Appendices

A Implementation Details

Neural Network Architecture Our implementation is based on AWD-LSTM (Merity et al., 2018).[18] It uses a three-layer LSTM, with carefully designed regularization techniques. PaLM includes the span attention after the second layer. Preliminary results show that it yields similar results, but is less sensitive to hyperparameters, compared to adding it to the last layer.

The context is concatenated to the hidden state ($\bar{\mathbf{h}}_t = [\mathbf{h}_t; \mathbf{a}_t]$), and then fed to a tanh-MLP controlled by a residual gate $\mathbf{g}_r$ (He et al., 2016), before fed onward into the next LSTM layer:

$$\hat{\mathbf{h}}_t = \mathbf{g}_r \odot \text{MLP}\left(\bar{\mathbf{h}}_t\right) + (\mathbf{1} - \mathbf{g}_r) \odot \mathbf{h}_t. \quad (6)$$

The rest of the architecture stays the same as AWD-LSTM. We refer the readers to Merity et al. (2018) for more details.

More details on PaLM-S. PaLM-S uses exactly the same architecture and hyperparameters as its unsupervised counterpart. We derive, from PTB training data, a m-dimensional 0-1 vector for each token. Each element specifies whether the corresponding span appears in the gold parse. Trivial spans (i.e., the ones over single tokens and full sentences) are ignored. The vector are normalized to sum to one, in order to facilitate the use of cross-entropy loss. λ in Eq. 5 is set to 0.01.

Hyperparameters. The regularization and hyperparameters largely follow Merity et al. (2018). We only differ from them by using smaller hidden size (and hence smaller dropout rate) to control for the amount of parameters in the PTB experiments, summarized in Table 5 For the WikiText-2 experiments, we use 200 rational RNN size and 400 dimensional context vectors. Other hyperparameters follow Merity et al. (2018). The max span length m is set to 20 for PTB experiments, and 10 for WikiText-2.

Merity et al. (2018) start by using SGD to train the model, and switch to averaged SGD (Polyak and Juditsky, 1992) after 5 nonimprovement-epochs. We instead use Adam (Kingma and Ba) with default PyTorch settings to train the model for 40 epochs, and then switch to ASGD, allowing for faster convergence.

[18] https://github.com/salesforce/awd-lstm-lm

Type	Values
Rational RNN size	200
Context Vector Size	400
LSTM Hidden Size	1020
Weight Dropout	0.45
Vertical Dropout	0.2

Table 5: The hyperparameters used in the PTB language modeling experiment.

B Span Representations

Below is the derivation for Eq. 2.

$$\begin{aligned}
\overrightarrow{\mathbf{c}}_{i,j} &= \overrightarrow{\mathbf{u}}_j + \sum_{k=i}^{j-1} \overrightarrow{\mathbf{u}}_k \bigodot_{\ell=k+1}^{j} \overrightarrow{\mathbf{f}}_\ell \\
&= \overrightarrow{\mathbf{u}}_j + \sum_{k=1}^{j-1} \overrightarrow{\mathbf{u}}_k \bigodot_{\ell=k+1}^{j} \overrightarrow{\mathbf{f}}_\ell - \sum_{k=1}^{i-1} \overrightarrow{\mathbf{u}}_k \bigodot_{\ell=k+1}^{j} \overrightarrow{\mathbf{f}}_\ell \\
&= \overrightarrow{\mathbf{c}}_j - \left(\overrightarrow{\mathbf{u}}_{i-1} + \sum_{k=1}^{i-2} \overrightarrow{\mathbf{u}}_k \bigodot_{\ell=k+1}^{i-1} \overrightarrow{\mathbf{f}}_\ell \right) \bigodot_{\ell=i}^{j} \overrightarrow{\mathbf{f}}_\ell \\
&= \overrightarrow{\mathbf{c}}_j - \overrightarrow{\mathbf{c}}_{i-1} \bigodot_{k=i}^{j} \overrightarrow{\mathbf{f}}_k
\end{aligned}$$