A. Appendix

A.1. General Learning Setup

The method described in Sections 3 and 4 provides a gradient of the loss with respect to the parameters of the model. To update the parameters, one can use any standard optimization method for neural networks. Our experiments use Adam (Kingma & Ba, 2015) with default settings. SPENs are vulnerable to overfitting, as the energy network is often very expressive. We reduce overfitting by performing early stopping, by taking the model that performs best on development data. Often, we have found that early stopping with a model that has a higher capacity energy (e.g., higher-dimensional hidden layers in the energy network) is superior to using a low-capacity energy.

A.2. Architectures for SRL Experiments

A.2.1. Baseline Arc-Factored Architecture

Our baseline is an arc-factored model for the conditional probability of the predicate-argument arc labels:

$$\mathbb{P}(\mathbf{r}|\mathbf{x},\mathbf{p},\mathbf{a}) = \Pi_i \mathbb{P}(r_i|\mathbf{x},\mathbf{p},\mathbf{a}). \tag{18}$$

where $\mathbb{P}(r_i|\mathbf{x},\mathbf{p},\mathbf{a}) \propto \exp\big(g(r_i,\mathbf{x},\mathbf{p},\mathbf{a})\big)$. Here, each conditional distribution is given by a logistic regression model. We compute $g(r_i,\mathbf{x},\mathbf{p},\mathbf{a})$ using a multi-layer perceptron (MLP) similar to FitzGerald et al. (2015). Its inputs are discrete features extracted from the argument span and the predicate (including words, pos tags, and syntactic dependents), and the dependency path and distance between the argument and the predicate. These features are transformed to a 300-dimensional representation linearly, where the embeddings of word types are initialized using newswire embeddings from (Mikolov et al., 2013). We map from 300 dimensions to 250 to 47 (the number of semantic roles in CoNLL) using linear transformations separated by tanh layers. We apply dropout to the embedding layer with rate 0.5 and a standard log loss.

A.2.2. Global Energy Term for SPEN

From the pre-trained model Eq. (18), we define $\mathbf{f}_r$ as the predicate-argument arc features, We also have predicate features $\mathbf{f}_p$ and argument feature $\mathbf{f}_a$, given by the average word embedding of the token spans. The hidden layers of any MLP below are 50-dimensional. Each MLP is two layers, with a SoftPlus in the middle. All parameters are trained discriminatively using end-to-end training.

Let $\mathbf{y}_p \in \Delta_A^m$ be the sub-tensor of $\mathbf{y}$ for a given predicate p and let $\mathbf{z}_p = \sum_k \mathbf{y}_p[:,k] \in [0,1]^m$, where $\mathbf{z}_p[a]$ is the total amount of mass assigned to the arc between predicate p and argument a, obtained by summing over possible labels. We also define $\mathbf{w}_p = \sum_k \mathbf{y}_p[k,:] \in \mathbb{R}_+^A$. This is a length-A vector containing how much total mass of each arc label is assigned to predicate p. Finally, define $\mathbf{s}_r = \sum_k \mathbf{y}[:,:,k]$. This is the total mass assigned to arc r, obtained by summing over the possible labels that the arc can take on.

The global energy is defined by the sum of the following terms. The first energy term scores the set of arguments attached to each predicate. It computes a weighted average of the features $\mathbf{f}_a$ for the arguments assigned to predicate p, with weights given by $\mathbf{z}_p$. It then concatenates this with $\mathbf{f}_p$, and passes the result through a two-layer multi-layer perceptron (MLP) that returns a single number. The total energy is the sum of the MLP output for every predicate. The second energy term scores the labels of the arcs attached to each predicate. We concatenate $\mathbf{f}_p$ with $\mathbf{w}_p$ and pass this through an MLP as above. The third energy term models how many arguments a predicate should take on. For each predicate, we predict how many arguments should attach to it, using a linear function applied to $\mathbf{f}_p$. The energy is set to the squared difference between this and the total mass attached to the predicate under $\mathbf{y}$, which is given by $\sum_k \mathbf{w}_p[k]$. The fourth energy term averages $\mathbf{w}_p$ over all p and applies an MLP to the result. The fifth term computes a weighted average of the arc features $\mathbf{f}_r$, with weights given by $\mathbf{s}_r$ and also applies an MLP to the result. The last two terms capture general topical coherence of the prediction.