Structured Bayesian Pruning with Log-Normal Multiplicative Noise

Key Results

Structured Bayesian Pruning is a new model that provides structured sparsity, e.g., removes neurons and convolutional filters.

Our contributions can be summarized as follows:

- a method of regularization of DNNs that results in structured sparsity
- a proper analog of sparsity-inducing log-uniform prior
- experiments that show that SBP regularizes well and leads to a high level of group sparsity (it removes up to 80% of all units on a VGG-like architecture) and acceleration (up to 4.5x measured speed-up) with small accuracy drop

The method is implemented as a separate dropout-like layer and an additional regularization term. TensorFlow implementation of our method is available.

Approximate posterior distribution over $\theta$ by Stochastic Variational Inference:

$$L = -\mathbb{E}_{q(\theta \mid \varphi)} \log p(Y \mid X, \theta) + D_{KL}(q(\theta \mid \varphi) \parallel p_{prior}(\theta)) \rightarrow \min \varphi$$

- The true posterior distribution over $\theta$ is approximated by $q(\theta \mid \varphi) \parallel p(\theta)$
- Just a slightly different loss function; implementation is basically the same

Structured Bayesian Pruning with Improper Log-Uniform Prior

The model injects multiplicative noise $\theta = x \cdot \eta_i \sim p_{model}(\eta_i)$

Log-uniform prior for sparsity:

$$p(\eta_i) = \log U_{[0,1]}(\theta) \propto \frac{1}{\eta_i}$$

Log-normal prior for sparsity:

$$\log \theta_i \sim N(\log \varphi_i, \sigma_i^2)$$

The approximated posterior is log-normal:

$$\log \theta_i \sim N(\log \varphi_i, \sigma_i^2)$$

The variational family has no “prior gap”

Log-normal noise does not change the sign of $x$

The KL-divergence term can be computed analytically

Due to the improper prior we obtain an ill-posed optimization problem

$$\text{KL}(\log N(\theta, \sigma^2) \parallel \log U_{[0,1]}(\theta)) \propto -\log(\sigma)$$

In order to obtain a proper probabilistic model, we truncate the prior and the posterior:

$$p(\theta_i) = \log U_{[0,\infty]}(\theta_i)$$

$$q(\theta_i) = \log N(\theta_i \mid \varphi_i) \Rightarrow \log N_{[\varphi_i,\sigma_i]}(\theta_i \mid \varphi_i)$$

$$\log P(\theta_i \mid \varphi_i) \propto \log P(\theta_i) - \log U_{[0,\infty]}(\theta_i)$$

All necessary statistics can be computed in closed form:

- KL-divergence for training
- Expectation $E_{\theta}$ for inference during testing
- Signal-to-noise ratio $\text{SNR} = E_{\theta} / \sqrt{D_{KL}}$ for pruning redundant neurons

Final Algorithm

Our final loss function is negative variational lower bound

$$L = -\mathbb{E}_{q(\theta \mid \varphi)} \log p(Y \mid X, \theta) + \alpha \cdot KL(q(\theta | \mu, \sigma) \parallel p(\theta)) \rightarrow \min \varphi$$

where $\alpha$ denotes all weights of DNN, $q$ and $p$ are truncated distributions.

Training procedure details:

- All models were pretrained with L2 regularization on parameters $W$
- Re-weight the KL term by $\alpha$, proportional to the computational complexity of each specific layer (SBP procedure)
- Remove neurons with low $\text{SNR} \beta$ after training; no fine-tuning needed!
- Tricks for numerically stable calculations are presented in the appendix

Experiments: LeNets on MNIST

In MNIST experiments we compare different structured sparsity-inducing techniques on LeNet-5-Caffe and LeNet-500-300 architectures.

- Our method provides the highest speed-up with the same accuracy.
- Table: SSL is based on group lasso regularization, SparseVD induces weight-wise sparsity and can coincidently remove all weights in filters or neurons, StructuredBP is our model. We report acceleration that was measured on CPU (Intel Xeon E5-2630), GPU (Tesla K40) and in terms of Floating Point Operations (FLOPs).

Experiments: VGG-like on CIFAR-10

- CIFAR-10 experiments were done on a VGG-like architecture[3]. The network consists of 12 convolutional and 2 fully connected layers with Batch Normalization and Binary Dropout
- With small accuracy drop our models provide significant acceleration and high structured sparsity. Presented speed-up was measured on CPU.

Experiments: Random Labels

Discussions

- Bayesian Learning framework is well known for providing non-structured sparse solutions. Usually sparsity is caused by Empirical Bayes which adjusts the prior distribution to the data. It can potentially lead to additional overfitting.
- In this work we utilize the Bayesian framework to obtain structured sparsity. We did not adjust the prior distribution, so the risk of overfitting is decreased.

Links and References