IEEE 2020 International Conference on Pattern Recognition (ICPR) - MiCo Milano Congress Center, ITALY, 10 – 15 Jan 2021

Revisiting the Training of Very Deep Neural Networks without Skip Connections

Oyebade K. Oyedotun, Abd El Rahman Shabayek, Djamila Aouada and Björn Ottersten
Interdisciplinary Centre for Security, Reliability and Trust (SnT),
University of Luxembourg, L-1855 Luxembourg
Email: {oyebade.oyedotun, abdelrahman.shabayek, djamila.aouada, bjorn.ottersten}@uni.lu

Abstract—**Deep neural networks (DNNs) with many layers of feature representations yield state-of-the-art results on several difficult learning tasks. However, optimizing very deep DNNs without shortcut connections known as PlainNets, is a notoriously hard problem. Considering the growing interest in this area, this paper investigates holistically two scenarios that plague the training of very deep PlainNets: (1) the relatively popular challenge of 'vanishing and exploding units' activations', and (2) the less investigated 'singularity' problem, which is studied in details in this paper. In contrast to earlier works that study only the saturation and explosion of units' activations in isolation, this paper harmonizes the inconspicuous coexistence of the aforementioned problems for very deep PlainNets. Particularly, we argue that the aforementioned problems would have to be tackled simultaneously for the successful training of very deep PlainNets. Finally, different techniques that can be employed for tackling the optimization problem are discussed, and a specific combination of simple techniques that allows the successful training of PlainNets having up to 100 layers is demonstrated.**

I. Introduction

The importance of DNN depth for learning some classes of functions is studied in [1]. However, training very deep DNNs without shortcut connections known as 'PlainNets' with over 15 layers is difficult [2, 3]. It has been posited that the poor test error rates obtained for very deep PlainNets arise from poor model optimization, and not overfitting [3]. For a successful training, very deep networks with over 15 hidden layers typically rely on shortcut (or skip) connections from lower to higher layers in the model, as seen in residual [3] and highway [2] networks, along with their variants [4]. Models with skip connections give remarkable results on different tasks [5]. However, a satisfactory account of their operation is still lacking [6, 7]. For example, [6] shows that residual networks (ResNets) simply rely on the shortcut connections as shorter paths through the network for propagating error gradients and therefore do not directly address the problem of vanishing or exploding gradients. Moreover, the classical-hierarchical feature representation concept does not strictly apply to ResNet as it has been shown that the effective depth is much smaller than the topological depth, e.g., 17 layers for a 110 layer ResNet [6]. This, for instance, can obfuscate feature transfer as feature representation level may become unclear.

As such, the unconventional objective that we pursue in this paper is investigating concretely the difficulty of training of very deep PlainNets. A major motivation for this is that for several decades, these classical models with strictly hierarchical representations have been studied, and being able to train very deep PlainNets would allow to study their representational capacity and obtainable performance gain. Namely, our contributions are as follows

1) Investigate the dual problem of *vanishing/exploding units' activations (i.e. outputs)*, and *units' singularity* that work together to make training difficult.
2) Provide interesting visualizations of the aforementioned problems using fully connected PlainNets with up to 100 layers on COIL-20 and USPS datasets.
3) Discuss and demonstrate using a 50-layer convolutional PlainNet on CIFAR-10 dataset the combination of different simple training concepts that can be employed for alleviating the identified problems.

The organization of the paper is as follows. Related works are discussed in Section II. Section III describes the datasets used. The background and problem statement are in Section IV. The problems of training very deep PlainNets are investigated in Section V. Experimental results, recommendations for successful training, and discussions are in Section VI. The conclusion is given in Section VII.

II. Related work

Vanishing and exploding gradients were identified as major problems for training recurrent neural networks [8]. Recently, they have been investigated for very deep PlainNets [9]. Very good results were reported [2, 3], addressing the problem of training very deep networks. ResNet [3] employs shortcut connections for alleviating this problem, and [2] proposes gating mechanisms for routing mostly untransformed features through the model highway at the start of training. Huang et al. [10] attribute the difficulty of training very deep PlainNets to the problem of feature reuse, i.e., feature dilution along the successive layers of feature transformations resulting in features at the output layer that are not that representative. It has been suggested that inspecting the forward pass during the training could reveal a lot about their optimization condition [7, 9]. For example, [11] showed that the hierarchical structure of deep networks results in a number of critical points; these points grow with depth and may increase the difficulty of training. Recent works attempt to unravel the problem of training very deep PlainNets by taking inspiration from how the ResNet alleviates the problem [7, 9]. The problem of shattered gradients where error gradients become uncorrelated and resemble white noise was studied in [9]; the authors showed that gradients do not shatter for shallow PlainNets, and that the ResNet largely circumnavigates this problem using shortcut connections. The work in [12] relied on information-theoretic analysis for studying the problem of volume conservation that results in vanishing gradients. In [13], the DiracNet, which relies on the reparameterization of layer weights was proposed as a very deep PlainNet. The reported results are promising; however, a careful inspection shows that the DiracNet does not qualify as a PlainNet, as it uses skip connections that are tucked into the weight tensors. Moreover, the same work [13] states that 'Dirac parameterization and ResNet differ only by the order of nonlinearities'.

III. Datasets

For studying the problems of training very deep PlainNets in this paper, simple datasets such as COIL-20 [14] and USPS [15] are used. We note that 'diagnosing' whether the source of poor test errors in DNNs is *overfitting* or *underfitting* can be tricky. Therefore, we emphasize that these two simple datasets are deliberately chosen to show the severity of the different problems that are studied. That is, a DNN that uses several millions of parameters, and still fails to successfully learn at least the training set shows incontrovertible optimization problem; model overfitting can be entirely ruled out as the problem for training. The COIL-20 dataset contains 1,440 128×128 pixels grayscale images of 20 different objects. The COIL-20 dataset is randomly divided into training and testing images in the ratio 70% and 30%, respectively. The USPS dataset contains 7,291 grayscale training images of 10 different handwritten digits, where all images are of size 16×16 pixels. For testing, there are 2,007 images of size 16×16 pixels. Finally, we use the benchmarking CIFAR-10 dataset [16] for showing that proposed solution gives interesting results on a challenging task. The dataset has 10 classes with 50,000 and 10,000 training and testing images, respectively. All the images are of size 32×32 pixels.

IV. Background and problem statement

A. Background

There is a recent interest in training very deep PlainNets [17] considering that their construction is considerably simpler than alternative architectures that use skip connections, where the suitable number of layers in a block, whether to add (i.e. in ResNet [3]), concatenate (i.e. DenseNet [18]) or add and concatenate (i.e. Inception-ResNet [19]) different hidden layers has to be determined. Subsequently, understanding the operation of PlainNets is straightforward, while the literature lacks clarity on the models that use skip connections. For example, controversies on models with skip connections include the useful number of layers [6], how optimization problem is alleviated [6, 7, 20] and why generalization is improved [6]. We observe that [17] proposed the Delta-Orthogonal initialization scheme for training PlainNets having up to 10,000 layers. A careful examination of the work [17] shows that though the proposed models reached roughly 0% error rate on the different training sets, they performed poorly on the test sets. For instance, their 32-layer model with 17.8M parameters achieved a poor test error rate of 18% on CIFAR-10 dataset. Interestingly, shallower models with significantly fewer model parameters readily achieve test error rates below 10% on CIFAR-10 dataset; Network in Network (NiN) [21] and Deeply supervised network (DSN) [21] both with 10 layers and about 1.8M parameters achieve 8.81% and 7.97%, respectively. Even worse, the 128-layer model with over 25M parameters in [17] gives a poorer test error rate of about 24%. As such, the usefulness of deeper PlainNets becomes questionable, when shallower PlainNets with fewer parameters clearly outperform deeper PlainNets with more parameters.

B. Problem statement

For a hypothetical trained PlainNet model, M, the test error, M_{err}^{test}, typically improves with the number of hidden layers, l, as in

$$M_{err}^{test} \propto 1/l. \tag{1}$$

As such, the objective of training deeper DNNs is popular in the literature [21, 22, 23]. However, it has been observed that increasing the depth of M, l, beyond 10-20 layers generally leads to a progressive reduction of model generalization [2, 3]. Similarly, Fig. 1 shows the performance degradation for fully connected PlainNets with 300 rectified linear units (ReLUs) per hidden layer at different depths for COIL-20 [14] and USPS [15] datasets. Namely, it is seen from Fig. 1 that as the PlainNet becomes deeper, both model optimization and generalization become progressively poorer. Table 1 shows the number of model parameters for the PlainNets given in Fig. 1. Namely, it is seen from Fig. 1 and Table 1 that as the PlainNet becomes deeper with more parameters, both optimization and generalization become progressively poorer.

V. Proposed analysis of PlainNet optimization problem

Herein, we give the proposed analysis of the two training problems of PlainNets, which are units' activations saturation/explosion and units' singularity.

A. Units's activations saturation and explosion

Model units that have very small or zero activations receive little or no error gradients for parameters update during the back-propagation phase; this scenario is referred to as unit saturation [8]. Consider the transformation learned at hidden layer l of a model for input $\boldsymbol{H}(\boldsymbol{x})^{l-1}$ as

$$\boldsymbol{H}(\boldsymbol{x})^{l} = f(W^{l}\boldsymbol{H}(\boldsymbol{x})^{l-1}), \tag{2}$$

$$\boldsymbol{H}(\boldsymbol{x})^{l} = f(\boldsymbol{Z}^{l}), \tag{3}$$

where f is the activation function, and $\boldsymbol{Z}^{l}$ is the pre-activation for layer l; the bias is omitted for simplicity. If E is the error function and f is a function that can saturate (e.g. the rectified linear function), then we may have in $\boldsymbol{H}(\boldsymbol{x})^{l}$ a dead unit $h_{j}^{l} \approx 0$ that receives zero (or negligible) error gradient (i.e. $\delta E/\delta h_{j}(x)^{l} \approx 0$) during the back-pass phase;

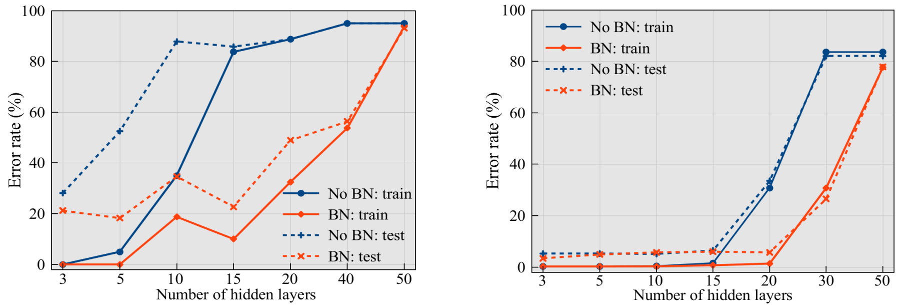


Fig. 1: PlainNet error rate performance with depth. Batch normalization (i.e. BN) alleviates, but do not resolve the optimization problem. Left: COIL-20 dataset. Right: USPS dataset

TABLE I: Details of model parameters for the fully connected PlainNets in Fig. 1

PlainNet depth	3	5	10	15	20	40	50
Number of parameters	0.50M	0.68M	1.13M	1.59M	2.04M	3.86M	4.78M

therefore, h_j^l neither learns nor allows the update of other units earlier in the model that are connected to it. Also, since ReLUs are unbounded in the upper end spectrum, moderate-valued activations can accumulate to result in extremely high activations for which resulting error gradients are very high [8, 9]; this can lead to *wild* weights update that make training unstable. Experimental results discussed in Section VI.A support these observations.

B. Singularity of units' activations

We present analysis into the singularity of units' activations and its effect on the error gradients and solution instability that result in optimization non-convergence. i.e. failure. Singularity implies that some of the units in a layer are linearly dependent or collinear; this can be quite severe for PlainNets. This means that the units respond in the same manner irrespective of the input received; that is, the colinearity of units' responses or representations. This problem grows with increased model depth, since the specialty (or uniqueness) of the units vanishes and symmetry dominates. Symmetry is used to refer to a scenario where swapping two units does not change the output of the model [24]. Importantly, it is known that singularity renders gradient descent an impotent solver [25], since the problem becomes ill-conditioned and lie in a degenerate solution space. For analysis, we rely on the following definitions, lemma and proposition.

Definition 1 The condition number of a matrix $Z \in \mathbb{R}^{m\times n}$ denoted $\kappa(Z)$ is

$$\kappa(\boldsymbol{Z}) = \sigma_{max}(\boldsymbol{Z})/\sigma_{min}(\boldsymbol{Z}), \tag{4}$$

where $\sigma_{max}(\boldsymbol{Z})$ and $\sigma_{min}(\boldsymbol{Z})$ are the maximum and minimum singular values of Z, respectively. Generally, problems with $\kappa(\boldsymbol{Z}) \gg 1$ are referred to as ill-posed.

Lemma 1 *Let $\boldsymbol{U} \in \mathbb{R}^{m\times n}$ and $\boldsymbol{V} \in \mathbb{R}^{n\times p}$ be matrices. If $\boldsymbol{U}$ or $\boldsymbol{V}$ is singular, then their product, $\boldsymbol{P} = \boldsymbol{UV}$, is also singular.*

Proof. Singularity implies that some row(s) or column(s) of $\boldsymbol{U}$ or $\boldsymbol{V}$ are linearly correlated. Thus, $\exists$ a vector $s \in \mathbb{R}^m$ with $\parallel s \parallel_2 \neq 0$: $s\boldsymbol{U} = 0$ or $q \in \mathbb{R}^p$ with $\parallel q \parallel_2 \neq 0$: $\boldsymbol{V}q = 0$. Hence, $s\boldsymbol{UV} = 0$ or $\boldsymbol{UV}q = 0$, and thus $\boldsymbol{UV}$ is singular. □

Assuming that $\boldsymbol{H}(\boldsymbol{x})^{l-1}$ in (2) is singular, then $\boldsymbol{H}(\boldsymbol{x})^{l}$ is rank singular (i.e. deficient) from Lemma 1 and thus collapses the space. Furthermore, it is straightforward to verify that the learned transformation at a hypothetical layer L, $\boldsymbol{H}(\boldsymbol{x})^{L} : L \geq l$ is singular using Lemma 1. Hence, subsequent layers in the model suffer singularity problems. Our experimental results show that this is indeed the case.

Definition 2 For a hypothetical DNN layer l with a batch of hidden units' activations $\boldsymbol{H}(\boldsymbol{x})^{l} \in \mathbb{R}^{m\times n}$, the local error gradients, $\boldsymbol{\Delta}^l$, for updating weight W^l is of the form

$$\boldsymbol{\Delta}^l = f'(\boldsymbol{H}(\boldsymbol{x})^l)\left(W^{l+1}\Delta^{l+1}\right), \tag{5}$$

where f' is the derivative of the activation function, f.

It is seen from (5) that $\boldsymbol{\Delta}^l$ depends on $f'(\boldsymbol{H}(\boldsymbol{x})^l)$; and likewise, $\boldsymbol{\Delta}^{l+1}$ depends on $f'(\boldsymbol{H}(\boldsymbol{x})^{l+1})$, and so on in similar fashion. Consequently, if $\boldsymbol{\Delta}^l$ is singular, then $\boldsymbol{\Delta}^L : L \geq l$ is singular using Lemma 1. Ultimately, singular error gradients are ineffective for driving optimization to convergence.

Proposition 1 *Given an L-layer DNN with input $\boldsymbol{X} \in \mathbb{R}^{n\times N}$, hidden representation at layer l, $\boldsymbol{H}(\boldsymbol{x})^l \in \mathbb{R}^{n\times N}$, and solution $\boldsymbol{\theta} = \{\boldsymbol{W}\}_{l=1}^{L}$, $\Delta\boldsymbol{H}(\boldsymbol{x})^l$ translates to a relative change in solution, $\Delta\boldsymbol{\theta}$, as follows*

$$\frac{\parallel \Delta\boldsymbol{\theta} \parallel}{\parallel \boldsymbol{\theta} \parallel} \leq \kappa(\boldsymbol{H}(\boldsymbol{x})^l)\frac{\parallel \Delta\boldsymbol{H}(\boldsymbol{x})^l \parallel}{\parallel \boldsymbol{H}(\boldsymbol{x})^l \parallel} \quad : 0 \leq l \leq L, \tag{6}$$

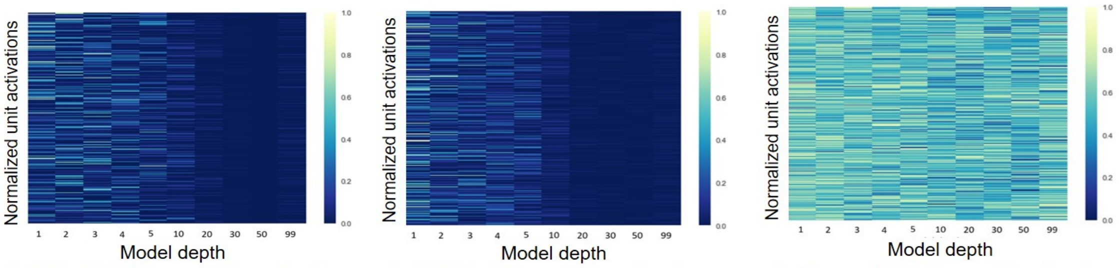
where $\boldsymbol{H}(\boldsymbol{x})^0 = X$ for $l = 0$.

Proof sketch. Let us consider a simple problem, $\boldsymbol{Y} = \boldsymbol{WX}$, where $\boldsymbol{W} \in \mathbb{R}^{n\times n}$, $\boldsymbol{X} \in \mathbb{R}^{n\times N}$ and $\boldsymbol{Y} \in \mathbb{R}^{n\times N}$. Our objective is to estimate the solution, $\boldsymbol{W}$, where $\boldsymbol{X}^{\dagger}$ is the pseudoinvrese of $\boldsymbol{X}$. In addition, let a *small* change of $\Delta\boldsymbol{X}$ translates into a *small* solution change, $\Delta\boldsymbol{W}$, such that we have

$$\boldsymbol{Y} = (\boldsymbol{W} + \Delta\boldsymbol{W})(\boldsymbol{X} + \Delta\boldsymbol{X}). \tag{7}$$

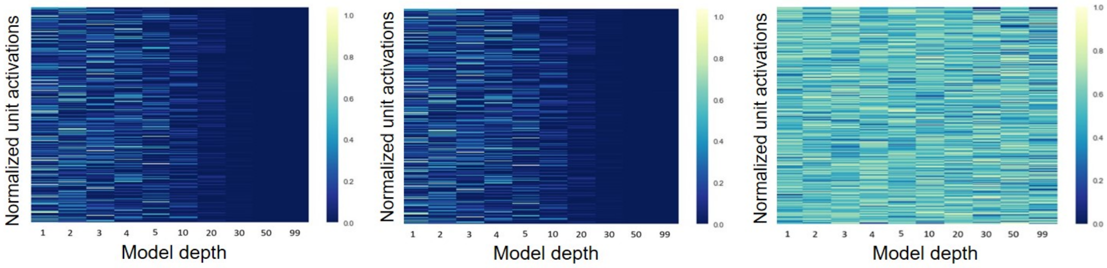
Considering that $\Delta\boldsymbol{W}\Delta\boldsymbol{X} \approx 0$ and $\boldsymbol{Y} = \boldsymbol{WX}$, (7) becomes

$$\frac{\Delta\boldsymbol{W}}{\boldsymbol{W}} = -\boldsymbol{X}^{\dagger}\Delta\boldsymbol{X}. \tag{8}$$



(a) Naive PlainNet: vanishing activations (b) PlainNet-BN: vanishing activations (c) Proposed PlainNet: stable activations

Fig. 2: Normalized mean layer activations for a 100 layer PlainNet over COIL-20 dataset



(a) Naive PlainNet: vanishing activations (b) PlainNet-BN: vanishing activations (c) Proposed PlainNet: stable activations

Fig. 3: Normalized mean layer activations for a 100 layer PlainNet over USPS dataset

Using Cauchy-Schwarz inequality for (8) yields

$$\frac{\| \Delta \boldsymbol{W} \|}{\| \boldsymbol{W} \|} \;\le\; \| \boldsymbol{X}^{\dagger} \| \| \Delta \boldsymbol{X} \| \;\le\; \| \boldsymbol{X}^{\dagger} \| \| \Delta \boldsymbol{X} \| \frac{\| \boldsymbol{X} \|}{\| \boldsymbol{X} \|}. \quad (9)$$

Finally, given that $\kappa(\boldsymbol{X}) \approx \| \boldsymbol{X}^{\dagger} \| \| \boldsymbol{X} \|$, we obtain

$$\frac{\| \Delta \boldsymbol{W} \|}{\| \boldsymbol{W} \|} \le \kappa(\boldsymbol{X}) \frac{\| \Delta \boldsymbol{X} \|}{\| \boldsymbol{X} \|}. \;\square \quad (10)$$

First, note that the singularity of $\boldsymbol{H}(\boldsymbol{x})^l$ means $\sigma_{min}(\boldsymbol{H}(\boldsymbol{x})^l) = 0$ so that $\kappa(\boldsymbol{H}(\boldsymbol{x})^l) = \infty$. Proposition 1 shows that the stability of the solution θ learned during training depends on $\kappa(\boldsymbol{H}(\boldsymbol{x})^l)$. Subsequently, an infinite $\kappa(\boldsymbol{H}(\boldsymbol{x})^l)$ means that small $\Delta \boldsymbol{H}(\boldsymbol{x})^l$ that typically arise from the slightly different samples of the same class in the training data is so amplified that $\| \Delta \boldsymbol{\theta} \| / \| \boldsymbol{\theta} \|$ in Proposition 1 is unbounded during training, and thus optimization convergence fails. In simple terms, the solution $\boldsymbol{\theta}$ continuously and bizarrely fluctuates for small changes in $\boldsymbol{H}(\boldsymbol{x})^l$ so that a decent local minima in the solution space cannot be reached during training.

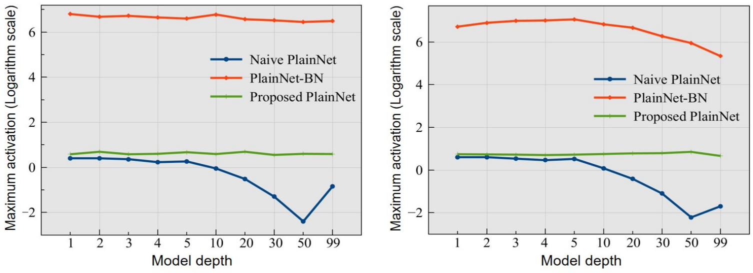
VI. Experiments

We present and discuss the results of our main exposition, which is the study of units' activations and weights conditions that reflect optimization problems. Using the datasets in Section III, we observe learning in the regime of (i) *vanishing/exploding units' activations* (ii) *singularities*. We consider three different training scenarios: (1) PlainNet without batch normalization, referred to as 'naive PlainNet'; and (2) PlainNet with batch normalization, referred to as 'PlainNet-BN', and (3) a proposed solution referred to as 'Proposed PlainNet' that is discussed in Section VI.C.2. For main experiments, we use (i) 100-layer fully connected PlainNet with 300 ReLUs in every hidden layer trained on COIL-20 and USPS datasets (ii) 50-layer convolutional PlainNet trained on CIFAR-10 dataset; see Section A1 in the supplementary material for achitectural details. Both models are trained using mini-batch gradient descent with a batch size of 128, learning rate in the range [0.0001-0.1], momentum rate of 0.9, weight decay of 10^{-4}. The 100-layer fully connected PlainNet and 50-layer convolutional PlainNet are trained for 300 and 400 epochs, respectively.

A. *Vanishing/exploding units' activations and weights condition*

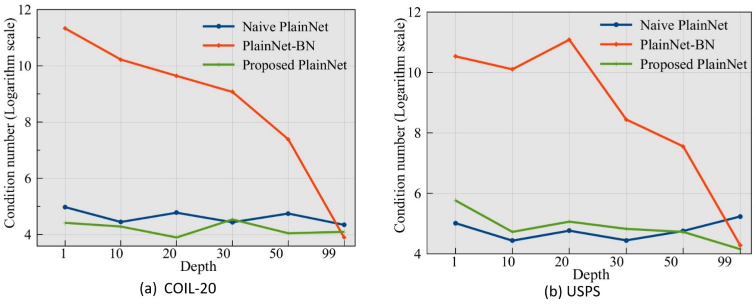
Fig. 2 (a) & (b) show the normalized mean layer activations for the naive PlainNet and PlainNet-BN over the entire COIL-20 dataset. Fig. 3 (a) & (b) show the normalized mean layer activations for the naive PlainNet and PlainNet-BN over the entire USPS dataset. It will be seen that for both naive PlainNet and PlainNet-BN, units' activations decrease with depth for both datasets. It can be observed that at about 20 layers, units' activations have considerably decayed. Importantly, it is emphasized that the PlainNet-BNs have extremely large activations, but global units' activations normalization obfuscate this fact. Fig. 4 shows the maximum absolute units' activations for both the naive PlainNet and PlainNet-BN. It is observed that the naive PlainNet operates with reasonably small activations (in the range 0–10); however, units' activations decrease rapidly with depth. In contrast, some units in the PlainNet-BN operate with extremely high activations (in the range 0–10^6) that are chaotic for a successful optimization. Ultimately, both scenarios result in the difficulty of training.

The analytical investigation on units' activations is further



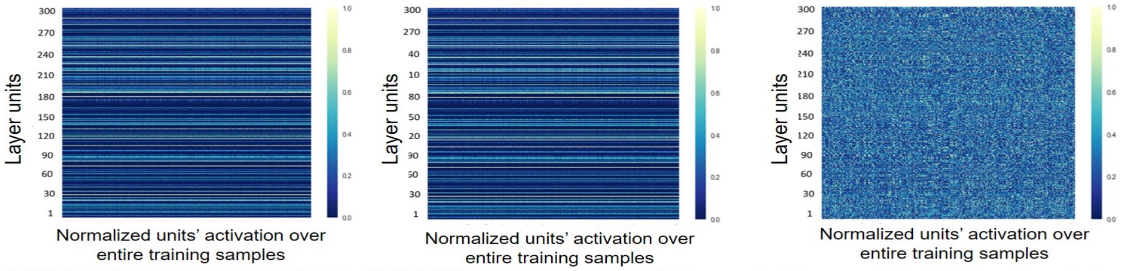
(a) COIL-20 (b) USPS

Fig. 4: Maximum absolute units' activations with depth



(a) COIL-20 (b) USPS

Fig. 5: Infinity-norm based condition number for model weights



(a) Naive PlainNet: near singularity (b) PlainNet-BN: near singularity (c) Proposed PlainNet: no singularity

Fig. 6: 50th layer activations in a 100 layer PlainNet for the entire COIL-20 training set

substantiated by examining the weights of the trained very deep PlainNets. Specifically, we expect that the naive PlainNets would have moderate condition numbers, since gradients typically vanish, weights are mostly not updated and remain close to the initialized values. For the PlainNet-BN, we expect that the extremely high units' activations would cause large error gradients that would subsequently result in wild weights update and therefore very high condition numbers. Fig. 5 shows the condition numbers for the naive PlainNet and PlainNet-BN; the plots confirm earlier analysis and expectations.

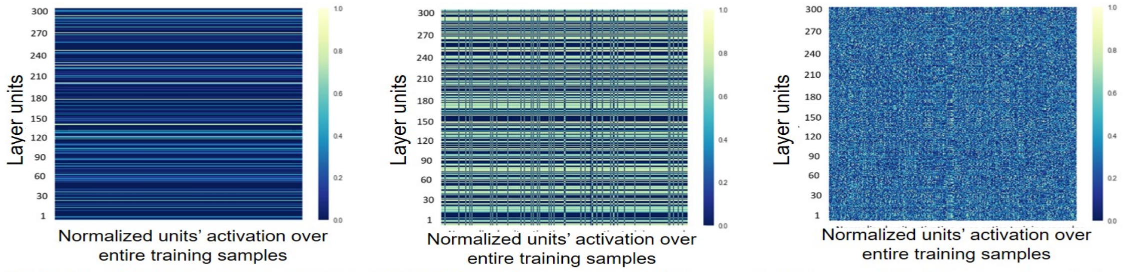
B. Singularities of hidden units' responses

Fig. 6 and Fig. 7 show units' activations over the entire COIL-20 dataset at the 50th and 99th layers, respectively, where it is observed that the units of the naive PlainNet have small and similar responses over the entire dataset (i.e. training examples), showing the singularity problem; see Fig. 6 (a) and Fig. 7 (a). The PlainNet-BN also exhibits singularity, albeit with very high units' activations; see Fig. 6 (b) and Fig. 7 (b). Fig. 8 shows the units' activations over the entire USPS dataset at 99th layer, where it is seen that the vanishing and exploding units' activations, along with singularity problems are more severe; see Fig. 8 (a) & 9 (8). See Section A2 in the supplementary material for additional experimental results on the percentage of units' activations that are above various threshold values across the different hidden layers.

C. Improving the training of PlainNets and Results

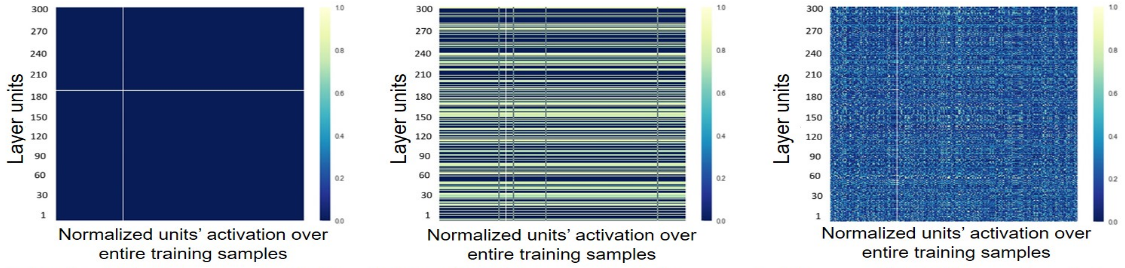
1) Alleviating training difficulties: Herein, we discuss different techniques for tackling the difficulty of training very deep PlainNets in relation to the combined problems of saturating/exploding units' activations and singularity.

- For tackling saturating/exploding units' activations, leaky ReLUs (LReLUs) [27] or parametric ReLUs (PReLUs) [28] can be used in place of ReLUs. In contrast to ReLUs that yield zero-valued outputs for negative pre-activation values (i.e. saturation), LReLUs simply scales the negative pre-activation values by a small constant



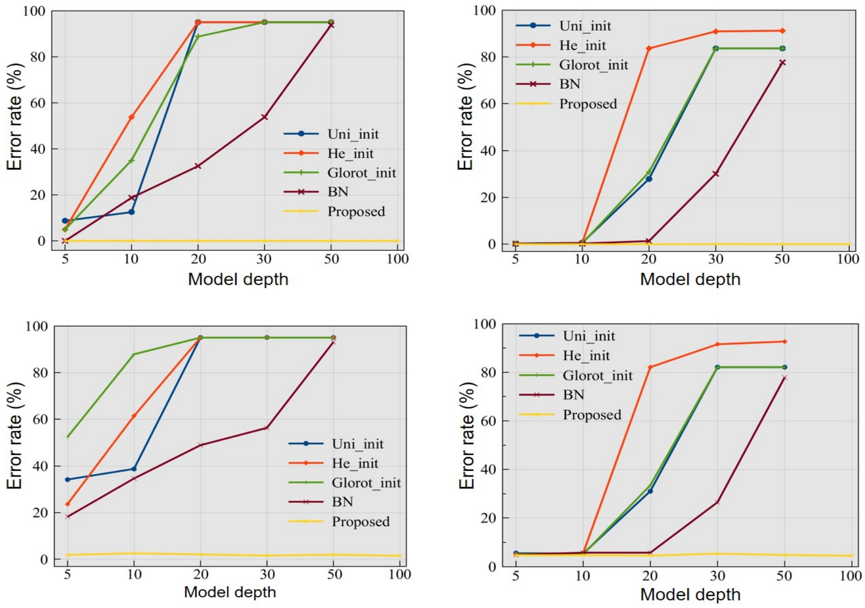
(a) Naive PlainNet: near singularity (b) PlainNet-BN: near singularity (c) Proposed PlainNet: no singularity

Fig. 7: 99th layer activations in a 100 layer PlainNet for the entire COIL-20 training set



(a) Naive PlainNet: near singularity (b) PlainNet-BN: near singularity (c) Proposed PlainNet: no singularity

Fig. 8: 99th layer activations in a 100 layer PlainNet for the entire USPS training set



(a) COIL-20 (b) USPS

Fig. 9: Top: PlainNet error rates on training set for COIL-20 (left) and USPS (right) datasets. Bottom: Error rates on test set for COIL-20 (left) and USPS (right) datasets [26]

factor as the output. The PReLU is similar to the LReLU except that the PReLU is parameterized so that it learns the scaling factors. That is, different PReLUs use different scaling factors, which are learned during training. In addition, from (2), having some units with negative activations in $\boldsymbol{H}(\boldsymbol{x})^{l-1}$ can help to reduce the growth of units' activations in $\boldsymbol{H}(\boldsymbol{x})^{l}$. Consequently, both the LReLU and PReLU, which can attain negative values can be also useful for ameliorating the problem of units' activations explosion.

- The singularity of hidden units can be addressed by regularizing layer weights so that they are never singular. For example, imposing orthogonality on the columns and rows of all layer weights [29]. If the input of the

TABLE II: Ablation study results for fully connected 100-layer PlainNets on USPS dataset

Model component	Train error	Test error
Batch normalization (BN)	84.56%	83.21%
LReLU	92.37%	92.03%
Max-norm	86.22%	86.85%
BN + LReLU	78.38%	79.52%
BN + max-norm	82.90%	81.86%
LReLU + max-norm	83.62%	82.11%
Proposed: BN + LReLU + max-norm	**0.11%**	**5.48%**

DNN is non-singular and all the layer weights are non-singular, then all the layer outputs are non-singular, since the product of two non-singular matrices is non-singular.

- The possible erratic weights updates that results from bad conditioning can be tackled by constraining the weight values using the max-norm constraint, which ensures that the norm of the weight vector of any hidden unit does not exceed the specified maximum norm.

2) Results: For validation, we implement some of the aforementioned approaches for improving PlainNet training. Specifically, we consider the worst case in our experimental settings by training 100-layer PlainNets on the COIL-20 and USPS datasets using batch normalization, LReLUs with a scaling factor of 0.3 for negative pre-activation values, weights initialized from a uniform distribution as in [28], and a max-norm constraint of 3 imposed on the weights' vector of every hidden unit; these models are referred to as the 'Proposed PlainNet'. Fig. 2 (c) and Fig. 3 (c) show evolution of units' activations across the layers. It is seen that unit's activations of the proposed PlainNets remain stable (i.e. neither diminish nor explode) even at the 99th layer for both datasets; compare this result with the naive PlainNet and PlainNet-BN in Fig. 2 (a) & (b) and Fig. 3 (a) & (b). Fig. 4 (a) & (b) show that the maximum absolute activations are within reasonable range, a compromise between the naive PlainNet and PlainNet-BN. Fig. 5 (a) & (b) show that the weights of the proposed PlainNet indeed have moderate condition numbers, and therefore optimization is easier. Importantly, an examination of the weights of the proposed PlainNet after training shows that they have been reasonably updated, as against the naive PlainNet where weights were mostly not updated. The absence of the unit's singularity in the proposed PlainNets is shown in Fig. 6 (c), Fig. 7 (c) and Fig. 8 (c). That is, the units respond in the different ways to inputs from the datasets.

TABLE III: Results on CIFAR-10 dataset

Model	Skip conn.	Layers	Parameters	Test error
Highway network [2]	Yes	19	2.30M	7.54%
ResNet [3]	Yes	56	0.85M	6.97%
ResNet [3]	Yes	110	1.7M	6.43%
All CNN [32]	No	8	1.30M	7.25%
NiN [33]	No	10	1.30M	8.81%
Delta init. [17]	No	32	17.80M	18.00%
PlainNet-BN [3]	No	56	0.85M	15.00%
Proposed PlainNet	**No**	**50**	**0.72M**	**6.65%**

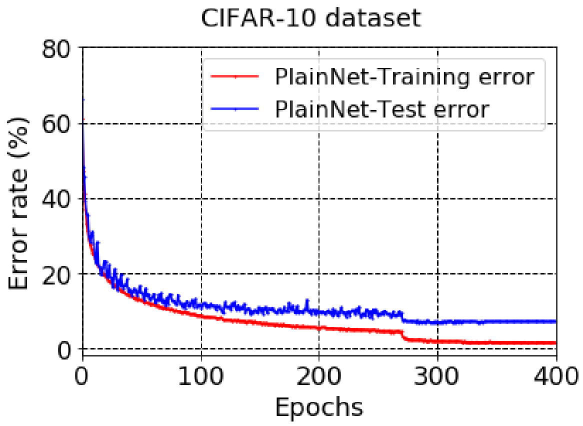


Fig. 10: Training curves for the proposed 50-layer convolutional PlainNet on CIFAR-10 dataset

Fig. 9 (a) & (b) show that initialization schemes such as Uniform, He et al. (He_init) [28], Glorot et al. (Glorot_init) [30], and batch normalization (BN) [31] do not resolve training difficulty beyond some certain depth. However, the proposed PlainNet (i.e shown as proposed) is trainable even with 100 layers; both training and test errors remain small with depth increase. In addition, it is seen that training the 100-layer PlainNets without batch normalization, LReLUs or max-norm constraint results in failed optimization; the training and testing accuracies obtained on both datasets are less than 80%. Furthermore, we perform ablation studies on the proposed fully connected 100-layer PlainNet to observe the individual contribution of batch normalization (BN), Leaky Rectified Linear Unit (LReLU) and max-norm for alleviating training problems. Furthermore, two out of the three components are applied at a time to see the impact of the component left out. Experimental results on the USPS dataset, which are reported in Table II show that the three components altogether are crucial for tackling the training problem of very deep PlainNets. Removing one or two of the three components gave worse training results.

Lastly, using the 50-layer convolutional PlainNet, we report experimental results on the popular CIFAR-10 dataset [16] in Table III, where it is seen that the proposed PlainNet with fewer parameters outperforms all the other PlainNets, and achieved a very competitive result in comparison with models that use skip connections. Note that fair comparison is among models that use similar number of parameters. The training curves for the proposed 50-layer PlainNet is shown in Fig. 10. Importantly, it is noted that the other PlainNets with over 30 layers have noticeably poor results, showing optimization problems.

VII. Conclusion and Future Work

Training very deep DNNs without skip connections typically lead to improved model generalization. However, it is common to observe poor generalization when model depth exceeds 15 layers. In this paper, we study two sources of optimization problem, which include the popular *vanishing/exploding activations* and somewhat obscure *hidden units singularity* when model depth exceeds 15 layers. Importantly, we show for the first time in the literature the interwoven interaction of the aforementioned problems for training very deep PlainNets. Our investigation results reveal that the successful training of very deep PlainNets would rely on simultaneously alleviating vanishing/exploding units' activations and singularity of units' activations. Lastly, we demonstrate an approach for alleviating the training problems. Considering the initial success of training very deep PlainNets in this paper, the different recommendations for improved training would be explored further on more challenging datasets as future work.

Acknowledgment

This work was funded by the National Research Fund (FNR), Luxembourg, under the project references R-AGR-0424-05-D/Bjorn Ottersten and CPPP17/IS/11643091/IDform/Aouada.

References

[1] M. Bianchini and F. Scarselli, "On the complexity of neural network classifiers: A comparison between shallow and deep architectures," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 8, pp. 1553–1565, 2014.

[2] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Advances in neural information processing systems*, 2015, pp. 2377–2385.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[4] O. K. Oyedotun, A. El Rahman Shabayek, D. Aouada, and B. Ottersten, "Highway network block with gates constraints for training very deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 1658–1667.

[5] O. K. Oyedotun, D. Aouada, B. Ottersten *et al.*, "Improving the capacity of very deep networks with maxout units," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2971–2975.

[6] A. Veit, M. J. Wilber, and S. Belongie, "Residual networks behave like ensembles of relatively shallow networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 550–558.

[7] D. Balduzzi, M. Frean, L. Leary, J. Lewis, K. W.-D. Ma, and B. McWilliams, "The shattered gradients problem: If resnets are the answer, then what is the question?" in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 342–350.

[8] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[9] G. Philipp, D. Song, and J. G. Carbonell, "Gradients explode-deep networks are shallow-resnet explained," *arXiv preprint arXiv:1712.05577*, 2017.

[10] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *European Conference on Computer Vision*. Springer, 2016, pp. 646–661.

[11] T. Nitta, "On the singularity in deep neural networks," in *International Conference on Neural Information Processing*. Springer, 2016, pp. 389–396.

[12] T. Unterthiner and S. Hochreiter, "Understanding very deep networks via volume conservation," in *International Conference on Learning Representations Workshop*, 2016.

[13] S. Zagoruyko and N. Komodakis, "Diracnets: training very deep neural networks without skip-connections," *arXiv preprint arXiv:1706.00388*, 2017.

[14] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (coil-20)," in *Technical Report CUCS-005-96*, 1996.

[15] J. J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, May 1994.

[16] A. Krizhevsky, "Learning multiple layers of features from tiny images," in *Technical Report*, 2009.

[17] L. Xiao, Y. Bahri, J. Sohl-Dickstein, S. Schoenholz, and J. Pennington, "Dynamical isometry and a mean field theory of cnns: How to train 10,000-layer vanilla convolutional neural networks," in *International Conference on Machine Learning*, 2018, pp. 5393–5402.

[18] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[19] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[20] K. Greff, R. K. Srivastava, and J. Schmidhuber, "Highway and residual networks learn unrolled iterative estimation," in *International Conference on Learning Representaions*, 2017.

[21] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Artificial Intelligence and Statistics*, 2015, pp. 562–570.

[22] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," in *International Conference on Learning Representations*, 2015.

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[24] H. Wei, J. Zhang, F. Cousseau, T. Ozeki, and S.-i. Amari, "Dynamics of learning near singularities in layered networks," *Neural computation*, vol. 20, no. 3, pp. 813–843, 2008.

[25] L. Elden, "Algorithms for the regularization of ill-conditioned least squares problems," *BIT Numerical Mathematics*, vol. 17, no. 2, pp. 134–145, 1977.

[26] O. K. Oyedotun, A. E. R. Shabayek, D. Aouada, and B. Ottersten, "Training very deep networks via residual learning with stochastic input shortcut connections," in *International Conference on Neural Information Processing*. Springer, 2017, pp. 23–33.

[27] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*. Citeseer, 2013.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[29] N. Bansal, X. Chen, and Z. Wang, "Can we gain more from orthogonality regularizations in training deep networks?" in *Advances in Neural Information Processing Systems*, 2018, pp. 4261–4271.

[30] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[31] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[32] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," in *International Conference on Learning Representations Workshop*, 2014.

[33] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

IEEE 2020 International Conference on Pattern Recognition (ICPR) - MiCo Milano Congress Center, ITALY, 10 – 15 Jan 2021

Revisiting the Training of Very Deep Neural Networks without Skip Connections

Oyebade K. Oyedotun, Abd El Rahman Shabayek, Djamila Aouada and Björn Ottersten
Interdisciplinary Centre for Security, Reliability and Trust (SnT),
University of Luxembourg, L-1855 Luxembourg
Email: {oyebade.oyedotun, abdelrahman.shabayek, djamila.aouada, bjorn.ottersten}@uni.lu

A1. 50-Layer Convolutional PlainNet Architectural Details

The architecture of the proposed 50-layer convolutional PlainNet reported in Table III of the main manuscript is given herein Table A1, where Conv_$n-m(r\times r)$ is the nth convolutional layer with m output filters of size $r\times r$; and MP_$n(r\times r)$ is the nth max-pooling layer of window size $r\times r$.

A2. Additional Investigation Results for Training Very Deep PlainNets

Fig. A1 shows the percentage of units with activations above different thresholds after training on the USPS dataset. We note that the number of units with zero activations may be misleading in observing training difficulties, as the percentage of units with exactly zero activations is roughly 50% across all layers in our experiments. The work in [1] corroborates our argument that having many units with very small activations in very deep PlainNets encourages pseudo-linearity and therefore singularity. Our experiments reveal that training difficulty arises when the activations of many units are smaller than $|0.05|$. Fig. A1 shows that the percentage of units' activations above $|0.05|$ begins to reduce rapidly from the 20th layer in the naive PlainNet, while many units in the PlainNet-BN maintain very high activations much deeper into the model, making training unstable. However, it is observed for the proposed PlainNet on both COIL-20 and USPS datasets that every units' activation value is above $|0.05|$ and less than $|50|$. This constitute a reasonable middle ground between the naive PlainNet and PlainNet-BN as shown in Fig. A1.

References

[1] G. Philipp, D. Song, and J. G. Carbonell, "Gradients explode-deep networks are shallow-resnet explained," *arXiv preprint arXiv:1712.05577*, 2017.

TABLE A1: Architecture of the Proposed 50-layer convolutional PlainNet for CIFAR-10 dataset

50-layer proposed PlainNet	
Output size	Input: 32×32×3
32×32	Conv_1-16(3×3)
32×32	Conv_2-16(1×1)
32×32	Conv_3-16(3×3)
32×32	Conv_4-32(1×1)
32×32	Conv_5-16(1×1)
32×32	Conv_6-16(3×3)
32×32	Conv_7-32(1×1)
32×32	Conv_8-16(1×1)
32×32	Conv_9-16(3×3)
32×32	Conv_10-32(1×1)
32×32	Conv_11-16(1×1)
32×32	Conv_12-16(3×3)
32×32	Conv_13-32(1×1)
32×32	Conv_14-16(1×1)
32×32	Conv_15-16(3×3)
32×32	Conv_16-32(1×1)
32×32	Conv_17-48(1×1)
16×16	MP__1(2×2); stride=2
16×16	Conv_18-48(1×1)
16×16	Conv_19-64(3×3)
16×16	Conv_20-64(1×1)
16×16	Conv_21-48(1×1)
16×16	Conv_22-48(3×3)
16×16	Conv_23-64(1×1)
16×16	Conv_24-48(1×1)
16×16	Conv_25-48(3×3)
16×16	Conv_26-64(1×1)
16×16	Conv_27-48(1×1)
16×16	Conv_28-48(3×3)
16×16	Conv_29-64(1×1)
16×16	Conv_30-48(1×1)
16×16	Conv_31-48(3×3)
16×16	Conv_32-64(1×1)
16×16	Conv_33-48(1×1)
16×16	Conv_34-48(3×3)
16×16	Conv_35-64(1×1)
16×16	Conv_36-80(3×3)
7×7	MP__2(3×3); stride=2
7×7	conv_37-80(1×1)
7×7	conv_38-80(3×3)
7×7	Conv_39-96(1×1)
7×7	conv_40-80(1×1)
7×7	conv_41-80(3×3)
7×7	Conv_42-96(1×1)
7×7	conv_43-80(1×1)
7×7	conv_44-80(3×3)
7×7	Conv_45-96(1×1)
7×7	conv_46-80(1×1)
7×7	conv_47-80(3×3)
7×7	Conv_48-96(1×1)
7×7	Conv_50-128(3×3)
1×1	Global averaging pool
10-way softmax	

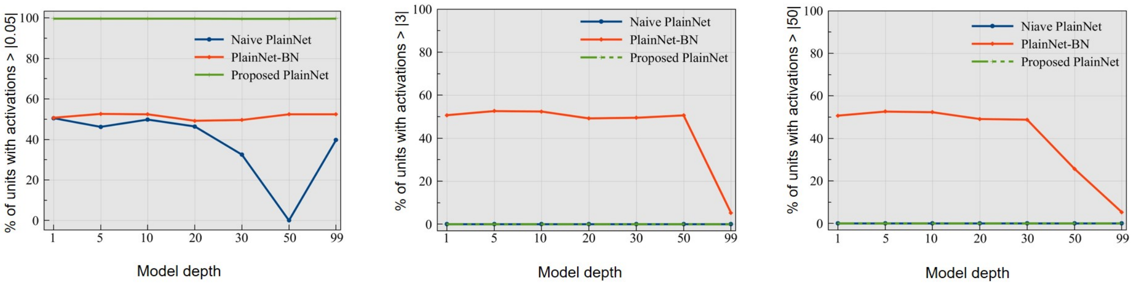


Fig. A1: % of units' unnormalized absolute activations above threshold for the USPS dataset