeling each semantic role as a discrete label, thus capturing fine-grained label similarity. Moreover, since predictions and inputs are embedded in the same vector space, and features extracted from inputs and outputs are decoupled, our approach is amenable to joint learning of multiple annotation conventions, such as PropBank and FrameNet, in a single model. Finally, as with other neural network approaches, our model obviates the need to manually engineer feature conjunctions.

Our underlying inference algorithm for SRL follows Täckström et al. [2015], who presented a dynamic program for structured SRL; it is targeted towards the prediction of full argument spans. Hence, we present empirical results on three span-based SRL datasets: CoNLL 2005 and 2012 data annotated with PropBank conventions, as well as FrameNet 1.5 data. We also evaluate our system on the dependency-based CoNLL 2009 shared task by assuming single word argument spans, that represent semantic dependencies, and limit our experiments to English. On all datasets, our model performs on par with a strong linear model baseline that uses hand-engineered conjunctive features. Due to random parameter initialization and stochasticity in the online learning algorithm used to train our models, we observed considerable variance in performance across datasets. To resolve this variance, we adopt a product-of-experts model that combines multiple randomly-initialized instances of our model to achieve state-of-the-art results on the CoNLL 2009 and FrameNet datasets, while coming close to the previous best published results on the other two. Finally, we present even stronger results for FrameNet data (which is scarce) by jointly training the model with PropBank-annotated data.

3.2 Model

Our model for SRL performs inference separately for each marked predicate in a sentence. It assumes that the predicate has been automatically disambiguated to a semantic frame drawn from a frame lexicon, and the semantic roles of the frame are used for labeling the candidate arguments in the sentence. Formally, we are given a sentence x in which a predicate t, with lexical unit ℓ, has been marked. Assuming that the semantic frame f of the predicate has already been identified (see subsection 3.3.2 for this step), we seek to predict the set of spans that correspond to its overt semantic arguments and to label each argument with its semantic role. Specifically, we model the problem as that of assigning each span $s \in \mathcal{S}$, from an over-generated set of candidate argument spans $\mathcal{S}$, to a semantic role $r \in \mathcal{R}$. The set of semantic roles $\mathcal{R}$ includes the special null role $\varnothing$, which is used to represent non-overt arguments. Thus, our algorithm performs the SRL task in one step for a single predicate frame. For the span-based SRL task, in a sentence of n words, there could be $O(n^2)$ potential arguments. For statistical and computational reasons we prune the set of spans $\mathcal{S}$ using a set of syntactically-informed heuristics from prior work (see subsection 3.3.2).

3.2.1 Graphical Model

We make use of a graphical model that represents global assignment of arguments to their semantic roles, subject to linguistic constraints Punyakanok et al. [2008]; Täckström et al. [2015]. Under this graphical model, we assume a parameterized potential function that assigns a real-valued compatibility score $g(s, r; \boldsymbol{\theta})$ to each span-role pair $(s, r) \in \mathcal{S} \times \mathcal{R}$, where $\boldsymbol{\theta}$ denotes the model parameters. Below, we consider two types of potential functions. As a baseline, similar to most prior work, one could use a simple linear function of discrete input features $g_{\text{L}}(s, r; \boldsymbol{\theta}) = \boldsymbol{\theta}^\top \cdot \boldsymbol{\phi}(r, s, x, t, \ell, f)$, where $\boldsymbol{\phi}(\cdot)$ denotes a feature function. In this work, we instead propose a multi-layer feed-forward neural network potential function, specified in subsection 3.2.2. Given these local factors, we employ the dynamic program of Täckström et al. to enforce global constraints

on the inferred output.

Let $\mathcal{R}^{|\mathcal{S}|}$ denote the set of all possible assignments of semantic roles to argument spans (s_i, r_i) for $s_i \in \mathcal{S}$ that satisfy the constraints. Given a potential function $g(s, r) \triangleq g(s, r; \boldsymbol{\theta})$, the probability of a joint assignment $\boldsymbol{r} \in \mathcal{R}^{|\mathcal{S}|}$, subject to the constraints, is given by

$$p(\boldsymbol{r} \mid x, t, \ell, f) = \exp\left(\sum_{s_i \in \mathcal{S}} g(s_i, r_i) - A(\mathcal{S})\right), \tag{3.1}$$

where the log-partition function $A(\mathcal{S})$ sums over all satisfying joint role assignments:

$$A(\mathcal{S}) = \log \sum_{\boldsymbol{r'} \in \mathcal{R}^{|\mathcal{S}|}} \exp\left(\sum_{s_i \in \mathcal{S}} g(s_i, r_i')\right). \tag{3.2}$$

3.2.2 Neural Network Potentials

Our approach replaces the standard linear potential function $g_{\text{L}}(s, r; \boldsymbol{\theta})$ with the real-valued output of a feed forward neural network with non-linear hidden units. The network structure is outlined in Figure 3.1. The frame f and role r are initially encoded using a one-hot encoding as $\boldsymbol{i}_f$ and $\boldsymbol{i}_r$. In other words, $\boldsymbol{i}_f$ and $\boldsymbol{i}_r$ have all zeros except for one position at f and r respectively. These are passed through fully connected linear layers to give $\boldsymbol{e}_f$ and $\boldsymbol{e}_r$. We call these linear layers the *embedding* layers since $\boldsymbol{i}_f$ selects the embedding of the frame f and $\boldsymbol{i}_r$ for r. Next, $\boldsymbol{e}_f$ and $\boldsymbol{e}_r$ are passed through a fully connected rectified linear layer Nair and Hinton [2010], to obtain the final frame-role representation $\boldsymbol{v}_{(f,r)}$. For the candidate span, the process is similar. Atomic features $\boldsymbol{\phi}(s, x, t, \ell)$ for the argument span s are extracted first. (These features are the non-conjoined features used in the linear model of Täckström et al.; see Table 3.1 for the list). These are next passed through a fully-connected linear embedding layer to get the span embedding $\boldsymbol{e}_s$, which is subsequently passed through a fully connected rectified linear layer to obtain $\boldsymbol{v}_s$, the final span

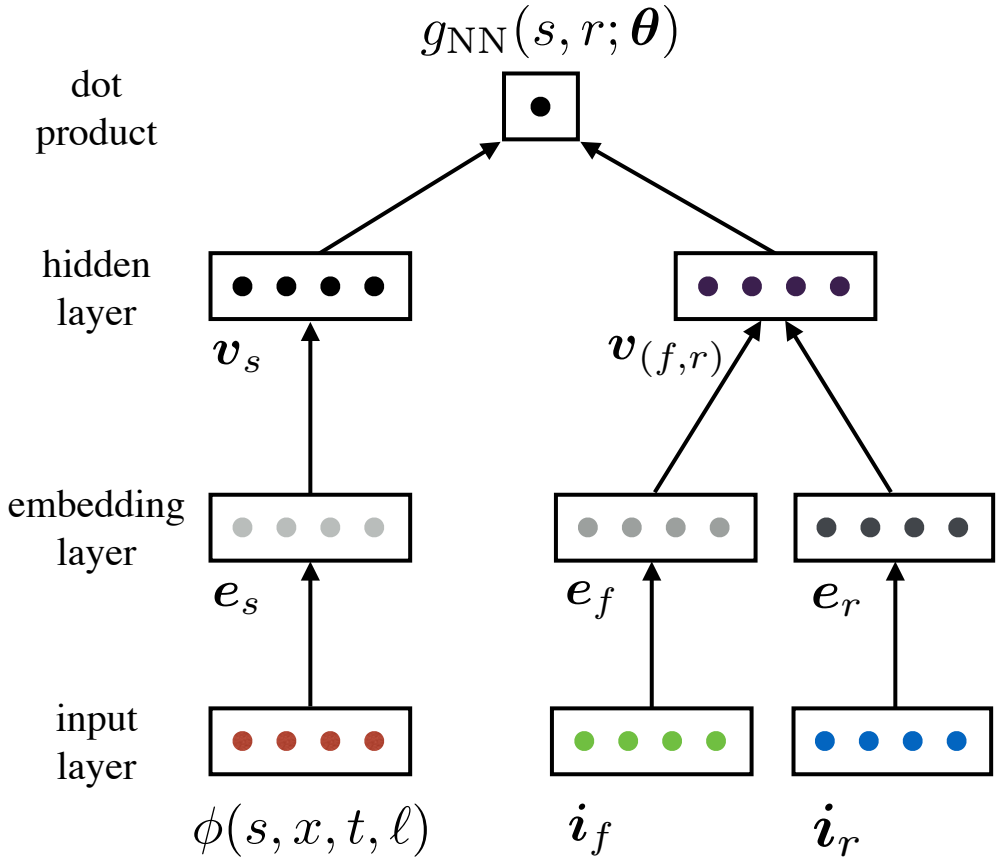


Figure 3.1: Neural network architecture.

representation. The final output is the dot product of $\boldsymbol{v}_s$ and $\boldsymbol{v}_{(f,r)}$:

$$g_{\text{NN}}(s, r; \boldsymbol{\theta}) = \boldsymbol{v}_s^\top \cdot \boldsymbol{v}_{(f,r)} \,. \tag{3.3}$$

The weights of all the layers constitute the parameters $\boldsymbol{\theta}$ of the neural network. We initialize $\boldsymbol{\theta}$ randomly, with the exception of embedding parameters corresponding to words, which are initialized from pre-trained word embeddings (see subsection 3.3.4 for details). We train the network as described in subsection 3.2.3.[2]

Note that unlike typical linear models, the atomic span features are not explicitly conjoined

[2]Various other network structures are worth investigating, such as concatenating the span, frame and role representations and passing them through fully connected layers. This treatment, for example, has been used by Chen and Manning [2014] for syntactic parsing. We leave these explorations to future work.

• first word of s	• tag of the first word of s
• last word of s	• tag of the last word of s
• head word of s	• tag of the head word of s
• bag of words in s	• bag of tags in s
• cluster of s's head	• linear *distance* of s from t
• t's children words	• word cluster of s's head
• dependency path between s's head and t	
• subcategorization frame of s	
• *position* of s w.r.t. t (*before, after, overlap or same*)	
• predicate use voice (*active*, *passive*, or *unknown*)	
• whether the subject of t is missing (*missing-subj*)	
• *position* of s w.r.t. t (*before, after, overlap or same*)	
• word, tag, dependency label and cluster of the words immediately to the left and right of s	

Table 3.1: Span features $\boldsymbol{\phi}(s, x, t, \ell)$ in Figure 3.1.

with each other, the frame or the role. Instead the hidden layers learn to emulate span feature conjunctions and frame and role feature conjunctions in parallel.[3] Moreover, note that span $\boldsymbol{v}_s$ and frame-role $\boldsymbol{v}_{(f,r)}$ representations are decoupled in this model. This decoupling is important as it allows us to train a single model in a multitask setting. We demonstrate this by successfully combining PropBank and FrameNet training data, as described in section 3.4.

3.2.3 Parameter Estimation

We consider two methods for parameter estimation.

Local Estimation In local estimation, we treat each span-role assignment pair $(s, r) \in \mathcal{S} \times \mathcal{R}$ as an individual binary decision problem and maximize the corresponding log-likelihood of the training set.[4] Denote by $z_{s,r} \in \{0, 1\}$ the decision variable, such that $z_{s,r} = 1$ iff span s is assigned role

[3] We found that adding feature conjunctions to the network's input layer did not improve performance in practice.

[4] An alternate way to locally train the neural network would be to treat the scores as potentials in a multiclass logistic regression model and optimize log-likelihood, as is done with the locally-trained linear model from Täckström et al.

r. By passing the potential $g_{\text{NN}}(s, r; \boldsymbol{\theta})$ through the logistic function, we obtain the log-likelihood $l(z_{s,r}; \boldsymbol{\theta}) \triangleq \log p(z_{s,r} \mid x, t, \ell, f)$ of an individual training example. Here,

$$p(z_{s,r} \mid x, t, \ell, f) = \begin{cases} \frac{1}{1+e^{-g_{\text{NN}}(s,r;\boldsymbol{\theta})}} & \text{if } z_{s,r} = 1 \\ \frac{e^{-g_{\text{NN}}(s,r;\boldsymbol{\theta})}}{1+e^{-g_{\text{NN}}(s,r;\boldsymbol{\theta})}} & \text{if } z_{s,r} = 0 \end{cases}$$

Thus, the gold role for a given span according to the training data serves as the positive example, while all the other potential roles serve as negatives. To maximize the log-likelihood, we use Adagrad Duchi et al. [2011]. This requires the gradient of the log-likelihood with respect to the parameters $\boldsymbol{\theta}$, which can be derived using the chain rule.

Structured Estimation In structured estimation, we instead learn a globally normalized probabilistic model that takes the structural constraints into account during training. This method is closely related to the linear approach of Täckström et al. [2015], as well as to the fine-tuning of a neural CRF described by Do and Artières [2010].

We train the model by maximizing the log-likelihood of the training data, again using Adagrad. From Equation 3.1, we have that the log-likelihood $l(\boldsymbol{r}; \boldsymbol{\theta}) \triangleq \log p(\boldsymbol{r} \mid x, t, \ell, f)$ of a single (structured) training example $(\boldsymbol{r}, \mathcal{S}, x)$ is given by

$$l(\boldsymbol{r}; \boldsymbol{\theta}) = \sum_{s_i \in \mathcal{S}} g(s_i, r_i) - A(\mathcal{S})\,. \tag{3.4}$$

By application of the chain rule, the gradient of the log-likelihood factorizes as

$$\frac{\partial l(\boldsymbol{r}; \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial l(\boldsymbol{r}; \boldsymbol{\theta})}{\partial g_{\text{NN}}} \frac{\partial g_{\text{NN}}}{\partial \boldsymbol{\theta}}\,, \tag{3.5}$$

where we have used the shorthand g_{NN} for brevity. It is easy to show that the first term $\partial l(\boldsymbol{r}; \boldsymbol{\theta}) / \partial g_{\text{NN}}$

[2015], but we did not investigate this alternative in this work.

factors into the marginals over edges in the DP lattice, which can be computed with the forward-backward algorithm (recall that the potentials are in simple correspondence with the edge scores in the DP lattice, see [Täckström et al., 2015, §4] for details). Again, the chain rule can be used to compute the gradient $\partial g_{\text{NN}}/\partial \boldsymbol{\theta}$ with respect to the parameters of each layer in the network.

3.2.4 Product of Experts

As we will observe in Table 3.2, random initialization of the neural network parameters $\boldsymbol{\theta}$ causes variance in the performance over different runs. We found that using a straightforward product-of-experts (PoE) model Hinton [2002] at inference time reduces this variance and results in significantly higher performance. This PoE model is a very simple ensemble, being the factor-wise sum of the potential functions from K independently trained neural networks:

$$g_{\text{PoE}}(s, r; \boldsymbol{\theta}) = \sum_{j=1}^{K} g_{\text{NN}}^{(j)}(s, r, \boldsymbol{\theta}) \,. \tag{3.6}$$

where $g_{\text{NN}}^{(j)}(s, r, \boldsymbol{\theta})$ is the score from model j.

3.3 Experimental Setup

In this section we describe the datasets used, the required preprocessing steps, the baselines compared to and the details of our experimental setup.

3.3.1 Datasets and Significance Testing

We evaluate our approach on four standard datasets. For span-based SRL using PropBank conventions Palmer et al. [2005b], we evaluate on both the CoNLL 2005 shared task dataset Carreras and Màrquez [2005b], and the larger CoNLL 2012 dataset derived from the OntoNotes 5.0 corpus Weischedel et al. [2011]. We also evaluate our model on the CoNLL 2009 shared task dataset Hajič

et al. [2009b], that annotates roles for semantic dependencies, rather than full argument spans. For the CoNLL datasets, we use the standard training, development and test sets. For frame-semantic parsing using FrameNet conventions Baker et al. [1998b], we follow Das et al. [2014] and Hermann et al. [2014] in using the full-text annotations of the FrameNet 1.5 release and follow their data splits.

We use the standard evaluation scripts for each task and use a paired bootstrap test Efron and Tibshirani [1994] to assess the statistical significance of the results. For brevity, we only give the p-values for the observed differences between our best and second best models on each of the test sets.

3.3.2 Preprocessing and Frame Identification

All datasets are preprocessed with a part-of-speech tagger and a syntactic dependency parser, both trained on the CoNLL 2012 training split, after converting the constituency trees to Stanford-style dependencies De Marneffe and Manning [2013]. The tagger is based on a second-order conditional random field Lafferty et al. [2001] with standard emission and transition features; for parsing, we use a graph-based parser with structural diversity and cube-pruning Zhang and McDonald [2014].

On the WSJ development set (section 22), the labeled attachment score of the parser is 90.9% while the part-of-speech tagger achieves an accuracy of 97.2%. On the CoNLL 2012 development set, the corresponding scores are 90.2% and 97.3%. Both the tagger and the parser, as well as the SRL models use word cluster features (see Table 3.1). Specifically, we use the clusters with 1000 classes from Turian et al. [2010], which are induced with the Brown algorithm Brown et al. [1992]. To generate the candidate arguments $\mathcal{S}$ (see subsection 3.2.2) for the CoNLL 2005 and 2012 span-based datasets, we follow Täckström et al. [2015] and adapt the algorithm of Xue and Palmer [2004] to use dependency syntax. For the dependency-based CoNLL 2009 experiments, we modify our approach to assume single length spans and treat every word of the sentence as a candidate argument. For FrameNet, we follow the heuristic of Hermann et al. [2014].

As mentioned in section 3.2, we automatically disambiguate the predicate frames. For FrameNet, we use an embedding-based model described by Hermann et al. [2014]. For PropBank, we use a multi-class log-linear model, since Hermann et al. did not observe better results with the embedding model.

To ensure a fair comparison with the closest linear model baseline, we ensured that the pre-processing steps, the argument candidate generation algorithm for the span-based datasets and the frame identification methods are identical to [Täckström et al., 2015, §3.2, §6.2-§6.3].

3.3.3 Baseline Systems

In addition to comparing to Täckström et al. [2015], whose setup is closest to ours, we also compare to prior state-of-the-art systems from the literature.

For CoNLL 2005, we compare to the best non-ensemble and ensemble systems of Surdeanu et al. [2007], Punyakanok et al. [2008] and Toutanova et al. [2008]. The ensemble variants of these systems use multiple parses and multiple SRL systems to leverage diversity. In contrast to these ensemble systems, our product-of-experts model uses only a single architecture and one syntactic parse; the constituent models differ only in random initialization. We also compare with the recent deep bidirectional LSTM model of Zhou and Xu [2015].

For CoNLL 2012, we compare to Pradhan et al. [2013], who report results with the (non-ensemble) ASSERT system Pradhan et al. [2005], and to the model of Zhou and Xu [2015].

For CoNLL 2009, we compare to the top system from the shared task Zhao et al. [2009], two state-of-the-art systems that employ a reranker Björkelund et al. [2010]; Roth and Woodsend [2014], and the recent tensor-based model of Lei et al. [2015]. We also trained the linear model of Täckström et al. on this dataset (their work omitted this experiment), as a baseline.

Finally, for the FrameNet experiments, we compare to the state-of-the-art system of Hermann et al. [2014], which combines a frame-identification model based on WSABIE Weston et al. [2011] with a log-linear role labeling model.

	WSJ Dev				WSJ Test				Brown Test			
Method	P	R	F1	Comp.	P	R	F1	Comp.	P	R	F1	Comp.
Surdeanu (Single)	–	–	–	–	79.7	74.9	77.2	52.0	–	–	–	–
Surdeanu (Ensemble)	–	–	–	–	87.5	74.7	80.6	51.7	81.8	61.3	70.1	34.3
Toutanova (Single)	–	–	77.9	**57.2**	–	–	79.7	**58.7**	–	–	67.8	39.4
Toutanova (Ensemble)	–	–	78.6	58.7	81.9	78.8	80.3	60.1	–	–	68.8	40.8
Punyakanok (Single)	–	–	–	–	77.1	75.5	76.3	–	–	–	–	–
Punyakanok (Ensemble)	80.1	74.8	77.4	50.7	82.3	76.8	79.4	53.8	73.4	62.9	67.8	32.3
Täckström (Local)	81.3	74.8	77.9	52.4	82.6	76.4	79.3	54.3	74.0	66.8	70.2	38.4
Täckström (Struct.)	81.2	76.2	78.6	54.4	82.3	77.6	79.9	56.0	74.3	68.6	71.3	39.8
Zhou	79.7	**79.4**	**79.6**	–	**82.9**	**82.8**	**82.8**	–	70.7	68.2	69.4	–
This work (Local)	81.4	75.6	78.4	53.7	82.3±0.4	76.8±0.5	79.4±0.1	55.1±0.6	74.1±0.6	68.0±0.7	70.9±0.3	39.1±0.8
This work (Struct.)	80.7	76.1	78.3	54.1	81.8±0.5	77.3±0.3	79.4±0.2	55.6±0.5	73.8±0.7	68.8±0.6	71.2±0.3	40.5±0.8
This work (Local, PoE)	**82.0**	76.6	79.2	55.2	**82.9**	77.8	80.3*	56.7	**75.2**	69.1	72.0	40.8
This work (Struct., PoE)	81.2	76.7	78.9	55.1	82.5	78.2	80.3*	57.3*	74.5	**70.0**	**72.2****	**41.3**

Table 3.2: PropBank-style SRL results on CoNLL 2005 data. Bold font indicates the best system using a single or no syntactic parse, while the best scores among all systems are underlined. Results from prior work are taken from the respective papers, and '–' indicates performance metrics missing in the original publication. Statistical significance was assessed for F1 and Comp. on the WSJ and Brown test sets with $p < 0.01$ ($*$) and $p < 0.05$ ($**$).

3.3.4 Hyperparameters and Initialization

There are several hyperparameters in our model (subsection 3.2.2). First, the span embedding dimension of $\boldsymbol{e}_s$ was fixed to 300 to match the dimension of the pre-trained GloVe word embeddings from Pennington et al. [2014] that we use to initialize the embeddings of the word-based features in $\boldsymbol{\phi}(s, x, t, \ell)$. Preliminary experiments showed random initialization of the word-based embeddings to be inferior to pre-trained embeddings. The remaining model parameters were randomly initialized. The frame embedding dimension was chosen from $\{100, 200, 300, 500\}$, while the hidden layer dimension was chosen from $\{300, 500\}$. For PropBank, we fixed the role embedding dimension to 27, which is the number of semantic roles in PropBank datasets (ignoring the AA role, that appears with negligible frequency). For FrameNet, the role embedding dimension was chosen from $\{100, 200, 300, 500\}$. In the Adagrad algorithm, the mini-batch size was fixed to 100 for local estimation (section 3.2.3). For structured estimation (section 3.2.3), a batch size of one was used, since each structured instance contains multiple local factors. The learning rate was chosen from $\{0.1, 0.2, 0.5, 1.0\}$ for local learning and from $\{0.01, 0.02, 0.05, 0.1\}$ for structured

	Excluding predicate senses			Including predicate senses	
	WSJ Dev	WSJ Test	Brown Test	WSJ Test	Brown Test
CoNLL-2009 1st place	–	82.1	69.8	86.2	74.6
Björkelund et al., 2010 + reranking	80.5	82.9	70.9	86.9	75.7
Roth and Woodsend, 2014 + reranking	–	82.1	71.1	86.3	**75.9**
Lei et al. 2015	81.0	82.5	70.8	86.6	75.6
Täckström et al. 2015 (Local)	81.4	83.0	71.2	86.9	74.8
Täckström et al. 2015 (Struct.)	82.4	83.7	72.3	87.3	75.5
This work (Local)	$81.2_{\pm 0.2}$	$82.7_{\pm 0.3}$	$71.9_{\pm 0.4}$	$86.7_{\pm 0.2}$	$75.2_{\pm 0.3}$
This work (Struct)	$82.3_{\pm 0.1}$	$83.6_{\pm 0.1}$	$71.9_{\pm 0.3}$	$87.3_{\pm 0.1}$	$75.2_{\pm 0.2}$
This work (Local, PoE)	82.4	83.8	**72.8**	87.5	**75.9**
This work (Struct., PoE)	**83.0***	**84.3***	72.4	**87.8***	75.5

Table 3.3: PropBank-style *semantic dependency* SRL results (labeled F1) on the CoNLL 2009 data set. Bold font indicates the best system. Statistical significance was assessed with $p < 0.01$ ($*$).

learning.[5] All hyperparameters were tuned on the respective development sets for each dataset with a straightforward grid-search procedure. In the product-of-experts setup, we train $K = 10$ models, each with a different random seed, and combine them at inference time (see Equation 3.6).

3.4 Empirical Results

Table 3.2 shows results on the CoNLL 2005 development set and the WSJ and Brown test sets. Our individual neural network models are on par with the best linear single-system baselines that use carefully chosen feature combinations, but has variance across reruns. On the WSJ test set, the product-of-experts model featuring neural networks trained with structured learning achieves higher F_1-score than all non-ensemble baselines, except the LSTM model of Zhou and Xu. It is on par and at times better than ensemble baselines that use diverse syntactic parses. The PoE model outperforms all baselines on the Brown test set, exhibiting its generalization power on out-of-domain text. Overall, using structured learning improves recall at a slight expense of precision when compared to local learning, leading to an increase in the complete argument structure accuracy (*Comp.* in the tables).

[5] We observed a strong interaction between learning rate and mini-batch size. Since the number of factors per frame structure is much larger than 100, lower learning rates are better suited for structured estimation.

	CoNLL 2012 Development			
Method	P	R	F1	Comp.
Täckström (Local)	80.6	77.1	78.8	59.0
Täckström (Struct.)	80.5	77.8	79.1	60.1
Zhou	–	–	**81.1**	–
This work (Local)	80.4	77.3	78.8	59.0
This work (Struct)	80.6	77.8	79.2	59.8
This work (Local, PoE)	**81.0**	78.3	79.6	60.6
This work (Struct., PoE)	**81.0**	**78.5**	79.7	**60.9**
	CoNLL 2012 Test			
Method	P	R	F1	Comp.
Pradhan	**81.3**	70.5	75.5	51.7
Pradhan, revised	78.5	76.6	77.5	55.8
Täckström (Local)	80.9	77.7	79.2	60.9
Täckström (Struct.)	80.6	78.2	79.4	61.8
Zhou	–	–	**81.3**	–
This work (Local)	80.6 ± 0.3	77.8 ± 0.2	79.2 ± 0.1	60.8 ± 0.3
This work (Struct.)	80.9 ± 0.2	78.4 ± 0.2	79.6 ± 0.1	61.7 ± 0.2
This work (Local, PoE)	**81.3**	78.8	80.0	62.4
This work (Struct., PoE)	81.2	**79.0**	80.1 *	**62.6***

Table 3.4: PropBank-style SRL results on the CoNLL 2012 development and test sets. Results from prior work are taken from the respective papers, and '–' indicates performance metrics missing in the original publication. Significance was assessed for F1 and Comp. on the test set with $p < 0.01$ ($*$).

Table 3.3 shows results on the CoNLL 2009 task. Following Lei et al. [2015], we present results using the official evaluation script, along with additional metrics that do not count frame predictions. Note that the linear baseline of Täckström et al. outperforms most prior work, including ones that employs rerankers, except on the Brown test set. Our neural network model performs even better, achieving state-of-the-art results on all metrics.

Table 3.4 shows the results on the span-based CoNLL 2012 data. The trends observed on the CoNLL 2005 data hold here as well, with structured training yielding an increase in precision at the cost of a small drop in recall. This leads to improvements in both F_1 score and complete structure accuracy. The product-of-experts model trained with structured learning here yields results better

than the ASSERT system Pradhan et al. [2013], but akin to CoNLL 2005, our system falls short in comparison to Zhou and Xu's F_1-score. In contrast to the smaller CoNLL 2005 data, even our single (non-PoE) model outperforms the linear model of Täckström et al. [2015] on the CoNLL 2012 data. We hypothesize that the relative abundance of the latter counteracts the risk for overfitting of the larger number of parameters in our model.

Finally, Table 3.5 shows the results on FrameNet data, which is very small in size. Here, structured learning does not help and in fact leads to a small drop in performance. Our locally-trained neural network model performs comparably to the linear model of Täckström et al. [2015]. However we achieve significant improvements in both F_1-score and full structure accuracy by training our model with a dataset composed of both FrameNet and CoNLL 2005 data.[6] The ability to train in this multitask setting is a unique capability of our approach, and yields state-of-the-art results for FrameNet.

Figure 3.3 shows the effect of adding increasing amount of CoNLL 2005 data to supplement the FrameNet training corpus in this multitask setting. The Y-axis plots F_1-score on the development data averaged across runs for the local non-PoE model. With increasing amount of PropBank data, performance increases in small steps, and peaks when all the data is added. This shows that with more PropBank data we could further improve performance on the FrameNet task; we leave further exploration of multitask learning of predicate argument structures, including multilingual settings, to future work.

3.4.1 Qualitative Analysis of Embeddings

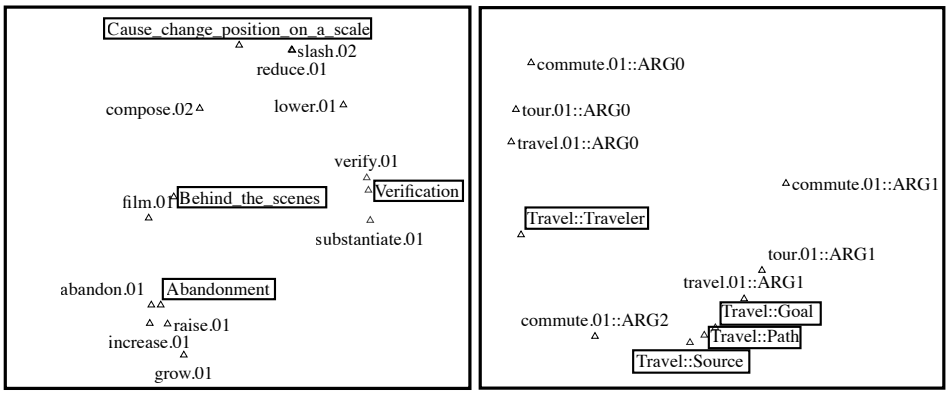
Figure 3.2 shows example embeddings from the model trained jointly on FrameNet and PropBank annotations. Figure 3.2a shows the proximity of the learned embeddings $\boldsymbol{e}_f$ of frames from both FrameNet and PropBank. Figure 3.2b shows the embeddings for frame-role pairs $\boldsymbol{v}_{(f,r)}$ (the output

[6]The joint model does not improve results for PropBank. This is likely due to the much larger CoNLL 2005 training set, compared to the FrameNet data (39,832 training sentences in the former as opposed to 3,256 sentences in the latter).

	FrameNet Development			
Method	P	R	F1	Comp.
Hermann	78.3	64.5	70.8	–
Täckström (Local)	**80.7**	62.9	70.7	31.2
Täckström (Struct.)	79.6	64.1	71.0	33.3
This work (Local)	78.6	64.6	70.9	32.0
This work (Struct.)	79.6	63.9	70.9	31.8
This work (Local, PoE)	79.0	65.0	71.3	33.1
This work (Struct., PoE)	79.0	64.4	71.0	32.3
This work (Local, PoE, Joint)	79.4	**65.8**	**72.0**	**34.5**
This work (Struct., PoE, Joint)	78.8	65.4	71.5	33.5
	FrameNet Test			
Method	P	R	F1	Comp.
Hermann	74.3	66.0	69.9	–
Täckström (Local)	**76.1**	64.9	70.1	33.0
Täckström (Struct.)	75.4	65.8	70.3	33.8
This work (Local)	73.9 $\pm$0.6	66.4 $\pm$0.4	69.9 $\pm$0.3	33.4 $\pm$0.6
This work (Struct.)	74.8 $\pm$0.2	65.5 $\pm$0.2	69.9 $\pm$0.1	33.0 $\pm$0.3
This work (Local, PoE)	74.3	66.9	70.4	33.9
This work (Struct., PoE)	74.6	66.3	70.2	33.3
This work (Local, PoE, Joint)	75.0	**67.3**	**70.9****	**35.4***
This work (Struct., PoE, Joint)	74.2	67.2	70.5	34.2

Table 3.5: Joint frame and argument identification results for FrameNet. Statistical significance was assessed for F1 and Comp. on the test set with $p < 0.01$ ($*$) and $p < 0.05$ ($**$).

of the hidden rectified linear layer). Here, we fix the FrameNet frame Travel and the similar PropBank frames commute.01, tour.01 and travel.01 are visualized along with their semantic roles. We observe that the model learns very similar embeddings for the semantically related roles across both datasets. Note that there is a clear separation of the agentive roles from the others for both conventions and how the FrameNet and PropBank counterparts of each type of role are proximate in vector space.



(a) (b)

Figure 3.2: Two-dimensional t-SNE projections Van der Maaten and Hinton [2008] of joint PropBank and FrameNet (boxed) embeddings of frames (a) and frame-role pairs (b).

3.5 Conclusion

We presented a neural network model for semantic role labeling that learns to embed both inputs and outputs in the same vector space. We considered both local and structured training methods for the network parameters from supervised SRL data. Empirically, our approach achieves state-of-the-art results on two standard datasets with a product of experts model, while approaching the performance of a recent deep recurrent neural network model on two other datasets. By training the model jointly on both FrameNet and PropBank data, we achieve the best result to date on the FrameNet test set. Finally, qualitative analysis indicates that the model represents semantically similar annotations with proximate vector-space embeddings.

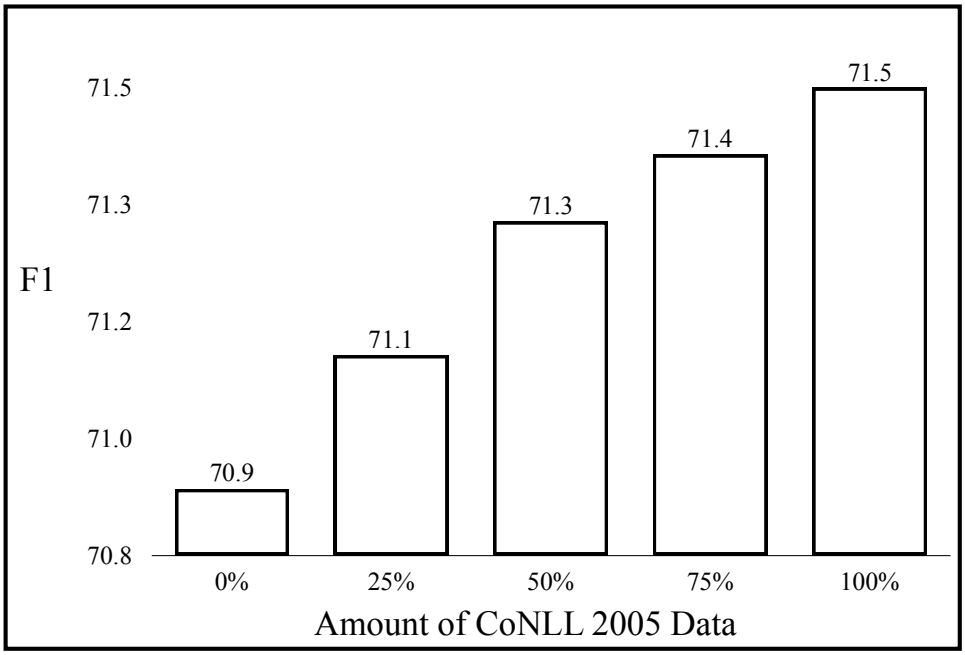


Figure 3.3: F_1 score on the FrameNet development data averaged over runs versus the percentage of CoNLL 2005 data used to append the FrameNet training corpus. For this plot, we used the locally trained non-PoE model.

Chapter 4

Large-Scale QA-SRL Parsing

4.1 Introduction

[1] Learning semantic parsers to predict the predicate-argument structures of a sentence is a long standing, open challenge [Palmer et al., 2005a; Baker et al., 1998a]. Such systems are typically trained from datasets that are difficult to gather,[2] but recent research has explored training non-experts to provide this style of semantic supervision [Abend and Rappoport, 2013; Basile et al., 2012; Reisinger et al., 2015; He et al., 2015]. In this chapter, we show for the first time that it is possible to go even further by crowdsourcing a large scale dataset that can be used to train high quality parsers at modest cost.

We adopt the Question-Answer-driven Semantic Role Labeling (QA-SRL) He et al. [2015] annotation scheme. QA-SRL is appealing because it is intuitive to non-experts, has been shown to closely match the structure of traditional predicate-argument structure annotation schemes He et al. [2015], and has been used for end tasks such as Open IE Stanovsky and Dagan [2016]. In QA-SRL, each predicate-argument relationship is labeled with a question-answer pair (see Figure 4.1). He

[1] The work presented in this chapter was done with Julian Michael, Luheng He and Luke Zettlemoyer and is published as FitzGerald et al. [2018].

[2] The PropBank [Bonial et al., 2010] and FrameNet [Ruppenhofer et al., 2016] annotation guides are 89 and 119 pages, respectively.

In 1950 Alan M. Turing ***published*** "Computing machinery and intelligence" in Mind, in which he ***proposed*** that machines could be ***tested*** for intelligence ***using*** questions and answers.

published	1	Who published something?	Alan M. Turing
	2	What was published?	“ Computing Machinery and
	3	When was something published?	In 1950
proposed	4	Who proposed something?	Alan M. Turing
	5	What did someone propose?	that machines could be tested for intelligent using
	6	When did someone propose something?	In 1950
tested	7	What can be tested?	machines
	8	What can something be tested for?	intelligence
	9	How can something be tested?	using questions
using	10	What was being used?	questions and
	11	Why was something being used?	tested for

Figure 4.1: An annotated sentence from our dataset. Question 6 was not produced by crowd workers in the initial collection, but was produced by our parser as part of Data Expansion (see section 4.6.)

et al. [2015] showed that high precision QA-SRL annotations can be gathered with limited training but that high recall is challenging to achieve; it is relatively easy to gather answerable questions, but difficult to ensure that every possible question is labeled for every verb. For this reason, they hired and trained hourly annotators and only labeled a relatively small dataset (3000 sentences).

Our first contribution is a new, scalable approach for crowdsourcing QA-SRL. We introduce a streamlined web interface (including an auto-suggest mechanism and automatic quality control to boost recall) and use a validation stage to ensure high precision (i.e. all the questions must be answerable). With this approach, we produce QA-SRL Bank 2.0, a dataset with 133,479 verbs from 64,018 sentences across 3 domains, totaling 265,140 question-answer pairs, in just 9 days. Our analysis shows that the data has high precision with good recall, although it does not cover every possible question. Figure 4.1 shows example annotations.

Using this data, our second contribution is a comparison of several new models for learning a QA-SRL parser. We follow a pipeline approach where the parser does (1) unlabeled *span detection* to determine the arguments of a given verb, and (2) *question generation* to label the relationship between the predicate and each detected span. Our best model uses a span-based representation similar to that introduced by Lee et al. [2016] and a custom LSTM to decode questions from a learned span encoding. Our model does not require syntactic information and can be trained directly from the crowdsourced span labels.

Experiments demonstrate that the model does well on our new data, achieving up to 82.2% span-detection F1 and 47.2% exact-match question accuracy relative to the human annotations. We also demonstrate the utility of learning to predict easily interpretable QA-SRL structures, using a simple data bootstrapping approach to expand our dataset further. By tuning our model to favor recall, we over-generate questions which can be validated using our annotation pipeline, allowing for greater recall without requiring costly redundant annotations in the question writing step. Performing this procedure on the training and development sets grows them by 20% and leads to improvements when retraining our models. Our final parser is highly accurate, achieving 82.6% question accuracy

Wh	Aux	Subj	Verb	Obj	Prep	Misc
Who			blamed	someone		
What	did	someone	blame	something	on	
Who			refused		to	do something
When	did	someone	refuse		to	do something
Who	might		put	something		somewhere
Where	might	someone	put	something		

Table 4.1: Example QA-SRL questions, decomposed into their slot-based representation. See He et al. [2015] for the full details. All slots draw from a small, deterministic set of options.

and 77.6% span-level precision in an human evaluation.

Our code and data is made publically available [3].

4.2 Data Annotation

A QA-SRL annotation consists of a set of question-answer pairs for each verbal predicate in a sentence, where each answer is a set of contiguous spans from the sentence. QA-SRL questions are defined by a 7-slot template shown in Table 4.1. We introduce a crowdsourcing pipeline to collect annotations rapidly, cheaply, and at large scale.

Pipeline Our crowdsourcing pipeline consists of a *generation* and *validation* step. In the generation step, a sentence with one of its verbs marked is shown to a single worker, who must write QA-SRL questions for the verb and highlight their answers in the sentence. The questions are passed to the validation step, where n workers answer each question or mark it as *invalid*. In each step, no two answers to distinct questions may overlap with each other, to prevent redundancy.

Instructions Workers are instructed that a *valid* question-answer pair must satisfy three criteria: 1) the question is grammatical, 2) the question-answer pair is asking about the time, place, participants, etc., of the target verb, and 3) all correct answers to each question are given.

[3] http://qasrl.org

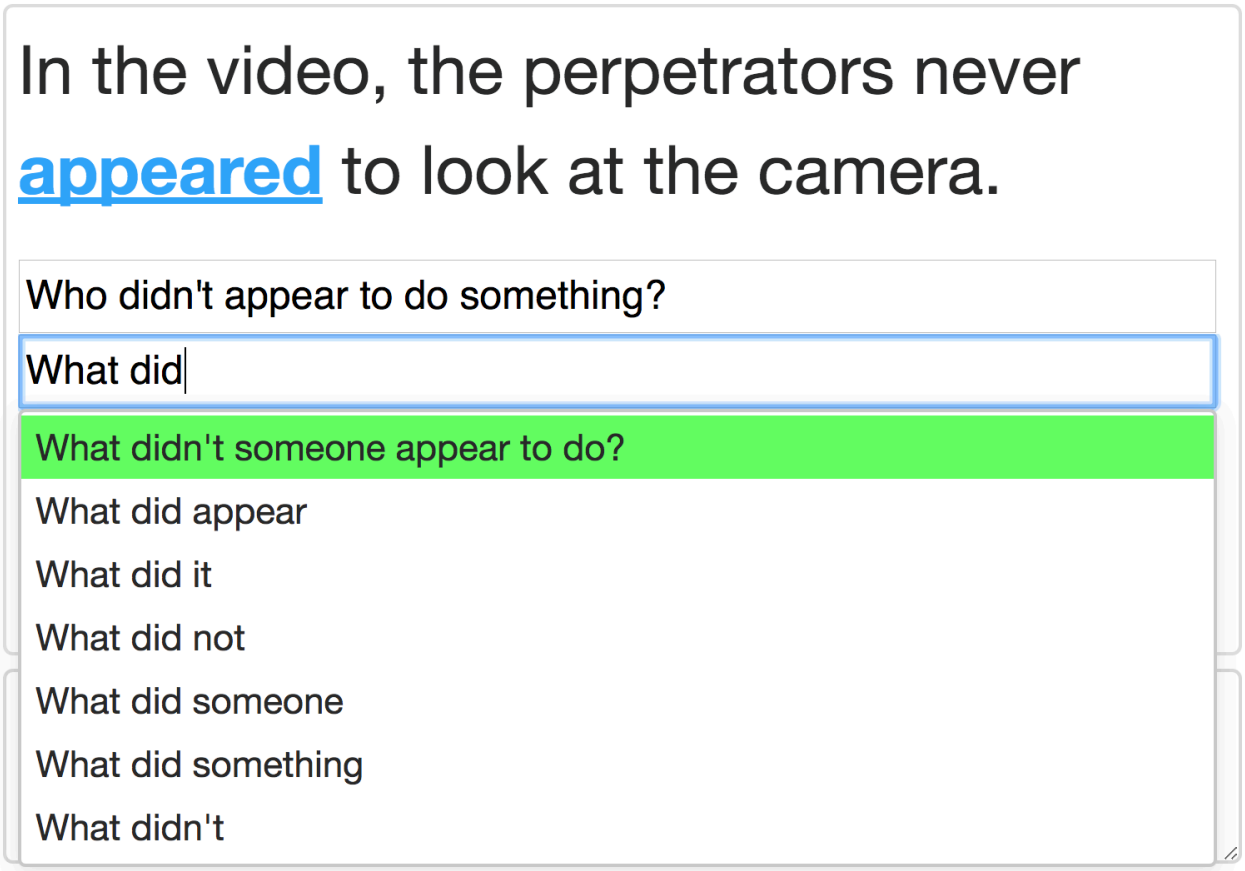


Figure 4.2: Interface for the generation step. Autocomplete shows completions of the current QA-SRL slot, and auto-suggest shows fully-formed questions (highlighted green) based on the previous questions.

Autocomplete We provide an autocomplete drop-down to streamline question writing. Autocomplete is implemented as a Non-deterministic Finite Automaton (NFA) whose states correspond to the 7 QA-SRL slots paired with a partial representation of the question's syntax. We use the NFA to make the menu more compact by disallowing obviously ungrammatical combinations (e.g., *What did been appeared?*), and the syntactic representation to auto-suggest complete questions about arguments that have not yet been covered (see Figure 4.2). The auto-suggest feature significantly reduces the number of keystrokes required to enter new questions after the first one, speeding up the annotation process and making it easier for annotators to provide higher recall.

Payment and quality control Generation pays 5c for the first QA pair (required), plus 5c, 6c, etc. for each successive QA pair (optional), to boost recall. The validation step pays 8c per verb, plus a 2c bonus per question beyond four. Generation workers must write at least 2 questions per verb and have 85% of their questions counted valid, and validators must maintain 85% answer span agreement with others, or they are disqualified from further work. A validator's answer is considered to agree with others if their answer span overlaps with answer spans provided by a majority of workers.

Preprocessing We use the Stanford CoreNLP tools Manning et al. [2014] for sentence segmentation, tokenizing, and POS-tagging. We identify verbs by POS tag, with heuristics to filter out auxiliary verbs while retaining non-auxiliary uses of "have" and "do." We identify conjugated forms of each verb for the QA-SRL templates by finding them in Wiktionary.[4]

Dataset We gathered annotations for 133,479 verb mentions in 64,018 sentences (1.27M tokens) across 3 domains: Wikipedia, Wikinews, and science textbook text from the Textbook Question Answering (TQA) dataset Kembhavi et al. [2017]. We partitioned the source documents into train, dev, and test, sampled paragraph-wise from each document with an 80/10/10 split by sentence.

[4] www.wiktionary.org

	Wikipedia	Wikinews	Science
Sentences	15,000	14,682	46,715
Verbs	32,758	34,026	66,653
Questions	75,867	80,081	143,388
Valid Qs	67,146	70,555	127,455

Table 4.2: Statistics for the dataset with questions written by workers across three domains.

Annotation in our pipeline with $n = 2$ validators took 9 days on Amazon Mechanical Turk.[5] 1,165 unique workers participated, annotating a total of 299,308 questions. Of these, 265,140 (or 89%) were considered valid by both validators, for an average of 1.99 valid questions per verb and 4.14 valid questions per sentence. See Table 4.2 for a breakdown of dataset statistics by domain. The total cost was $43,647.33, for an average of 32.7c per verb mention, 14.6c per question, or 16.5c per valid question. For comparison, He et al. [2015] interviewed and hired contractors to annotate data at much smaller scale for a cost of about 50c per verb. Our annotation scheme is cheaper, far more scalable, and provides more (though noisier) supervision for answer spans.

To allow for more careful evaluation, we validated 5,205 sentences at a higher density (up to 1,000 for each domain in dev and test), re-running the generated questions through validation with $n = 3$ for a total of 6 answer annotations for each question.

Quality Judgments of question validity had moderate agreement. About 89.5% of validator judgments rated a question as valid, and the agreement rate between judgments of the same question on whether the question is invalid is 90.9%. This gives a Fleiss's Kappa of 0.51. In the higher-density re-run, validators were primed to be more critical: 76.5% of judgments considered a question valid, and agreement was at 83.7%, giving a Fleiss's Kappa of 0.55.

Despite being more critical in the denser annotation round, questions marked valid in the original dataset were marked valid by the new annotators in 86% of cases, showing our data's relatively high precision. The high precision of our annotation pipeline is also backed up by our small-scale manual evaluation (see Coverage below).

[5] www.mturk.com

Answer spans for each question also exhibit good agreement. On the original dataset, each answer span has a 74.8% chance to exactly match one provided by another annotator (up to two), and on the densely annotated subset, each answer span has an 83.1% chance to exactly match one provided by another annotator (up to five).

Coverage Accurately measuring recall for QA-SRL annotations is an open challenge. For example, question 6 in Figure 4.1 reveals an inferred temporal relation that would not be annotated as part of traditional SRL. Exhaustively enumerating the full set of such questions is difficult, even for experts.

However, we can compare to the original QA-SRL dataset He et al. [2015], where Wikipedia sentences were annotated with 2.43 questions per verb. Our data has lower—but loosely comparable—recall, with 2.05 questions per verb in Wikipedia.

In order to further analyze the quality of our annotations relative to He et al. [2015], we reannotate a 100-verb subset of their data both manually (aiming for exhaustivity) and with our crowdsourcing pipeline. We merge the three sets of annotations, manually remove bad questions (and their answers), and calculate the precision and recall of the crowdsourced annotations and those of He et al. [2015] against this pooled, filtered dataset (using the span detection metrics described in section 3.4). Results, shown in Table 4.3, show that our pipeline produces comparable precision with only a modest decrease in recall. Interestingly, re-adding the questions rejected in the validation step greatly increases recall with only a small decrease in precision, showing that validators sometimes rejected questions considered valid by the authors. However, we use the filtered dataset for our experiments, and in section 4.6, we show how another crowdsourcing step can further improve recall.

	P	R	F
He et al. [2015]	97.5	86.6	91.7
This work	95.7	72.4	82.4
This work (unfiltered)	94.9	85.4	89.9

Table 4.3: Precision and recall of our annotation pipeline on a merged and validated subset of 100 verbs. The unfiltered number represents relaxing the restriction that none of 2 validators marked the question as invalid.

4.3 Models

Given a sentence $\boldsymbol{X} = x_0, \ldots, x_n$, the goal of a QA-SRL parser is to produce a set of tuples $(v_i, \boldsymbol{Q_i}, \mathcal{S}_i)$, where $v \in \{0, \ldots, n\}$ is the index of a verbal predicate, $\boldsymbol{Q}_i$ is a question, and $\mathcal{S}_i \in \{(i, j) \mid i, j \in [0, n], j \geq i\}$ is a set of spans which are valid answers. Our proposed parsers construct these tuples in a three-step pipeline:

1. *Verbal predicates* are identified using the same POS-tags and heuristics as in data collection (see section 4.2).
2. *Unlabeled span detection* selects a set $\mathcal{S}_v$ of spans as arguments for a given verb v.
3. *Question generation* predicts a question for each span in $\mathcal{S}_v$. Spans are then grouped by question, giving each question a set of answers.

We describe two models for unlabeled span detection in subsection 4.3.1, followed by question generation in subsection 4.3.2. All models are built on an LSTM encoding of the sentence. Like He et al. [2017], we start with an input $\boldsymbol{X}_v = \{\boldsymbol{x}_0 \ldots \boldsymbol{x}_n\}$, where the representation $\boldsymbol{x}_i$ at each time step is a concatenation of the token w_i's embedding and an embedded binary feature $(i = v)$ which indicates whether w_i is the predicate under consideration. We then compute the output representation $\boldsymbol{H}_v = \text{BiLSTM}(\boldsymbol{X}_v)$ using a stacked alternating LSTM Zhou and Xu [2015] with highway connections [Srivastava et al., 2015] and recurrent dropout [Gal and Ghahramani, 2016]. Since the span detection and question generation models both use an LSTM encoding, this component could in principle be shared between them. However, in preliminary experiments we found that sharing hurt performance, so for the remainder of this work each model is trained

independently.

4.3.1 Span Detection

Given an encoded sentence $\boldsymbol{H}_v$, the goal of span detection is to select the spans $\mathcal{S}_v$ that correspond to arguments of the given predicate. We explore two models: a sequence-tagging model with BIO encoding, and a span-based model which assigns a probability to every possible span.

BIO Sequence Model

Our BIO model predicts a set of spans via a sequence $\boldsymbol{y}$ where each $y_i \in \{\boldsymbol{B}, \boldsymbol{I}, \boldsymbol{O}\}$, representing a token at the beginning, interior, or outside of any span, respectively. Similar to He et al. [2017], we make independent predictions for each token at training time, and use Viterbi decoding to enforce hard BIO-constraints[6] at test time. The resulting sequences are in one-to-one correspondence with sets $\mathcal{S}_v$ of spans which are pairwise non-overlapping. The locally-normalized tag distributions are computed from the BiLSTM outputs $\boldsymbol{H}_v = \{\boldsymbol{h}_{v0}, \ldots, \boldsymbol{h}_{vn}\}$:

$$p(y_t \mid \boldsymbol{x}) \propto exp(\boldsymbol{w}_{\text{tag}}^{\mathsf{T}} \text{MLP}(\boldsymbol{h}_{vt}) + \boldsymbol{b}_{\text{tag}}) \tag{4.1}$$

Span-based Model

Our span-based model makes independent binary decisions for all $O(n^2)$ spans in the sentence. Following Lee et al. [2016], the representation of a span (i, j) is the concatenation of the BiLSTM output at each endpoint:

$$\boldsymbol{s}_{vij} = [\boldsymbol{h}_{vi}, \boldsymbol{h}_{vj}]. \tag{4.2}$$

[6] E.g., an I-tag should only follow a B-tag.

The probability that the span is an argument of predicate v is computed by the sigmoid function:

$$p(y_{ij} \mid \boldsymbol{X}_v) = \sigma(\boldsymbol{w}_{\text{span}}^{\intercal}\text{MLP}(\boldsymbol{s}_{vij}) + \boldsymbol{b}_{\text{span}}) \tag{4.3}$$

At training time, we minimize the binary cross entropy summed over all n^2 possible spans, counting a span as a positive example if it appears as an answer to any question.

At test time, we choose a threshold τ and select every span that the model assigns probability greater than τ, allowing us to trade off precision and recall.

4.3.2 Question Generation

We introduce two question generation models. Given a span representation $\boldsymbol{s}_{vij}$ defined in section 4.3.1, our models generate questions by picking a word for each question slot (see section 4.2). Each model calculates a joint distribution $p(\boldsymbol{y} \mid \boldsymbol{X}_v, \boldsymbol{s}_{vij})$ over values $\boldsymbol{y} = (y_1, \ldots, y_7)$ for the question slots given a span $\boldsymbol{s}_{\boldsymbol{vij}}$, and is trained to minimize the negative log-likelihood of gold slot values.

Local Model

The local model predicts the words for each slot independently:

$$p(y_k \mid \boldsymbol{X}_v, \boldsymbol{s}_{vij}) \propto \exp(\boldsymbol{w}_k^{\intercal}\text{MLP}(\boldsymbol{s}_{vij}) + \boldsymbol{b}_k). \tag{4.4}$$

Sequence Model

The sequence model uses the machinery of an RNN to share information between slots. At each slot k, we apply a multiple layers of LSTM cells:

$$\boldsymbol{h}_{l,k}, \boldsymbol{c}_{l,k} = \text{LSTMCELL}_{l,k}(\boldsymbol{h}_{l-1,k}, \boldsymbol{h}_{l,k-1}, \boldsymbol{c}_{l,k-1}) \tag{4.5}$$

where the initial input at each slot is a concatenation of the span representation and the embedding of the previous word of the question: $\boldsymbol{h}_{0,k} = [\boldsymbol{s}_{vij}; \boldsymbol{y}_{k-1}]$. Since each question slot predicts from a different set of words, we found it beneficial to use separate weights for the LSTM cells at each slot k. During training, we feed in the gold token at the previous slot, while at test time, we use the predicted token. The output distribution at slot k is computed via the final layers' output vector $\boldsymbol{h}_{Lk}$:

$$p(y_k \mid \boldsymbol{X}_v, \boldsymbol{s}_{vij}) \propto \exp(\boldsymbol{w}_k^{\intercal} \text{MLP}(\boldsymbol{h}_{Lk}) + \boldsymbol{b}_k) \tag{4.6}$$

4.4 Experimental Setup

Hyperparameters The parameters of our LSTMs are initialized with random orthonormal matrices as described by Saxe et al. [2013]. Input tokens are lower-cased, and the word vectors are pre-initialized with the 100-dimensional Glove embeddings trained on 6B tokens Pennington et al. [2014] and fine-tuned during training. Tokens which are not covered by the Glove embeddings are assigned to the UNK vector. The embedding of the binary predicate indicator feature is also 100 dimensions. The text-encoder BiLSTM consists of 4 layers, uses a hidden size of 300 and . The output prediction feed-forward neural network for each model consists of a single 100 dimensional hidden layer with the non-rectified linear unit nonlinearity. For the sequential question generation model, each timestep consists of 4 layers of LSTMCells with a hidden size of 200.

Training All models are trained using Adadelta Zeiler [2012] with $\epsilon = 1e^{-6}$ and $\rho = 0.95$ and a mini-batch size of 80. The span encoding BiLSTM uses a recurrent dropout rate of 0.1, and we clip gradients with a norm greater than 1. All models were trained until performance on the development set did not improve for 10 epochs[7]. Our models were implemented in PyTorch[8] using the AllenNLP toolkit Gardner et al..

[7] All models completed training within 40 epochs, which took less than 4 hours on a single Titan X Pascal GPU.
[8] `http://pytorch.org/`

4.5 Initial Results

Automatic evaluation for QA-SRL parsing presents multiple challenges. In this section, we introduce automatic metrics that can help us compare models. In section 4.7, we will report human evaluation results for our final system.

4.5.1 Span Detection

Metrics We evaluate span detection using a modified notion of precision and recall. We count predicted spans correct if they match any of the labeled spans in the dataset. Since each predicted span could potentially be a match to multiple questions (due to overlapping annotations) we map each predicted span to one matching question in the way that maximizes measured recall using maximum bipartite matching. We use both exact match and intersection-over-union (IOU) greater than 0.5 as matching criteria.

Results Table 4.4 shows span detection results on the development set. We report results for the span-based models at two threshold values τ: $\tau = 0.5$, and $\tau = \tau^*$ maximizing F1. The span-based model significantly improves over the BIO model in both precision and recall, although the difference is less pronounced under IOU matching.

4.5.2 Question Generation

Metrics Like all generation tasks, evaluation metrics for question generation must contend with the fact that there are in general multiple possible valid questions for a given predicate-argument pair. For instance, the question "What did someone blame something on?" may be rephrased as "Who was blamed for something?" However, due to the constrained space of possible questions defined by QA-SRL's slot format, accuracy-based metrics can still be informative. In particular, we report the rate at which the predicted question exactly matches the gold question, as well as

Exact Match			
	P	R	F
BIO	69.0	75.9	72.2
Span ($\tau = 0.5$)	81.7	80.9	81.3
Span ($\tau = \tau*$)	80.0	84.7	82.2
IOU ≥ 0.5			
	P	R	F
BIO	80.4	86.0	83.1
Span ($\tau = 0.5$)	87.5	84.2	85.8
Span ($\tau = \tau*$)	83.8	93.0	88.1

Table 4.4: Results for Span Detection on the dense development dataset. Span detection results are given with the cutoff threshold τ at 0.5, and at the value which maximizes F-score. The top chart lists precision, recall and F-score with exact span match, while the bottom reports matches where the intersection over union (IOU) is ≥ 0.5.

	EM	PM	SA
Local	44.2	62.0	83.2
Seq.	47.2	62.3	82.9

Table 4.5: Question Generation results on the dense development set. **EM** - Exact Match accuracy, **PM** - Partial Match Accuracy, **SA** - Slot-level accuracy

a relaxed match where we only count the question word (WH), subject (SBJ), object (OBJ) and Miscellaneous (Misc) slots (see Table 4.1). Finally, we report average slot-level accuracy.

Results Table 4.5 shows the results for question generation on the development set. The sequential model's exact match accuracy is significantly higher, while word-level accuracy is roughly comparable, reflecting the fact that the local model learns the slot-level posteriors.

4.5.3 Joint results

Table 4.6 shows precision and recall for joint span detection and question generation, using exact match for both. This metric is exceedingly hard, but it shows that almost 40% of our model's predictions are exactly correct in both span and question. In section 4.7, we use human evaluation to get a more accurate assessment of our model's true accuracy.

	P	R	F
Span + Local	37.8	43.7	40.6
Span + Seq. ($\tau = 0.5$)	39.6	45.8	42.4

Table 4.6: Joint span detection and question generation results on the dense development set, using exact-match for both spans and questions.

4.6 Data Expansion

Since our trained parser can produce full QA-SRL annotations, its predictions can be validated by the same process as in our original annotation pipeline, allowing us to focus annotation efforts towards filling potential data gaps.

By detecting spans at a low probability cutoff, we over-generate QA pairs for already-annotated sentences. Then, we filter out QA pairs whose answers overlap with answer spans in the existing annotations, or whose questions match existing questions. What remains are candidate QA pairs which fill gaps in the original annotation. We pass these questions to the validation step of our crowdsourcing pipeline with $n = 3$ validators, resulting in new labels.

We run this process on the training and development partitions of our dataset. For the development set, we use the trained model described in the previous section. For the training set, we use a relaxed version of jackknifing, training 5 models over 5 different folds. We generate 92,080 questions at a threshold of $\tau = 0.2$. Since in this case many sentences have only one question, we restructure the pay to a 2c base rate with a 2c bonus per question after the first (still paying no less than 2c per question).

Data statistics 46,017 (50%) of questions run through the expansion step were considered valid by all three annotators. In total, after filtering, the expansion step increased the number of valid questions in the train and dev partitions by 20%. However, for evaluation, since our recall metric identifies a single question for each answer span (via bipartite matching), we filter out likely question paraphrases by removing questions in the expanded development set whose answer spans have

two overlaps with the answer spans of one question in the original annotations. After this filtering, the expanded development set we use for evaluation has 11.5% more questions than the original development set.

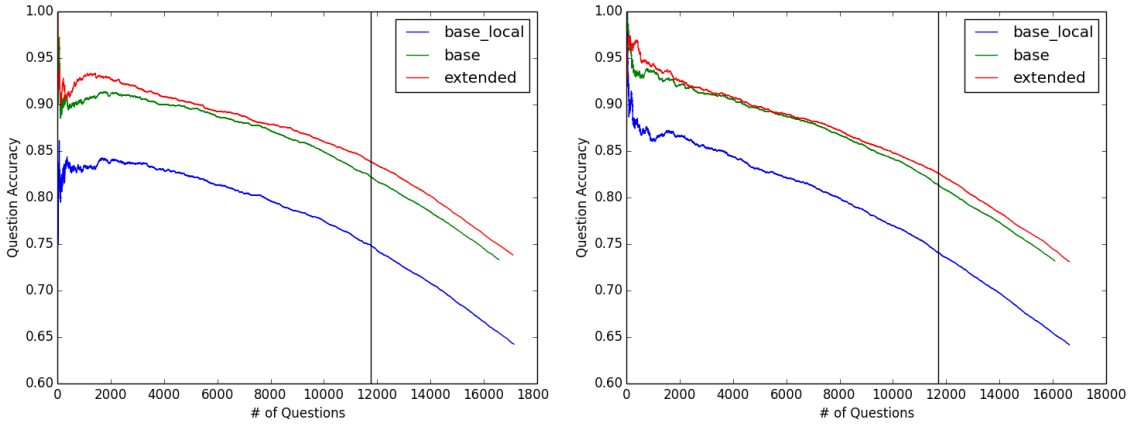
The total cost including MTurk fees was $8,210.66, for a cost of 8.9c per question, or 17.8c per valid question. While the cost per valid question was comparable to the initial annotation, we gathered many more negative examples (which may serve useful in future work), and this method allowed us to focus on questions that were missed in the first round and improve the exhaustiveness of the annotation (whereas it is not obvious how to make fully crowdsourced annotation more exhaustive at a comparable cost per question).

Retrained model We retrained our final model on the training set extended with the new valid questions, yielding modest improvements on both span detection and question generation in the development set (see Table 4.7). The span detection numbers are higher than on the original dataset, because the expanded development data captures true positives produced by the original model (and the resulting increase in precision can be traded off for recall as well).c

4.7 Final Evaluation

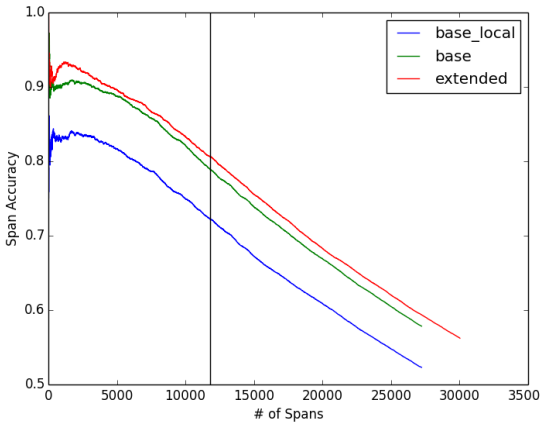
We use the crowdsourced validation step to do a final human evaluation of our models. We test 3 parsers: the span-based span detection model paired with each of the local and sequential question generation models trained on the initial dataset, and our final model (span-based span detection and sequential question generation) trained with the expanded data.

Methodology On the 5,205 sentence densely annotated subset of dev and test, we generate QA-SRL labels with all of the models using a span detection threshold of $\tau = 0.2$ and combine the questions with the existing data. We filter out questions that fail the autocomplete grammaticality check (counting them invalid) and pass the data into the validation step, annotating each question

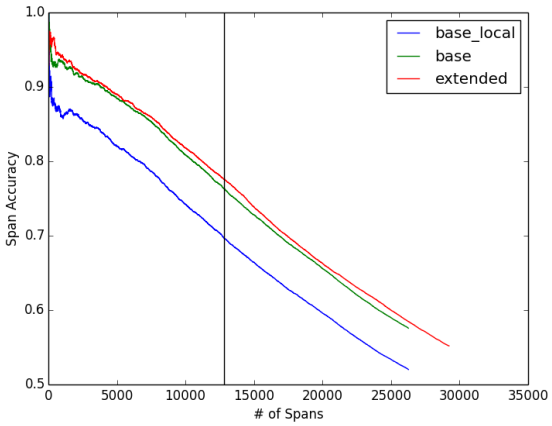


(a) Question accuracy on Dev

(b) Question accuracy on Test



(c) Span accuracy on Dev



(d) Span accuracy on Test

Figure 4.3: Human evaluation accuracy for questions and spans, as each model's span detection threshold is varied. Questions are considered correct if 5 out of 6 annotators consider it valid. Spans are considered correct if their question was valid, and the span was among those labeled by human annotators for that question. The vertical line indicates a threshold value where the number of questions per sentence matches that of the original labeled data (2 questions / text).

Exact Match				
	P	R	F	AUC
Original	80.8	86.8	83.7	.906
Expanded	82.9	86.4	84.6	.910
IOU ≥ 0.5				
	P	R	F	AUC
Original	87.1	93.2	90.1	.946
Expanded	87.9	93.1	90.5	.949

(a) Span Detection results with $\tau*$.

	EM	PM	WA
Original	50.5	64.4	84.1
Expanded	50.8	64.9	84.1

(b) Question Generation results

	P	R	F
Original	47.5	46.9	47.2
Expanded	44.3	55.0	49.1

(c) Joint span detection and question generation results with $\tau = 0.5$

Table 4.7: Results on the expanded development set comparing the full model trained on the original data, and with the expanded data.

to a total of 6 validator judgments. We then compute question and span accuracy as follows: A question is considered correct if 5 out of 6 annotators consider it valid, and a span is considered correct if its generated question is correct and the span is among those selected for the question by validators. We rank all questions and spans by the threshold at which they are generated, which allows us to compute accuracy at different levels of recall.

Results Figure 4.3 shows the results. As expected, the sequence-based question generation models are much more accurate than the local model; this is largely because the local model generated many questions that failed the grammaticality check. Furthermore, training with our expanded data results in more questions and spans generated at the same threshold. If we choose a threshold value which gives a similar number of questions per sentence as were labeled in the original data annotation (2 questions / verb), question and span accuracy are 82.64% and 77.61%, respectively.

A much larger super eruption in Colorado **produced** over 5,000 cubic kilometers of material.	Produced	What produced something?	A much larger super eruption
		Where did something produce something?	in Colorado
		What did something produce?	over 5,000 cubic kilometers of material
In the video, the perpetrators never **appeared** to **look** at the camera.	appeared	Where didn't someone appear to do something?	In the video
		Who didn't appear to do something?	the perpetrators
		When did someone appear?	never
		What didn't someone appear to do?	look at the camera
			to look at the camera
	look	Where didn't someone look at something?	In the video
		Who didn't look?	the perpetrators
		What didn't someone look at?	the camera
Some of the vegetarians he **met** were members of the Theosophical Society, which had been **founded** in 1875 to further universal brotherhood, and which was **devoted** to the study of Buddhist and Hindu literature.	met	Who met someone?	Some of the vegetarians
			vegetarians
		Who met?	he
		What did someone meet?	members of the Theosophical Society
	founded	What had been founded?	members of the Theosophical Society
			the Theosophical Society
		When was something founded?	in 1875
			1875
		Why has something been founded?	to further universal brotherhood
	devoted	What was devoted to something?	members of the Theosophical Society
		What was something devoted to?	the study of Buddhist and Hindu literature

Table 4.8: System output on 3 randomly sampled sentences from the development set (1 from each of the 3 domains). Spans were selected with $\tau = 0.5$. Questions and spans with a red background were marked incorrect during human evaluation.

Table 4.8 shows the output of our best system on 3 randomly selected sentences from our development set (one from each domain). The model was overall highly accurate—only one question and 3 spans are considered incorrect, and each mistake is nearly correct,[9] even when the sentence contains a negation.

4.8 Conclusion

In this chapter, we demonstrated that QA-SRL can be scaled to large datasets, enabling a new methodology for labeling and producing predicate-argument structures at a large scale. We present a new, scalable approach for crowdsourcing QA-SRL, which allowed us to collect a new dataset covering over 250,000 question-answer pairs from 75,000 sentences in just 9 days. We demonstrate the utility of this data by training the first parser which is able to produce high-quality QA-SRL structures. Finally, we demonstrated that the validation stage of our crowdsourcing pipeline, in combination with our parser tuned for recall, can be used to add new annotations to the dataset, increasing recall.

[9]The incorrect question "When did someone appear?" would be correct if the Prep and Misc slots were corrected to read "When did someone appear to do something?"

Chapter 5

Multitask Training with QA-SRL

5.1 Introduction

In chapter 3, we showed that it was possible to leverage a large SRL dataset in order to improve performance on problem with fewer labelled examples, even if the annotation scheme is different. In chapter 4, we collected a large dataset of Question-Answer driven SRL (QA-SRL), and showed that we could learn a high-quality parser using state-of-the-art modeling techniques. QA-SRL labels are similar in structure to traditional SRL label schemes such as PropBank or FrameNet, with the exception that arguments are labeled with questions, rather than a discrete label. Given this similar structure, a natural question arises: can we combine QA-SRL and PropBank annotations in a multitask objective, and achieve improved performance on the latter?

A related question emerges from recent work in deep, contextualized word representations. ELMo [Peters et al., 2018] is a recent method for learning deep contextualized word representations trained to perform language modeling on an unlabeled text corpus consisting of billions of tokens. Incorporating these representations into existing models has proven to significantly improve the state of the art on a wide range of NLP tasks. These improvements in many cases exceed what has been achieved by way of multitask training. It is unknown whether the improvements obtained by

pretraining a deep word representation to perform language modeling are complementary to those achievable by multitask training with a more similar scaffold task.

In this chapter, we investigate both questions. We investigate multitask training for PropBank Semantic Role Labelling on the CoNLL 2005 dataset, with the addition of QA-SRL as a scaffold task. We additionally experiment with the inclusion of ELMo. Our results indicate that while multitask training with QA-SRL can provide improvements when there is little in-domain data, these improvements are overshadowed by those obtained by incorporating ELMo embeddings into the model, suggesting that multitask training may simply be learning improved lexical representations which can be more effectively obtained by language modeling on large unlabeled corpora.

5.2 Model

For these multitask training experiments, we employ a model similar to the LSGN model for end-to-end SRL [He et al., 2018]. The probability of an output structure $P(Y \mid X, v)$ is defined as:

$$P(Y \mid X, v) = \prod_{r \in \mathcal{R}} P(y_r \mid X) \tag{5.1}$$

$$P(y_r \mid X, v) = \frac{\exp(\phi(r, y_r))}{\sum_{l' \in \mathcal{L} \cup \{\epsilon\}} \exp(\phi(r, l'))} \tag{5.2}$$

$\phi(r, t)$ is a function which scores each pair of *part* and *target* in the structure. Since we assume that predicates are provided, or identified earlier in the pipeline, we do not need to score the predicate ourselves as is done by He et al. [2018], and instead our part scores are factored into an argument score (ie., *unlabeled argument detection*, which measures the likelihood that a span is an argument), and role score which scores each label for a given span.

$$\begin{aligned}\phi(r,t) &= \phi(s,l) \\ &= \Phi_{arg}(s) + \Phi_{role}(s,l)\end{aligned} \tag{5.3}$$

These scores are conditioned on the sentence x and predicate index v, by way of the same predicate-contextualized LSTM encoding $\boldsymbol{H}_v$ as described in section 4.3. These scores are computed with feed-forward networks, using the span representations $\boldsymbol{s}_{vij}$ defined in Equation 4.2 as input:

$$\Phi_{arg}(s) = \boldsymbol{w}_{arg}^T \text{MLP}(\boldsymbol{s}_{vij}) + \boldsymbol{b}_{arg} \tag{5.4}$$

$$\Phi_{role}(s,l) = \boldsymbol{w}_{arg}^T \text{MLP}([\boldsymbol{s}_{vij}, \boldsymbol{g}(l)]) \tag{5.5}$$

where $\boldsymbol{g}(l)$ is an embedding of the role label l. A further benefit of this factorization of $\phi(s,t)$ is that the argument score $\Phi_{arg}(s)$ can be used to perform an initial pruning of spans to those with the highest score, reducing the number of span-role scores $\Phi_{role}(s,l)$ which must be computed.

5.2.1 Using QA-SRL

There are three possible methods of incorporating our QA-SRL annotations which we consider: pretraining, scaffolding, and joint embedding. Here, we describe each method in turn. However, preliminary experiments showed the joint embedding method consistently worked the best, so we only report results for this method.

Pretraining The first method is to pretrain the shared LSTM on the *question prediction* task described in subsection 4.3.2, and using the subsequent LSTM and word-embedding parameters to initialize a model which is then fine-tuned on the SRL task.

Scaffold Training The second approach is to use QA-SRL question prediction (described in subsection 4.3.2) as a *scaffold task* – an auxiliary task which is only used during training in a multitask setup. Similar to Swayamdipta et al. [2017], who used syntactic parsing as a scaffold task for SRL, we simply sum the loss contributed by the main SRL task and our auxiliary task (weighted by a hyperparameter δ).

$$loss(s,t) = loss_{srl}(s,t) + \delta \cdot loss_{qasrl}(s,t) \tag{5.6}$$

Like that work, our datasets are disjoint – we do not have QA-SRL and SRL annotations for the same sentences. Therefore, each example only receives loss for one of the two tasks, and the loss for the other task is set to 0.

Multitask Training We can leverage the close structural similarity of SRL and QA-SRL labels in order to learn a multitask model by embedding both sets of labels in the same space, in a manner analogous to that described in chapter 3. We treat the set of questions for a given predicate as given, and use these as the set of possible labels $\mathcal{L}$ in place of the SRL roles. We can form the label embedding $\boldsymbol{g}_{qa}(l)$ for a QA-SRL question to be used in the role score (Equation 5.5) using any function which embeds a sequence. For this experiment we use a summed bag-of-embeddings:

$$\boldsymbol{g}_{qa}(\boldsymbol{y}) = \sum_k \boldsymbol{g}_k(y_k) \tag{5.7}$$

Since SRL and QA-SRL have quite different notions of what constitutes a valid argument,[1] we allow the model to make different span decisions for each task. We do this by augmenting the argument score Φ_{arg} from Equation 5.4 with an an embedding indicating which problem (or

[1]QA-SRL annotations often have multiple overlapping spans for a given question, since the answers were provided by multiple annotators, and because inferences such as coreference resolution are often made. Additionally, different questions can have overlapping answers. For example, in the sentence "While John was in college, he played football." could have the question "Who played something?" with the answer "John" and the question "When did someone play something" with the answer "When John was in college". In contrast, PropBank arguments do not overlap, and are more closely restricted by syntax, and therefore do not allow for coreferences to be resolved.

domain) the current example comes from:

$$\Phi_{arg}(s) = \boldsymbol{w}_{arg}^T \text{MLP}([\boldsymbol{s}_{vij}, \boldsymbol{g}_{dom}(d)] + \boldsymbol{b}_{arg} \tag{5.8}$$

5.2.2 Incorporating ELMo

In order to incorporate the ELMo embeddings, we simply concatenate the output of the ELMo network at each timestep to the word and predicate indicator embeddings which are the input to the LSTM in our baseline model. We also fine-tune the ELMo network during training.

5.3 Experimental Setup

Data We use the entire initial (unexpanded) QA-SRL Bank 2.0 dataset for the QA-SRL task. Our target data is the CoNLL 2005 [Carreras and Màrquez, 2005a] PropBank shared task. In order to explore the relative effect of pretraining in regimes with varying amounts of target-problem data, we vary the number of training sections used.

ELMo Word Embeddings We use the large 2-layer ELMo model trained on the 1 Billion Word Benchmark [Chelba et al., 2013].

5.4 Results

Table 5.1 and Figure 5.1 shows the results from our multitask experiment. In regimes with less training data, training the baseline model on QA-SRL together with CoNLL gives a modest improvement over the baseline. These improvements are roughly evenly split between improved span detection (as indicated by the unlabeled argument prediction scores) and improved label classification (as indicated by results of decoding with gold spans). With the full CoNLL 2005

	# CoNLL 2005 Training Sections				
Method	1	3	5	10	21
Predicted Spans					
CoNLL only	62.9	69.2	73.0	78.0	81.8
CoNLL & QASRL	66.9	73.5	75.3	78.9	81.8
CoNLL only w/ ELMo	75.4	79.5	81.8	84.0	85.6
CoNLL & QASRL w/ ELMo	75.6	79.6	81.7	83.9	85.6
Gold Spans					
CoNLL only	81.2	83.8	85.7	89.1	91.1
CoNLL & QA-SRL	83.6	85.5	87.2	89.6	91.4
CoNLL only w/ ELMo	85.5	88.4	89.8	91.7	92.9
CoNLL & QA-SRL w/ ELMo	85.6	88.0	89.6	91.3	92.7
Unlabeled Argument Prediction					
CoNLL only	73.7	79.4	81.9	84.7	87.8
CoNLL & QA-SRL	77.6	82.9	83.5	85.7	87.5
CoNLL only w/ ELMo	85.5	87.5	88.7	90.0	90.7
CoNLL & QA-SRL w/ ELMo	84.8	87.8	88.8	90.2	90.9

Table 5.1: Argument detection F-score for the multitask training experiment, with varying amounts of CoNLL 2005 data (measured in sections of the training set). When QA-SRL is included, we use the entire unexpanded training set from section 4.2. In addition to predicted argument detection, we report performance when gold spans are detected, as well as the unlabeled argument prediction scores.

training set included, no further gain is obtained from adding QA-SRL.

Adding ELMo embeddings to the full model results in a much larger improvement across all amounts of training data. Again, these gains come both from improved unlabeled span detection and better label classification. Furthermore, with the inclusion of ELMo, no further improvement is obtained by the addition of QA-SRL.

5.5 Discussion

Our preliminary experiment described above shows only a modest improvement from the incorporation of QA-SRL annotations. Here we describe several potential next steps, inspired by recent work in multitask learning and domain adaptation.

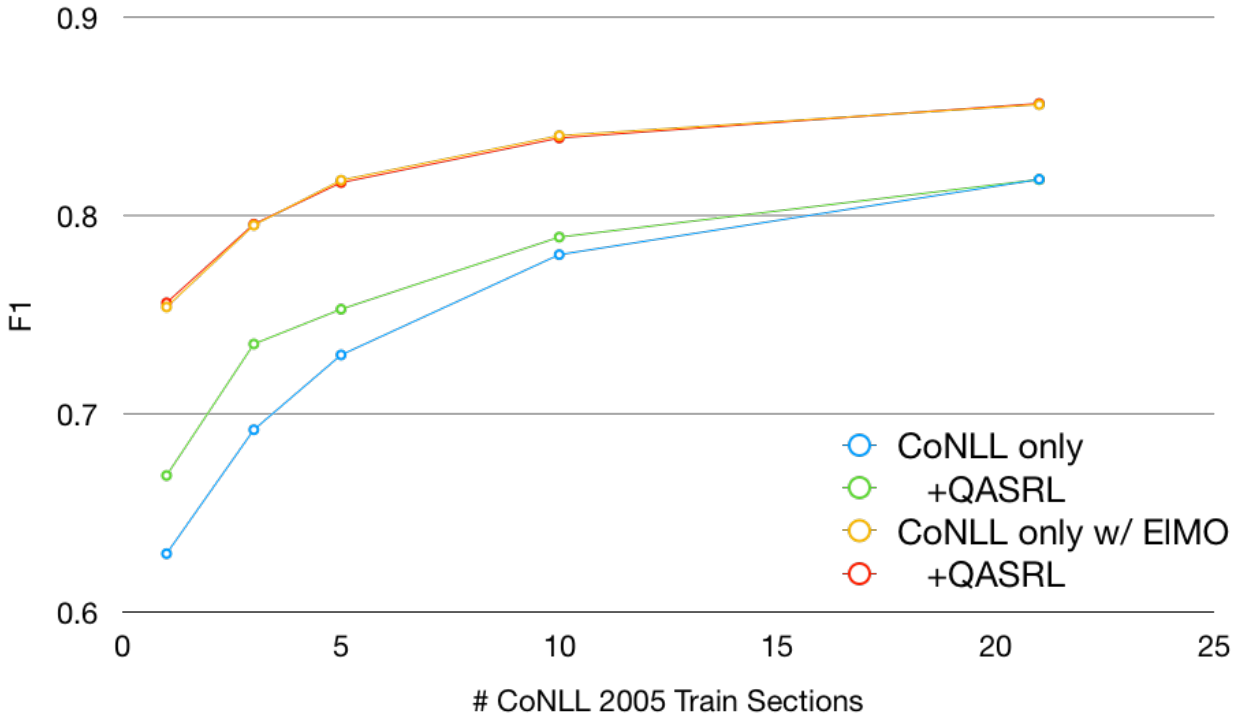


Figure 5.1: Argument detection F-score plotted as a function of increasing number of CoNLL training sections. These are the same results shown for Predicted Spans in Table 5.1. When QA-SRL is included, we use the entire unexpanded training set from section 4.2.

Recall-Oriented Softmax-Margin Swayamdipta et al. [2017] use a recall-oriented softmax-margin loss [Mohit et al., 2012; Gimpel and Smith, 2010] for scaffold training of frame semantic parsing using syntactic structures as the auxiliary task. This loss function augments the partition function of the maximum-likelihood objective with a cost function that more strongly penalizes false negatives, which is appropriate for our task, since the majority of spans are null. This loss was also used by Peng et al. [2018], who additionally multiple semantic formalisms on disjoint data by treating unobserved formalisms as latent structure variables.

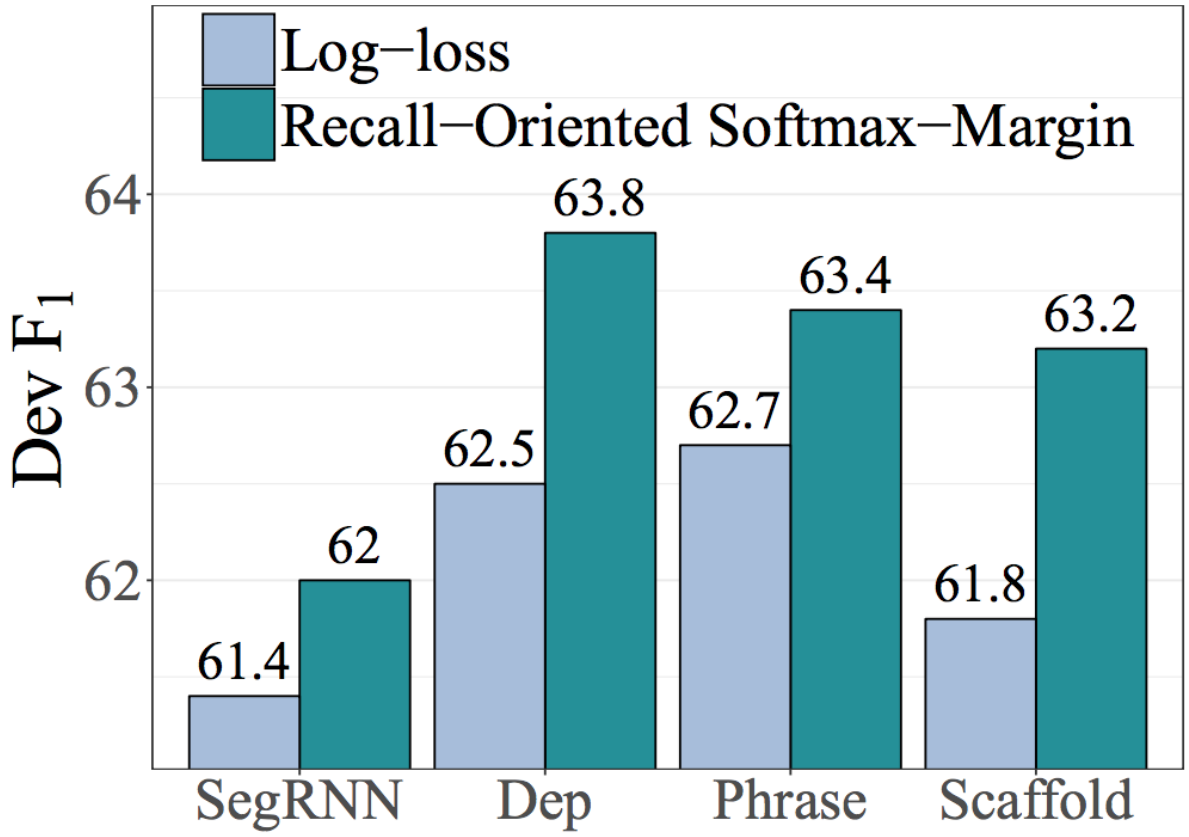


Figure 5.2: Figure taken from Swayamdipta et al. [2017]. Relative performance of the Log-loss objective and the Recall-Oriented Softmax-Margin objective. Not only does the latter perform better, but leads to larger gains from multi-task training with syntactic parsing.

Swayamdipta et al. [2017] not only find that using this loss function leads to higher performance overall, but their results also show that the improvement which results from multitask training with syntactic parsing as a scaffold task is larger when using this loss than when the standard maximum-likelihood loss is used (see Figure 5.2). If this trend proves to be the case for multitask training with QA-SRL as well, we might be able to improve the extent to which multitask training improves over the baseline, potentially even finding additional gains over ELMo.

Domain Adversarial Training Since QA-SRL Bank 2.0 is collected over a different set of sentences than the target PropBank domain, it is possible that the LSTM is learning to produce different

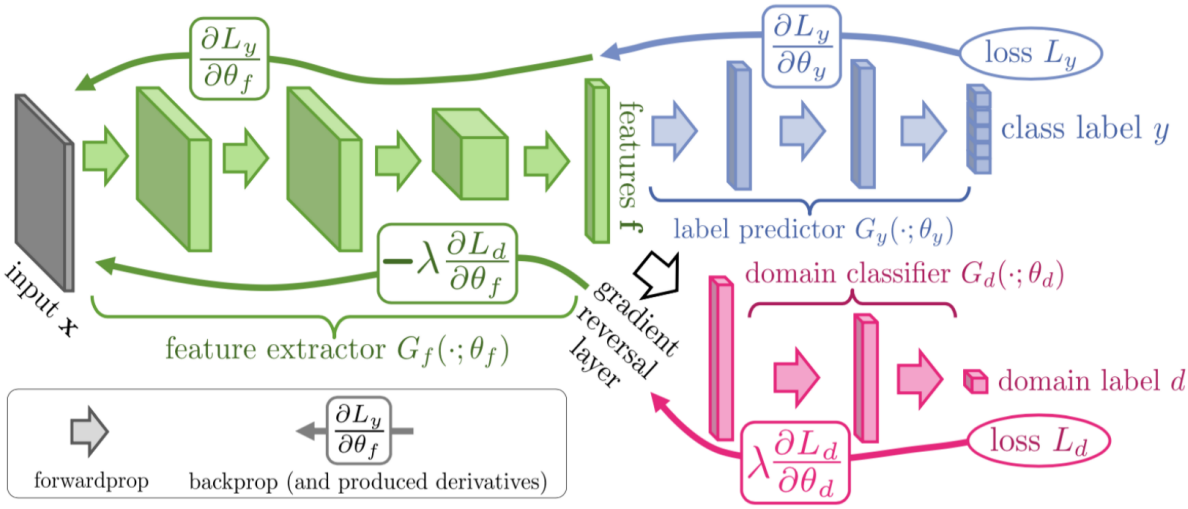


Figure 5.3: Figure taken from Ganin et al. [2016], demonstrating their approach for domain-adversarial training. The blue section represents the target task (in our case either SRL or QA-SRL parsing), while the red section represents a classifier which attempts to predict which domain the input came from. The gradients from this domain classifier are reversed, discouraging the feature learner below from producing features which are predictive of the domain.

features for each domain, reducing the sharing effect between the labels of each problem. This could help explain why less improvement was observed over models trained with the ELMo [Peters et al., 2018] contextualized word embeddings.

To solve this, we take inspiration from recent work on *domain adversarial training* for domain adaptation [Ganin et al., 2016]. Here, the goal is to encourage a model to learn feature representations which capture the information needed to solve the target problem, while ignoring superficial domain differences. To achieve this, a domain classifier is added as an auxiliary task, and the model is penalized to the extend that this classifier is able to accurately predict which domain the example belongs to (see Figure 5.3). Applying this auxilliary loss to our model could potentially force the model to more effectively model the similarity between PropBank and QA-SRL labels, rather than relying on simply learning the differences between the labelled text.

5.6 Conclusion

We proposed using QA-SRL in order to improve perfomance on traditional SRL formalisms, as well as additional tasks such as factuality prediction. Our experiment shows that in regimes with smaller amounts of training data, multi-task training with QA-SRL does indeed improve performance on PropBank parsing. However, the improvement is inferior to that which can be obtained by using the deep contextualised ELMo word embeddings trained for language modelling on large amounts of unlabeled text. Furthermore, after the addition of the ELMo embeddings, no additional improvement is obtained by multitask training with QA-SRL. This negative result raises important questions for future work as to the utility of multitask training with supervised data for related tasks, relative to simply pretraining a language model on large amounts of unlabeled text.

Chapter 6

Conclusion

We presented several works using neural network models in order to solve the problem of sparsity in existing SRL data. Our main contributions are:

1. First, in chapter 3, we presented a neural network model for semantic role labeling in which arguments and semantic roles are jointly embedded in a shared vector space. This model allows us to train jointly using multiple different annotation schemes (namely PropBank and FrameNet), which allows us to leverage datasets with a lot of annotations in order to improve performance on those with less.

2. Next, in chapter 4, we introduced a large, crowdsourced dataset of QA-SRL annotations, and the first high-quality QA-SRL parser. This dataset rivals the size of the largest PropBank dataset [Weischedel et al., 2011], but is collected at a fraction of the cost by utilizing non-expert crowd workers. Our model demonstrates that a high-quality semantic parser can be learned from this data.

3. Finally, in chapter 5, we investigated multitask training for PropBank SRL, using our large QA-SRL dataset as an auxilliary task. While we found that using the QA-SRL data improved performance in regimes with small amounts of in-domain PropBank data, these improvements

were overshadowed by those obtained by using deep contextual word representations trained on large amounts of unlabeled text. This experiment raises important questions as to the utility of multitask training relative to these unsupervised approaches.

6.1 Future Work

Further Modelling with QA-SRL The current version of QA-SRL utilized in chapter 4 covers verbal predicates within a single sentence, but could be extended to cover additional semantic phenomena. Events can also be expressed through nominal predicates, as covered by the NomBank dataset [Meyers et al., 2004]. The key challenge here would be identifying which nouns are evoking a predicate, as we cannot simply assume that every noun does so, as is the case with verbs.

Other domains, particularly those which involve *processes*, such as cooking recipes [Kiddon et al., 2015] and science processes [Louvan et al., 2016], often involve cross-sentence predicate-argument structures. For example, in the text:

"Combine flour and eggs in a large bowl. Pour into the baking sheet."

the span "flour and eggs" are the object of the predicate "pour". Such cross-sentence structures are not typically captured by traditional SRL annotation schemes, but could be easily labelled via QA-SRL simply by allowing annotators to select answer spans from multiple sentences. Similarly the *implicit* subject argument of "pour" could be labeled by allowing the annotator to indicate that the answer to "Who should pour something?" is the reader.

Factuality Prediction While our focus to this point has been on the task of Semantic Role Labelling, the QA-SRL dataset contains annotations that go beyond traditional SRL frameworks such as PropBank and FrameNet, encoding additional infomation that these formalisms do not. One such task is factuality predition [Rudinger et al., 2018; Lee et al., 2015], where the goal is to predict whether the event described by a given predicate happened or not. As can be seen in Table 4.8,

QA-SRL encodes factuality – our model correctly predicts, for example, that the predicates "appear" and "look" in sentence 2 did not occur, which suggests that QA-SRL could be used as a scaffold task for this problem.

Multitask Training Finally, our experiment in chapter 5 raises important questions about the utility of multitask training relative to approaches which use a language modelling objective to pretrain key parts of the network on large amounts of unlabeled data. In light of these results, an important avenue for future work should be investigating which past multitask results will continue to hold once deep contextualized word representations like ELMo are incorporated into the model.

Bibliography

Omri Abend and Ari Rappoport. 2013. Universal conceptual cognitive annotation (ucca). In *Proceedings of the ACL*, volume 1, pages 228–238.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual Question Answering. In *ICCV*.

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998a. The berkeley framenet project. In *Proceedings of the 17th International Conference on Computational Linguistics*.

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

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. 2012. Developing a large semantically annotated corpus. In *LREC 2012, Eighth International Conference on Language Resources and Evaluation*.

Anders Björkelund, Bernd Bohnet, Love Hafdell, and Pierre Nugues. 2010. A high-performance syntactic and semantic dependency parser. In *Proceedings of ICCL: Demonstrations*.

Claire Bonial, Olga Babko-Malaya, Jinho D Choi, Jena Hwang, and Martha Palmer. 2010. Propbank annotation guidelines. *Center for Computational Language and Edu-cation Research Institute of Cognitive Science Uni-versity of Colorad at Boulder*.

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

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

Xavier Carreras and Lluís Màrquez. 2005a. Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the ninth conference on computational natural language learning*, pages 152–164. Association for Computational Linguistics.

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

Rich Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *ICML*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

Danqi Chen and Christopher D Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of EMNLP*.

Ronan Collobert and Jason Weston. 2007a. Fast semantic extraction using a novel neural network architecture. In *Proceedings of ACL*.

Ronan Collobert and Jason Weston. 2007b. Fast semantic extraction using a novel neural network architecture. In *ACL 2007*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of ICML*, pages 160–167. ACM.

Marina Danilevsky, Yunyao Li, Frederick Reiss, Huaiyu Zhu, et al. 2018. Systemt: Declarative text understanding for enterprise. In *Proceedings of the NAACL, Industry Track*.

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

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

Trinh-Minh-Tri Do and Thierry Artières. 2010. Neural conditional random fields. In *Proceedings of AISTATS*.

Manjuan Duan, Ethan Hill, and Michael White. 2016. Generating disambiguating paraphrases for structurally ambiguous sentences. In *Proceedings of the 10th Linguistic Annotation Workshop*.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 12.

Greg Durrett and Dan Klein. 2015. Neural CRF parsing. In *Proceedings of ACL*.

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

Parvin Sadat Feizabadi and Sebastian Padó. 2014. Crowdsourcing annotation of non-local semantic roles. In *Proceedings of EACL*, pages 226–230.

Nicholas FitzGerald, Julian Michael, Luheng He, and Luke Zettlemoyer. 2018. Large-scale QA-SRL parsing. In *Proceedings of ACL*.

Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In *Proceedings of EMNLP*.

Marco Fossati, Claudio Giuliano, and Sara Tonelli. 2013. Outsourcing framenet to the crowd. In *Proceedings of ACL*, volume 2, pages 742–747.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *NIPS*, pages 1019–1027.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. Allennlp: A deep semantic natural language processing platform.

Kevin Gimpel and Noah A Smith. 2010. Softmax-margin crfs: Training log-linear models with cost functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics.

Ross Girshick. 2015. Fast r-cnn. In *Proceedings of ICCV*, pages 1440–1448.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, et al. 2009a. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–18. Association for Computational Linguistics.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, et al. 2009b. The CoNLL-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL*.

Luheng He, Kenton Lee, Omer Levy, and Luke Zettlemoyer. 2018. Jointly predicting predicates and arguments in neural semantic role labeling. In *ACL 2018*.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and whats next. In *Proceedings of ACL*.

Luheng He, Mike Lewis, and Luke S. Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of EMNLP*.

Luheng He, Julian Michael, Mike Lewis, and Luke Zettlemoyer. 2016. Human-in-the-loop parsing. In *Proceedings of EMNLP 2016*.

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

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2017. A transition-based directed acyclic graph parser for ucca. *arXiv preprint arXiv:1704.00552*.

Geoffrey E. Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *CVPR*.

Chloé Kiddon, Ganesa Thandavam Ponnuraj, Luke Zettlemoyer, and Yejin Choi. 2015. Mise en place: Unsupervised interpretation of instructional recipes. In *Proceedings of EACL*, pages 982–992.

Meghana Kshirsagar, Sam Thomson, Nathan Schneider, Jaime Carbonell, Noah A. Smith, and Chris Dyer. 2015. Frame-semantic role labeling with heterogeneous annotations. In *Proceedings of ACL*.

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

Kenton Lee, Yoav Artzi, Yejin Choi, and Luke Zettlemoyer. 2015. Event detection and factuality assessment with non-expert supervision. In *Proceedings of EMNLP*.

Kenton Lee, Luheng He, Mike Lewis, and Luke S. Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of EMNLP*.

Kenton Lee, Shimi Salant, Tom Kwiatkowski, Ankur Parikh, Dipanjan Das, and Jonathan Berant. 2016. Learning recurrent span representations for extractive question answering. *arXiv preprint arXiv:1611.01436*.

Tao Lei, Yuan Zhang, Lluís Márquez, Alessandro Moschitti, and Regina Barzilay. 2015. High-order low-rank tensors for semantic role labeling. In *Proceedings of NAACL*.

Mike Lewis, Luheng He, and Luke Zettlemoyer. 2015. Joint A* CCG parsing and semantic role labelling. In *Proceedings of EMNLP*.

Ding Liu and Daniel Gildea. 2010. Semantic role features for machine translation. In *Proceedings of the ICCL*.

Samuel Louvan, Chetan Naik, Sadhana Kumaravel, Heeyoung Kwon, Niranjan Balasubramanian, and Peter Clark. 2016. Cross sentence inference for process knowledge. In *Proceedings of EMNLP*, pages 1442–1451.

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(85).

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of ACL: System Demonstrations*.

Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The nombank project: An interim report. In *HLT-NAACL 2004 workshop: Frontiers in corpus annotation*.

Julian Michael, Gabriel Stanovsky, Luheng He, Ido Dagan, and Luke Zettlemoyer. 2017. Crowdsourcing question-answer meaning representations. *CoRR*, abs/1711.05885.

Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of ICML*.

Behrang Mohit, Nathan Schneider, Rishav Bhowmick, Kemal Oflazer, and Noah A Smith. 2012. Recall-oriented learning of named entities in arabic wikipedia. In *Proceedings of EACL*. Association for Computational Linguistics.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of ICML*.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajic, and Zdenka Uresova. 2015. Semeval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of SemEval*.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005a. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*.

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

Ellie Pavlick, Travis Wolfe, Pushpendre Rastogi, Chris Callison-Burch, Mark Dredze, and Benjamin Van Durme. 2015. Framenet+: Fast paraphrastic tripling of framenet. In *Proceedings of ACL*, volume 2, pages 408–413.

Hao Peng, Sam Thomson, Swabha Swayamdipta, and Noah A Smith. 2018. Learning joint semantic parsers from disjoint data. *Proceedings of NAACL*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of EMNLP*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

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

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

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

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Drew Reisinger, Rachel Rudinger, Francis Ferraro, Craig Harman, Kyle Rawlins, and Benjamin Van Durme. 2015. Semantic proto-roles. *Transactions of the Association for Computational Linguistics*, 3:475–488.

Matthew Richardson, Christopher JC Burges, and Erin Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of EMNLP*, pages 193–203.

Michael Roth and Mirella Lapata. 2015. Efficient inference and structured learning for semantic role labeling. *Transactions of the Association for Computational Linguistics*, 3:449–460.

Michael Roth and Kristian Woodsend. 2014. Composition of word representations improves semantic role labelling. In *Proceedings of EMNLP*.

Rachel Rudinger, Aaron Steven White, and Benjamin Van Durme. 2018. Neural models of factuality. *arXiv preprint arXiv:1804.02472*.

Josef Ruppenhofer, Michael Ellsworth, Miriam RL Petruck, Christopher R Johnson, and Jan Scheffczyk. 2016. *FrameNet II: Extended theory and practice*. Institut für Deutsche Sprache, Bibliothek.

Andrew M Saxe, James L McClelland, and Surya Ganguli. 2013. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*.

Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of EMNLP*.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of ACL*.

Vivek Srikumar and Christopher D Manning. 2014. Learning distributed representations for structured output prediction. In *Proceedings of NIPS*.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *NIPS*, pages 2377–2385.

Gabriel Stanovsky and Ido Dagan. 2016. Creating a large benchmark for open information extraction. In *Proceedings of EMNLP*.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of NAACL*.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008a. 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. Association for Computational Linguistics.

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

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

Swabha Swayamdipta, Sam Thomson, Chris Dyer, and Noah A Smith. 2017. Frame-semantic parsing with softmax-margin segmental rnns and a syntactic scaffold. *arXiv preprint arXiv:1706.09528*.

Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Efficient inference and structured learning for semantic role labeling. *Transactions of the Association for Computational Linguistics*, 3:29–41.

Adam R Teichert, Adam Poliak, Benjamin Van Durme, and Matthew R Gormley. 2017. Semantic proto-role labeling. In *AAAI*.

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

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

Chenguang Wang, Alan Akbik, laura chiticariu, Yunyao Li, Fei Xia, and Anbang Xu. 2017. Crowd-in-the-loop: A hybrid approach for annotating semantic roles. In *Proceedings of EMNLP*.

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

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

Aaron Steven White, Kyle Rawlins, and Benjamin Van Durme. 2017. The semantic proto-role linking model. In *Proceedings of the EACL*, volume 2, pages 92–98.

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

Bishan Yang and Tom Mitchell. 2017. A joint sequential and relational model for frame-semantic parsing. In *Proceedings of EMNLP*, pages 1247–1256.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

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

Hai Zhao, Wenliang Chen, Chunyu Kity, and Guodong Zhou. 2009. Multilingual dependency learning: A huge feature engineering method to semantic dependency parsing. In *Proceedings of CoNLL*.

Jie Zhou and Wei Xu. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of ACL*.