SUPPLEMENTARY MATERIAL: ICLR 2023 SUBMISSION ID 2106

Anonymous authors
Paper under double-blind review

A HYPERPARAMETER VALUES

We show all hyperparameters related to Stacked LSTM and RHN experiments in Table. 1. These hyperparameters are not optimized by hyperpruning.

Table 1: Hyperparameters list: dimension of hidden units (H-dim), number of layers (Layers), dimension of input embedding (Emb), Optimization (Opt), Learning rate (LR), Non-monotone interval for SNT-ASGD (Non-mono), Training batch size (BS), Back-Propagation Through Time (BPTT), Gradient Clip (Clip), LS computation batch size (LS-BS), Sparsity decay schedule (Decay Sche), Encoder and Decoder weight tied (Tied).

Model	H-dim	Layers	Emb	Opt	LR	Non-mono	BS
Stacked LSTM	1500	2	1500	SNT-ASGD	40	5	20
RHN	830	1	830	SNT-ASGD	15	5	20

Model	BPTT	Dropout	Epochs	Clip	LS-BS	Decay Sche	Tied
Stacked LSTM	35	0.65	100	0.25	2	Cosine	False
RHN	35	0.65	500	0.25	2	Cosine	True

B JACOBIAN MATRIX DERIVATION

B.1 STACKED LSTM

The LSTM is defined by the following equations:

$$
\begin{aligned}
f_t &= \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \\
i_t &= \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \\
o_t &= \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \\
c'_t &= \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \\
c_t &= f_t \cdot c_{t-1} + i_t \cdot c'_t \\
h_t &= o_t \cdot \sigma_c(c_t),
\end{aligned}
$$

where σ_g and σ_c are element-wise sigmoid function and $tanh$ function, respectively. For simplicity, we use the following notation when calculating the derivatives:

$$
y_* = W_* x_t + U_* h_{t-1} + b_*,
$$

where $*$ can be f, i, o, c, depending on the gate. The derivative of each of those gates/states with respect to the hidden states is shown below:

$$
\begin{aligned}
\frac{\partial f_t}{\partial h_{t-1}} &= [\sigma_g(y_f)(1-\sigma_g(y_f))]^T U_f \\
\frac{\partial i_t}{\partial h_{t-1}} &= [\sigma_g(y_i)(1-\sigma_g(y_i))]^T U_i \\
\frac{\partial o_t}{\partial h_{t-1}} &= [\sigma_g(y_o)(1-\sigma_g(y_o))]^T U_o \\
\frac{\partial c_t}{\partial h_{t-1}} &= \frac{\partial f_t}{\partial h_{t-1}} c_{t-1} + \frac{\partial i_t}{\partial h_{t-1}} tanh(y_c) + i_t sech^2(y_c) U_c \\
\frac{\partial h_t}{\partial h_{t-1}} &= \frac{\partial o_t}{\partial h_{t-1}} tanh(c_t) + o_t sect^2(c_t) \frac{c_t}{\partial h_{t-1}},
\end{aligned}
$$

The derivative of each of those gates/states with respect to the inputs is shown below:

$$
\begin{aligned}
\frac{\partial f_t}{\partial x_t} &= [\sigma_g(y_f)(1-\sigma_g(y_f))]^T W_f \\
\frac{\partial i_t}{\partial x_t} &= [\sigma_g(y_i)(1-\sigma_g(y_i))]^T W_i \\
\frac{\partial o_t}{\partial x_t} &= [\sigma_g(y_o)(1-\sigma_g(y_o))]^T W_o \\
\frac{\partial c_t}{\partial x_t} &= \frac{\partial f_t}{\partial x_t} c_{t-1} + \frac{\partial i_t}{\partial x_t} tanh(y_c) + i_t sech^2(y_c) W_c \\
\frac{\partial h_t}{\partial x_t} &= \frac{\partial o_t}{\partial x_t} tanh(c_t) + o_t sect^2(c_t) \frac{c_t}{\partial x_t},
\end{aligned}
$$

B.2 RHN

An RHN layer with a recurrent depth of L is described by:

$$
\begin{aligned}
h_l^{[t]} &= p_l^{[t]} \cdot e_l^{[t]} + h_{l-1}^{[t]} \cdot c_l^{[t]}, \\
& l \in [1, L],
\end{aligned}
$$

where

$$
\begin{aligned}
p_l^{[t]} &= tanh(W_P x^{[t]} \mathbb{1}_{\{l=1\}} + R_{P_l} h_{l-1}^{[t]} + b_{P_l}) \\
e_l^{[t]} &= \sigma(W_E x^{[t]} \mathbb{1}_{\{l=1\}} + R_{E_l} h_{l-1}^{[t]} + b_{E_l}) \\
c_l^{[t]} &= \sigma(W_C x^{[t]} \mathbb{1}_{\{l=1\}} + R_{C_l} h_{l-1}^{[t]} + b_{C_l})
\end{aligned}
$$

and $\mathbb{1}$ is the indicator function. The output of the RNH layer is the output of the L^{th} Highway layer, i.e., $y^{[t]} = h_L^{[t]}$. The derivative is shown below:

$$
\frac{h_1^{[t]}}{h_1^{[t-1]}} = \frac{\partial h_1^{[t]}}{\partial h_L^{[t-1]}} \prod_{l=1}^{L-1} \frac{\partial h_{l+1}^{[t-1]}}{\partial h_l^{[t-1]}}
$$

where

$$
\frac{\partial h_{l+1}^{[t-1]}}{\partial h_l^{[t-1]}} = P_{l+1}^{\prime[t-1]} diag(e_{l+1}^{[t-1]}) + E_{l+1}^{\prime[t-1]} diag(p_{l+1}^{[t-1]}) + diag(c_{l+1}^{[t-1]}) + C_{l+1}^{\prime[t-1]} diag(h_l^{[t-1]}),
$$

with

$$P_{l+1}'^{[t-1]} = R_{P_{l+1}}^T diag[tanh'(R_{P_{l+1}} h_l^{[t-1]} + b_{P_{l+1}})]$$
$$E_{l+1}'^{[t-1]} = R_{E_{l+1}}^T diag[tanh'(R_{E_{l+1}} h_l^{[t-1]} + b_{E_{l+1}})]$$
$$C_{l+1}'^{[t-1]} = R_{C_{l+1}}^T diag[tanh'(R_{C_{l+1}} h_l^{[t-1]} + b_{C_{l+1}})],$$

and

$$\frac{\partial h_1^{[t]}}{\partial h_L^{[t-1]}} = P_1'^{[t]} diag(e_1^{[t]}) + E_1'^{[t]} diag(p_1^{[t]}) + diag(c_1^{[t]}) + C_1'^{[t]} diag(h_L^{[t-1]})$$

with

$$P_1'^{[t]} = R_{P_1}^T diag[tanh'(W_P x^{[t]} + R_{P_1} h_L^{[t-1]} + b_{P_1})]$$
$$E_1'^{[t]} = R_{E_1}^T diag[tanh'(W_E x^{[t]} + R_{E_1} h_L^{[t-1]} + b_{E_1})]$$
$$C_1'^{[t]} = R_{C_1}^T diag[tanh'(W_C x^{[t]} + R_{C_1} h_L^{[t-1]} + b_{C_1})]$$

If it is coupled, c = 1 - e. The RHN becomes

$$h_l^{[t]}. = p_l^{[t]} \cdot e_l^{[t]} + h_{l-1}^{[t]} \cdot (1 - e_l^{[t]})$$

C CORRELATION COMPARISON

To show that LS-based distance is more informative and predictive than the loss-based method, we show the scatter plot of early LS-based distance vs. final perplexity (Top) and early perplexity vs. final perplexity (Bottom) in Fig. 1. For a quantitative comparison, we use Pearson Coefficient which evaluates to what degree a monotonic function fits the relationship between two random variables, the higher the better. The Pearson Coefficient is defined below.

$$\rho = \frac{cov(r, s)}{\sigma_r \sigma_s},$$

where $cov(\cdot, \cdot)$ is the covariance of two variables, and σ_r and σ_s are the standard deviations of r and s, respectively.

As we can see from Fig. 1, LS-based distance has a much higher Pearson Coefficient than early perplexity at the beginning of training (epoch 3: 0.68 vs. 0.45) and at early training (epoch 15: 0.73 vs. 0.27) which means LS-based distance is more informative and predictive than early perplexity.

D TIME EFFICIENCY

In Figure 4 of the main paper, we showed the time efficiency of LS-based hyperpruning and loss-based full training search by estimating their time to reach a particular target perplexity. Specifically, the entire training of each candidate takes 100 epochs, including 70 epochs of SGD optimization and 30 epochs of SNT-ASGD optimization. Experiments are conducted on an NVIDIA GeFORCE RTX 2080 Ti GPU, and one epoch of SGD and SNT-ASGD optimization takes about 1.5 and 3.5 minutes, respectively. Therefore, extensive training for each candidate takes $70 \times 1.5 + 30 \times 3.5 = 210$ min = 3.5 hr. For a target testing perplexity of 70, we find that about 40 candidates are need to be fully trained to reach it which takes about $150hrs$. For LS-based Hyperpruning with the initial candidate pool size $n = 40$, initial epoch $e_0 = 3$ and incremental epoch $e_i = 3$, the candidate selection process

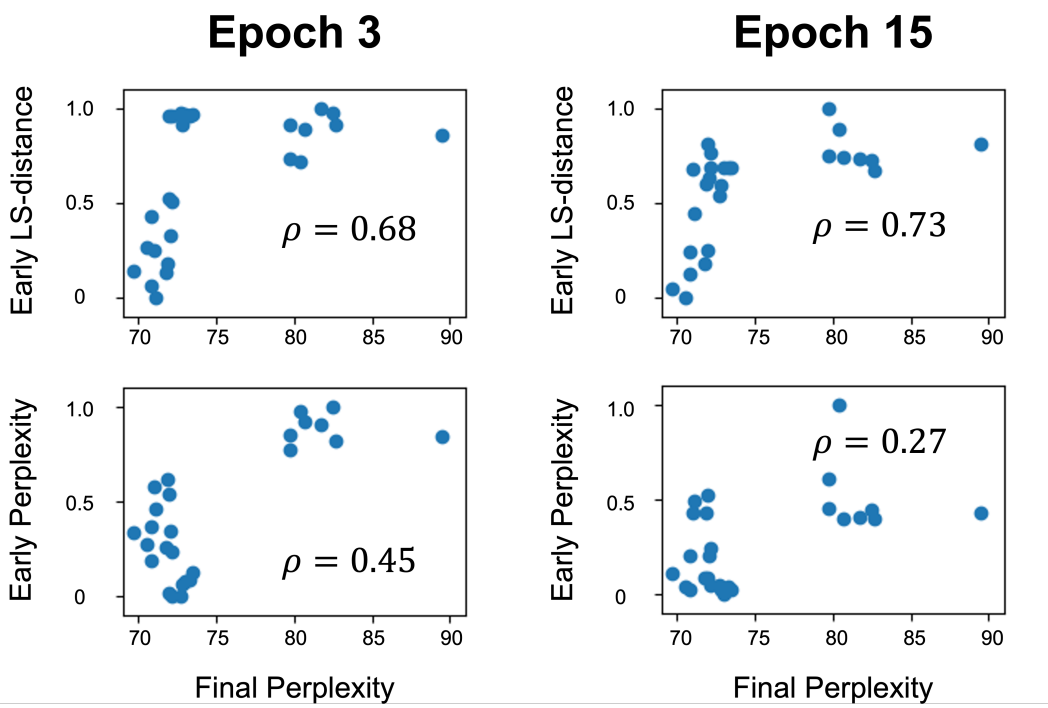

Figure 1: (Top) scatter plot of Early LS-distance (at epoch 3 and 15) vs. Final perplexity. (Bottom) scatter plot of Early perplexity (at epoch 3 and 15) vs. Final perplexity.

and the extensive training process take about 8 and 7 hours, respectively. Thus, the LSH is more time efficient than Perplexity-based full training (15hr vs. 150hr).

One concern of LSH is that the LS computation is computationally expensive and memory heavy since it involves QR decomposition and Jacobian matrix multiplication. It takes about 6s to compute the LS of each sample. However, as we mentioned in Table. 4 of the main paper, the difference between using 2 and 10 samples is not significant, so in our experiments, we just use 2 samples for the LS computation which take about 12s. Compared with the time of training one epoch with SGD (90s), the LS computation only slows down the candidate selection process by 13% compared to the loss-based one, and the extensive training process does not require LS computation. Therefore, for larger problems, even if the LS computation is expensive since only a few samples are required, the overall computation overhead is still reasonable.

E EMBEDDING SPACE

As mentioned in the main paper, we use the following equations to calculate the distance between the candidate model t^i in the candidate pool $\mathcal{T}$ and the reference dense model $\hat{t}$:

$$[\hat{v}_0, \ldots, \hat{v}_E, v_0^i, \ldots, v_E^i] = embedding([\hat{\Lambda}, \Lambda^i])$$
$$s = distance(\hat{v}_E, v_E^i),$$

where Λ_j^i stands for the LS of the i-th candidate at the j-th epoch and Λ^i groups all LS of t^i till the the latest epoch E, i.e.,$\Lambda^i \equiv \{\Lambda_j^i\}_{j=0}^E$. The embedded LS of t^i and $\hat{t}$ at epoch E, i.e., $\hat{v}_E$ and v_E^i, is used to calculate the distance. We use PCA as the embedding function and L_2 distance but there are no special constraints on this particular embedding function and distance metric. We perform an ablation study on three embedding spaces, namely, PCA, original LS, and T-SNE, and two kinds of distance metrics, namely L_2 and Cosine distance which is defined in the following equations:

$$L_2(p,q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$$

$$Cos(p,q) = 1 - \frac{p \cdot q}{|p||q|} = 1 - \frac{\sum_{i=1}^{n}(p_i q_i)}{\sqrt{\sum_{i=1}^{n}(p_i)^2} + \sqrt{\sum_{i=1}^{n}(q_i)^2}},$$

The results of the selected model based on those embedding functions and distance metrics are shown in Table. 2. As we can see, each combination achieves a decent result, and even using the original LS with Cosine distance obtains a comparable performance which means LS is informative of the model performance even at early training. Furthermore, PCA with L_2, used by LSH outperforms other combinations.

Table 2: Perplexities of the final model selected based on different combinations of embedding functions and distances metrics.

Embedding	PCA		LS		T-SNE	
Distance	L2	Cos	L2	Cos	L2	Cos
Perplexity	**69.9**	72.99	72.56	70.3	70.5	72.6

F LSH-SELECTED MODEL

To better understand the LSH-selected model, we show it on the LS-Space with the dense reference model and its testing perplexity over the entire training. In Fig. 2 (Left), we use the PCA as the embedding function and show the 2d-PCA space. Each point corresponds to the model at one epoch, i.e., v_e^i,. The arrows in the figure point along with the training, i.e., from pre-training to post-training, pruned model's color gradually changes from blue to green and the dense model's from black to gray. In Fig. 2 (Right), we show the perplexities of LSH selected model at each point during training. As noticed, the sudden drop of perplexity at around epoch 70 is due to the change of optimizer from SGD to SNT-ASGD.

the LSH-selected pruned model

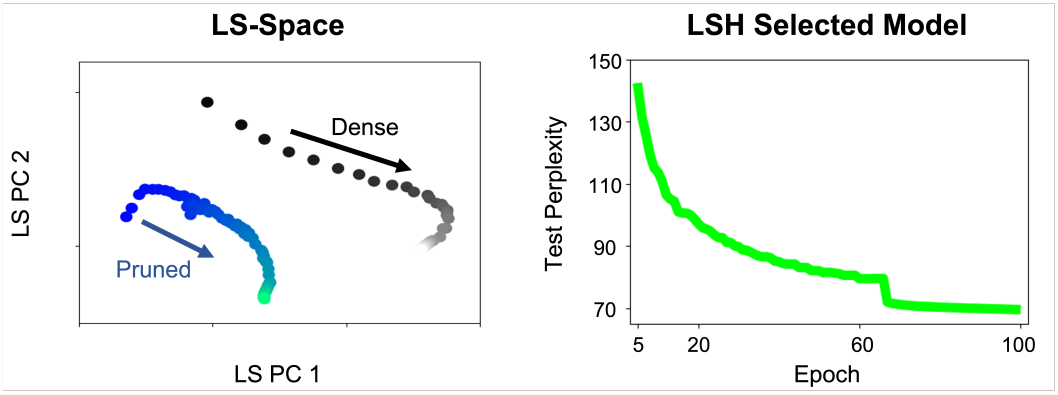

Figure 2: (Left) LS-space: Each dot represents one model at one epoch and the arrows point to later training (Pruned: blue to green; Dense: black to gray). (Right) Epoch vs. Test perplexity of the LSH selected model