

BATCH NORMALIZATION and LAYER NORMALIZATION

Anna Binoy

CS461: Advanced Machine Learning, School of Computer Sciences
National Institute of Science Education and Research, Bhubaneswar, India

1st November 2023

OUTLINE

- QUICK RECAP
- BATCH NORMALIZATION
 - Internal Covariate Shift
 - Smoothness of the optimization landscape
- LAYER NORMALIZATION
 - How layer normalization differs from batch normalization?
- OTHER NORMALIZATION TECHNIQUES
 - Group Normalization
 - Instance Normalization
- CONCLUSION
- REFERENCE

Why Normalization ?

- Neural networks may struggle when working with input features that have different value ranges, causing longer training times and convergence issues.
- Training models on unnormalized data leads to slower convergence due to gradient descent difficulties.
- The problem of exploding gradients arises, causing instability as high values propagate through network layers.
- Preprocessing techniques like normalization and standardization help overcome scaling issues.

Quick Recap :

NORMALIZATION

- Normalization is a data preprocessing technique that transforms input feature values to the range [0,1].
- The transformation for a particular input feature x , that has values in the range $[x_{\min}, x_{\max}]$ is represented using the equation ,

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Quick Recap : STANDARDIZATION

- Standardization is a data preprocessing technique that transforms input values to follow a distribution with a zero mean (μ) and unit variance (σ^2).
- It achieves this by using a mathematical formula ,

$$x_{std} = \frac{x - \mu}{\sigma}$$

Throughout this presentation, we will refer to the process of standardization as normalization for consistency.

What is Batch Normalization ?

- Batch normalization (BN) involves normalizing activation vectors in hidden layers using the mean and variance of the current batch's data.
- This normalization is implemented right before or after the application of the nonlinear function.
- Here n = the number of samples in a batch
- At each hidden layer, Batch Normalization transforms the signal as follow :

$$\mu = \frac{1}{n} \sum_i Z^{(i)} \quad \text{-- (1)}$$

$$\sigma^2 = \frac{1}{n} \sum_i (Z^{(i)} - \mu)^2 \quad \text{-- (2)}$$

$$Z_{norm}^{(i)} = \frac{Z^{(i)} - \mu}{\sqrt{\sigma^2 - \epsilon}} \quad \text{-- (3)}$$

$$\hat{Z}^{(i)} = \gamma Z_{norm}^{(i)} + \beta \quad \text{-- (4)}$$

What is Batch Normalization ?

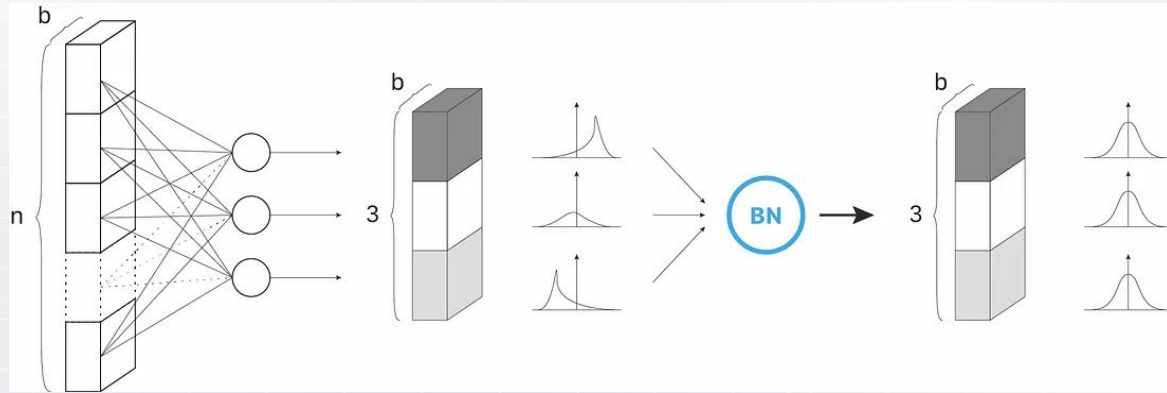
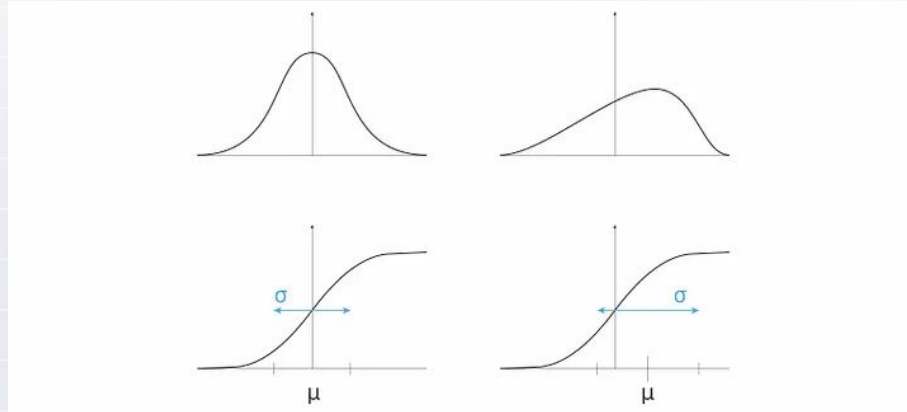


Fig 1: Batch Normalization first three steps [3]

- Example of a 3-neurons hidden layer, with a batch of size b .
- The BN layer first determines the mean μ and the variance σ^2 of the activation values across the batch, using equation (1) and (2) .
- It then normalizes the activation vector $Z^{(i)}$ using equation (3). Each neuron's output follows a standard normal distribution. [3]

What is Batch Normalization ?



$$\hat{Z}^{(i)} = \gamma Z_{norm}^{(i)} + \beta$$

Fig 2: Benefits of γ and β parameters. [3]

- It finally calculates the layer's output $\hat{Z}^{(i)}$ is calculated by applying a linear transformation with γ (scaling factor, adjust the standard deviation) and β (offset, adjust the bias), two trainable parameters as mentioned in equation(4).
- Modifying the distribution (on the top) allows us to use different regimes of the nonlinear functions (on the bottom).

Why Batch Normalization ?

The three main hypothesis on why Batch Normalization actually work is as follows :

- Batch normalization (BN) reduces the “internal covariate shift” [1].
- BN influences the smoothness of the optimization landscape, and hence simplifies the training process. [2]
- BN removes the interdependency between layers during training [5].

Internal Covariate Shift

- Internal Covariate Shift (ICS) is the shifting of the model input distribution happening in the hidden layers of a deep neural network.
- Normalizing the input signal makes the points closer to each other in the feature space during training : it is now easier to find a well generalizing function.

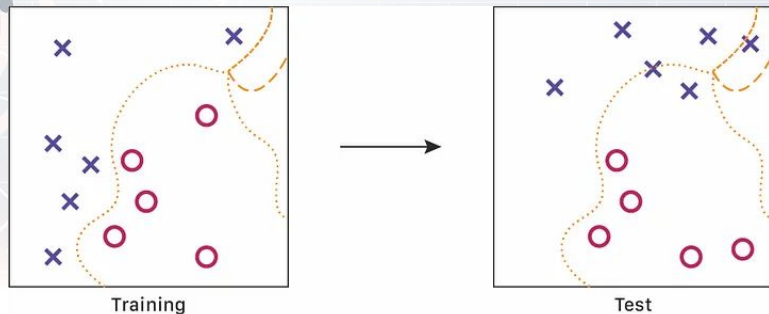


Fig 3: Plot of the extracted feature in the feature space. [3]

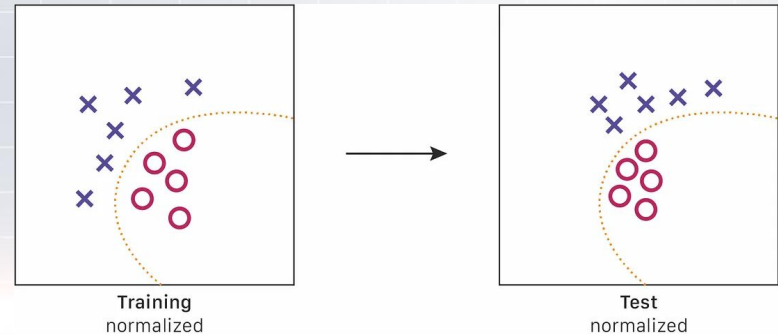


Fig 4: Normalizing the input signal makes the points closer to each other in the feature space during training : it is now easier to find a well generalizing function. [3]

Internal Covariate Shift

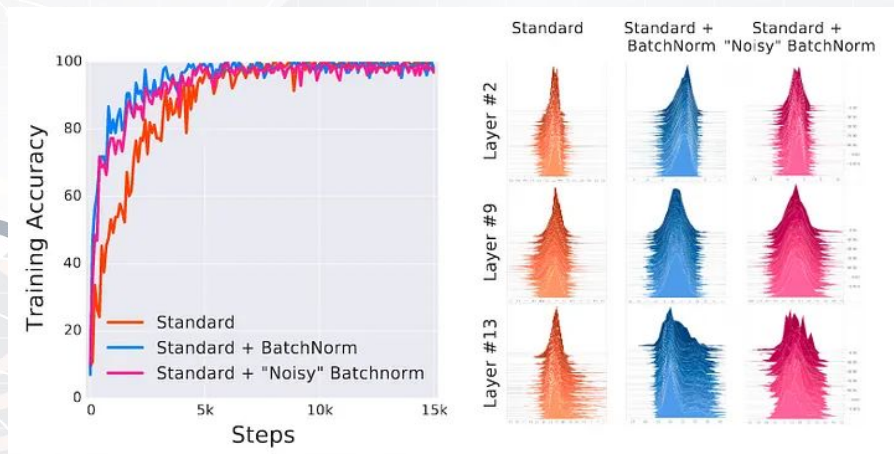


Fig 5: Networks with BN are trained faster than the standard one ; explicitly adding ICS_distrib on a regulated network does not deteriorate BN benefits. The trained model is a VGG networks (on CIFAR-10) . [2][3]

- The first network doesn't have any BN layer, while the second layer have BN layers.
- In the third one, they have explicitly added some ICS_distrib inside the hidden unit right before the activation (by adding random bias & variance).
- The noisy network is still trained faster than the standard one. Its reached performances are comparable to the ones obtained with a standard BN network.

Smoothness of the optimization landscape

- Normalization of the input signal inside hidden units , which makes the optimization landscape smoother.
- BN makes the optimization landscape smoother while preserving all of the minima of the normal landscape.

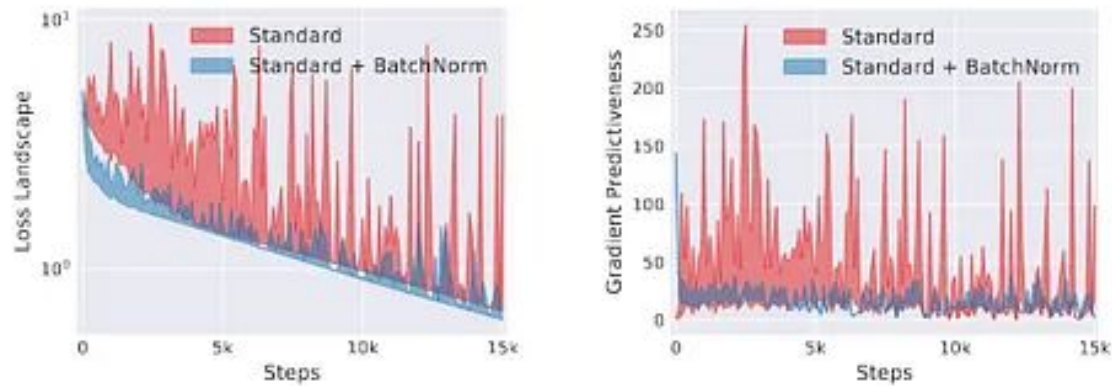


Fig 6 : Batch normalization impact on optimization landscape smoothing [2]

Interdependency between hidden layers

- From the fig.5 the gradient from the networks output:
$$\text{grad}(a) = \text{grad}(b) * \text{grad}(c) * \text{grad}(d) * \text{grad}(e)$$
- This interdependency is problematic for the input stability.
- Batch Normalization also has a beneficial effect on the gradient flow through the network, by reducing the dependence of gradients on the scale of the parameters or of their initial values. This allows for use of much higher learning rates without the risk of divergence.

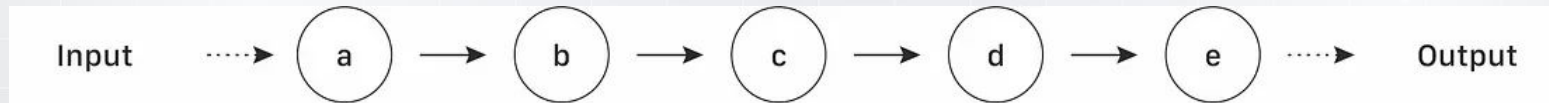


Fig 3: A simple DNN , which consists of linear transformation and (a), (b), (c), (d) and (e) are the sequential layers of the network. [3] [5]

Batch Normalization : Results

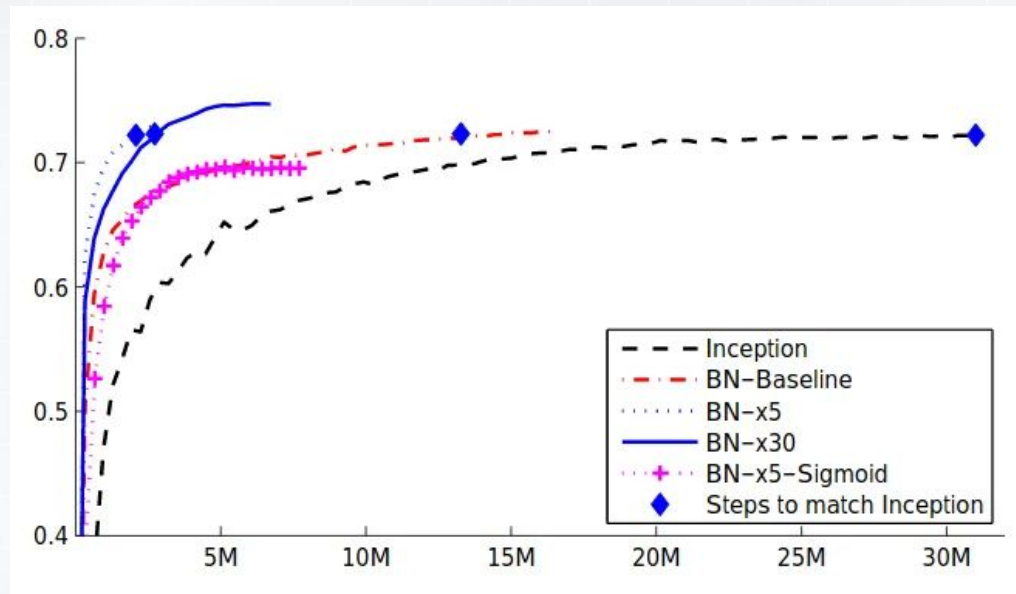
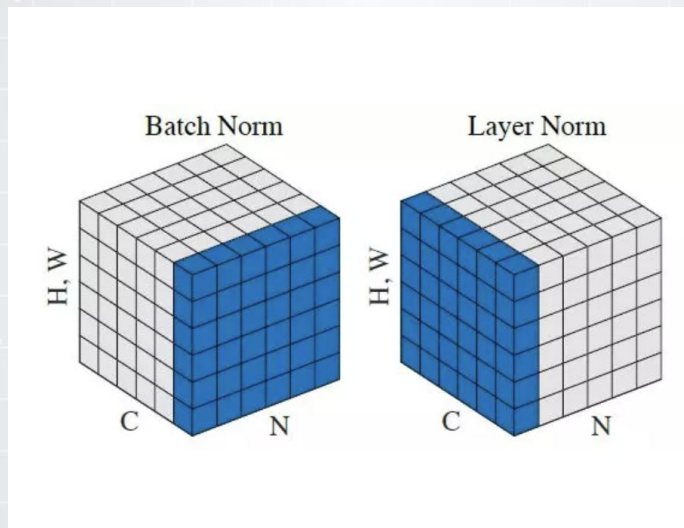


Fig : Batch normalization impact on training (ImageNet) : “Inception” : original network ; “BN-Baseline” : same as Inception with BN, same learning rate (LR) ; “BN-x5” : same as Inception with BN, LR x5 ; “BN-x30” : same as Inception with BN, LR x30 ; “BN-x5-Sigmoid” : same as Inception with BN, LR x5 and sigmoid instead of ReLU [1]

Limitations of Batch Normalization

- When using batch normalization, the sample mean and standard deviation from the current mini-batch may not be representative, particularly with small batch sizes. This can hinder the network from learning effectively. [9] [10]
- Batch normalization is less suitable for sequence models, especially when handling sequences of varying lengths and smaller batch sizes, as it relies on batch statistics for normalization. [6]

Why Layer Normalization ?



- Layer normalization is independent of the batch size, so it can be applied to batches with smaller sizes as well.
- Layer normalization has been shown to improve both the training time and the generalization performance of several existing RNN models.

Fig 1: A feature map tensor, with N as the batch axis, C as the channel axis, and (H, W) as the spatial axes. The pixels in blue are normalized by the same mean and variance.

What is Layer Normalization ?

- In Layer Normalization, all neurons within a layer share the same distribution across all input features.
- Here n = dimension of the input feature vector.
- The normalization is done along the length of the d dimensional vector and not the batch size.

$$\mu = \frac{1}{n} \sum_i Z^{(i)} \quad \text{-- (1)}$$

$$\sigma^2 = \frac{1}{n} \sum_i (Z^{(i)} - \mu)^2 \quad \text{-- (2)}$$

$$Z_{norm}^{(i)} = \frac{Z^{(i)} - \mu}{\sqrt{\sigma^2 - \epsilon}} \quad \text{-- (3)}$$

$$\hat{Z}^{(i)} = \gamma Z_{norm}^{(i)} + \beta \quad \text{-- (4)}$$

How Layer Normalization different from Batch Normalization ?

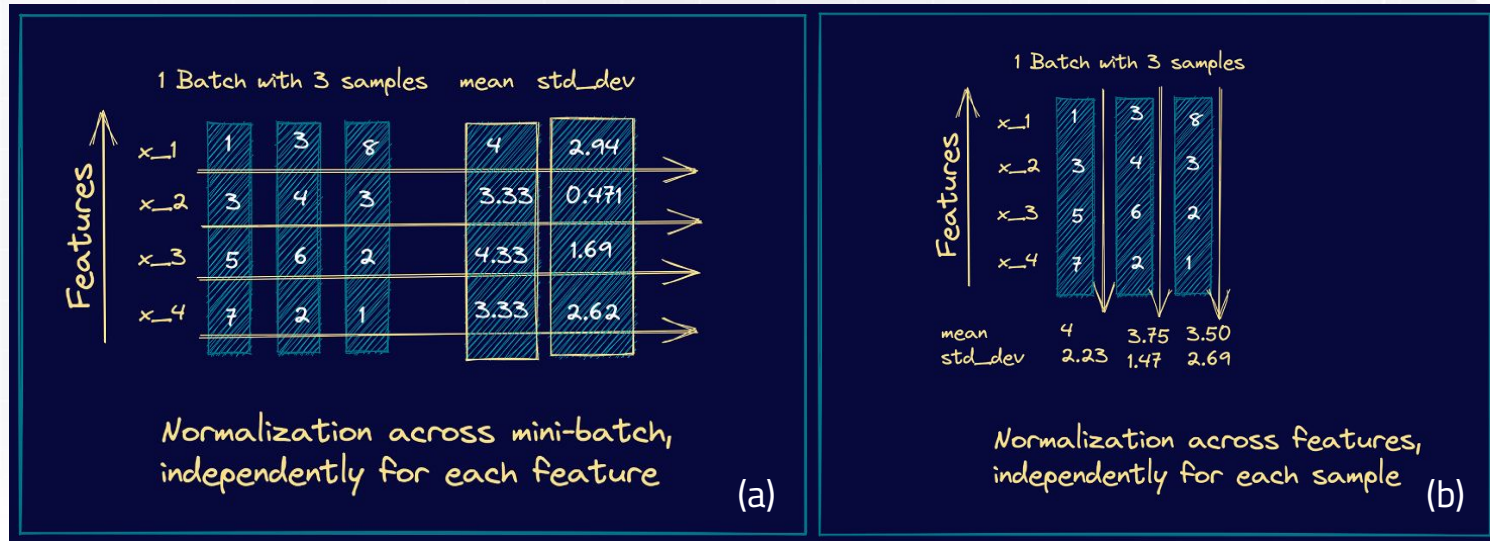


Fig 1: a) How Batch Normalization Works - An Example ,b) How Layer Normalization Works - An Example. [4]

Some other Normalizations

Instance Normalization

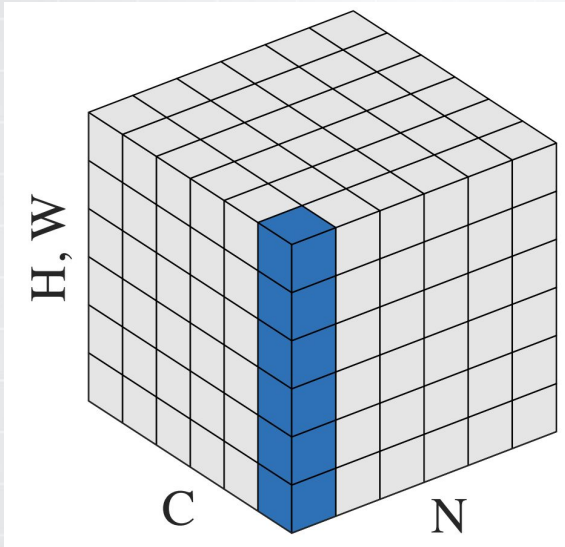


Fig 1: A feature map tensor, with N as the batch axis, C as the channel axis, and (H, W) as the spatial axes. The pixels in blue are normalized by the same mean and variance, computed across a single instance.

- Instance Normalization operates at the instance level, independently normalizing each sample (or instance) in a batch.
- It is frequently used in tasks related to style transfer, image-to-image translation, and generative models.
- It helps maintain the statistical properties of each instance, making it suitable for preserving the style or characteristics of individual images in these tasks.
- Instance Normalization is less dependent on batch statistics, which can be advantageous in scenarios where the batch size varies or is very small.

Group Normalization

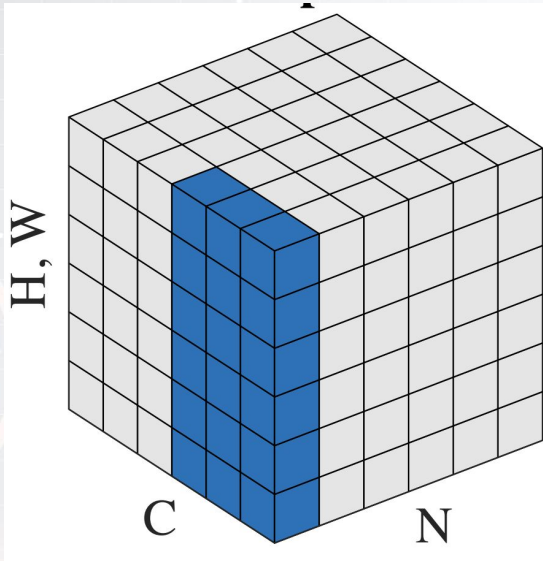


Fig 1: A feature map tensor, with N as the batch axis, C as the channel axis, and (H, W) as the spatial axes. The pixels in blue are normalized by the same mean and variance, computed across a multiple instances.

- Group Normalization divides the channels into groups and normalizes each group independently.
- It is useful when working with smaller batch sizes, as it reduces the dependence on batch statistics.
- Group Normalization is commonly applied in various computer vision tasks and is well-suited for convolutional neural networks (CNNs).

Comparison

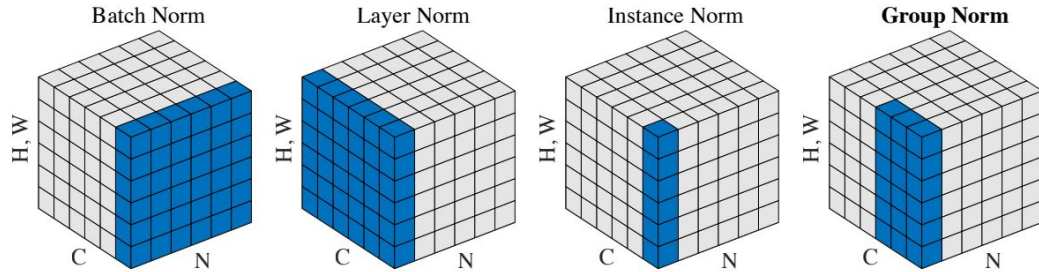
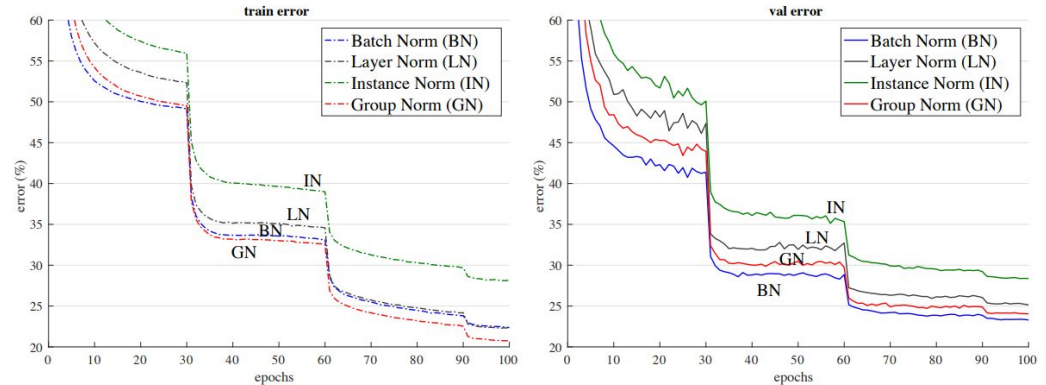


Fig 1: A feature map tensor, with N as the batch axis, C as the channel axis, and (H, W) as the spatial axes. The pixels in blue are normalized by the same mean and variance, computed by aggregating the values of these pixels. [8]

Fig 1: . Comparison of error curves with a batch size of 32 images/GPU for ResNet-50. Given aside is plot of the ImageNet training error (left) and validation error (right) vs. numbers of training epochs. [8]



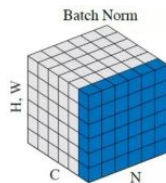
Some cool statistics!

Normalization

General • 38 methods







Edit

Normalization layers in deep learning are used to make optimization easier by smoothing the loss surface of the network. Below you will find a continuously updating list of normalization methods.

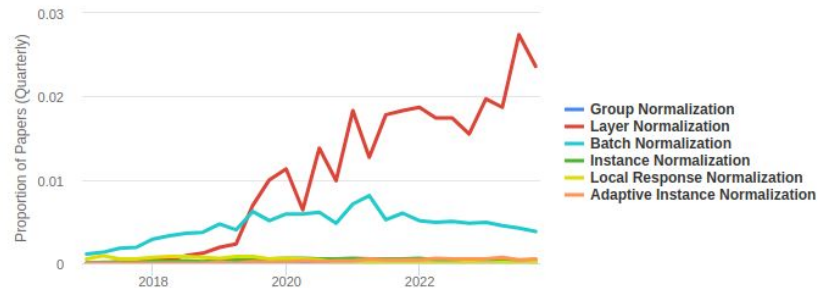


Methods

Add a Method

Method	Year	Papers
 Layer Normalization  Layer Normalization	2016	13958
 Batch Normalization  Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift	2015	5233
 Instance Normalization  Instance Normalization: The Missing Ingredient for Fast Stylization	2016	434

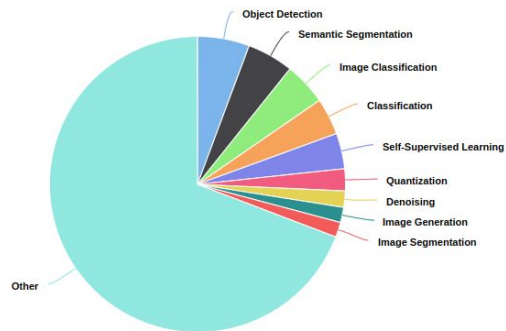
Usage Over Time



This feature is experimental; we are continuously improving our matching algorithm.

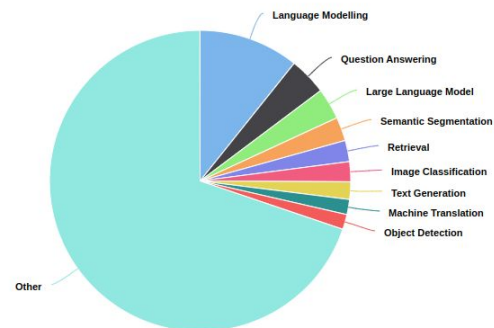
Fig.1. The snippet of the page on normalization on paperswithcode.com and the one above shows the usage of various normalization techniques over time [7]

Batch Normalization



Task	Papers	Share
Object Detection	38	5.68%
Semantic Segmentation	34	5.08%
Image Classification	31	4.63%
Classification	27	4.04%
Self-Supervised Learning	26	3.89%
Quantization	16	2.39%
Denoising	12	1.79%
Image Generation	11	1.64%
Image Segmentation	11	1.64%

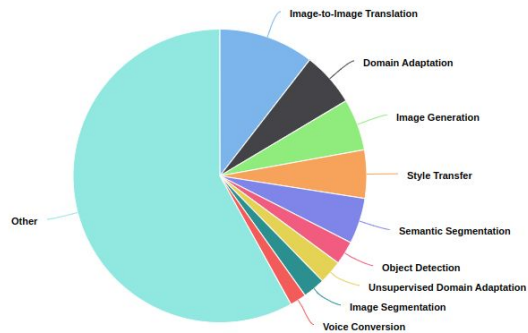
Layer Normalization



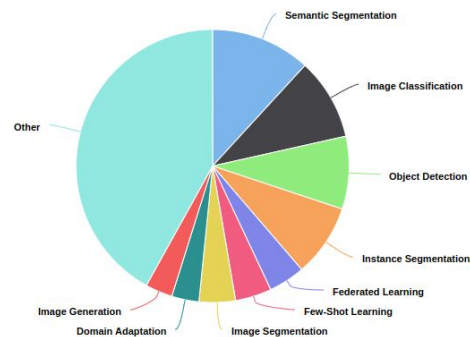
Task	Papers	Share
Language Modelling	80	10.74%
Question Answering	30	4.03%
Large Language Model	25	3.36%
Semantic Segmentation	19	2.55%
Retrieval	17	2.28%
Image Classification	16	2.15%
Text Generation	14	1.88%
Machine Translation	12	1.61%
Object Detection	12	1.61%

Instance Normalization

Group Normalization



Task	Papers	Share
Image-to-Image Translation	81	10.49%
Domain Adaptation	46	5.96%
Image Generation	44	5.70%
Style Transfer	41	5.31%
Semantic Segmentation	39	5.05%
Object Detection	20	2.59%
Unsupervised Domain Adaptation	20	2.59%
Image Segmentation	19	2.46%
Voice Conversion	14	1.81%



Task	Papers	Share
Semantic Segmentation	11	11.83%
Image Classification	9	9.68%
Object Detection	8	8.60%
Instance Segmentation	8	8.60%
Federated Learning	4	4.30%
Few-Shot Learning	4	4.30%
Image Segmentation	4	4.30%
Domain Adaptation	3	3.23%
Image Generation	3	3.23%

CONCLUSION

BATCH NORMALIZATION

- Normalizes each feature independently across the mini-batch.
- Since it is dependent on batch size, it's not effective for small batch sizes.
- It requires different processing at training and inference times.

LAYER NORMALIZATION

- Normalizes each of the inputs in the batch independently across all features.
- It is independent of the batch size, so it can be applied to batches with smaller sizes as well.
- The same set of operations can be used at both training and inference times.

References

- [1] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167.
- [2] Santurkar, S., Tsipras, D., Ilyas, A., & Madry, A. (2018). How does batch normalization help optimization?, Advances in Neural Information Processing Systems .
- [3] Johann Huber(2020), Batch normalization in 3 levels of understanding, Towards Data Science.
- [4] Bala Priya C, Build Better Deep Learning Models with Batch and Layer Normalization, pinecone.io
- [5] Ian Goodfellow, Ch 9: Convolutional Networks, Youtube.
- [6] Jimmy Lei Ba, Jamie Ryan Kiros, Geoffrey E. Hinton(2016), Layer Normalization, arXiv preprint arXiv:1607.06450.
- [7] Normalization, paperswithcode.com
- [8] Yuxin Wu, Kaiming He (2018), Group Normalization, arXiv:1803.08494v3
- [9] Elad Hoffer, , Itay Hubara, , Daniel Soudry (2018) ,Train longer, generalize better: closing the generalization gap in large batch training of neural networks, arXiv:1705.08741v2
- [10] Cecilia Summers, Michael J. Dinneen (2020), Four Things Everyone should know to improve Batch Normalization, arXiv:1906.03548v2