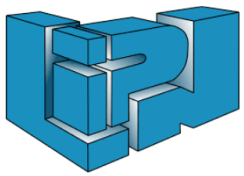
# Normalisation Contextuelle : Nouvelle Approche pour la Stabilité et l'Amélioration des Performances des Réseaux de Neurones

*Context Normalization: New Approach for the Stability and Improvement of Neural Network Performance*

Bilal FAYE, Hanane AZZAG, Mustapha LEBBAH, Fangchen FENG

faye@lipn.univ-paris13.fr





# Table of contents

# Introduction to Normalization

## Foundation

For a given set of samples $X \in \mathbb{R}^{n \times d}$, the normalization operation, represented by the function $\phi : x \to \hat{x}$, is employed to guarantee that the transformed data $\hat{X}$ exhibits specific desired statistical properties.

# Normalization Techniques

There are several normalization techniques ($\phi$) :

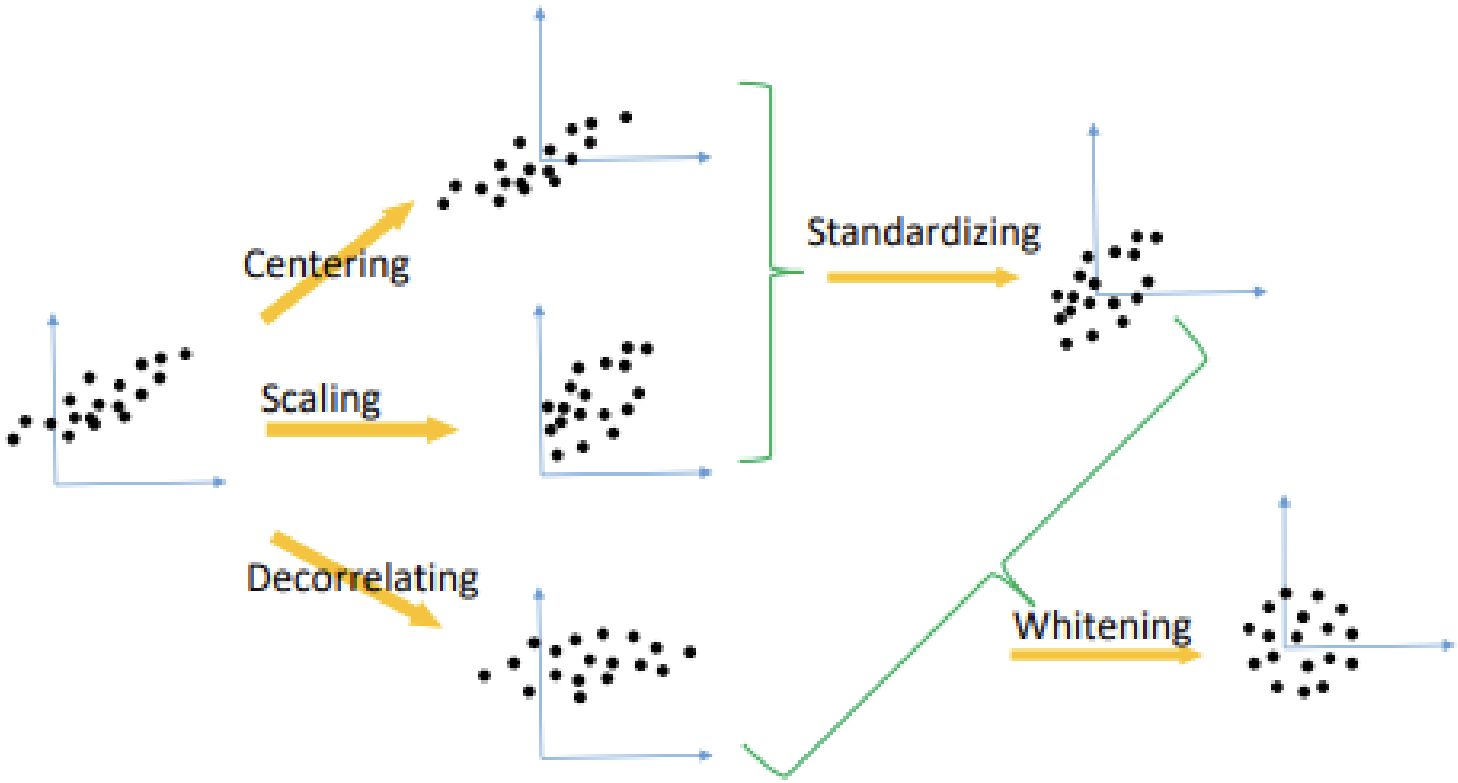
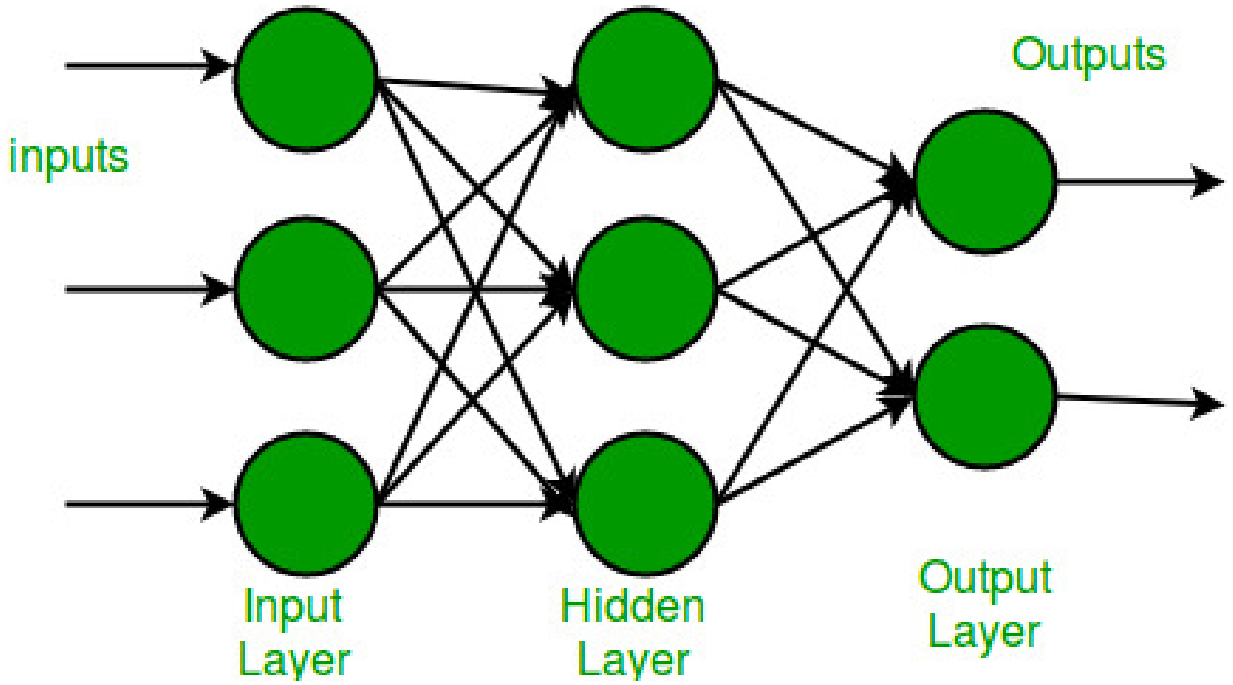



Figure – Various Normalization Methods

# Normalization in Single-Layer Networks

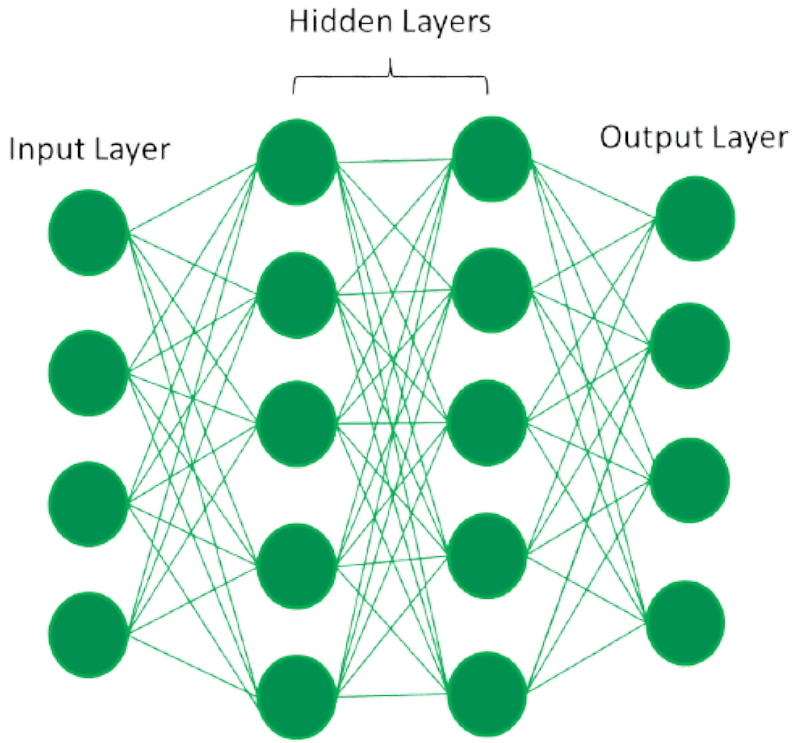
Normalization is instrumental in equalizing variable amplitudes, enhancing the convergence of single-layer networks [1]




1. LeCun et al. Efficient Backpropagation, 1998. In Neural networks

# Normalization in Multi-Layer Networks

In a multi-layer neural network, since the input is connected only to the first layer, the hidden layers may not necessarily benefit from input normalization.



# Normalization Techniques in Training

To achieve similar benefits to normalizing inputs, it's crucial to normalize activations during training.

Various normalization techniques include :

- Activation Normalization : Normalizing the output of each layer's activation function.
- Weight Normalization : Normalizing the weights within a neural network layer.
- Gradient Normalization : Normalizing the gradients during the backpropagation process.

# Batch Normalization (BN)

To normalize activations, the most common technique is Batch Normalization (BN) [2].

## Batch Normalizing Technique

In Batch Normalization (BN), a mini-batch of samples denoted as $B$ is normalized for each sample $x$ in $B$ using the following formula :

$$\hat{x} = \frac{x - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}, \ \tilde{x} = \gamma.\hat{x} + \beta \tag{1}$$

where $\mu_B$ and $\sigma_B^2$ represent the mean and variance of $B$, respectively, $\epsilon > 0$ a small value that handles numerical instabilities, and $\gamma$, $\beta$ a learnable parameters.

2. Ioffe et al. Batch normalization : Accelerating deep network training by reducing internal covariate shift, 2015. In International conference on machine learning (ICML).

# Limitations of Batch Normalization (BN)

**Some limitations of BN :**

- Performance dependency on batch size.
- Assumption that samples within the mini-batch are from the same distribution.

# Addressing Batch Size Dependencies

**To mitigate batch size dependencies, several methods have been proposed :**

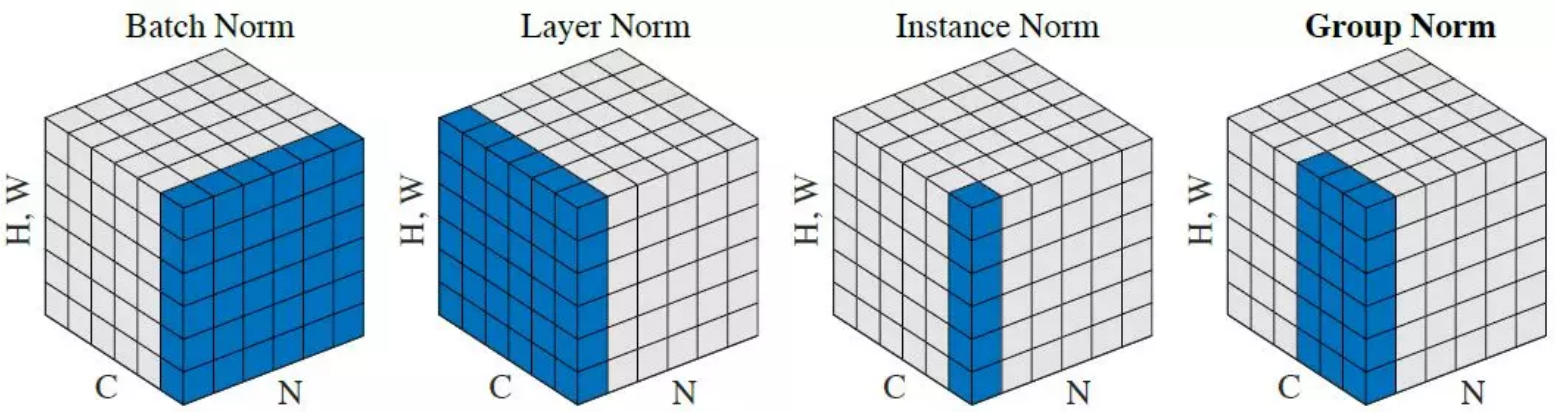



Figure – Variants of BN addressing batch size challenges : LN [3], IN [4], GN [5]

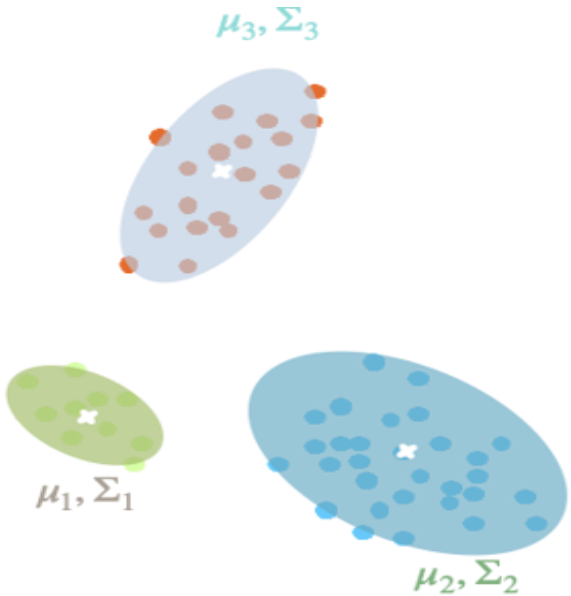
3. Ba et al., Layer Normalization, 2016. In arXiv preprint arXiv :1607.06450.

4. Ulyanov et al. Instance Normalization : The Missing Ingredient for Fast Stylization, 2016. In arXiv preprint arXiv :1607.08022.

5. Wu et al. Group Normalization, 2018. In Proceedings of the European Conference on Computer Vision (ECCV).

# Advancements in Addressing Batch Size Distribution Hypotheses

To address hypotheses related to batch size distribution, methods like **Mixture Normalization (MN)** [6] have been proposed.




6. Kalayeh et al. Training faster by separating modes of variation in batch-normalized models, 2019. In IEEE transactions on pattern analysis and machine intelligence.

# Current Challenges in Mixture Normalization (MN)

**Limitations of MN :**

- The utilization of the computationally expensive EM algorithm.
- Activation normalization independent of the target task, relying on constant parameters.

**Solution : Context Normalization (CN).**

# Introduction to CN

CN, can be summarized in three key concepts :

- Hypothesis : Samples within the mini-batch may not be from the same distribution.
- Introduction of the concept of "context" : Grouping data with similar characteristics (clustering).
- Normalization of activations from the same context using learned parameters during deep neural network backpropagation.

# CN Layer applied to a given activation $x_i$.

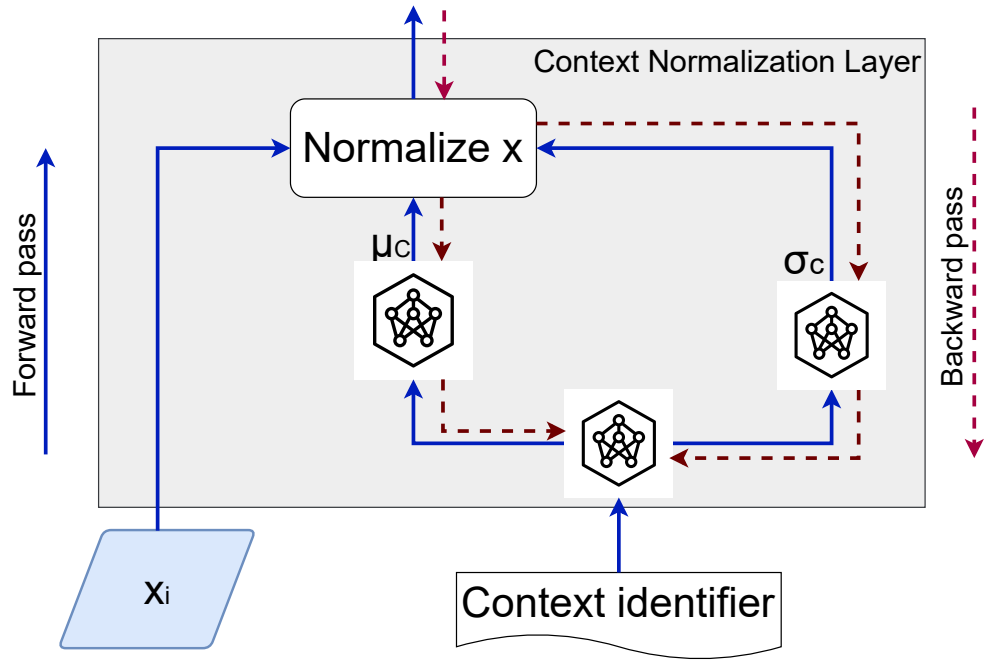



Figure – The context identifier ($c$) is encoded by a neural network. The output is then used as input to two different neural networks, generating a mean ($\mu_c$) and a standard deviation ($\sigma_c$) for normalizing $x_i$ ($\frac{x_i - \mu_c}{\sqrt{\sigma_c^2 + \epsilon}}$).

# Contributions of CN

**Key Contributions :**

- Elimination of the costly EM algorithm through innovative use of the context concept.
- Improved model convergence and performance by estimating parameters for each context during backpropagation, tailoring activation representation to the target task.

# Applications of CN in classification task

- CN vs. BN and MN on shallow Convolutional Neural Network (CNN).
- CN in domain adaptation.

*First application : CN vs. BN and MN on shallow Convolutional Neural Network (CNN).*

# CN vs. BN and MN on Shallow CNN

- Evaluation conducted on three datasets : CIFAR-10, CIFAR-100 [7] and Tiny ImageNet [8].
- EM algorithm applied with three components ($k = 3$) for each dataset.
- Components obtained with EM are used in the MN layer and serve as contexts in the CN layer.

7. Krizhevsky, A. Canadian Institute for Advanced Research, 2009. In Learning Multiple Layers of Features from Tiny Images

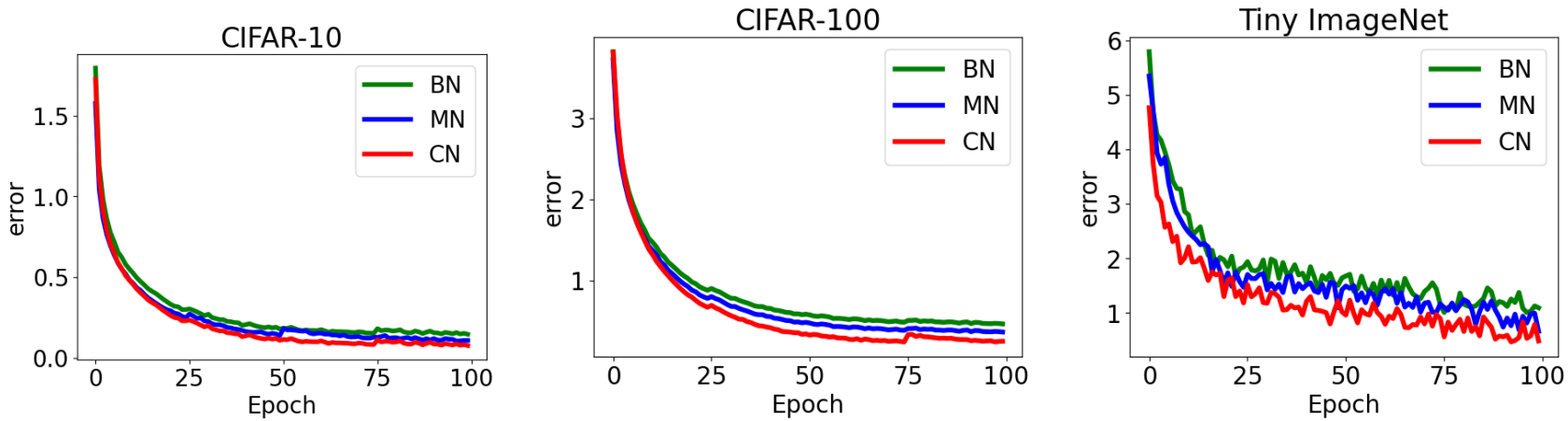
8. Deng et al. Tiny ImageNet, 2015. In ImageNet Large Scale Visual Recognition Challenge.

# Architecture of the Shallow CNN

| layer | type | size | kernel | (stride, pad) |
|---|---|---|---|---|
| input | input | $3 \times 32 \times 32$ | _ | _ |
| conv1 | conv+bn+relu | $64 \times 32 \times 32$ | $5 \times 5$ | (1, 2) |
| pool1 | max pool | $64 \times 16 \times 16$ | $3 \times 3$ | (2, 0) |
| conv2 | conv+bn+relu | $128 \times 16 \times 16$ | $5 \times 5$ | (1, 2) |
| pool2 | max pool | $128 \times 8 \times 8$ | $3 \times 3$ | (2, 0) |
| conv3 | conv+bn+relu | $128 \times 8 \times 8$ | $5 \times 5$ | (1, 2) |
| pool3 | max pool | $128 \times 4 \times 4$ | $3 \times 3$ | (2, 0) |
| conv4 | conv+bn+relu | $256 \times 4 \times 4$ | $3 \times 3$ | (1, 1) |
| pool4 | avg pool | $256 \times 1 \times 1$ | $4 \times 4$ | (1, 0) |
| linear | linear | 10 or 100 | _ | _ |

Table – Shallow Convolutional Neural Network

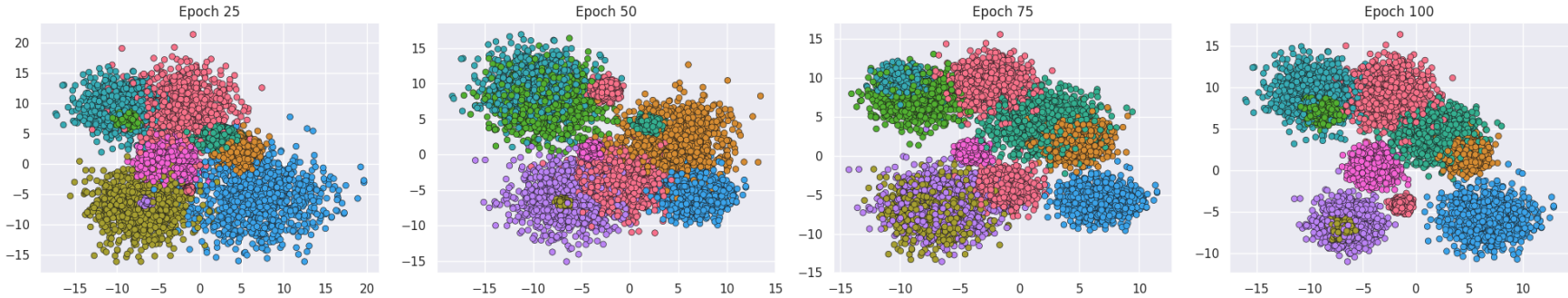
# Error Evolution during Training




(a) CIFAR-10 (b) CIFAR-100 (c) Tiny ImageNet

Figure – Validation Error Evolution during Shallow CNN Training.

# Visualization of Random Minibatch Activation during Shallow CNN Training (CIFAR-10)



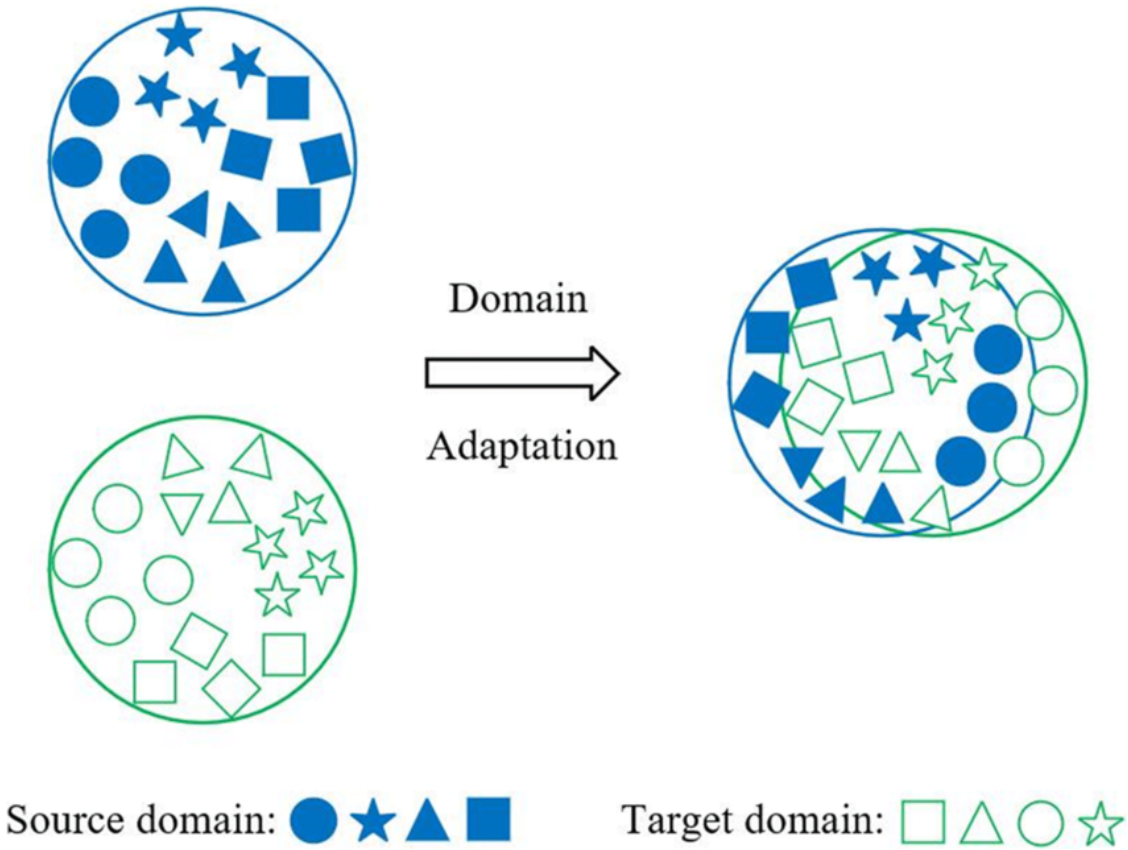
# CN vs. BN and MN on Shallow CNN

In this experiment, we observe that :

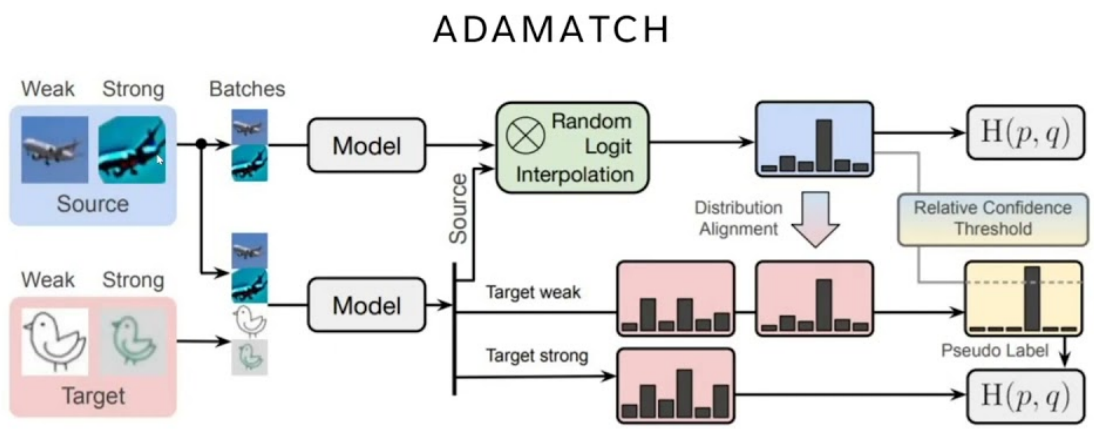
- CN accelerates model convergence more effectively compared to BN and MN.
- CN results in improved accuracy, showcasing an average enhancement of **2%** on CIFAR-10, **3%** on CIFAR-100 and **4%** on Tiny ImageNet.

## Second Application : CN on Domain Adaptation

# Definition of Domain Adaptation



# CN in Domain Adaptation

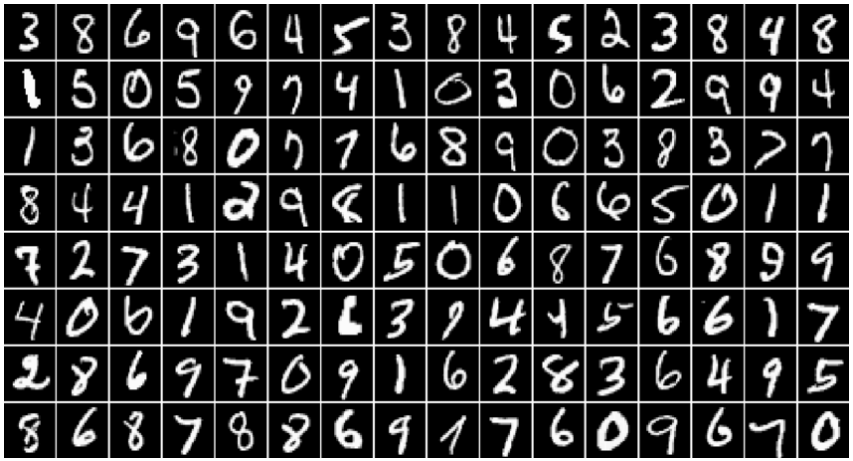



| Task | Labeled | Unlabeled | Distributions |
|---|---|---|---|
| SSL | source | target | source = target |
| UDA | source | target | source ≠ target |
| SSDA | source+target | target | source ≠ target |

Figure – AdaMatch [9] architecture.

9. Berthelot et al. Adamatch : A unified approach to semi-supervised learning and domain adaptation, 2021. In arXiv preprint arXiv :2106.04732.

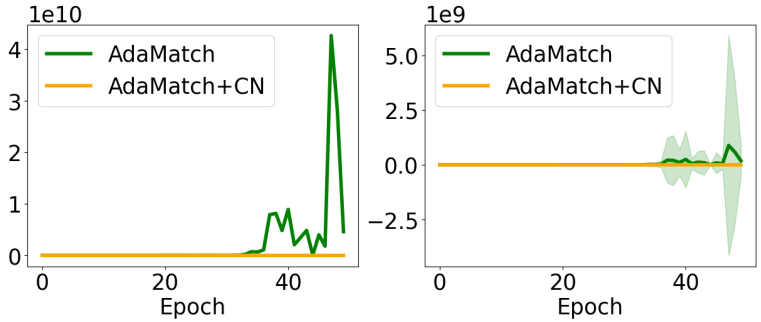
# Dataset Visualization : MNIST and SVHN



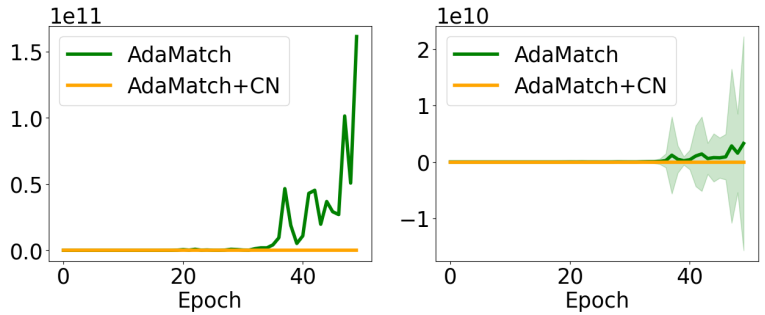
(a) MNIST Digit Samples

(b) Street View House Numbers (SVHN) Dataset Samples

# Context Normalization Impact on Domain Adaptation




(a) Source Domain (MNIST)




(b) Target Domain (SVHN)

Figure – Evolution of Gradient Variance : Comparison between AdaMatch and AdaMatch+CN models during training on the source (MNIST) and target (SVHN) domains. Left : Maximum gradient variance per epoch. Right : Average gradient variance per epoch.

# Impact of Context Normalization on Domain Adaptation : Test Accuracy Comparison

| Model | Source Data (MNIST) | Target Data (SVHN) |
|---|---|---|
| AdaMatch | 79.39% | 20.46% |
| **AdaMatch+CN-Channels** | **99.21%** | **43.80%** |

Table – Test accuracy comparison of AdaMatch and AdaMatch with context normalization (AdaMatch+CN), using the source domain (MNIST) as a context identifier.

# Conclusion

- CN overcomes these limitations by grouping similar observations into "contexts" for local representation without the need for context construction algorithms.

- Normalization parameters are learned similarly to model weights, ensuring speed, convergence, and superior performance compared to BN and MN.

# Future Work

- Enhance the versatility of CN by adapting it to different data types, allowing for a broader range of applications.
- Developing an unsupervised variant of CN, eliminating context as an input and allowing dynamic acquisition during training, potentially broadening its applicability.

Thank you for your attention.