# Bag of Tricks for Faster & Stable Image Classification

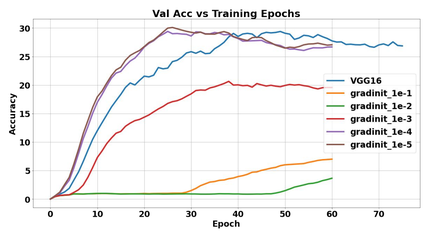
**Aman Bansal, Shubham Anand Jain, Bharat Khandelwal**
**CS231N Spring 2022**

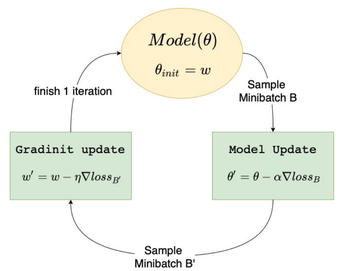


Stanford University

## GradInit

- **Objective:** Effective initialization of weights => a smooth optimization landscape
- **Idea:** Train the weights to maximize gradients while avoiding exploding gradients
- **Pros:** Avoids unstable training, allows faster learning
- **Con:** Loss increases quickly after overfitting because of smooth landscape




- Gradinit Cost
  <1% of overall cost for large models
  Can be amortized over underlying training hyperparameter search

Reference: ``GradInit: Learning to Initialize Neural Networks for Stable Efficient Training'', Zhu, Chen, et al., NeurIPS 2021

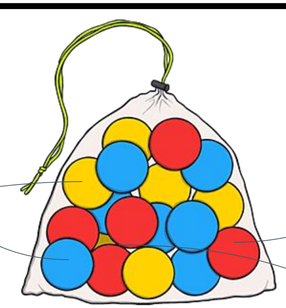
## Problem Statement

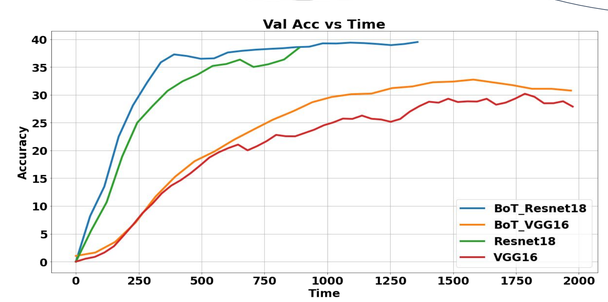
We work on **image classification** tasks. We try various (recently published) ideas that help **learning be faster and more stable**.

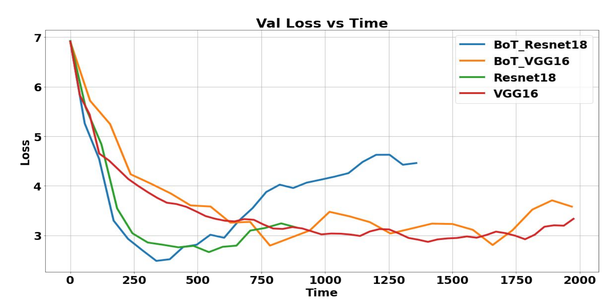
**Dataset:** CIFAR-100 + Preprocessing

**Objective:** Analyze interesting papers in top conferences, combine to get better training performance, create **a module that can train any pytorch model with any combination of these!**

*Our GitHub Repository: Scan the QR code!*

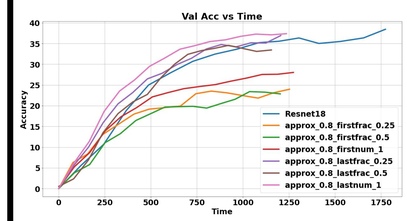


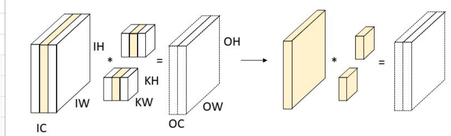



## Approximate Tensor Multiplication

- **Objective:** Reduce cost of matrix multiplications by approximation
- **Idea:** Do approximate multiplication using column-row sampling
- **Pros:** Faster, Can potentially act similar to dropout
- **Con:** Can impact performance and learning if not used correctly



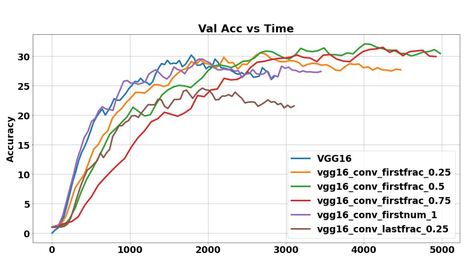



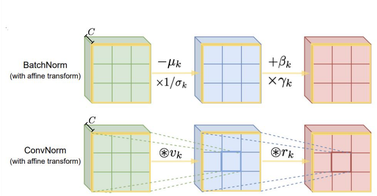

**Novel Idea:** Use CRS-Sampling in last few (pre-final) layers!
**Justification:** Linear Layers at the end are more parameter-heavy, and matter less in final output compared to layers near the beginning (ripple effect)

Reference: ``Faster Neural Network Training with Approximate Operations'', Adelman, Menecham, et al., NeurIPS 2021

## Convolutional Normalization

- **Objective:** Promote orthogonality in kernels of convolutional layers
- **Idea:** Can do more easily in Fourier domain!
- **Pros:** Promotes Lipschitzness, More stable
- **Con:** Slower than Conv2d



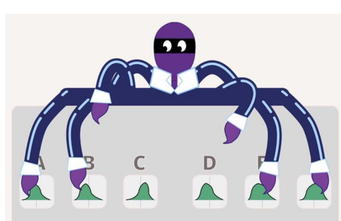



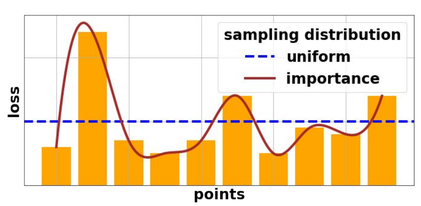

**Novel Idea:** Use ConvNorm for starting layers!
**Justification:** Starting layers more important -> more stable leads to better training
Adding more ConvNorm layers follows marginally decreasing returns in time v/s acc

Reference: ``Convolutional Normalization: Improving Deep Convolutional Network Robustness and Training'', Liu, Sheng, et al., CVPR 2021

## Importance Sampling

- **Objective:** Sample some data points more than others
- **Idea:** Assign an importance score and sample with probability proportional to score
- **Pro:** Better accuracy in same number of epochs
- **Con:** Increases epoch training time




**Novel Idea:** Weigh points proportional to running momentum of the loss!
**Justification:** Loss is a good approximation to norm of gradient for importance sampling
**Results:** Marginally better accuracy results with an uptick in training time

Ref: 1) ``Not all samples are created Equal'', Katharopoulos Angelos, ICML 2018
2) ``Adam with Bandit Sampling for Deep Learning'', Liu, Riu, et al., NeurIPS 2020