Improving Batch Normalization with Skewness Reduction for Deep Neural Networks

Pak Lun Kevin Ding, Sarah Martin, Baoxin Li
School of Computing, Informatics, Decision Systems Engineering, Arizona State University
Arizona State University
{kevinding, samart44, baoxin.li}@asu.edu

ICPR 2020
25th INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION
Milan, Italy 10 | 15 January 2021

Introduction

In this paper, we demonstrate that the performance of the network can be improved, if the distributions of the features of the output in the same layer are similar. As normalizing based on mean and variance does not necessarily make the features to have the same distribution, we propose a new normalization scheme: Batch Normalization with Skewness Reduction (BNSR). Our contributions are summarized as follows:

- We propose a new batch normalization scheme.
- The scheme introduces a nonlinear function, which not only decreases the skewness of the feature distributions, but also increases the flexibility of the network.
- We demonstrate that our approach outperforms other normalization approaches on visual recognition tasks.

Batch Normalization with Skewness Reduction

To encourage the distributions of the features to be further similar, we propose BNSR, which adds a nonlinear function between the two parts of original BN: the feature normalization and the scaling and shifting part. We first start by giving the definitions:

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 $p > 1$.

As a result, after applying the step of feature normalization, we operate the step of skewness reduction, which can be described as:

$$\hat{x} \leftarrow \varphi_p(\hat{x}) \tag{7}$$

Although applying this function always leads to non-zero means and non-unit variances, these oscillations are still acceptably small if we choose a small p, and conceptually can be absorbed by the linear transformation right after this step.

Algorithm

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

$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$
$\sigma_{\mathcal{B}}{}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2$
$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sigma_{\mathcal{B}}}$
$\hat{x}_i \leftarrow \varphi_p(\hat{x}_i)$
$y_i \leftarrow \gamma\hat{x}_i + \beta \equiv \mathrm{BNSR}_{\gamma,\beta}(x_i)$

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 μ, σ 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

We first determine the value of p by using VGG-19 network to evaluate the performance on CIFAR-100 for p = 1.01, 1.02, 1.03, 1.04 or 1.05. We choose the p corresponding to the least final error, which is p = 1.01 We then analyze how the similarity of the feature distributions impact the performance of the neural network, by using the same network to evaluate different settings of normalization on CIFAR-100. After that, we investigate the histogram for the features from different layers. We also use BNSR for only 33% of the total number of normalization layers (that is, for all the normalization layers, we use BNSR for 33% of them, and original BN for 66% of them), and analyze where BNSR is more effective. We then evaluate BNSR with BN, LN, IN on CIFAR-100, and with BN on ImageNet. Experimental results show that the proposed approach can outperform other state-of-the-arts that are not equipped with BNSR.

	BNSR	BN	Noise(μ, σ)	Noise(ρ)
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