Machine Learning (CE 40477)

Fall 2024

Ali Sharifi-Zarchi

CE Department
Sharif University of Technology

October 9, 2024

What is Batch Normalization Concept?

- Batch Normalization main purpose: **Smoothing the optimization space**
- Batch Normalization Opt.: Normalizing activations in a network.

Smoothing the optimization space

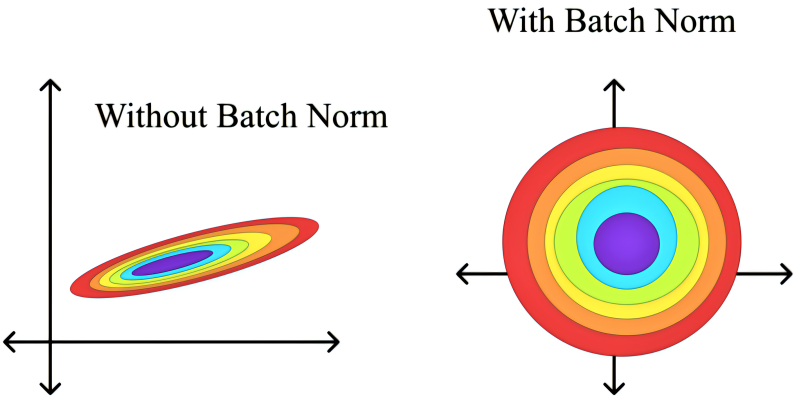


Figure 1: Using Batch Normalization causes the optimization space to become smoother. **Source**

Why Batch Normalization?

Problem: Internal Covariate Shift

- Wait! What does it mean, in simple language?
- Let's say that we want to train a model and the ideal target output function that the model needs to learn is as below.

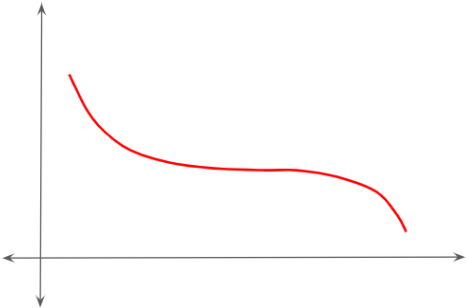


Figure 2: Target function **Source**

Problem: Internal Covariate Shift

- Suppose that the training data values input to the model cover only a part of the range of output values. Therefore, the model can only learn a subset of the target function.

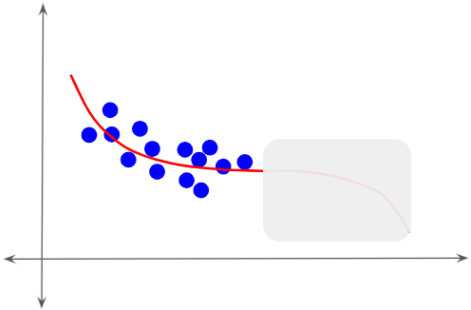


Figure 3: Training data distribution **Source**

Problem: Internal Covariate Shift

- The model has no idea about the rest of the target curve. It could be anything.

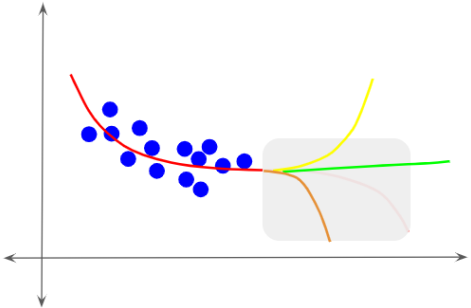


Figure 4: Rest of the target curve **Source**

Problem: Internal Covariate Shift

- Suppose we feed the model to the testing data as below.
- This has a very different distribution from the data that the model was initially trained with.
- The model cannot simply generalize its predictions for this new data.

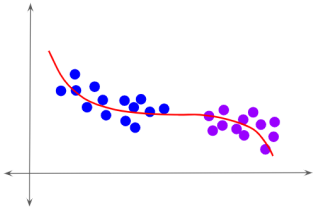


Figure 5: Test data has different distribution **Source**

Problem: Internal Covariate Shift

- This is the problem of Covariate Shift the model is fed data with a different distribution than what it was previously trained with, even though that new data still conforms to the same target function.
- For the model to figure out how to adapt to this new data, it has to re-learn some of its target output functions.
- **This slows down the training process.**

Problem: Internal Covariate Shift

What happens inside Deep Network Layers?

- Think about what happens when we train a deep network: As we update the weights of earlier layers, the data distribution in the deeper layers keeps shifting.
- This means that deeper layers see new and varying patterns every time we update the weights in previous layers.
- As a result, the network must keep readjusting, which makes the learning process slower and more difficult.
- **In other words:** The network is constantly “re-learning” how to make predictions because the data it sees is never consistent!

Batch Normalization Solution

- **Goal:** Normalize inputs so that the mean is near 0 and the variance is close to 1.
- **How it helps:** Stabilizes learning, allowing for higher learning rates and faster convergence.

Magic Effect of Batch Normalization

Magic Effect!

- Batch Normalization helps the network train faster and achieve higher accuracy.
- Batch Normalization **makes the distribution more stable** and reduces the internal covariate shift.

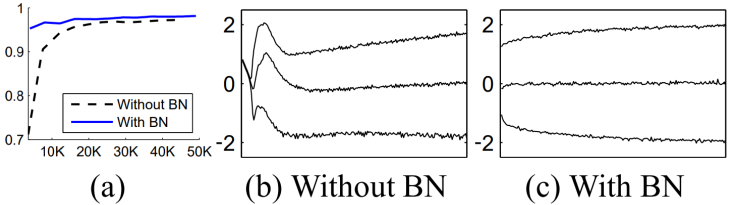


Figure 6: (a) The test accuracy of the MNIST network trained with and without Batch Normalization, vs the number of training steps. (b, c) The evolution of input distributions to a typical sigmoid, for training, shown as 15, 50, 85th percentiles [1].

How Batch Normalization Works

Process Overview

- For each mini-batch during training, batch normalization normalizes the inputs to a layer by adjusting their mean and variance.

How Batch Normalization Works

Steps in Batch Normalization

1. **Compute the Mean and Variance**
   For a given mini-batch, compute the mean μ_B and variance σ_B^2 of the inputs:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i, \quad \sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

How Batch Normalization Works

Steps in Batch Normalization

1. **Compute the Mean and Variance**
   For a given mini-batch, compute the mean μ_B and variance σ_B^2 of the inputs:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i, \quad \sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$$

2. **Normalize the Inputs**
   Subtract the mean and divide by the standard deviation to get normalized activations:

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

   where ϵ is a small constant added for numerical stability.

How Batch Normalization Works (Continued)

Steps in Batch Normalization

3. **Scale and Shift**
   After normalization, introduce learnable parameters γ and β that allow the model to scale and shift the normalized output:

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

   This ensures that the model can recover the original data distribution if needed.

How Batch Normalization Works

Inference Mode: Hint!!!

- During inference (when predicting new data), batch statistics (mean and variance) are replaced with moving averages collected during training.

Effect of Batch Normalization on Gradients

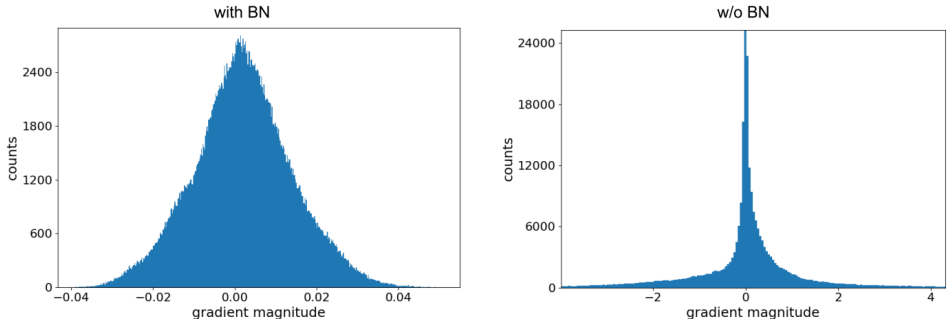


Figure 7: Gradient magnitudes at initialization for layer 55 of a network with and without Batch Normalization. **Source**

Effect of Batch Normalization on Gradients

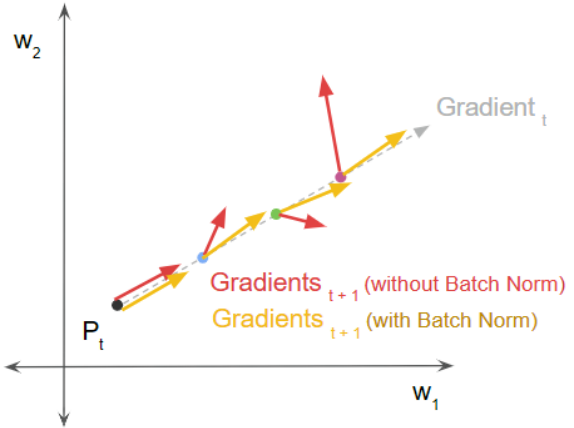


Figure 8: Gradients with Batch Norm are smoother. **Source**

Effect of Batch Normalization on Gradients

The main benefit of batch normalization is that it reduces the dependency of the gradient on the scale of the input and parameters:

$$\frac{\partial \mathcal{L}}{\partial x} = \frac{\partial \mathcal{L}}{\partial y} \cdot \frac{\partial y}{\partial \hat{x}} \cdot \frac{\partial \hat{x}}{\partial x} \tag{1}$$

Where:

- $\frac{\partial y}{\partial \hat{x}} = \gamma$
- $\frac{\partial \hat{x}}{\partial x} = \frac{1}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$

Thus, the gradient becomes:

$$\frac{\partial \mathcal{L}}{\partial x} = \frac{\gamma}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \cdot \frac{\partial \mathcal{L}}{\partial y} \tag{2}$$

Loss Landscape is not Smooth in Typical Neural Networks

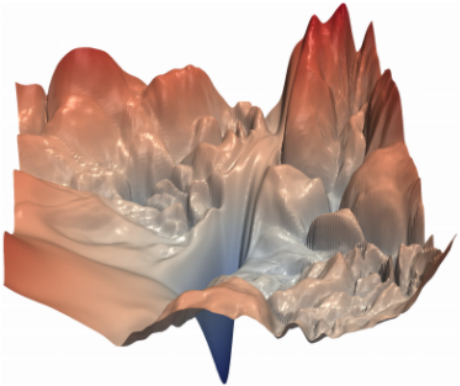


Figure 9: A neural network loss-landscape. **Source**

How Batch Normalization Smooth the Loss Landscape

The smoothing effect of batch normalization can be understood by observing how it constrains the gradient magnitudes. The expression shows that:

$$\frac{\partial \mathscr{L}}{\partial x} \text{ is scaled by } \frac{1}{\sqrt{\sigma_{\mathscr{B}}^2 + \epsilon}} \tag{3}$$

This consistent scaling leads to a smooth loss landscape because:

- It stabilizes the gradient flow, ensuring controlled optimization step sizes.
- Reduces the risk of large oscillations or abrupt changes in the loss landscape.
- Makes the optimization process less likely to be trapped in local minima or saddle points.

Batch Normalization Pros

Pros

- **Faster Convergence:** Empirical results support that models with batch normalization converge faster and achieve higher accuracy, even with higher learning rates.
- **Reduced Sensitivity to Weight Initialization:** Helps mitigate the dependency on careful weight initialization.
- **Acts as Regularization:** Batch normalization can help reduce overfitting.
- **Reduces Vanishing/Exploding Gradients:** Helps maintain stable gradients throughout deep networks.

Batch Normalization Pros

Why Using Batch Normalization Reduces Sensitivity to Weight Initialization?

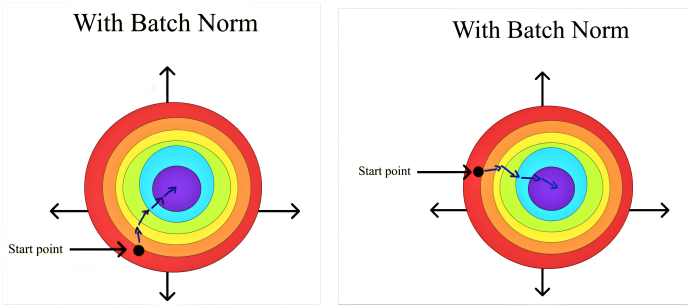


Figure 10: Start point doesn't matter! **Source**

Batch Normalization Pros

Why Does Batch Normalization Reduce Sensitivity to Weight Initialization?

- Batch normalization smooths the optimization landscape, reducing the dependency on initial weights.
- This allows the model to converge to a minimum efficiently, regardless of where optimization begins.

Batch Normalization Cons

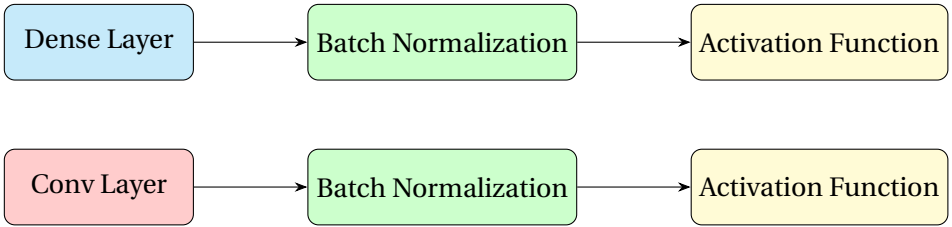
Cons

- **Batch Size Sensitivity:** Performance can depend on batch size, and very small batches may not provide stable statistics.
- **Computational Overhead:** Adds extra computation during training.
- **Behavior During Inference:** The shift from batch statistics to moving averages during inference may lead to slight discrepancies.

Batch Normalization in Practice

Where to Apply

- **Typical Location:** Apply after the linear transformation (e.g., after a dense or convolutional layer) but before the activation function.
- **Layer Placement:**



Batch Normalization in Practice

- **Key Point:** Batch normalization stabilizes and accelerates training while providing regularization benefits.
- **Impact on Training:** Enables efficient training of deeper networks with reduced hyperparameter tuning.

[1] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” *CoRR*, vol. abs/1502.03167, 2015.

[2] A. Ng and K. Katanforoosh, *CS230 Lecture Notes*.
Stanford University, 2018.

[3] F.-F. Li and Z. Durante, *CS231n Lectures*.
Stanford University, 2024.
Updated June 3, 2024.

[4] J. Bjorck, C. P. Gomes, and B. Selman, “Understanding batch normalization,” *CoRR*, vol. abs/1806.02375, 2018.

[5] H. Li, Z. Xu, G. Taylor, and T. Goldstein, “Visualizing the loss landscape of neural nets,” *CoRR*, vol. abs/1712.09913, 2017.

Any Questions?