# Improving Batch Normalization with Skewness Reduction for Deep Neural Networks

Pak Lun Kevin Ding, Sarah Martin and Baoxin Li

School of Computing, Informatics, and Decision Systems Engineering

Arizona State University

# Motivation

- BN layer normalizes the batch input to zero mean and unit variance
  - Smoother loss landscape
  - Faster convergence
- Making the distributions of the features in the same layer more similar would make the network perform better
  - The third moment, Skewness
    - More non-linearity

# Batch Normalization with Skewness Reduction

**Definition 2.** *Let* $\varphi_p : \mathbb{R} \to \mathbb{R}$ *be a function, the skewness correction function are defined as follows:*

$$\varphi_p(x) = \begin{cases} x^p & \text{if} \quad x \geq 0 \\ -(-x)^p & \text{if} \quad x < 0 \end{cases} \tag{6}$$

*where* $p > 1$.

# Batch Normalization with Skewness Reduction

**Algorithm 1:** Training stage of BNSR, applied to features $x$ over a mini-batch

**Input** : Values of $x$ over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;

**Parameters:** Parameters to be learned: $\gamma, \beta$

**Output** : $y_i = \mathrm{BN}_{\gamma,\beta}(x_i)$

$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$

$\sigma_{\mathcal{B}}{}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2$

$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sigma_{\mathcal{B}}}$

$\hat{x}_i \leftarrow \varphi_p(\hat{x}_i)$

$y_i \leftarrow \gamma\hat{x}_i + \beta \equiv \mathrm{BNSR}_{\gamma,\beta}(x_i)$

# Batch Normalization with Skewness Reduction

**Algorithm 2:** Testing stage of BNSR, applied to features $x$ over a mini-batch

**Input** : Values of $x$ over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;

**Output** : $y_i = \mathrm{BN}_{\gamma,\beta}(x_i)$

Calculate the population $\mu$, $\sigma$ by unbiased estimation or exponential moving average

**for** $i = 1 \ldots m$ **do**

  $\hat{x}_i \leftarrow \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$

  $\hat{x}_i \leftarrow \varphi_p(\hat{x}_i)$

**end**

$y_i = \gamma \hat{x}_i + \beta$

# Experiments

- Determine p
  - VGG-19 on CIFAR-100, p in {1.01, 1.02, 1.03, 1.04, 1.05}
- Impact of the similarity of the feature distributions
  - $x \leftarrow x$ (identity mapping)
  - $x \leftarrow ax + b$ where $a, b \sim N_m(0, 0.5)$
  - $x \leftarrow \varphi_p(x)$ where $p \sim Unif_m(1, 1.05)$
  - $x \leftarrow \varphi_p(x)$ where $p = 1.01$

| | BNSR | BN | Noise($\mu$, $\sigma$) | Noise($\rho$) |
|---|---|---|---|---|
| error | 30.61 | 31.35 | 33.52 | 32.1 |

TABLE I

COMPARISON OF ERROR RATES (%) OF BNSR, BN, BN WITH NOISY MEAN AND VARIANCE, BN WITH NOISY SKEWNESS ON CIFAR-100. THE TRAINING LOSS AND ERROR RATE CURVES ARE IN FIG. 2

# Experiments

- Features in the earlier layers
  - Analyze where BNSR is more effective
    - BNSR is used for all layers
    - BNSR is used only for the earlier layers
    - BNSR is used only for the later layers

| | 100% | 33%(uni) | 33%(early) | 33%(late) |
|---|---|---|---|---|
| error | 23.49 | 23.40 | 23.74 | 25.20 |

TABLE III

COMPARISON OF ERROR RATES (%) OF BNSR UNDER DIFFERENT PERCENTAGE OF USAGE ON CIFAR-100. THE TRAINING LOSS AND TESTING ERROR PLOTS CAN BE FOUND IN FIG. 4.

# Experiments

- Comparison with other normalization schemes

| | BNSR | BN | LN | IN |
|---|---|---|---|---|
| error | 23.49 | 25.51 | 39.78 | 28.72 |

TABLE II

COMPARISON OF ERROR RATES (%) OF BNSR, BN, LN, IN ON CIFAR-100. THE TRAINING LOSS AND ERROR RATE CURVES ARE IN FIG. 3

| | BNSR | BN |
|---|---|---|
| error | 22.73 | 23.17 |

TABLE IV

COMPARISON OF ERROR RATES (%) OF BN AND BNSR ON IMAGENET DATASET. THE TRAINING LOSS AND ERROR RATE CURVES ARE IN FIG. 7