Normalization Propagation: A Parametric Technique for Removing Internal Covariate Shift in Deep Networks

ICML 2016

Devansh Arpit, Yingbo Zhou, Bhargava U. Kota, Venu Govindaraju
Task:

Train deep networks using gradient descent based methods
Task:

Train deep networks using gradient descent based methods

Goal:

Remove internal covariate shift: leads to slow convergence
Motivation

- **Covariate Shift** (Shimodaira, 2000)

\[ \mathbf{X} \xrightarrow{P(Y|X)} \mathbf{Y} \]

- Domain 1
  - \( P(Y|X) \) identical for both Domain
  - Learn best approximation \( P_\theta(Y|X) \approx P(Y|X) \)

- Domain 2
Motivation

- **Covariate Shift** (Shimodaira, 2000)

\[ P(Y|X) \xrightarrow{X} Y \]

- Domain 1
- \( P(Y|X) \) identical for both Domain
- Learn best approximation \( P_\theta(Y|X) \approx P(Y|X) \)
- Maximum Likelihood focuses on high density of \( X \)
Motivation

- **Covariate Shift** (Shimodaira, 2000)

\[ \mathbf{X} \xrightarrow{P(Y|X)} Y \]

Domain 1
- \( P(Y|X) \) identical for both Domain
- Learn best approximation \( P_\theta(Y|X) \approx P(Y|X) \)
- Maximum Likelihood focuses on high density of \( X \)
- Low density regions of \( X \) are artifacts in \( P_\theta^*(Y|X) \)

Domain 2
Motivation

- **Covariate Shift** *(Shimodaira, 2000)*
  
  $\mathbf{X} \xrightarrow{P(\mathbf{Y}|\mathbf{X})} \mathbf{Y}$

  $P(Y|X)$ for Domain 1 $\neq P(\theta^*(Y|X))$ for Domain 2

  - $P(Y|X)$ identical for both Domain
  - Learn best approximation $P_{\theta}(Y|X) \approx P(Y|X)$
  - Maximum Likelihood focuses on high density of $X$
  - Low density regions of $X$ are artifacts in $P_{\theta^*}(Y|X)$
Motivation

- **Internal Covariate Shift** (Ioffe and Szegedy, 2015)
  \[ X_i \xrightarrow{P(X_{i+1}|X_i)} X_{i+1} \]

- Multi-layer end-to-end learning model

![Diagram of a multi-layer neural network](image)

- Learn the best approximation
  \[ P_\theta(X_{i+1}|X_i) \approx P(X_{i+1}|X_i) \]
Motivation

- **Internal Covariate Shift** (Ioffe and Szegedy, 2015)
  \[ X_i \xrightarrow{P(X_{i+1}|X_i)} X_{i+1} \]

- Multi-layer end-to-end learning model

- Learn the best approximation \( P_\theta(X_{i+1}|X_i) \approx P(X_{i+1}|X_i) \)
- During SGD updates, hidden layer \( P(X_i) \) keeps shifting
**Motivation**

- **Internal Covariate Shift** (Ioffe and Szegedy, 2015)
  \[
  X_i \xrightarrow{P(X_{i+1}|X_i)} X_{i+1}
  \]

- **Multi-layer end-to-end learning model**

  ![Multi-layer model diagram]

- Learn the best approximation \( P_\theta(X_{i+1}|X_i) \approx P(X_{i+1}|X_i) \)
- During SGD updates, hidden layer \( P(X_i) \) keeps shifting
- \( \implies P_\theta^*(X_{i+1}|X_i) \) keeps shifting
Motivation

- Internal Covariate Shift (Ioffe and Szegedy, 2015)
  \[ X_i \xrightarrow{P(X_{i+1}|X_i)} X_{i+1} \]

- Multi-layer end-to-end learning model

- Learn the best approximation \( P_\theta(X_{i+1}|X_i) \approx P(X_{i+1}|X_i) \)
- During SGD updates, hidden layer \( P(X_i) \) keeps shifting
- \( \implies P_{\theta^*}(X_{i+1}|X_i) \) keeps shifting
- Learning \( P_\theta(X_{i+1}|X_i) \) using SGD is slow
Batch Normalization (Ioffe and Szegedy, 2015)
Non-parametric normalization of layer input distribution
A traditional vs. Batch Normalized (BN) ReLU layer:

**Traditional:**
\[ x \quad u_i = w_i^T x + \beta_i \quad y = \text{ReLU}(u) \quad o = y \]

**BN:**
\[ x \quad u_i = \frac{\gamma_i (w_i^T (x - \mathbb{E}_B[x]))}{\sqrt{\text{var}_B(w_i^T x)}} + \beta_i \quad y = \text{ReLU}(u) \quad o = y \]
- **Batch Normalization** (Ioffe and Szegedy, 2015)
- Non-parametric normalization of layer input distribution
- A traditional vs. Batch Normalized (BN) ReLU layer:

  **Traditional:**

  $$x \quad u_i = w_i^T x + \beta_i \quad y = \text{ReLU}(u) \quad o = y$$

  ![Diagram of traditional ReLU layer]

  **BN:**

  $$x \quad u_i = \frac{\gamma_i(w_i^T(x - \mathbb{E}_B[x]))}{\sqrt{\text{var}_B(w_i^T x)}} + \beta_i \quad y = \text{ReLU}(u) \quad o = y$$

  ![Diagram of Batch Normalized ReLU layer]

- **Drawbacks:**
  - Hidden layers’ (global) distribution mean/std estimates shift (used for validation for early stopping)
  - Training with batch size 1 not possible
Table of Contents

1 Introduction
2 Motivation
3 Related Work
4 Our Approach
5 Experiments
6 Conclusion
Our Approach: Normalization Propagation

- Exploit normalization in data by propagating it to higher layers
- Assume data and pre-activations are Gaussian
- A traditional vs. NormProp ReLU layer:

**Traditional:**

\[ x \quad u_i = W_i^T x + \beta_i \quad y = \text{ReLU}(u) \quad o = y \]

**NormProp:**

\[ x \quad u_i = \frac{\gamma_i (W_i^T x)}{\|W_i\|_2} + \beta_i \quad y = \text{ReLU}(u) \quad o = \frac{y - \sqrt{\frac{1}{2\pi}}}{\sqrt{\frac{1}{2} (1 - \frac{1}{\pi})}} \]

- **Condition:** \( \|W_i\|_2 = 1 \) and \( W \) incoherent

\[ \text{coherence} = \max_{W_i, W_j, i \neq j} \frac{|W_i^T W_j|}{\|W_i\|_2 \|W_j\|_2} \]
Our Approach: Normalization Propagation

Analysis:

- Singular values of $\mathbf{J} \approx 1$ prevents gradient problems (Saxe et al. 2014)
- Let layer Jacobian $\mathbf{J} = \frac{\partial o}{\partial x}$
- If $\mathbf{x}$ is Normal, $\|\mathbf{W}_i\|_2 = 1$ and and $\mathbf{W}$ incoherent:
  - $\mathbb{E}_x[\mathbf{J}\mathbf{J}^T] \approx 1.47\mathbf{I} \implies$ Singular values of $\mathbf{J} \approx 1.2$

Extension to other Activation functions ($\sigma$):

- $o_i = \frac{1}{c_1} \left[ \sigma \left( \frac{\gamma_i(\mathbf{W}_i \ast \mathbf{x})}{\|\mathbf{W}_i\|_F} + \beta_i \right) - c_2 \right]$
  - $c_1 = \sqrt{\text{var}(\sigma(\mathbf{Y}))}$, $c_2 = \mathbb{E}[\sigma(\mathbf{Y})]$
  - $\mathbf{Y}$ has Standard Normal distribution
1 Introduction
2 Motivation
3 Related Work
4 Our Approach
5 Experiments
6 Conclusion
Datasets:

- **CIFAR-10** (Krizhevsky, 2009)
  - 32x32 color images
  - 45k train, 5k validation, 10k test samples
  - 10 classes

- **CIFAR-100** (Krizhevsky, 2009)
  - 32x32 color images
  - 45k train, 5k validation, 10k test samples
  - 100 classes

- **SVHN** (Netzer et al., 2011)
  - 32x32 color images
  - ≈528k train, 6k validation, ≈26k test samples
  - 10 classes
Global vs. Batch Data Normalization

- Global data normalization: data mean/variance calculated using all data points
- Batch data normalization: data mean/variance calculated using training mini-batch
Experimental Results

Effect of Batch-size on NormProp

- Validation accuracy vs. epochs
- Different batch sizes: 1, 50, 100, 250
Experimental Results

Hidden unit distribution for validation set:

- **NormProp**
- **Batch Normalization**
- **No Normalization**
Experimental Results

NormProp vs. BN Convergence (batch-size 50)

- Validation accuracy over epochs for NormProp and BN.
- The graph shows the convergence of validation accuracy for both methods.
- The y-axis represents the validation accuracy, ranging from 60 to 95.
- The x-axis represents the epochs, ranging from 10 to 50.

Experiments
Normalization Propagation
### Experimental Results

<table>
<thead>
<tr>
<th>Datasets and Methods</th>
<th>Test Error (%)</th>
<th>Datasets and Methods</th>
<th>Test Error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CIFAR-10</strong></td>
<td></td>
<td><strong>CIFAR-100</strong></td>
<td></td>
</tr>
<tr>
<td>without data augmentation</td>
<td></td>
<td>without data augmentation</td>
<td></td>
</tr>
<tr>
<td>NormProp</td>
<td><strong>9.11</strong></td>
<td>NormProp</td>
<td><strong>32.19</strong></td>
</tr>
<tr>
<td>Batch Normalization</td>
<td>9.41</td>
<td>Batch Normalization</td>
<td>35.32</td>
</tr>
<tr>
<td>NIN + ALP units</td>
<td>9.59</td>
<td>NIN + ALP units</td>
<td>34.40</td>
</tr>
<tr>
<td>NIN</td>
<td>10.47</td>
<td>NIN</td>
<td>35.68</td>
</tr>
<tr>
<td>DSN</td>
<td>9.69</td>
<td>DSN</td>
<td>34.57</td>
</tr>
<tr>
<td>Maxout</td>
<td>11.68</td>
<td>Maxout</td>
<td>38.57</td>
</tr>
<tr>
<td>with data augmentation</td>
<td></td>
<td>with data augmentation</td>
<td></td>
</tr>
<tr>
<td>NormProp</td>
<td>7.47</td>
<td>NormProp</td>
<td><strong>29.24</strong></td>
</tr>
<tr>
<td>Batch Normalization</td>
<td><strong>7.25</strong></td>
<td>Batch Normalization</td>
<td>30.26</td>
</tr>
<tr>
<td>NIN + ALP units</td>
<td>7.51</td>
<td>NIN + ALP units</td>
<td>30.83</td>
</tr>
<tr>
<td>NIN</td>
<td>8.81</td>
<td>NIN</td>
<td>-</td>
</tr>
<tr>
<td>DSN</td>
<td>7.97</td>
<td>DSN</td>
<td>-</td>
</tr>
<tr>
<td>Maxout</td>
<td>9.38</td>
<td>Maxout</td>
<td>-</td>
</tr>
</tbody>
</table>
### Experimental Results

<table>
<thead>
<tr>
<th>Datasets and Methods</th>
<th>Test Error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVHN</td>
<td></td>
</tr>
<tr>
<td>NormProp</td>
<td><strong>1.88</strong></td>
</tr>
<tr>
<td>Batch Normalization</td>
<td>2.25</td>
</tr>
<tr>
<td>NIN + ALP units</td>
<td>-</td>
</tr>
<tr>
<td>NIN</td>
<td>2.35</td>
</tr>
<tr>
<td>DSN</td>
<td>1.92</td>
</tr>
<tr>
<td>Maxout</td>
<td>2.47</td>
</tr>
</tbody>
</table>
1 Introduction
2 Motivation
3 Related Work
4 Our Approach
5 Experiments
6 Conclusion
- Parametric approach for removing internal covariate shift
- Computing mini-batch statistics not required
- Robust to choice of batch size (even with batch size 1)
- Stable Convergence