Unsupervised Adaptive Normalization

Bilal FAYE, Hanane AZZAG, Mustapha LEBBAH, Fangchen FENG

faye@lipn.univ-paris13.fr

Table of contents

What is Normalization ?

Foundation

For a given set of samples (or activations) $X \in \mathbb{R}^{N\times C\times H\times W}$, 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

- **Centering** : $\hat{X} = X - \mu_X \implies \mu_{\hat{X}} = 0.$
- **Scaling** : $\hat{X} = \frac{X}{\sigma_X} \implies \sigma_{\hat{X}} = 1.$
- **Standardizing** : $\hat{X} = \frac{X - \mu_X}{\sigma_X} \implies \mu_{\hat{X}} = 0, \sigma_{\hat{X}} = 1.$
- **Decorrelating** : $\hat{X} = DX \implies \Sigma_{\hat{X}}$ is diagonal.
- **Whitening** : $\hat{X} = \hat{\Lambda}^{1/2} DX \implies \Sigma_{\hat{X}} = I.$

Σ : Covariance matrix ; D : Eigenvectors matrix ; Λ : Eigenvalues matrix.

Normalization in Neural Networks

To stabilize the training of neural networks, several normalization strategies are employed :

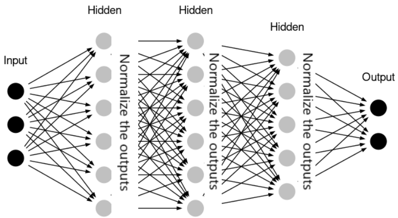
- **Activation Normalization (AN)**
- Weight Normalization
- Gradient Normalization

This presentation will focus on AN.

Normalization in Neural Networks

AN enhances neural network **stability** and **performance** by normalizing neuron activations during training. There are two main methods of AN :

- Single Mode Normalization
- Multiple Mode Normalization



Single Mode Normalization

Definition

Given a mini-batch of activations $X \in \mathbf{R}^{N \times H \times W \times C}$, single mode normalization (SMN) normalizes all activations in the mini-batch using the **same parameters**.

The pioneer of SMN is **Batch Normalization (BN)** [1].

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

BN normalization technique

$$\hat{X} = \gamma(\frac{X - \mu}{\sqrt{\sigma^2 + \epsilon}}) + \beta$$

γ and β are learnable parameters. μ and σ^2 represent the mean and variance, respectively. ϵ is a small constant added to prevent division by zero.

Limitations of SMN

SMN methods demonstrate effectiveness in certain scenarios but has limitations :

- Using a single mean and variance is inaccurate for heterogeneous data.
- SMN methods perform poorly with small mini-batches.

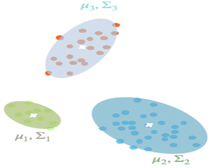
To address these issues, an alternative AN strategy is employed : Multiple Modes Normalization.

Multiple Modes Normalization

Definition

Given a mini-batch of activations $X \in \mathbf{R}^{N \times H \times W \times C}$, multiple mode normalization (MMN) normalizes all activations in the mini-batch using **multiple parameters**.

One existing method used MMN approach is **Mixture Normalization (MN)** [2].



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

MN normalization technique

Using Gaussian distribution hypothesis, each x_n in X is normalized as follow :

$$\hat{x}_n = \gamma(\sum_{k=1}^{K} \frac{p(k|x_n)}{\sqrt{\lambda_k}} . \frac{x_n - \alpha_k}{\sqrt{\delta_k^2 + \epsilon}}) + \beta$$

where

$$\alpha_k = \sum_n \frac{p(k|x_n)}{\sum_j p(j|x_n)} \cdot x_n$$

and

$$\delta_k^2 = \sum_n \frac{p(k|x_n)}{\sum_j p(j|x_n)} \cdot (x_n - \alpha_k)^2$$

MN normalization technique

$$p(k|x_n) = \frac{\lambda_k p(x_n|k)}{\sum_{j=1}^{K} \lambda_j p(x_n|j)}$$

$p(x_n|k)$ represents the density function of the Gaussian distribution.

To estimate the parameters of this density function ($\{\lambda_k, \mu_k, \sigma_k^2\}_{k=1}^{K}$), MN utilizes the Expectation-Maximization (EM) algorithm during the training process.

Limitations of MMN

MMN methods improve upon SMN for heterogeneous data but have drawbacks :

- Algorithms like EM add significant computational cost.
- Estimating normalized parameters less frequently due to cost can impact the normalization process.

Solution : Unsupervised Adaptive Normalization.

Unsupervised Adaptive Normalization (UAN)

Unsupervised Adaptive Normalization (UAN)

UAN normalization strategy :

- Activations are normalized across multiple modes.
- Parameters for each mode are learned as neural network weights during backpropagation.

UAN normalization technique

During training :

$$\hat{x}_n = \gamma(\sum_{k=1}^{K} \frac{p(k|x_n)}{\sqrt{\lambda_k}} . \frac{x_n - \alpha_k}{\sqrt{\delta_k^2 + \epsilon}}) + \beta$$

The parameters $(\lambda_k, \mu_k, \sigma_k^2)$ are learned as neural network weights during backpropagation, with constraints $\sum \lambda_k = 1$ and $\sigma_k^2 \geq 0$ on each update.

UAN normalization technique

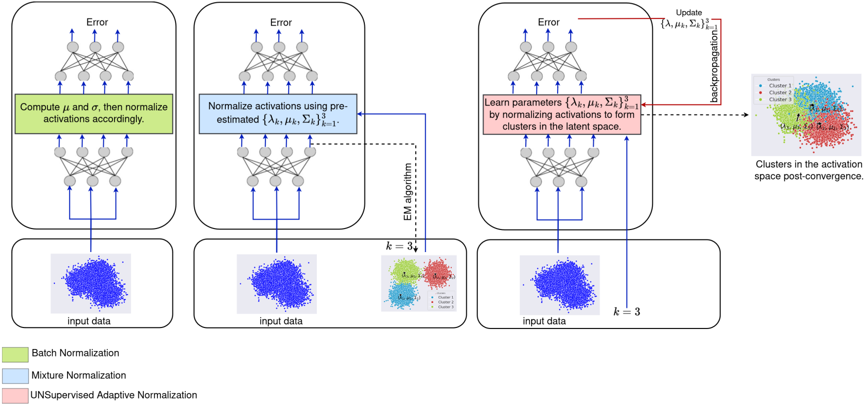
During inference :

$$\hat{x}_n = \gamma(\sum_{k=1}^{K} \frac{p(k|x_n)}{\sqrt{\lambda_k}} . \frac{x_n - \mu_k}{\sqrt{\sigma_k^2 + \epsilon}}) + \beta$$

UAN vs. MN

Compared to MN, UAN give 2 advantages :

- UAN doesn't depend to the costly EM algorithm for mixture component parameters estimation.
- Updating mixture component parameters as weights allow to have parameters that describe more the activations.


Error
Compute μ and σ, then normalize activations accordingly.
input data
Error
Normalize activations using pre-estimated $\{\lambda_k, \mu_k, \Sigma_k\}_{k=1}^{3}$.
EM algorithm
$k = 3$
input data
Error
Update $\{\lambda, \mu_k, \Sigma_k\}_{k=1}^{3}$
backpropagation
Learn parameters $\{\lambda_k, \mu_k, \Sigma_k\}_{k=1}^{3}$ by normalizing activations to form clusters in the latent space.
input data
$k = 3$
Cluster 1
Cluster 2
Cluster 3
Clusters in the activation space post-convergence.
Batch Normalization
Mixture Normalization
UNSupervised Adaptive Normalization

Experiments

Experiments

Experiments

- UAN IN A SIMPLIFIED SCENARIO
- UAN in DOMAIN ADAPTATION

Experiments

UAN IN A SIMPLIFIED SCENARIO

UAN IN A SIMPLIFIED SCENARIO

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

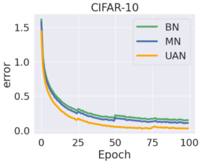
Table – Shallow Convolutional Neural Network

UAN IN A SIMPLIFIED SCENARIO

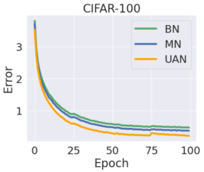
- Evaluation conducted on three datasets : CIFAR-10, CIFAR-100 [3] and Tiny ImageNet [4].
- EM algorithm applied with three components ($k = 3$) for each dataset.
- To ensure a fair comparison, we utilize $k = 3$ in UAN.

3. Krizhevsky, A. Canadian Institute for Advanced Research, 2009. In Learning Multiple Layers of Features from Tiny Images
4. Deng et al. Tiny ImageNet, 2015. In ImageNet Large Scale Visual Recognition Challenge.

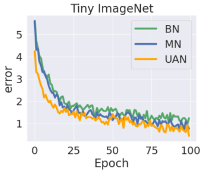
UAN IN A SIMPLIFIED SCENARIO



(a) CIFAR-10

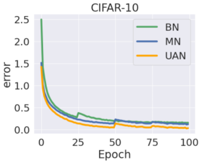


(b) CIFAR-100

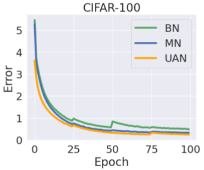


(c) Tiny ImageNet

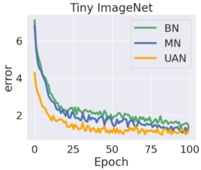
Learning rate = 0.001



(d) CIFAR-10



(e) CIFAR-100



(f) Tiny ImageNet

Learning rate = 0.005

Figure – Validation Error Evolution during Shallow CNN Training.

UAN IN A SIMPLIFIED SCENARIO

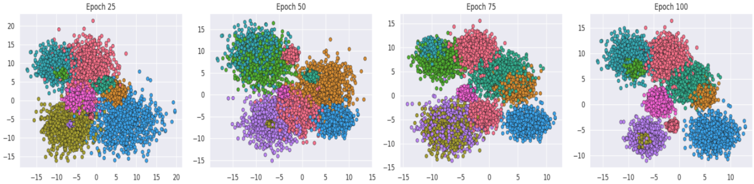


Figure – Visualization of Random Mini-batch Activation during Shallow CNN Training (CIFAR-10).

UAN IN A SIMPLIFIED SCENARIO

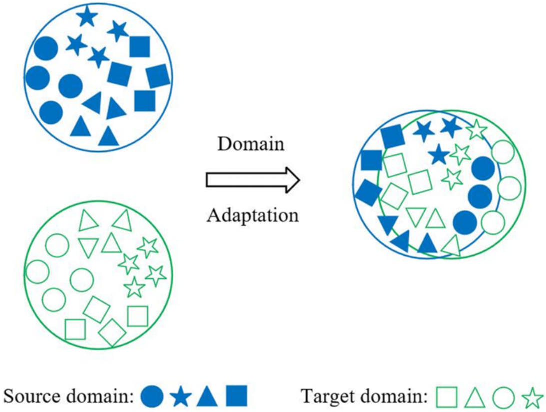
In this experiment, we observe that :

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

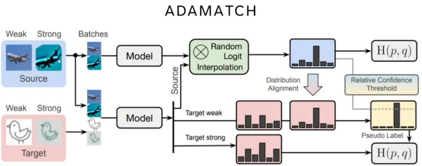
Experiments

UAN IN A Domain Adaptation

UAN IN A Domain Adaptation



UAN IN A Domain Adaptation

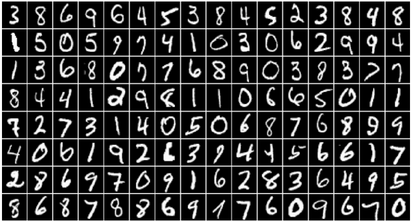


Task	Labeled	Unlabeled	Distributions
SSL	source	target	source = target
UDA	source	target	source ≠ target
SSDA	source+target	target	source ≠ target

Figure – AdaMatch [5] architecture.

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

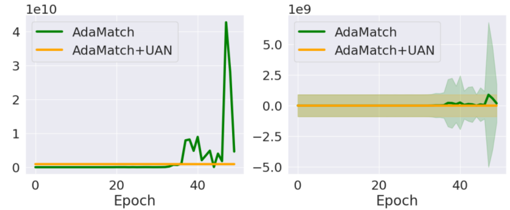
UAN IN A Domain Adaptation



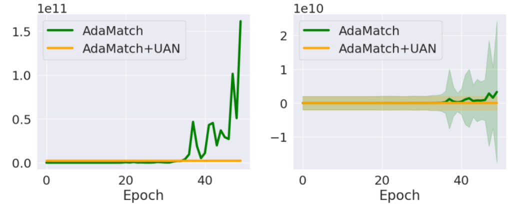
(a) MNIST Digit Samples

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

UAN IN A Domain Adaptation



(a) Source domain (MNIST)



(b) Target domain (SVHN)

Figure – Evolution of Gradient Variance : Comparison between

UAN IN A Domain Adaptation

Model	Source Data (MNIST)	Target Data (SVHN)
AdaMatch	97.36%	25.08%
AdaMatch+UAN	**98.9%**	**33.4%**

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

Conclusion and Future Works

Conclusion

- We propose a new versatile normalization method : UAN.
- UAN is a multiple modes normalization method that can be used as a layer in neural networks.
- UAN is less costly compared to existing multiple modes normalization methods.
- Estimating mode parameters as neural network weights leads to better representation.

Thank you for your attention.