Context Normalization Layer with Applications

* Bilal FAYE - Hanane AZZAG - Mustapha Lebbah - Mohamed-Djallel DILMI - Djamel BOUCHAFFRA

* faye@lipn.univ-paris13.fr

1er décembre 2023

Table of contents

State of art

State of art

Concept

Given a set of samples $X \in \mathbb{R}^{n \times d}$, the normalization operation is a function $\phi : x \to \hat{x}$ which ensures that the transformed data $\hat{X}$ has certain statistical properties.

State of art

There are several normalization techniques (ϕ) :

- Centering : $\hat{X}$ has zero-mean property.
- Scaling : $\hat{X}$ has a unit-variance property.
- Standardizing : $\hat{X}$ has a zero-mean and unit-variance property.
- Decorrelating : $\Sigma_{\hat{X}}$ is diagonal matrix.
- Whitening : $\Sigma_{\hat{X}} = I$, identity matrix.

State of art

Normalization can equalizes variable amplitudes, aiding single-layer network convergence (LeCun et al., Efficient backprop) [8].

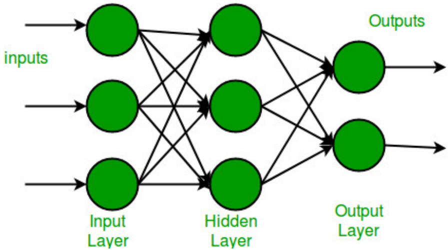


Figure – Single-Layer Perceptron

State of art

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.

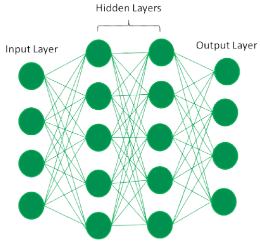


Figure – Multi-Layer Perceptron

State of art

From this perspective, it is important to normalize activations during training, to obtain similar benefits of normalizing inputs.

Different normalization techniques can be used :

- activation normalization
- weight normalization
- gradient normalization

State of art

To normalize activations, the most common technique is Batch Normalization (BN) [4].

Batch Normalizing Technique

Let's take a mini-batch of samples denoted as B. Batch Normalization (BN) normalizes each sample x in B as follows :

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

where μ_B and σ_B^2 represent, respectively, the mean and variance of B, while $\epsilon > 0$ is a small value that handles numerical instabilities.

State of art

Some limits of BN :

- the performance depends on the batch size.
- samples within the mini-batch are from the same distribution.

State of art

To address the batch size dependencies, several methods are proposed :

- Layer Normalizatin (LN) [1]
- Instance Normalization (IN) [9]
- Group Normalization (GN) [10]
- etc.

State of art

To address the hypothesis related to batch size distribution, some methods, such as **Mixture Normalization (MN)** [5], have been proposed.

MN employs a **Gaussian Mixture Model (GMM)** to assign each mini-batch sample, normalizing with respect to multiple means and standard deviations associated with different modes of variation in the data distribution.

State of art

MN concept

Let $x \in \mathbb{R}^D$, and $\theta = \{\lambda_k, \mu_k, \Sigma_k : k = 1, ..., K\}$, then we have : $p(x) = \sum_{k=1}^{K} \lambda_k p(x|k)$, s.t. $\forall_k \ : \ \lambda_k \ \geq 0, \ \sum_{k=1}^{K} \lambda_k = 1$, where $p(x|k) = \frac{1}{(2\pi)^{D/2}|\Sigma_k|^{1/2}} \exp\left(-\frac{(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)}{2}\right)$, is the k-th component, μ_k is the mean vector, and Σ_k is the covariance matrix.

The probability that x was generated by the k-th Gaussian component can be defined as follows : $\tau_k(x) = p(k|x) = \frac{\lambda_k p(x|k)}{\sum_{j=1}^{K} \lambda_j p(x|j)}$.

State of art

MN concept

MN normalizes each x_i as follow :

$$\hat{x}_i = \sum_{k=1}^{K} \frac{\tau_k(x_i)}{\sqrt{\lambda_k}} \hat{x}_i^k, \tag{2}$$

with

$$v_i^k = x_i - \mathbb{E}_{B_i}[\hat{\tau}_k(x).x], \; \hat{x}_i^k = \frac{v_i^k}{\sqrt{\mathbb{E}_{B_i}[\hat{\tau}_k(x).(v^k)^2] + \epsilon}}, \tag{3}$$

where $\hat{\tau}_k(x_i) = \frac{\tau_k(x_i)}{\sum_{j \in B_i} \tau_k(x_j)}$, is the normalized contribution of x_i in estimating the statistics of the k-th Gaussian component.

State of art

MN limitations :

- The use of EM algorithm that is too costly.
- Activation normalization that does not depend on the target task, with constant parameters.

To address these issues, we propose another normalization method called **Context Normalization (CN)**.

Our proposed normalization method (CN)

Our proposed normalization method (CN)

CN can be summarized in three main concepts :

- Introduction of the concept of 'context', which involves grouping data with similar characteristics.
- Hypothesis : Samples within the mini-batch are not necessarily from the same distribution.
- Activations from the same context are normalized using parameters that are learned during deep neural network backpropagation.

Our proposed normalization method (CN)

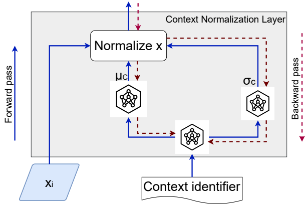


Figure – Context Normalization Layer applied to a given activation x_i. The context identifier (r) is encoded by a neural network, the output of which is then used as input to two different neural networks to generate a mean (μ_r) and a standard deviation (σ_r), respectively, for normalizing x_i ($\frac{x_i - \mu_r}{\sqrt{\sigma_r^2 + \epsilon}}$).

Our proposed normalization method (CN)

CN contributions :

- The use of the context concept eliminates the costly EM algorithm.
- The estimation of parameters for each context through normalization during the backpropagation phase enables the representation of activations according to the target task, contributing to improved model convergence and performance.

Some applications of Context Normalization

Some applications of Context Normalization

- Context Normalization vs. Mixture Normalization on shallow convolutional neural network (CNN).
- Context Normalization on Vision Transformer (ViT) [3].
- Context Normalization in domain adaptation.

Context Normalization vs. Mixture Normalization on shallow convolutional neural network (CNN)

- We use two datasets : CIFAR-10 [6] and CIFAR-100 [7].
- For each dataset, we apply the EM algorithm with three components ($k = 3$).
- The components obtained with EM are used in the MN layer and serve as contexts in the CN layer.

Context Normalization vs. Mixture Normalization on shallow convolutional neural network (CNN)

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

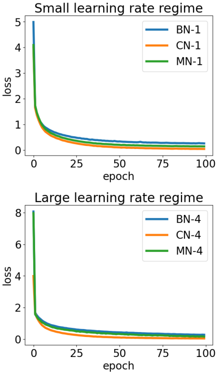
Table – Shallow Convolutional Neural Network

Context Normalization vs. Mixture Normalization on shallow convolutional neural network (CNN)

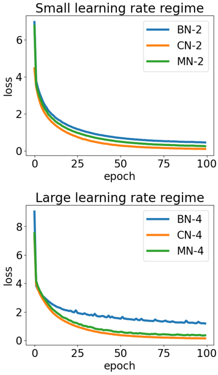
To compare the three normalization methods :

- We use the base architecture with only a BN layer as the baseline.
- We replace "bn" in the "conv2" layer of the base architecture with the MN layer.
- We replace "bn" in the "conv2" layer of the base architecture with the CN layer.
- Models are trained using both small and large learning rates.

Context Normalization vs. Mixture Normalization on shallow convolutional neural network (CNN)



(a) CIFAR-10



(b) CIFAR-100

Figure – Validation loss evolution : BN vs. MN vs. CN

Context Normalization vs. Mixture Normalization on shallow convolutional neural network (CNN)

In this experiment, we can see that :

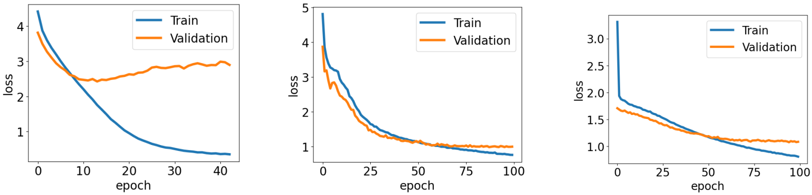
- Compared to MN and BN, CN helps accelerate the model's convergence more effectively.
- It leads to an increase in performance in terms of accuracy, with an average improvement of 3% on CIFAR-10 and 4% on CIFAR-100.

Context Normalization on Vision Transformer (ViT)

The main idea of this experiment is to demonstrate a simple approach for constructing a context :

- We use the CIFAR-100 dataset, which consists of 100 classes.
- We consider the 20 superclasses (grouping of classes) of CIFAR-100 as contexts.
- We use a ViT (Vision Transformer) architecture.
- In this experiment, MN cannot be applied, so we are only comparing with BN.

Context Normalization on Vision Transformer (ViT)



(a) BN (b) CN-Patches (c) CN-Channels

Figure – Comparison of validation loss curves for CIFAR-100 during training the ViT architecture with various normalization methods. "CN-Channels" denotes the application of the CN layer after the input layer, while "CN-Patches" represents the application of the CN layer after the embedding layer.

Context Normalization on Vision Transformer (ViT)

model	test accuracy	test top-5-accuracy
ViT	52.37%	80.98%
ViT+BN	53.35%	79.68%
ViT+CN-Patches	**63.80%**	**99.74%**
ViT+CN-Channels	**62.48%**	**99.83%**

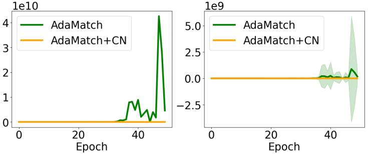
Table – Comparison of ViT performance with different normalization methods : CN (CN-Channels and CN-Patches) and BN.

Context Normalization in domain adaptation

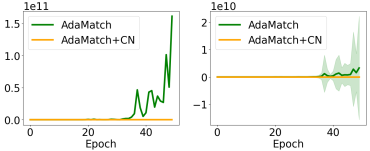
This experiment is motivated by the use of domains (source and target) as contexts :

- We use the AdaMatch algorithm [2] as the baseline model.
- We use the MNIST digits dataset as the source domain and SVHN as the target dataset.
- Each dataset is treated as a context.

Context Normalization in domain adaptation



(a) Source domain (MNIST)



(b) Target domain (SVHN)

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

Context Normalization in domain adaptation

model	source data (MNIST)	target data (SVHN)
AdaMatch	79.39%	20.46%
AdaMatch+CN-Channels	**99.21%**	**43.80%**

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

Feature works

- Merging seamlessly a gradient free optimization algorithm with a gradient-based error optimizer in order to reach global convergence.

- Create an unsupervised variant of CN with the aim of broadening its applicability. This entails excluding context as an input and acquiring it throughout the training process.

Thank you for your attention !

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv :1607.06450*, 2016.

[2] David Berthelot, Rebecca Roelofs, Kihyuk Sohn, Nicholas Carlini, and Alex Kurakin. Adamatch : A unified approach to semi-supervised learning and domain adaptation. *arXiv preprint arXiv :2106.04732*, 2021.

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words : Transformers for image recognition at scale. *arXiv preprint arXiv :2010.11929*, 2020.

[4] Sergey Ioffe and Christian Szegedy. Batch normalization : Accelerating deep network training by reducing internal