References

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

Diederik Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

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

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Andrew M Saxe, James L McClelland, and Surya Ganguli. 2014. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint*.

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

A Supplemental Material

A.1 Hyperparameters

Representation sizes The word embeddings are fixed 300-dimensional GloVe embeddings (Pennington et al., 2014) (context window size of 2 for head word embeddings, and window size of 10 for LSTM inputs), normalized to be unit vectors. Out-of-vocabulary words are represented by a vector of zeros. In the character CNN, characters are represented as learned 8-dimensional embeddings. The convolutions have window sizes of 3, 4, and 5 characters, each consisting of 50 filters.

Network sizes We use 3 stacked bidirectional LSTMs with highway connections and 200 dimensional hidden states. Each MLP consists of two hidden layers with 150 dimensions and rectified linear units (Nair and Hinton, 2010).

Inference We model spans up to length 30. We use $\lambda_a = 0.8$ for pruning arguments, $\lambda_p = 0.4$ for pruning predicates. At decoding time, we use dynamic programming (a simplified version of Täckström et al. (2015)) to predict a set of non-overlapping arguments for each predicate [1].

[1]This is mainly a constraint enforced by the official CoNLL evaluation script.

	CoNLL 2012			OntoNotes5	
	Train	Dev	Test	Train	Dev
Docs	2.8	0.3	0.3	11	1.5
Sentences	75	9.6	9.5	116	16
Predicates	190	24	27	253	35

Table 1: Data statistics (in number of thousands) for the CoNLL 2012 split and the train/dev split of OntoNotes5.

Training We use Adam (Kingma and Ba, 2015) with initial learning rate 0.001 and decay rate of 0.1% every 100 steps. The LSTM weights are initialized with random orthonormal matrices (Saxe et al., 2014). We apply 0.5 dropout to the word embeddings and character CNN outputs and 0.2 dropout to all hidden layers and feature embeddings. In the LSTMs, we use variational dropout masks that are shared across timesteps (Gal and Ghahramani, 2016), with 0.4 dropout rate.

Batching At training time, we randomly shuffle all the documents and then batch at sentence level. Each batch contains at most 40 sentences and 700 words. All models are trained for at most 320,000 steps with early stopping on the development set, which takes less than 48 hours on a single Titan X GPU.

A.2 OntoNotes Data Statistics

Table 1 shows the data statistics on various splits of OntoNotes. We found that some sentences in the OntoNotes 5.0 train/dev split have missing predicates, which is unsuitable for training end-to-end SRL systems. Therefore, our end-to-end SRL models are trained on the smaller but cleaner CoNLL 2012 splits. For experiments with gold predicates, we use the full OntoNotes 5.0 train/dev split and the CoNLL 2012 test set, following previous work.