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

Authors: Yeming Wen, Paul Vicol,Jimmy Ba & Dustin Tran

Review by Thibault Kollen

April 2020

Summary

- Introduction & Problematic
- Flipout Method
- Experiments
- Conclusion

- Stochasticity is a key component of many modern neural net architectures and training algorithms. Stochastic neural networks are a type of artificial neural networks built by introducing random variations into the network, either by giving the network's neurons stochastic transfer functions, or by giving them stochastic weights.
- The most widely used regularization methods are based on randomly perturbing a network's computations.
- In many cases, a network has many more weights than units, and it is very expensive to compute and store separate weight perturbations for every example in a mini-batch.

- Stochastic weight methods are typically done with a single sample per mini-batch
- Weight perturbation algorithms suffer from high variance of the gradient because they share the same perturbation (one sample per mini-batch)
- Actually, sharing the perturbation induces correlations between the gradients,implying that the variance can't be eliminated by averaging.

- Define f, the output of a network: $f(x, W)$, with x the input and W the weight
- We can also define W, the weight as: $W = \overline{W} + \Delta W$, with $\overline{W}$ the mean weights and ΔW, the stochastic perturbation.
- Define q_θ, the distribution, for the samples of weights
- Define E, the expected loss: $E_{(x,y)\sim D, W\sim q_\theta}[L(f(x, W), y)]$, with L, a loss function and D representing the data distribution

- We can perturb a network's activations, in this case it is easy to sample independently for different training examples within a mini-batch.
- Then, variance of the stochastic gradients decays as $1/N$, where N is the size of mini-batch
- It can be possible to reformulate weight perturbations as activation perturbations in some cases. It is the Local Reparameterization Tric or LRT method
- It samples B, the matrix of activation rather than the weight: $B = xW$, with x, the input and W, the weight matrix

Flipout Method

Objetives & Description

- Flipout, should be an efficient way to perturb the weights quasi-independently within a mini-batch, and have variance reduction effect for large mini-batches
- More precisely, it should be an efficient method for decorrelating the gradients between different examples without biasing the gradient estimates
- Two important hypothesis for the weight distribution: Independency of the perturbations of different weights and Symmetry around 0 of this perturbation

Flipout Method

How it works?

- We define $\widehat{\Delta W}$, a base perturbation used by all examples in mini-batch, to match the previous weight distribution and E a random sign matrix, independant of $\widehat{\Delta W}$.
- We define ΔW, the stochastic perturbation, as: $\Delta W = \widehat{\Delta W} \circ E$. The distribution is preserved for ΔW.
- Actually, Flipout multiply $\widehat{\Delta W}$, the base perturbation, with a different rank-one sign matrix for each example: $\Delta W_n = \widehat{\Delta W} \circ r_n s_n{}^T$.
- The distribution grants then an unbiased estimator for the loss gradients (marginal distribution over gradients for individual will be identical to the distribution using shared weight perturbations)

Flipout Method

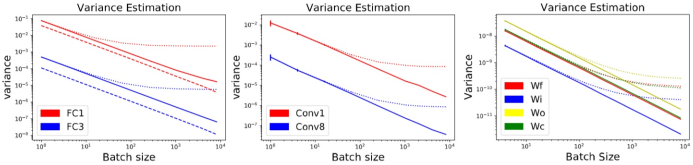
Advantages & Cost

- Flipout simplify computation in mini-batch, by using matrix multiplication instead of explicit perturbations, and the implementation will be more efficient.
- Flipout is guaranteed to reduce the variance of the gradient estimates compared to using naïve shared perturbations.
- In general, the most expensive operation in the forward pass is matrix multiplication.
- Here two matrix multiplications are required instead of one, but the tasks can be parallelized for optimization.

Experiments

Variance reduction

Name	Network Type	Data Set
ConvLe	(Shallow) Convolutional	MNIST (LeCun et al., 1998)
ConVGG	(Deep) Convolutional	CIFAR-10 (Krizhevsky & Hinton, 2009)
FC	Fully Connected	MNIST
LSTM	LSTM Network	Penn Treebank (Marcus et al., 1993)



(a) Fully-connected Net (FC) (b) Convolutional Net (conVGG) (c) LSTM

Figure: Empirical variance of gradients with respect to mini-batch size for several architectures. Dotted: Shared perturbations. Solid: Flipout. Dashed: LRT.

Experiments

Language model

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 ($\sigma = 1$) (ours)	1.257	1.230
Mult. Gauss + Flipout (ours)	**1.256**	**1.227**
RBN (2017)	–	1.32
H-LSTM + LN (2016)	1.281	1.250

Table 2: Bits-per-character (BPC) for the character-level PTB task. The RBN and H-LSTM+LN results are from the respective papers. All other results are from our own experiments.

Model	Valid	Test
Unregularized LSTM	132.23	128.97
Zaremba (2014)	80.40	76.81
Semeniuta (2016)	81.91	77.88
Gal (2016)	78.24	75.39
Zoneout (2016)	78.66	75.45
WD (2017)	78.82	75.71
WD + Flipout (ours)	**76.88**	**73.20**

Table 3: Perplexity on the PTB word-level validation and test sets. All results are from our own experiments.

Figure: Evaluation of the regularization effect of Flipout on the character-level and word-level language modeling tasks with the PTB. Comparison of Flipout to several other methods for regularizing RNNs

Conclusion

- We discover Flipout, an efficient method for decorrelating the weight gradients between different examples in a mini-batch.
- We see that Flipout is guaranteed to reduce the variance compared with shared perturbations
- Flipout also makes it practical to apply GPUs to evolution strategies.
- Flipout should make weight perturbations practical in the large batch setting favored by modern accelerators such as Tensor Processing Units

Thank you for listening!