Semantic Role Labeling with Iterative Structure Refinement

Chunchuan Lyu[1] **Shay B. Cohen**[1] **Ivan Titov**[1,2]
[1]ILCC, School of Informatics, University of Edinburgh
[2]ILLC, University of Amsterdam
chunchuan.lv@gmail.com scohen@inf.ed.ac.uk ititov@inf.ed.ac.uk

Abstract

Modern state-of-the-art Semantic Role Labeling (SRL) methods rely on expressive sentence encoders (e.g., multi-layer LSTMs) but tend to model only local (if any) interactions between individual argument labeling decisions. This contrasts with earlier work and also with the intuition that the labels of individual arguments are strongly interdependent. We model interactions between argument labeling decisions through *iterative refinement*. Starting with an output produced by a factorized model, we iteratively refine it using a refinement network. Instead of modeling arbitrary interactions among roles and words, we encode prior knowledge about the SRL problem by designing a restricted network architecture capturing non-local interactions. This modeling choice prevents overfitting and results in an effective model, outperforming strong factorized baseline models on all 7 CoNLL-2009 languages, and achieving state-of-the-art results on 5 of them, including English.

1 Introduction

Semantic role labeling (SRL), originally introduced by Gildea and Jurafsky (2000), involves the prediction of predicate-argument structure, i.e., identification of arguments and their assignment to underlying *semantic roles*. Semantic-role representations have been shown to be beneficial in many NLP applications, including question answering (Shen and Lapata, 2007), information extraction (Christensen et al., 2011) and machine translation (Marcheggiani et al., 2018). In this work, we focus on dependency-based SRL (Hajič et al., 2009), a popular version of the task which involves identifying syntactic heads of arguments rather than marking entire argument spans (see the graph in red in Figure 1). Edges in the dependency graphs are annotated with semantic roles (e.g., A0:PLEASER) and the predicates are labeled with their senses from a given sense inventory (e.g., SATISFY.01 in the example).

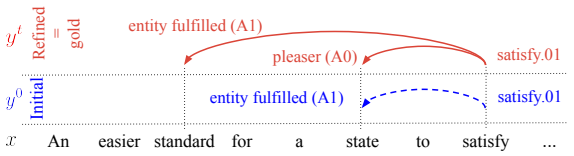

Figure 1: An example of structured refinement, the sentence fragment is from CoNLL-2009: the initial prediction by the factorized model in blue, the refined one (identical to the gold standard) in red.

Before the rise of deep learning methods, the most accurate SRL methods relied on modeling high-order interactions in the output space (e.g., between arguments or arguments and predicates) (Watanabe et al., 2010; Toutanova et al., 2008). Earlier neural methods can model such output interactions through a transition system, and achieve competitive performance (Henderson et al., 2013). However, current state-of-the-art SRL systems use powerful sentence encoders (e.g., layers of LSTMs (Li et al., 2018; He et al., 2017) or multi-head self-attention (Strubell et al., 2018)) and factorize over small fragments of the predicted structures. Specifically, most modern models process individual arguments and perform predicate disambiguation independently. The trend towards more factorizable models is not unique to dependency-based SRL but common for most structured prediction tasks in NLP (Kiperwasser and Goldberg, 2016; Dozat and Manning, 2017, 2018). The only major exception is language generation tasks, especially machine translation and language modeling, where larger amounts of text are typically used in training.

Powerful encoders, in principle, can capture long-distance dependencies and hence alleviate the need for modeling high-order interactions in

the output. However, capturing these interactions in the encoder would require substantial amounts of data. Even if we have domain knowledge about likely interactions between components of the predicted graphs, it is hard to inject this knowledge in an encoder.

Consider the example in Figure 1. The argument *'state'* appears in the highly ambiguous syntactic position '[.] *to satisfy*'. All three core semantic roles of the predicate SATISFY.01 can in principle appear here: patient (A1:ENTITY_FULFILLED, as in '*a sweet tooth to satisfy'*), instrument (A2:METHOD, as in '*a little dessert to satisfy your sweet tooth*') and agent (A0: PLEASER, as in our actual example). The basic factorized model got it wrong, assigning A1 to the argument *'state'*. However, taking into account other arguments, the model can correct the label. The configuration 'A1 *to satisfy*' is more likely when an agent (A0) is present in the sentence. The lack of an agent boosts the score for the correct configuration 'A0 *to satisfy*'.

Our *iterative refinement* approach encodes the above intuition. In iterative refinement (Lee et al., 2018), a refinement network repeatedly takes previous output as input and produces its refined version. Formally, we have

$$y^{t+1} = \text{Refine}(x, y^t).$$

Naturally, such refinement strategy also requires an initial prediction y^0, which is produced by a ('base') factorized model.

Refinement strategies have been successful in machine translation (Lee et al., 2018; Novak et al., 2017; Xia et al., 2017; Hassan et al., 2018), but their effectiveness in other NLP tasks is yet to be demonstrated.[1] We conjecture that this discrepancy is due to differences in data availability. Given larger amounts of training data typically used in machine translation, their base models and refinement networks overfit to a lesser extent. Overfitting in (1) the base model and (2) the refinement network are both problematic. The first implies that either there are no mistakes in the base models in the training set or their distribution is very different from that in the test regime, so the training material for the inference networks ends up being misleading. The second naturally means that refinement will fail at test time. We address both these issues by designing restricted inference networks and adding a specific form of noise when training them.

Our structured refinement network is simple but encodes non-local dependencies. Specifically, it takes into account the information about the role distributions on the previous iteration aggregated over the entire sentences but not the information what the other arguments are. It is a coarse compressed representation of the prediction, yet it represents long-distance information not readily available within the factorized base model. While this is not the only possible design, we believe that the empirical gains from using this simple refinement network, demonstrate the viability of our general framework of iterative refinement with restricted inference networks. They also suggest that intuitions underlying declarative constraints used in early work on SRL (Punyakanok et al., 2008; Das et al., 2012) can be revisited but now encoded in a flexible soft way to provide induction biases for the refinement networks. We leave this for future work.

We consider the CoNLL-2009 dataset (Hajič et al., 2009). We start with a strong factorized baseline model, which already achieves state-of-the-art results on a subset of the languages. Then, using our structure refinement network, we improve on this baseline on all 7 CoNLL-2009 languages. The model achieves best-reported results in 5 languages, including English. We also observe improvements on out-of-domain test sets, confirming the robustness of our approach. We perform experiments demonstrating the importance of adding noise, and ablation studies showing the necessity of incorporating output interactions. Furthermore, we provide analysis on constraint violations and errors on the English test set.[2]

2 Related Work

Learning to refine predictions from neural structured prediction models has recently received significant attention. Our approach bears similarity to methods used in machine translation (Lee et al., 2018; Novak et al., 2017; Xia et al., 2017). All these methods refine a translated sentence produced by a seq2seq model with another seq2seq model. Among them, the deliberation networks

[1] See extra discussion and related work in section 2.

[2] The code and experiment settings can be accessed at https://github.com/ChunchuanLv/Iterative_Inference

by Xia et al. (2017) rely on BiLSTMs and improve initial predictions from an competitive baseline and obtain state-of-art-results on English-to-French translation. Later, it has been shown that the deliberation networks can improve translation when used within the Transformer framework (Hassan et al., 2018).

Certain approaches, not necessarily directly optimized for refinement, can nevertheless be regarded as iterative refinement methods. Structured prediction energy networks (SPENs) are trained to assign global energy scores to output structures, and the gradient descent is used during inference to minimize the global energy (Belanger and McCallum, 2016). As the gradient descent involves iterative optimization, its steps can be viewed as iterative refinement. In particular, Belanger et al. (2017) build a SPEN for SRL, but for the span-based formalism, not the dependency one we consider in this work. While they improve over their baseline model, their baseline model used multi-layer perceptron to encode local factors, thus the encoder power is limited. Moreover their refined model performs worse in the out-of-domain setting than their baseline model, indicating overfitting (Belanger et al., 2017).

In the follow-up work, Tu and Gimpel (2018, 2019) introduce inference networks to replace gradient descent. Their inference networks directly refine the output. Improvements over competitive baselines are reported on part-of-speech tagging, named entity recognition and CCG supertagging (Tu and Gimpel, 2019). However, their inference networks are distilling knowledge from a tractable linear-chain conditional random field (CRF) model. Thus, these methods do not provide direct performance gains. More importantly, the interactions captured in these models are likely local, as they learn to mimic Markov CRFs.

Denoising autoencoders (Vincent et al., 2008) can also be used to refine structure. Indeed, image segmentation can be improved through iterative inference with denoising autoencoders (Romero et al., 2017; Drozdzal et al., 2018). Their framework is very similar to ours, albeit we are working in a discrete domain. One other difference is that by using a convolutional architecture in the refinement network, they are still modeling only local interactions. At a more conceptual level, Bengio et al. (2013) argued that a denoising autoencoder should not be too robust to the input variations as to ignore the input. This indicates that we should not expect refinement networks to correct all the errors, even in theory, and hence, the refinement networks do not need to be particularly powerful.

Very recently, Wang et al. (2019) used high order statistical model for Semantic Dependency Parsing (Oepen et al., 2015), and obtain improvements over strong baseline using BiLSTM. They attempted loopy belief propagation and mean field variational inference for inference, and train the model end to end. Such inference steps are well motivated. This work is similar to energy network approach (Belanger and McCallum, 2016), while a global score function is provided, and approximate inference steps are used. Comparing to ours, the inference can also be regarded as iterative structure refinement. Yet, we do not provide a global score and directly try to model the refinement. In principle, our formalization should give us more liberty in terms of designing the refinement network.

3 Dependency Semantic Role Labeling

In this section, we introduce the notation and present our factorized baseline model.

3.1 Notation

In dependency SRL, for each sentence of length n, we have a sequence of words w, dependency labels dep, part-of-speech tags pos, each being a discrete sequence of length n. To simplify notation, we consider one predicate at a time. We denote the number of roles by r, it includes the 'null' role, signifying that the corresponding word is not an argument of the predicate. Formally, $\mathrm{P} \in \Delta_{m-1}$ is the probability distribution over m predicate senses, and Δ_{m-1} represents the corresponding probability simplex. We also have predicate sense embeddings $\Pi \in \mathbb{R}^{m \times d_\pi}$, and index j, throughout the discussion, refers to the position of the target predicate in the sentence. $\mathrm{R} \in \Delta_{r-1}^{n}$ is a matrix of size $n \times r$ such that each row sums to 1, corresponding to a probability distribution over roles. In particular $\mathrm{R}_{i,0}$ is the probability of i-th word not being an argument of the predicate.

We index role label and sense predictions from different refinement iterations ('time steps') with t, i.e. P^t and R^t. The index t ranges from 0 to T, and P^0 and R^0 denotes the predictions from the factorized baseline model. Details (e.g., hyperparameters) are provided in the appendix.

3.2 Factorized Model

Similarly to recent approaches to SRL and semantic graph parsing (He et al., 2017; Li et al., 2018; Dozat and Manning, 2018), our factorized baseline model starts with concatenated embeddings x. Then, we encode the sentence with a BiLSTM, further extract features with an MLP (multilayer perceptron) and apply a bi-affine classifier to the resulting features to label the words with roles. We also use a predicate-dependent linear layer for sense disambiguation.

More formally, we start with getting a sentence representation by concatenating embeddings. We have $x^{\mathrm{w}} \in \mathbb{R}^{n \times d_w}$, $x^{\mathrm{dep}} \in \mathbb{R}^{n \times d_\delta}$, $x^{\mathrm{pos}} \in \mathbb{R}^{n \times d_p}$ for words, dependency labels and part-of-speech tags, respectively. We concatenate them to form a sentence representation:

$$x = x^{\mathrm{w}} \circ x^{\mathrm{dep}} \circ x^{\mathrm{pos}} \in \mathbb{R}^{n \times d_x} \tag{1}$$

We further encode the sentence with a BiLSTM:

$$\mathrm{h} = \textbf{BiLSTM}(x) \in \mathbb{R}^{n \times d_h} \tag{2}$$

From these context-aware word representations, we produce features for argument identification and role labeling that will be used by a bi-affine classifier. Note that, for every potential predicate-argument dependency (i.e. a candidate edge), we need to produce representations of both endpoints: the argument and the predicate 'sides'. For the argument side, h^{ρ_0} will be used to compute the logits for argument identification and h^{ρ_1} will be used for deciding on its role:

$$\mathrm{h}^{\rho_0} = \textbf{MLP}(\mathrm{h}) \in \mathbb{R}^{n \times d_{\rho_0}} \tag{3}$$

$$\mathrm{h}^{\rho_1} = \textbf{MLP}(\mathrm{h}) \in \mathbb{R}^{n \times d_{\rho_1}} \tag{4}$$

Similarly, for the predicate side, we also extract two representations h^{ϱ}_0 and h^{ϱ_1} (recall that the predicate is at position j):

$$\mathrm{h}^{\varrho_0} = \textbf{MLP}(\mathrm{h}_j) \in \mathbb{R}^{d_{\rho_0}} \tag{5}$$

$$\mathrm{h}^{\varrho_1} = \textbf{MLP}(\mathrm{h}_j) \in \mathbb{R}^{d_{\rho_1}} \tag{6}$$

We then obtain logits I^{ρ_0} corresponding to the decision to label arguments as *null*, and logits I^{ρ_1} for other roles. So, we have:

$$\mathrm{I}^{\rho_0} = \textbf{BiAffine}(\mathrm{h}^{\varrho_0}, \mathrm{h}^{\rho_0}) \in \mathbb{R}^{n} \tag{7}$$

$$\mathrm{I}^{\rho_1} = \textbf{BiAffine}(\mathrm{h}^{\varrho_1}, \mathrm{h}^{\rho_1}) \in \mathbb{R}^{n \times (r-1)} \tag{8}$$

Unlike Dozat and Manning (2018), where argument identification and role labeling are trained with two losses,[3] we feed them together into a single softmax layer to compute the semantic-role distribution R^0:

$$\mathrm{I}^{\alpha} = \mathrm{I}^{\rho_0} \circ \mathrm{I}^{\rho_1} \in \mathbb{R}^{n \times r} \tag{9}$$

$$\mathrm{R}^0 = \mathrm{Softmax}(\mathrm{I}^{\alpha}) \in \Delta^{n}_{r-1} \tag{10}$$

Now, for sense disambiguation, we need to extract yet another predicate representation h^{π}:

$$\mathrm{h}^{\pi} = \textbf{MLP}(\mathrm{h}_j) \in \mathbb{R}^{d_\pi} \tag{11}$$

In the formalism we use (PropBank), senses are predicate-specific, so we use predicate-specific sense embedding matrices Π. The matrix Π acts as a linear layer before softmax:

$$\mathrm{I}^{\pi} = \Pi \cdot \mathrm{h}^{\pi} \in \mathbb{R}^{m} \tag{12}$$

$$\mathrm{P}^0 = \mathrm{Softmax}(\mathrm{I}^{\pi}) \in \Delta_{m-1} \tag{13}$$

This ends the description of our baseline model, which we also use to get initial predictions for iterative refinement.

4 Structured Refinement Network

In this section, we introduce the structured refinement network for dependency SRL. When doing refinement, it has access to the roles distribution $\mathrm{R}^t \in \Delta^{n}_{r-1}$ and the sense distribution $\mathrm{P}^t \in \Delta_{m-1}$ computed at the previous iteration (i.e. time t). In addition, it exploits the sentence representation $x \in \mathbb{R}^{n \times d_x}$. Our refinement network is limited and structured, in the sense that it only has access to a compressed version of the previous prediction, and the network itself is a simple MLP.

Similarly to our baseline model, we extract feature vectors g from input x and further separately encode the argument representation g^{α} and the predicate token representation g^{π}:

$$\mathrm{g} = \textbf{BiLSTM}(x) \in \mathbb{R}^{n \times d_h} \tag{14}$$

$$\mathrm{g}^{\alpha} = \textbf{MLP}(\mathrm{g}) \in \mathbb{R}^{n \times d_g} \tag{15}$$

$$\mathrm{g}^{\pi} = \textbf{MLP}(\mathrm{g}_j) \in \mathbb{R}^{d_g} \tag{16}$$

To simplify the notation, we omit indexing them by t, except for R^t and P^t. We use two refinement networks, one for roles and another one for predicate senses.

[3] The separate processing of I^{ρ_0} and I^{ρ_1} rather than using a single MLP for all roles, including *null*, results in extra representation power allocated for the argument identification subtask.

4.1 Role Refinement Network

First, we describe our structured refinement network for role labeling. We use i to index arguments. We obtain a compressed representation o_i used for refining R_i^t by summing up the probability mass for all roles, excluding the null role:

$$\mathrm{o}_{i,u} = \sum_{k \neq i} \mathrm{R}_{k,u}^t \in \mathbb{R} \quad (17)$$

$$\mathrm{o}_i = [\mathrm{o}_{i,u}] \in \mathbb{R}^{r-1} \quad (18)$$

Intuitively, o_i is the aggregation of all other roles being labeled by the current predicate. We concatenate o_i with feature vectors of the current argument g^α, predicate g^π, the relaxed predicate sense embedding $\Pi^\intercal \cdot \mathrm{P}^t$ and the role probability itself (R_i) to form the input to a two-layer network:

$$\mathrm{z}_i^\alpha = \mathrm{R}_i^t \circ \mathrm{o}_i \circ \mathrm{g}_i^\alpha \circ \mathrm{g}^\pi \circ (\Pi^\intercal \cdot \mathrm{P}^t) \quad (19)$$

$$\mathrm{z}_i^\alpha \in \mathbb{R}^{2r-1+2d_g+d_\pi}$$

$$\mathrm{M}_i^\alpha = W^\alpha \cdot \sigma(W_\alpha \cdot \mathrm{z}_i^\alpha) \in \mathbb{R}^r \quad (20)$$

$$\mathrm{M}^\alpha = [\mathrm{M}_{i,u}^\alpha] \in \mathbb{R}^{n \times r}, \quad (21)$$

where σ is the logistic sigmoid function, $W_\alpha \in \mathbb{R}^{d_r \times (2r-1+2d_g+d_\pi)}$, $W^\alpha \in \mathbb{R}^{r \times d_r}$ are learned linear mappings. We obtain our refined logits M_i^α for the i-th argument; M^α refers to the stacked matrix of logits for all arguments. To obtain the refined role distribution, we add up M^α and I^α that we got from the baseline model, and follow that by a softmax layer:

$$\mathrm{R}^{t+1} = \mathrm{Softmax}(\mathrm{M}^\alpha + \mathrm{I}^\alpha) \in \Delta_{r-1}^n \quad (22)$$

4.2 Sense Refinement Network

To build a representation for sense disambiguation, we simply compute the probability mass for each role (excluding the null role) to obtain r^π, and concatenate this with g^π and $\Pi^\intercal \cdot \mathrm{P}^t$:

$$\mathrm{r}^\pi = \sum_k \mathrm{R}_{k,1:}^t \in \mathbb{R}^{r-1} \quad (23)$$

$$\mathrm{z}^\pi = (\Pi^\intercal \cdot \mathrm{P}^t) \circ \mathrm{r}^\pi \circ \mathrm{g}^\pi \in \mathbb{R}^{r-1+d_g+d_\pi} \quad (24)$$

Differently from the role refinement network, sense prediction is predicate-specific. Therefore, we first map z^π to $\mathbb{R}^{d_\pi}$, and then take the inner product with the predicate-specific sense embeddings Π to get the refined logits:

$$\mathrm{M}^\pi = \Pi \cdot W^\pi \cdot \sigma(W_\pi \cdot \mathrm{z}^\pi) \in \mathbb{R}^m \quad (25)$$

Similarly to role refinement, σ is the logistic function, $W_\pi \in \mathbb{R}^{d_r \times (r-1+d_g+d_\pi)}$, $W^\pi \in \mathbb{R}^{m \times d_r}$ are learned linear mappings. Again, we combine the logits M^π and I^π before the softmax layer:

$$\mathrm{P}^{t+1} = \mathrm{Softmax}(\mathrm{M}^\pi + \mathrm{I}^\pi) \in \Delta_{m-1} \quad (26)$$

4.3 Weight Tying

Our refinement networks are similar to the denoising autoencoders (DAEs; Vincent et al. 2008), so we use the weight-tying technique popular with DAEs. We believe that the technique may be even more effective here as the amount of labeled data for SRL is lower than in many usual applications of DAEs. We tie W_α with a subset of W^α rows: specifically with the rows acting on R_i^t in the computation of M_i^α (see equations 19 and 20). Similarly, we tie W_π with the part of W^π corresponding to $\Pi^\intercal \cdot \mathrm{P}^t$ (see equations 24 and 25). Formally,

$$W^\alpha = W_\alpha^\intercal[: r] \quad (27)$$

$$W^\pi = W_\pi^\intercal[: d_\pi] \quad (28)$$

where $W[: k]$ takes the first k rows of matrix W.

4.4 Self Refinement

We describe a simpler version of the refinement network which we will use in experiments to test whether the improvements with the structured refinement network over the factorized baseline are genuinely coming from modeling interaction between arguments rather than from simply combining multiple classifiers. This simpler refinement network does not account for any interactions between arguments. Instead of equations 19 and 24, we have:

$$\mathrm{z}_i^\alpha = \mathrm{R}_i^t \circ \mathrm{g}_i^\alpha \circ \mathrm{g}^\pi \in \mathbb{R}^{r+2d_g+d_\pi} \quad (29)$$

$$\mathrm{z}^\pi = (\Pi^\intercal \cdot \mathrm{P}^t) \circ \mathrm{g}^\pi \in \mathbb{R}^{d_g+d_\pi} \quad (30)$$

Everything else is kept the same as in the full model, expect that the size of W^α and W^π needs to be adjusted. We refer to this ablated network as the *self-refinement network*.

5 Training for Iterative Structure Refinement

In this section, we describe our training procedure.

5.1 Two-Stage Training

We have two models: the baseline model, producing the initial predictions, and the iterative refinement network, correcting them. While it is possible to train them jointly, we find joint training slow

to converge. Instead, we train the factorized baseline model first and then optimize the refinement networks while keeping the baseline model fixed.

5.2 Stochastic Training

Our baseline model overfits to the training set, and, if simply training on its output, our refinement network would learn to copy the base predictions. Instead, we perturb the baseline prediction during training. Naturally, we can add dropout (Srivastava et al., 2014) and recurrent dropout (Gal and Ghahramani, 2016) to our neural networks. However, for the smaller data set we use, we find this not sufficient. In particular, we use Gumbel-Softmax instead of Softmax. $\text{Gumbel-Softmax}(\mathrm{I}) = \text{Softmax}(\mathrm{I} + \lambda_g \epsilon)$, where the random variable ϵ is drawn from the standard Gumbel distribution (Maddison et al., 2017; Jang et al., 2017), and λ_g is a hyper-parameter controlling decoding stochasticity.[4]

5.3 Loss for Iterative Refinement

Let us denote gold-standard labels for roles and predicates as R^* and P^*. We use two separate losses $\mathcal{L}_{\text{base}}(\mathrm{R}^*, \mathrm{P}^*, x)$ and $\mathcal{L}_{\text{refine}}(\mathrm{R}^*, \mathrm{P}^*, x)$ for our two-stage training. We define losses for predictions from each refinement iteration and sum them up:

$$\mathcal{L}_{\text{base}}(\mathrm{R}^*, \mathrm{P}^*, x) = \mathcal{L}(\mathrm{R}^*, \mathrm{R}^0) + \mathcal{L}(\mathrm{P}^*, \mathrm{P}^0) \tag{31}$$

$$\mathcal{L}_{\text{refine}}(\mathrm{R}^*, \mathrm{P}^*, x) = \sum_{t=1}^{T} \mathcal{L}(\mathrm{R}^*, \mathrm{R}^t) + \mathcal{L}(\mathrm{P}^*, \mathrm{P}^t) \tag{32}$$

We adopt the Softmax-Margin loss (Gimpel and Smith, 2010; Blondel et al., 2019) for individual $\mathcal{L}$. Effectively, we subtract 1 from the logit of the gold label, and apply the cross entropy loss.

6 Experiments

Datasets We conduct experiments on CoNLL-2009 (Hajič et al., 2009) data set for all languages, including Catalan (Ca), Chinese (Zh), Czech (Cz), English (En), German (De), Japanese (Jp) and Spanish (Es). We use the predicted part-of-speech tags, dependency labels, and pre-identified predicate, provided with the dataset. The statistics of datasets are shown in Table 2.

[4] A more canonical way of controlling stochasticity is to use the temperature but we prefer not to scale the gradient.

Hyperparameters We use ELMo (Peters et al., 2018) for English, and FastText embeddings (Bojanowski et al., 2017; Grave et al., 2018) for all other languages. We train and run the refinement networks for two iterations. All other hyperparameters are the same for all languages, except BiLSTMs for English is larger than others.

Training Details Training the refinement network takes roughly 2 times more time than the baseline models, as it requires running BiLSTMs. The extra computation for the structured refinement network is minimal. For English, training the iterative refinement model for 1 epoch takes about 6 minutes on one 1080ti GPU. Adam is used as the optimizer (Kingma and Ba, 2015), with the learning rate of 3e-4. We use early stopping on the development set. We run 600 epochs for all baseline models, and 300 epochs for the refinement networks. Batch sizes are chosen from 32, 64, or 128 to maximize GPU memory usage. Our implementation is based on PyTorch and AllenNLP (Paszke et al., 2017; Gardner et al., 2018).

6.1 Results and Discussions

Test Results Results for all CoNLL-2009 languages on the standard (in-domain) datasets are presented in Table 1. We compare our best model to the best previous single model for the corresponding language (excluding ensemble ones). Most research has focused on English, but we include results of recent models which were evaluated on at least 3 languages. When compared to the previous models, both our models are very competitive, with the exception of German. On the German dataset, Mulcaire et al. (2018) also report a relatively weak result, when compared to Roth and Lapata (2016). The German dataset is the smallest one in terms of the number of predicates. Syntactic information used by Roth and Lapata (2016) may be very beneficial in this setting and may be the reason for this discrepancy. Our structured refinement approach improves over the best previous results on 5 out of 7 languages. Note that hyper-parameters of the refinement network are not tuned for individual languages, suggesting that the proposed method is robust and may be easy to apply to new languages and/or new base models. The only case where the refinement network was not effective is Chinese, where it achieved only a negligible improvement.

Out-of-Domain Results on the out-of-domain

Model	Ca	Cz	De	En	Ja	Es	Zh	Avg.
Roth and Lapata (2016)	-	-	**80.10**	86.7	-	80.20	79.4	-
Marcheggiani et al. (2017)	-	86.00	-	87.7	-	80.30	81.2	-
Mulcaire et al. (2018)*	79.45	85.14	69.97	87.24	76.00	77.32	81.89	79.57
Previous best single model	80.32	86.02	**80.10**	90.40	78.69	80.50	**84.30**	82.90
Baseline model	80.69	87.30	75.06	90.65	81.97	79.87	83.26	82.69
Structured refinement	**80.91**	**87.62**	75.87	**90.99**	**82.54**	**80.53**	83.31	**83.11**

Table 1: Labeled F1 score (including senses) for all languages on the CoNLL-2009 in-domain test set. For previous best result, Catalan is from Zhao et al. (2009), Japanese is from Watanabe et al. (2010), Czech is from Henderson et al. (2013), German and Spanish are from Roth and Lapata (2016), English is from Li et al. (2018) and Chinese is from Cai et al. (2018). We report the best testing results from Mulcaire et al. (2018).

	#sent	#pred	#pred/#sent
Ca	13200	37444	2.84
Cz	38727	414133	10.69
De	36020	17400	0.48
En	39279	179014	4.56
Ja	4393	25712	5.85
Es	14329	43828	3.06
Zh	22277	102827	4.62

Table 2: Number of sentences and predicates in training set of different languages.

testing sets are presented in Table 4.[5] We observe improvements from using refinement in all the cases. This shows that our refinement approach is robust against domain shift.

Ablations We report development set results in different settings in Table 3. Our full model performs 2 refinement iterations, uses weight tying, and the Gumbel noise.[6] We select the best configuration for each language to report the test set performance in Table 1 and Table 4.

As expected, weight tying is beneficial for lower-resource languages such as Catalan, Japanese and Spanish (see Table 2 for dataset characteristics). The Gumbel noise helps for all languages except for Czech and English, the two largest datasets. In particular, we observe almost no improvement on the Spanish dataset without using the Gumbel noise. We note relatively consistent but small gains from using 2 refinement iterations. The magnitude of the gains may be an artifact of us having the loss terms $\mathcal{L}(\mathrm{R}^*, \mathrm{R}^t)$ and $\mathcal{L}(\mathrm{P}^*, \mathrm{P}^t)$, encouraging not only the final (second), but also the first, iteration to produce accurate predictions. A potential alternative explanation is that our refinement network is restricted to simple interactions, resulting in the fixed point reachable in one step.

Constraints Violation We consider violation of unique core roles (U), continuation roles (C) and reference roles (R) constraints from Punyakanok et al. (2008); FitzGerald et al. (2015) in Table 6. U is violated if a core role (A0 - A5, AA) appears more than once; C is violated when the C-X role is not preceded by the X role (for some X); R is violated if R-X role does not appear. Our approach results in a large reduction in the uniqueness constraint violations. Our model slightly reduces the number of R violations, while He et al. (2017) reported that deterministically enforcing constraints is not helpful (albeit in span-based SRL). However learning those constraints in a soft way might be beneficial.

Argument Interaction vs. No Argument Interaction We compare the structured refinement network and the self-refinement network in Table 5. Both networks share the same hyperparameters. The structured refinement network consistently outperforms the self-refinement counterpart. This suggests that the refinement model benefits from accessing information about other arguments when doing refinement. In other words, modeling argument interaction appears genuinely useful.

Improvement Decomposition We report labeled role precision, recall and sense disambiguation accuracy in Table 7. Our structured refinement approach consistently improves over the baseline model in all metrics. While we cannot assert the improvements on all metrics are significant, this suggests that it learns some non-trivial interactions instead of merely learning to balance precision and

[5] Roth and Lapata (2016) has better in-domain testing score, but did not report the out-of-domain score.

[6] We set $\lambda_g^{\alpha} = 5$ for role and $\lambda_g^{\pi} = 50$ for sense, so that initial predictions contain around 20% errors.

Model	Ca	Cz	De	En	Ja	Es	Zh	Avg.
Baseline	81.69	88.43	73.97	89.60	82.96	80.49	85.27	83.20
Full	**82.11**	88.62	74.95	89.82	**83.60**	**81.19**	**85.52**	**83.69**
1 iteration	82.07	88.62	**75.07**	89.93	83.49	81.03	85.47	83.40
un-tied	81.99	88.61	75.04	89.79	83.47	80.89	85.49	83.61
no Gumbel	82.07	**88.71**	74.62	**90.08**	83.33	80.55	85.42	83.54

Table 3: Labeled F1 score (including senses) for all languages on development set for different configurations.

English	Test	Ood
Li et al. (2018)	90.4	81.5
Baseline	90.65	81.98
Structured Refinement	**90.99**	**82.18**
German	Test	Ood
Zhao et al. (2009)	**76.19**	**67.78**
Baseline	75.06	65.25
Structured Refinement	75.87	65.69
Czech	Test	Ood
Marcheggiani et al. (2017)	86.0	**87.2**
Baseline	87.30	85.80
Structured Refinement	**87.62**	86.04

Table 4: Labeled F1 scores (including senses) on English, German, Czech in-domain and out-of-domain test sets; we chose the previous models achieving the best scores on the out-of-domain test sets.

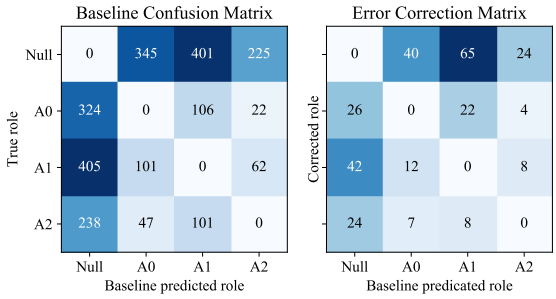

Figure 2: Confusion matrix for the baseline model, and a correction matrix where the errors were corrected by the refinement network. Only Null, A0, A1, A2 are presented here.

recall.

Error Correction Analysis We show the errors that the structured refinement network corrects in Figure 2. In the baseline confusion matrix, we see the errors are fairly balanced for all the roles we consider here. In the error correction matrix, the corrections are also fairly evenly distributed. Yet, this is not completely uniform. There is a tendency towards filtering out arguments rather than generating new ones.

7 Conclusions and Future Work

We propose the structured refinement network for dependency semantic role labeling. The structured refinement network corrects predictions made by a strong factorized baseline model while modeling interactions in the predicated structure. The resulting model achieves state-of-the-art results on 5 out of 7 languages in the CoNLL-2009 data set, and substantially outperforms the factorized model on all of these languages.

For the future work, the structured refinement network can be further improved. For example, we can take an inspiration from either declarative constraints used in the previous work (Punyakanok et al., 2008) or from literature on lexical semantics of verbs, studying patterns of event and argument realization (e.g., Levin 1993). Indeed, the unique role constraint as a declarative constraint is one of the motivation for the concurrent work on modeling argument interaction in SRL (Chen et al., 2019). That work relies on capsule networks (Sabour et al., 2017) and focuses primarily on enforcing the role uniqueness constraint.

The framework can be extended to other tasks. For example, in syntactic dependency parsing: the refinement network can rely on representations of grandparent nodes, siblings and children to propose a correction. In general, structure refinement networks should allow domain experts to incorporate prior knowledge about output dependencies and improve model performance.

Acknowledgments

We thank the anonymous reviewers for their suggestions. The project was supported by the European Research Council (ERC StG BroadSem

Model	Ca	Cz	De	En	Ja	Es	Zh	Avg.
Baseline Model	80.69	87.30	75.06	90.65	81.97	79.87	83.26	82.69
Self Refinement	80.65	87.32	74.83	90.71	82.27	80.08	**83.32**	82.74
Structured Refinement	**80.91**	**87.47**	**75.83**	**90.83**	**82.54**	**80.53**	83.31	**83.06**

Table 5: Labeled F1 score (including senses) for all languages on the CoNLL-2009 in-domain test set.

Model	U	C	R
Gold	55	0	88
Baseline	301	2	114
Structured Refinement	142	2	111

Table 6: Unique core roles violations (U), continuation roles violations (C) and reference roles violations (R) on English in-domain test set.

Model	RP	RR	Sense
Baseline	88.1	88.3	96.2
Structured Refinement	88.7	88.5	96.3

Table 7: Labeled roles precision (RP), recall (RR) and sense disambiguation accuracy (Sense) on English in-domain test set.

678254), the Dutch National Science Foundation (NWO VIDI 639.022.518) and Bloomberg L.P.

References

David Belanger and Andrew McCallum. 2016. Structured prediction energy networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pages 983–992. JMLR.org.

David Belanger, Bishan Yang, and Andrew McCallum. 2017. End-to-end learning for structured prediction energy networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 429–439. JMLR.org.

Yoshua Bengio, Li Yao, Guillaume Alain, and Pascal Vincent. 2013. Generalized denoising auto-encoders as generative models. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'13, pages 899–907, USA. Curran Associates Inc.

Mathieu Blondel, André FT Martins, and Vlad Niculae. 2019. Learning with fenchel-young losses. *arXiv preprint arXiv:1901.02324*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Jiaxun Cai, Shexia He, Zuchao Li, and Hai Zhao. 2018. A full end-to-end semantic role labeler, syntactic-agnostic over syntactic-aware? In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2753–2765, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Xinchi Chen, Chunchuan Lyu, and Ivan Titov. 2019. Capturing argument interaction in semantic role labeling with capsule networks. In *EMNLP*.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2011. An analysis of open information extraction based on semantic role labeling. In *Proceedings of the Sixth International Conference on Knowledge Capture*, K-CAP '11, pages 113–120, New York, NY, USA. ACM.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. 2016. Fast and accurate deep network learning by exponential linear units (elus). In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Dipanjan Das, André FT Martins, and Noah A Smith. 2012. An exact dual decomposition algorithm for shallow semantic parsing with constraints. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 209–217. Association for Computational Linguistics.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Timothy Dozat and Christopher D. Manning. 2018. Simpler but more accurate semantic dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 484–490, Melbourne, Australia. Association for Computational Linguistics.

Michal Drozdzal, Gabriel Chartrand, Eugene Vorontsov, Mahsa Shakeri, Lisa Di Jorio, An Tang, Adriana Romero, Yoshua Bengio, Chris Pal, and Samuel Kadoury. 2018. Learning normalized inputs for iterative estimation in medical image segmentation. *Medical Image Analysis*, 44:1 – 13.

Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 960–970, Lisbon, Portugal. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 1027–1035, USA. Curran Associates Inc.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew E. Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2018. Allennlp: A deep semantic natural language processing platform. *CoRR*, abs/1803.07640.

Daniel Gildea and Daniel Jurafsky. 2000. Automatic labeling of semantic roles. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 512–520, Hong Kong. Association for Computational Linguistics.

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*, pages 733–736, Los Angeles, California. Association for Computational Linguistics.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the 11th Language Resources and Evaluation Conference*, Miyazaki, Japan. European Language Resource Association.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*, CoNLL '09, pages 1–18, Stroudsburg, PA, USA. Association for Computational Linguistics.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, Shujie Liu, Tie-Yan Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou. 2018. Achieving human parity on automatic chinese to english news translation. *CoRR*, abs/1803.05567.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483. Association for Computational Linguistics.

James Henderson, Paola Merlo, Ivan Titov, and Gabriele Musillo. 2013. Multilingual joint parsing of syntactic and semantic dependencies with a latent variable model. *Computational Linguistics*, 39(4):949–998.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Eliyahu Kiperwasser and Yoav Goldberg. 2016. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1173–1182, Brussels, Belgium. Association for Computational Linguistics.

Beth Levin. 1993. *English verb classes and alternations: A preliminary investigation*. University of Chicago press.

Zuchao Li, Shexia He, Hai Zhao, Yiqing Zhang, Zhuosheng Zhang, Xi Zhou, and Xiang Zhou. 2018. Dependency or span, end-to-end uniform semantic role labeling. *CoRR*, abs/1901.05280.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Diego Marcheggiani, Joost Bastings, and Ivan Titov. 2018. Exploiting semantics in neural machine translation with graph convolutional networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 486–492, New Orleans, Louisiana. Association for Computational Linguistics.

Diego Marcheggiani, Anton Frolov, and Ivan Titov. 2017. A simple and accurate syntax-agnostic neural model for dependency-based semantic role labeling. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 411–420, Vancouver, Canada. Association for Computational Linguistics.

Phoebe Mulcaire, Swabha Swayamdipta, and Noah A. Smith. 2018. Polyglot semantic role labeling. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 667–672, Melbourne, Australia. Association for Computational Linguistics.

Roman Novak, Michael Auli, and David Grangier. 2017. Iterative refinement for machine translation. *CoRR*, abs/1610.06602.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, and Zdeňka Urešová. 2015. SemEval 2015 task 18: Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 915–926, Denver, Colorado. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

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

Adriana Romero, Michal Drozdzal, Akram Erraqabi, Simon Jégou, and Yoshua Bengio. 2017. Image segmentation by iterative inference from conditional score estimation. *CoRR*, abs/1705.07450.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1192–1202, Berlin, Germany. Association for Computational Linguistics.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. In *Advances in neural information processing systems*, pages 3856–3866.

Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*, pages 12–21.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2377–2385.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium. Association for Computational Linguistics.

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

Lifu Tu and Kevin Gimpel. 2018. Learning approximate inference networks for structured prediction. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*.

Lifu Tu and Kevin Gimpel. 2019. Benchmarking approximate inference methods for neural structured prediction. *CoRR*, abs/1904.01138.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, pages 1096–1103.

Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019. Second-order semantic dependency parsing with end-to-end neural networks. In *ACL*.

Yotaro Watanabe, Masayuki Asahara, and Yuji Matsumoto. 2010. A structured model for joint learning of argument roles and predicate senses. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 98–102, Uppsala, Sweden. Association for Computational Linguistics.

Yingce Xia, Fei Tian, Lijun Wu, Jianxin Lin, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2017. Deliberation networks: Sequence generation beyond one-pass decoding. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 1782–1792.

Hai Zhao, Wenliang Chen, Jun'ichi Kazama, Kiyotaka Uchimoto, and Kentaro Torisawa. 2009. Multilingual dependency learning: Exploiting rich features for tagging syntactic and semantic dependencies. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL 2009): Shared Task*, pages 61–66, Boulder, Colorado. Association for Computational Linguistics.