Flipout: Efficient Pseudo-Independent Weight Perturbations on Mini-Batches

Yeming Wen[†‡], Paul Vicol[†‡], Jimmy Ba[†‡], Dustin Tran[*♦], Roger Grosse[†‡]

[†]University of Toronto [‡]Vector Institute [*]Google [♦]Columbia University

Motivation

- Stochastic weights are used in many settings:
  - Regularization (DropConnect)
  - Training BNNs (Gaussian perturbations)
  - Evolution Strategies
  - Exploration in reinforcement learning
- Due to the large number of weights, it is very expensive to compute and store separate weight perturbations for every example in a mini-batch.
- All examples in a mini-batch typically share the same weight perturbation, thereby limiting the variance reduction effect of large mini-batches.

Summary

- We developed a method called Flipout that allows us to sample pseudo-independent weight perturbations efficiently for each example in a mini-batch.
- Flipout decorrelates the gradients between examples and achieves a $1/N$ variance reduction effect in practice.
- Flipout applies to any perturbation distribution that factorizes by weight and is symmetric around 0.
- Flipout speeds up training neural networks with multiplicative Gaussian perturbations, is effective at regularizing LSTMs, and enables us to vectorize evolution strategies.

Theoretical Results

- Flipout gives unbiased stochastic gradients.
- Flipout is guaranteed to have smaller variance than shared perturbations.

Independent: $\frac{\alpha}{N}$

Shared: $\frac{\alpha}{N} + \beta + \gamma$

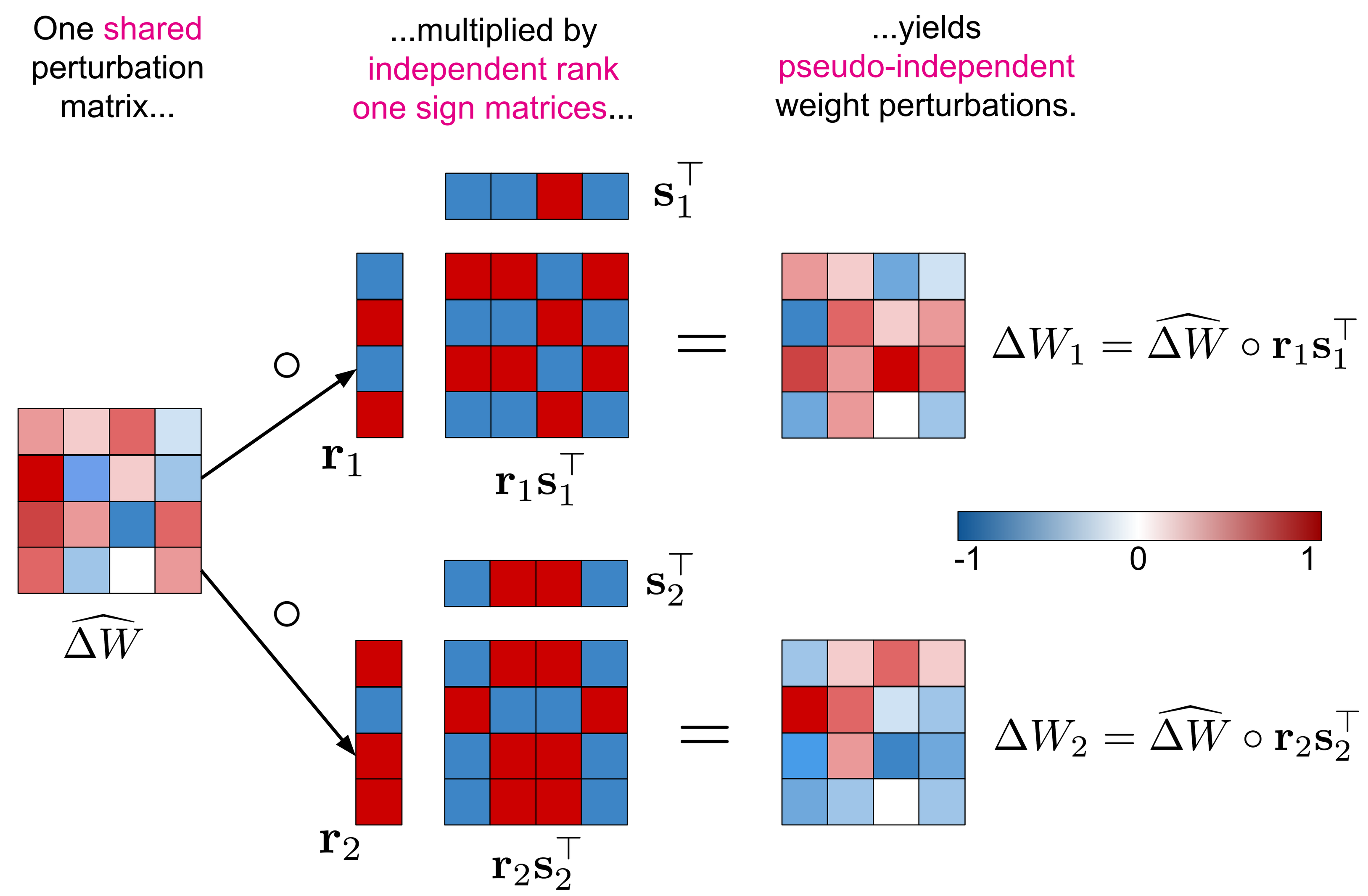
Flipout: $\frac{\alpha}{N} + \gamma$

$\alpha =$ variance of gradients on individual examples

$\beta =$ covariance from sampling r and s

$\gamma =$ covariance from sampling $\widehat{\Delta W}$

Method



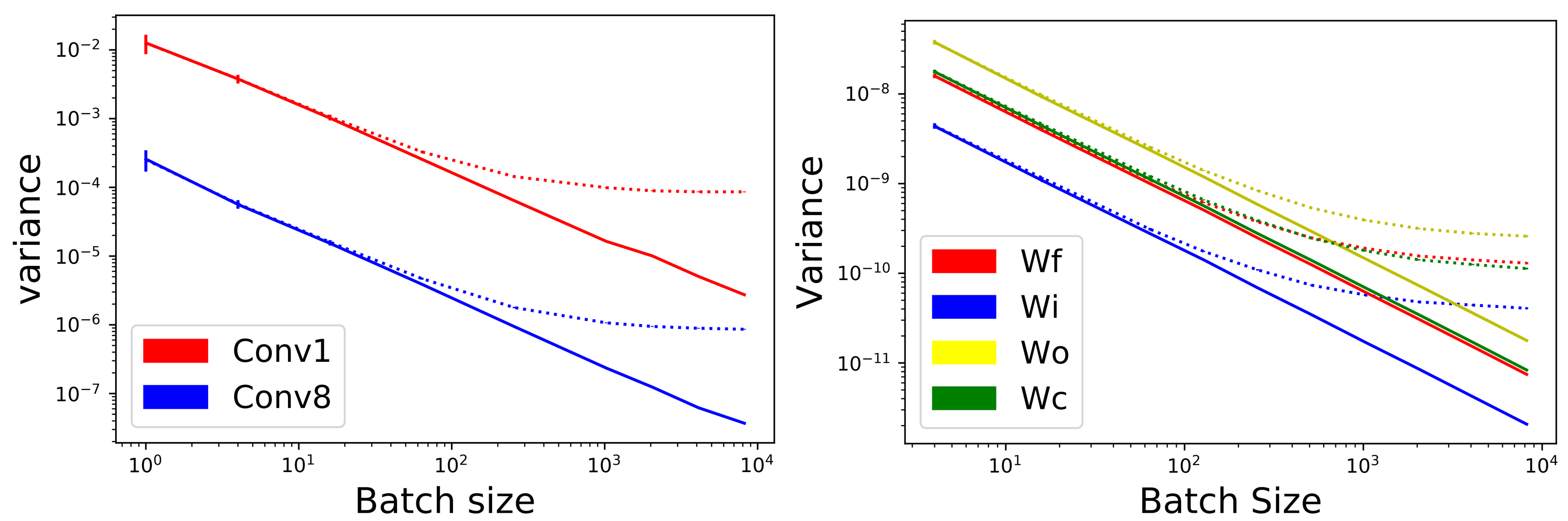
- To vectorize these computations, we define matrices R and S whose rows correspond to the random sign vectors r_n and s_n for all examples in the mini-batch. Let X denote the batch activations in one layer of a neural net. The next layer's activations are given by:

$$Y = \phi\left(X\overline{W} + \left((X \circ S)\widehat{\Delta W}\right) \circ R\right).$$

where ϕ denotes the activation function.

Variance Reduction

- Flipout achieves the ideal linear variance reduction with increasing mini-batch size for FC-NNs, CNNs, and RNNs.



ConVGG on CIFAR-10 LSTM on Penn Treebank

Dotted: shared perturbations. Solid: flipout

LSTM Regularization

- Character-level Penn Treebank: Flipout achieves the best reported results for a 1-layer, 1000 hidden unit architecture.

Model	Valid	Test
Unregularized LSTM	1.468	1.423
Semeniuta (2016)	1.337	1.300
Zoneout (2016)	1.306	1.270
Gal (2016)	1.277	1.245
Mult. Gauss. (ours)	1.257	1.230
Mult. Gauss + Flipout (ours)	**1.256**	**1.227**

Large Batch Training

- Flipout converges in ∼3x fewer iterations than shared perturbations and is ∼2x as expensive, yielding a 1.5x speedup overall.

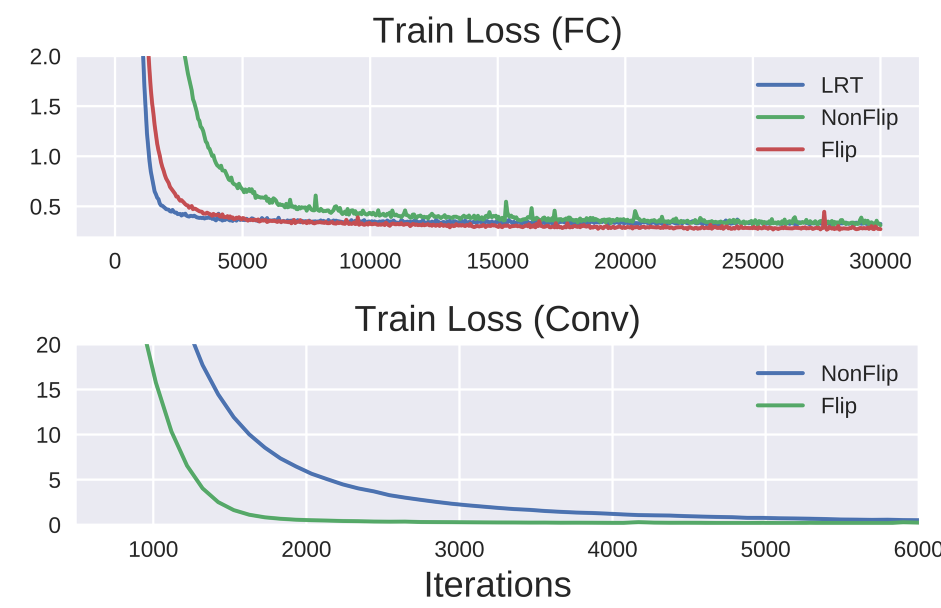


Figure: MNIST training using Bayes By Backprop with batch size 8192

Vectorizing Evolution Strategies

- FlipES is as sample-efficient as using fully-independent perturbations. One GPU with Flipout can handle the same throughput as at least 40 CPU cores using existing methods.