Block-Sparse Recurrent Neural Networks

Sharan Narang, Eric Undersander, Greg Diamos

Baidu Research

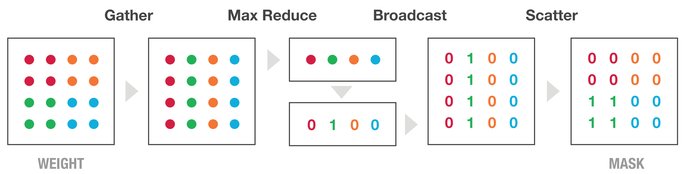
Abstract

Recurrent Neural Networks (RNNs) excel in domains such as speech recognition, machine translation, and language modelling. Sparsity is a technique to reduce compute and memory requirements of deep learning models. Sparse RNNs are easier to deploy on devices and high-end server processors.

Despite the compute and memory savings of sparse operations relative to their dense counterparts, their observed speed-up is less than expected on different hardware platforms. To address this issue, we investigate two approaches to induce **block** sparsity in RNNs: **block pruning** and **group lasso** regularization.

Sparsity Algorithm		Description
Block Pruning	BP	Blocks of weights are pruned throughout training by periodically comparing each block's largest weight against a gradually increasing threshold.
Group Lasso	GL	Group lasso regularization with a large lambda (λ_g) drives less-important blocks of weights to zero.
Group Lasso + Block Pruning	GLP	Group lasso regularization with a small lambda (λ_g) guides pruning by shrinking less-important blocks of weights.
Weight Pruning	WP	Our earlier work: pruning individual weights.

Using these techniques, we create block-sparse RNNs with sparsity ranging from 80% to 90% with small loss in accuracy. This is a roughly 10x model size reduction. Additionally, we can prune a larger dense network to recover this loss in accuracy while maintaining high block sparsity and reducing the overall parameter count. Block-sparse RNNs eliminate overheads related to data storage and irregular memory accesses and better utilize modern hardware compared to unstructured sparsity.



Block pruning. This toy example shows a 4×4 weight matrix. We gather blocks of size 2×2, perform a max reduction, compare to a prune threshold, and update a mask. A zero mask value indicates a pruned weight.

$$L = L_{\text{training}} + \lambda_g \sum_{g=1}^{G} \|w^{(g)}\|_2$$

Group lasso regularization. λ_g is regularization lambda, $w^{(g)}$ is a block of weights, $\|w^{(g)}\|_2$ is the L^2 norm of the block, and G is the total number of blocks.

Future Work

- Explore other block reduction operations besides max, including average, RMS, geometric mean, and median.
- Explore sparse model training. We want to be able to train deep learning models that are sparse throughout the entire training run. This will allow us to exploit computation and memory savings for training.
- Improve the performance of sparse matrix dense vector libraries for GPUs and ARM processors that would speed up deployment.

Results

Model	Algorithm	# Params (millions)	Sparsity	CER	Relative Perf
RNN Dense 1760	N/A	67	0.0%	15.36	0.0%
RNN Dense 704	N/A	11.6	0.0%	18.95	-23.4%
RNN Sparse 1760	BP	7.3	89.2%	17.93	-16.7%
RNN Sparse 2560	GLP	12.9	90.8%	15.89	-3.4%
RNN Sparse 3072	BP	25.8	87.3%	15.66	-1.9%
GRU Dense 2560	N/A	115	0.0%	15.42	0.0%
GRU Dense 704	N/A	11.0	0.0%	21.26	-37.9%
GRU Sparse 2560	GLP	10.8	90.6%	16.78	-8.8%
GRU Sparse 3584	BP	25.6	88.4%	16.23	-5.2%

Block sparsity experiments. RNN model has 2 convolution layers, 7 bidirectional layers, and softmax followed by CTC cost layer. GRU model has 2 convolution layers, 3 gated recurrent unit layers, row convolution, and softmax followed by CTC cost layer. Block size is 4×4.

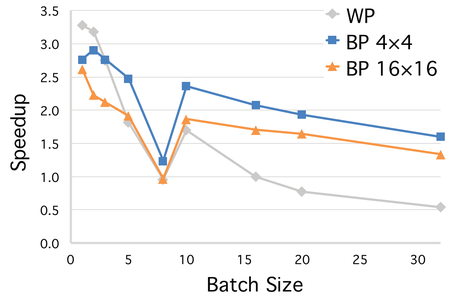
Model	Algorithm	# Params (millions)	Sparsity	CER	Relative Perf
RNN Sparse 1760	GL	10.9	83.3%	30.14	-96.0%
RNN Sparse 1760	GLP	6.2	90.8%	19.24	-25.3%
RNN Sparse 2560	GL	24.4	82.8%	27.40	-78.4%
RNN Sparse 2560	GLP	12.9	90.8%	15.89	-3.4%

Comparing group lasso variants. Block size is 4×4. Group lasso without pruning (GL) does much worse than group lasso with pruning (GLP).

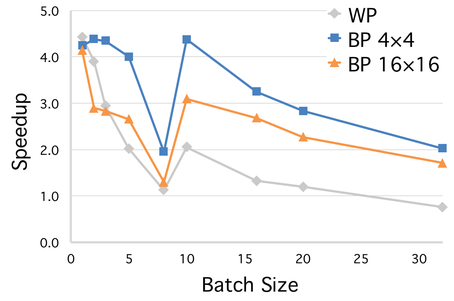
Model	Block Size	# Params (millions)	Sparsity	CER	Relative Perf
RNN Sparse	1×1	7.3	89.2%	17.32	-12.8%
RNN Sparse	4×4	7.3	89.2%	17.93	-16.7%
RNN Sparse	12×2	10.8	84.1%	16.96	-10.0%
RNN Sparse	8×8	10.7	84.1%	17.66	-14.9%
RNN Sparse	16×16	11.1	83.6%	17.10	-11.3%
RNN Sparse	32×32	14.1	79.1%	16.67	-8.5%
GRU Sparse	1×1	13.1	88.7%	16.55	-7.3%
GRU Sparse	4×4	16.2	86.0%	16.97	-10.5%
GRU Sparse	16×16	20.8	81.9%	16.84	-9.2%

Block pruning (BP), varying block size. All results have similar accuracy, but models with smaller block sizes achieve this with fewer weights (higher sparsity).

Speedup

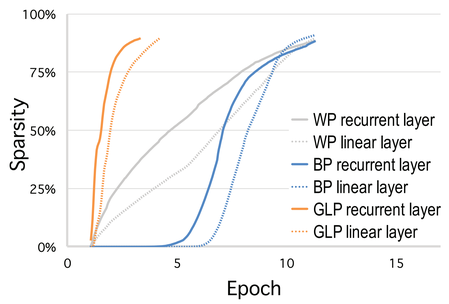


Speedup for RNN 1760 matrix multiply. TitanX Maxwell with CuSparse. Sparse matrices are in CSR format. RNN matrix sizes are (1760, 1760) with 90% sparsity and (1760, batch size).

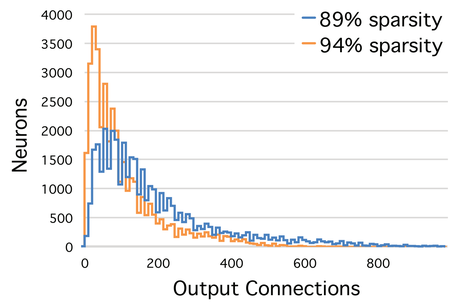


Speedup for GRU 2560 matrix multiply. GRU matrix sizes are (7680, 2560) with 95% sparsity and (2560, batch size).

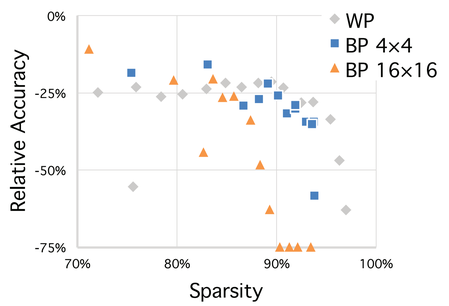
Analysis



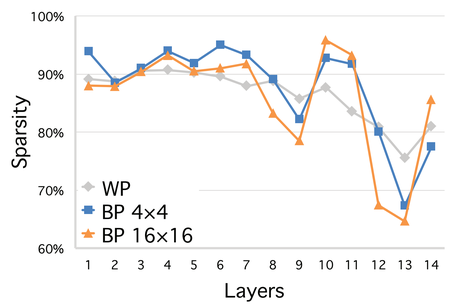
Pruning schedule. Two layers in the RNN model are shown. Layer sparsity increases gradually throughout early training.



Histogram of number of output connections for all neurons in the network. Block pruning (BP) with 4×4 blocks.



Accuracy vs. sparsity. Baseline RNN model, varying hyperparameters to produce a spectrum of results ranging from 70% to 97% sparsity. Results worse than -75% are capped at 75%.



Sparsity by layer. Recurrent layers in the RNN model are shown. Layers closer to the input tend to be more sparse than layers near the output.