Delve into Neural Activations: Towards Understanding Dying Neurons

Ziping Jiang, Yunpeng Wang, Chang-Tsun Li, Plamen Angelov and Richard Jiang

***Abstract*—Theoretically, a deep neuron network with non-linear activation is able to approximate any function, while empirically the performance of the model with different activations varies widely. In this work, we investigate the expressivity of the network from an activation perspective. In particular, we introduce a *generalized activation region/pattern* to describe the functional relationship of the model with an arbitrary activation function and illustrate its fundamental properties. We then propose a metric named *pattern similarity* to evaluate the practical expressivity of neuron networks regarding datasets based on the neuron level reaction toward the input. We find an undocumented *dying neuron* issue that the post-activation value of most neurons remain in the same region for data with different labels, implying that the expressivity of the network with certain activations is greatly constrained. For instance, around 80% of post-activation values of a well-trained Sigmoid net or Tanh net are clustered in the same region given any test sample. This means most of the neurons fail to provide any useful information in distinguishing the data with different labels, suggesting that the practical expressivity of those networks are far below the theoretical. By evaluating our metrics and the test accuracy of the model, we show that the seriousness of the *dying neuron* issue is highly related to the model performance. At last, we also discussed the cause of the *dying neuron* issue, providing an explanation of the model performance gap caused by choice of activation.**

***Impact Statement*—The activation function is a crucial component of deep learning models. Many previous works focus on developing efficient activations to promote model performance. However, how the activation function affects the performance of the model is still an unsolved question. In this work, we document a novel *dying neuron* issue that affects the model performance. We first introduce a *generalized activation pattern* to analyze the neuron level response of a model with arbitrary activation. Based on that, we further propose a *pattern similarity* that illustrates the expressivity of a model on a dataset. We show that the pattern similarity can be used to investigate the severity of the *dying neuron* issue and explain the performance gap caused by different activations.**

***Index Terms*—Artificial intelligence, neural computation, neural activation, neural networks, dying neurons**

This manuscript was submitted on 15/03/2022. This work was supported in part by the Huawei Technologies Co., Ltd under Grant HIRP2019041002010, the UK EPSRC under Grant EP/P009727/1, and the Leverhulme Trust under Grant RF-2019-492. (Correspondent author: Dr Richard Jiange-mail: r.jiang2@lancaster.ac.uk).

Ziping Jiang, Plamen Angelov and Richard Jiang are with the LIRA Center, Lancaster University, Lancaster, LA1 4YW, United Kingdom.

Yunpeng Wang is with the Huawei Technologies Co., Ltd, Huawei Base, Bantian, Longgang District, Shenzhen, Guangdong, 518129, China.

Chang-Tsun Li is with Deakin University, 75 Pigdons Rd, Waurn Ponds VIC 3216, Australia.

Here we thank Huawei Technologies Co., Ltd. for providing computational resources on MindSpore platform.

I. Introduction

In recent years, deep neural network methods have achieved state-of-art results on applications in various fields, from computer vision to natural language processing [1], [2], [3]. Theoretically, a non-linear model with the suitable depth and width is able to approximate an objective function at an arbitrarily precision [4], [5], while the empirical studies show that the choice of activation can greatly affect the potential performance of a model. In this work, we attempt to provide a fundamental analysis of the expressive ability and generalization of deep neuron networks to better understand the performance gap caused by different activations.

The most notable case of model performance gap caused by activation function can be observed from the comparison between ReLU net and Sigomid net [6]. It is widely recognized that the poor performance of the Sigmoid net is caused by the vanishing gradient issue attributed to the over-saturation of the hidden units. However, we find that issue is already fixed by the lately proposed techniques, such as batch-normalization [7] and weight initialization techniques [6], [8]. Then what else remains that cause the difference in the performance of the network with different activations?

We start with investigating the training dynamic of networks. We find that training of networks with non-piecewise linear activation can be affected by the current weight, therefore, is less stable. Then we use a simple model to track the change of weights as well as the ability in approximating an objective function of each neuron during training and find an undocumented *dying neuron* issue. Different from the gradient vanishing issue, the gradient of *dead neurons* is non-zero while the model still fails to update.

To better understand the issue and investigate the expressivity of networks, we extend the linear activation region for analyzing the complexity of ReLU networks to the general case. Intuitively, a network with ReLU activation computes a piecewise linear function consisting of a set of local linear functions. For a network N with an arbitrary activation, the input domain can be split into many regions in a similar way where each region is described by an activation pattern. The proposed generalized activation region allow us to investigate the expressivity and generalization of deep network with different activation functions.

We then discuss the properties of the activation region. In particular, we introduce a metric named *pattern similarity* to evaluate the topological distance of data on space defined by network N. By means of the transition density of trajectory proposed in [9], we build connections between pattern sim-

ilarity, transition density and prediction difference of data. We show that the pattern similarity metrics can be used as an indicator of network expressivity and generalization. By monitoring the change of pattern similarity of network change of networks, we find that the *dying neuron* issue is severe for certain networks. For instance, we find that for data with different labels, the pattern similarity of Sigmoid net and TanH-net was given threshold $\lambda = 80\%$ is still around 1. This means for any two data, there are 80% of the neurons have similar post-activation values, suggesting that the practical expressivity of the network is far below that of the theoretical. At last, we also discussed the cause of the *dying neuron* issue and show that the seriousness of the *dying neuron* issue is highly related to the model performance.

The contributions of this work can be summarized as follows:

- We introduce a generalized activation region/pattern for analyzing the neuron network with different activations and illustrate its properties;
- We propose a metric named *pattern similarity* to evaluate expressivity and generalization of network dynamically based on the neuron-level responses towards dataset;
- We document an undocumented *dying neuron* issue that greatly affects the model performance and discuss the cause of the issue.

This work is organized as follows. In Section II, we review previous research of activation function and neuron network expressivity. In Section III, we review the gradient vanishing issue and introduce the *dying neuron* issue by analyzing the learning dynamic from an activation perspective. Section IV-A introduces the generalized activation pattern as a tool for investigating the model expressivity and discusses its basic properties. Section V proposes a metrics named *pattern similarity* that is able to evaluate the model expressivity and severity of *dying neuron* issue. In Section VI, we present experiments to support the arguments proposed in this work and illustrate that the model performance is highly related to the severity of *dying neuron* issue.

II. Relatd Works

The research of activation function has a long history. The Sigmoid activation once was widely adopted on neural network models due to its statistical explainability but was proven less suitable for training models with a deep structure. On the other hand, the rectified linear unit (ReLU) activation is considered as a suitable replacement due to its linearity [10], [11]. The Leaky ReLU [12] then addresses the *dying ReLU* [13] issue that occurs when too many neurons are deactivated. The PReLU [8] makes the slope of the negative part trainable by introducing a coefficient controlling factor, followed by an adaptive piecewise linear unit [14]. Other attempts on developing piecewise linear activation including the including ReLU-6 [15] and hard Sigmoid [16]. At the same time, research in continuous activation function also provide satisfying result, including penalized Tanh [17], penalized Tanh [12], SiLU [18], ELU [19], Swish activation [20] and state-of-art GeLU activation [18].

Theoretically, many works provide discussion regarding the activation functions. One of the famous findings is the vanishing gradient issue [6], [21], [22]. The widely adopted solution to the issue is to initialize the model with weights that maintain the activation variance and the gradient variance [8], [6], adopting batch normalization to standardise the input signal of each layer [7], [23], [24], [25], [26].

Another strand of literature focus on the expressive power of the deep network. The first topic is how deep learning is able to approximate an objective function [27], [4], [5] and the relationship between performance and model complexity[28], [29]. Following work then evaluate the expressive power of deep model complexity [30], [31]. In particular, due to the outstanding performance of the ReLU net, many works focus on investigating the number of linear regions provided by a ReLU net [32], [33], [34]. In particular, recent works investigate the properties of linear activation regions [35], [36] and adopt it as a metric to evaluate ReLU models [9], [37].

III. Learning dynamic of Model

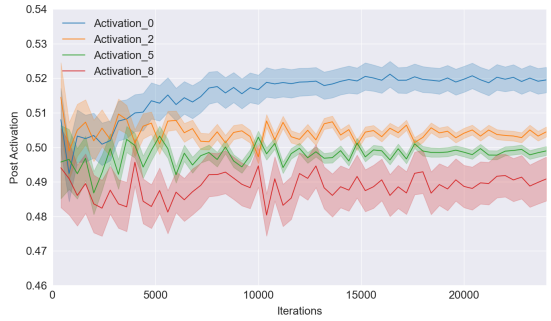
In this section, we investigate the learning dynamic of a deep network. We first show that the gradient vanishing issue is fixed with lately proposed techniques. Then we discuss the training dynamic of a network with different activations to investigate the undiscovered issue. At last, we use a toy model to illustrate the neuron level reaction of the network regarding inputs.

A. Review of Gradient Vanishing

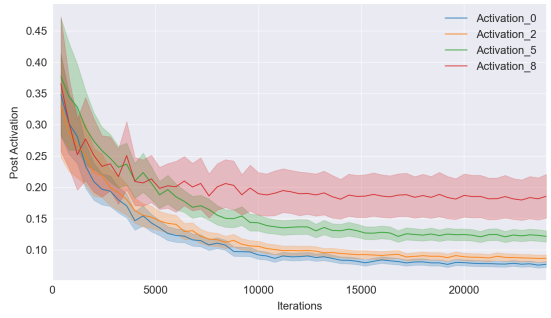
The gradient vanishing issue of Sigmoid activation refers to the phenomenon that when the inputs of neurons are extremely large or small, the Sigmoid function saturates at regions where gradients are almost zero. This results in a failure in the backpropagation of the network. Thus, the network fails to converge. Since it was proposed, it is widely accepted as the cause of the poor performance of Sigmoid and other activations, such as Tanh. However, the issue was observed in the early days before the introduction of other deep neuron network techniques, such as [7] and weight initialization techniques [6], [8]. In this section, we re-examine the issue using the same network settings but with batch-normalization layer and weight initialization.

Figure 1 compares the post-activation values of Sigmoid net and ReLU net trained on MNIST dataset. We notice that the gradient vanishing issue no longer exists with the lately proposed techniques introduced. To be specific, for the Sigmoid net the post-activation value of neurons are evenly distributed around 0.5 at which region the gradient of Sigmoid activation does not vanish. Moreover, Figure 4 shows that the unit-wise average gradient of the networks with different activations has similar properties during training. Detailed experiment settings and discussion can be found in section VI.

The above observations indicate that when batch-normalization is introduced to the model, the post-activation values and gradients of the Sigmoid net are no longer saturated. This suggests that the vanishing gradient issue of the



(a) Post Activatoin of Sigmoid net



(b) Post Activatoin of ReLU net

Fig. 1. The mean and variance of post-activation values of a network with 9 layers, with 256 neurons within each layer.

Sigmoid net can be fixed by re-centring and re-scaling the values of each layer. However, empirically, the performance of the Sigmoid net is still far behind the state-of-art activations. We contend that there are other issues that prevent the Sigmoid activation from performing well.

B. Learning Dynamic of Neuron Networks

Theoretically, a deep neuron network with non-linear activation is able to approximate any function. However, the performance of models with different activations varies widely. To provide an explanation to the conflict, we start with investigating the training dynamic of neuron network.

A stacked neural network can be expressed as $F = f_1 \circ f_2 \circ f \ldots f_n$, where $F : R^{in} \rightarrow R^{out}$ is a mapping function from input domain R^{in} to output domain R^{out}, and f_k is the activation function of layer k. Denote x^k, y^k and $\theta^k = \{W^k, b^k\}$ are the input, output and parameters of layer k, respectively. Therefore, we have

$$x^k = W^{k-1}y^{k-1} + b^{k-1}, y^k = f^k(x^k), \tag{1}$$

where weight $W^{i-1} \in R^{n_i \times n_{i-1}}$ and the bias $b^{i-1} \in R^{n_i}$. Given a loss function L and training sample (x, y), the change of weight ΔW^{i-1} regarding learning rate η is:

$$\Delta W^{i-1} = \frac{\partial L}{\partial y^i}\frac{\partial y^i}{\partial x^i}\frac{\partial x^i}{\partial W^{i-1}}\eta = \frac{\partial L}{\partial y^i}\frac{\partial y^i}{\partial x^i}y^{i-1}\eta. \tag{2}$$

where ΔW^{i-1} is a function of x^i, y^i, θ. Denote $C_1 = y^{i-1}\eta$, now we consider the derivative of ΔW^{i-1}:

$$\frac{\partial \Delta W^{i-1}}{\partial W^{i-1}} = C_1\left[\frac{\partial^2 L}{\partial y^i \partial W^{i-1}}\frac{\partial y_i}{\partial x^i} + \frac{\partial L}{\partial y^i}\frac{\partial^2 y_i}{\partial x^i \partial W^{i-1}}\right] \tag{3}$$

Since,

$$\frac{\partial L}{\partial y^i} = \frac{\partial L}{\partial y^n}\prod_{j=i}^{n-1}\frac{\partial y^{j+1}}{\partial x^j}W^j, \tag{4}$$

for a fixed model F_θ, given training sample (x, y),

$$\frac{\partial L}{\partial y^i} = C_2\frac{\partial L}{\partial y^n} \tag{5}$$

where C_2 is defined as the value of $C_2(x, y, \theta) = \prod_{j=i}^{n-1}\frac{\partial y^{j+1}}{\partial x^j}W^j$ given sample (x, y) and model parameters θ. Equation 3 then can be formatted as:

$$\frac{\partial \Delta W^{i-1}}{\partial W^{i-1}} = C_1C_2\left[\frac{\partial^2 L}{\partial y^n \partial W^i}\frac{\partial y^i}{\partial x^i} + \frac{\partial L}{\partial y^n}\frac{\partial^2 y_i}{\partial x^i \partial W^{i-1}}\right] \tag{6}$$

Now we consider the loss function L. For the activations function with linear derivatives, such as widely adopted mean square error (MSE) and softmax cross-entropy loss, their derivatives can be written as

$$\frac{\partial L}{\partial y^n} = a \times y^n + b, \tag{7}$$

where $\hat{y}$ and y are predicted vector and ground truth vector, a and b are coefficients of the derivative of the loss function.

We first consider the former part of Equation 6 given a piecewise linear activation. Since F is a stacked model, W^i is multiple once during the forward propagation, so $y^n \propto W^{i-1}$. By combining Equation 7 we know that the first multiplier of the former part is inrelevant with W^{i-1}. On the other hand, since $\frac{\partial y^i}{\partial x^i}$ is a constant, the latter part of Equation 6 is zero. Recall that the variable C_1 and C_2 are also inrelevant with W^{i-1}. For a network with piecewise linear activation, Equation 6 can be written as

$$\frac{\partial \Delta W^{i-1}}{\partial W^{i-1}} = C_0(x, y, \theta \backslash \{W^{i-1}\}) \tag{8}$$

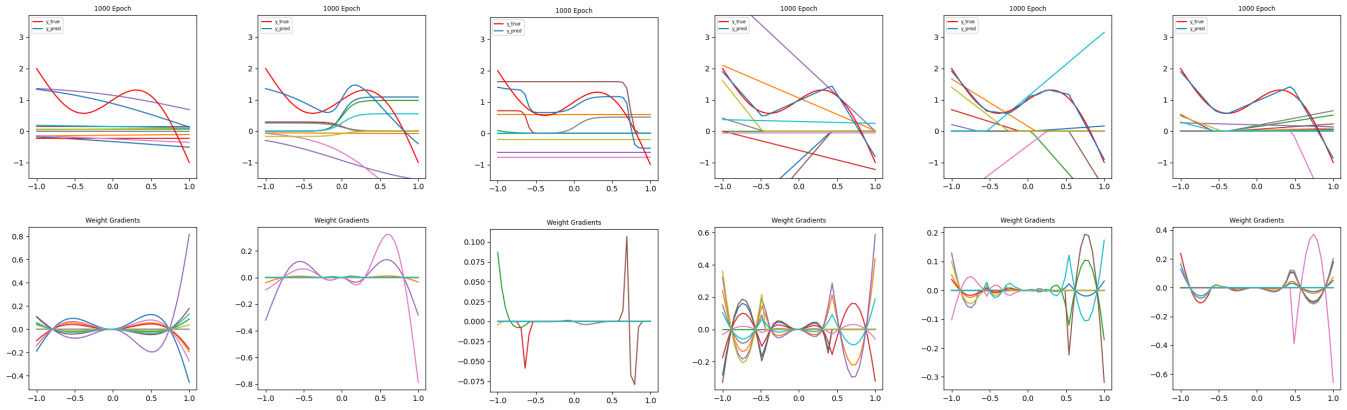
where

$$C_0 = C_1C_2\frac{\partial^2 L}{\partial y^n \partial W^i}\frac{\partial y^i}{\partial x^i}. \tag{9}$$

is a function of x, y and $\theta \backslash \{W^{i-1}\}$. Based on above analysis, we now can yield our theorem for comparing the learning dynamic of two activations.

Theorem 1. *Given a deep neural network with piecewise linear activation, if the derivative of loss function $L(\hat{y}, y)$ is linear, then for a given training sample (x, y), $\Delta W^i \propto W^i$, $i = 1, 2, ...n$, where W^i is the weigth of ith layer.*

Theorem 1 suggests that, if the loss function of a model has a linear derivative and is built with the piecewise linear activation function, the gradient of each neuron is *linear* to its current weight value, which implies that the weight W^i can be viewed as a scale of learning rate. In particular, with other



(a) Sigmoid(1, 1) (b) Sigmoid(15, 1) (c) Sigmoid(15, 0) (d) ReLU(1, 1) (e) ReLU(15, 1) (f) ReLU(15, 0)

Fig. 2. The performance and gradients of the model with ReLU activation. The upper graphs show the objective function, predicted results and the post-activation of each neuron of 1000 epoch training. In each of the figures, the red and blue lines represent the objective function and the predicted result, while each other line represents the post-activation (upper) and gradient (lower) value of a neuron.

layers frozen, given training sample (x, y), the weight update can be written as

$$\Delta W^i = C(x, y, W^i) \cdot \eta \tag{10}$$

where C is a function of the training sample and current weight. By the nature of the deep neuron network, the complexity of function C complexity grows as the model goes deeper. For a network with piecewise linear activations, according to Theorem 1, the ΔW^i can be reformated as $C'(x, y) \cdot W^i \cdot \eta$. In other words, the training of such networks is dependent only on the provided sample thus more stable.

C. Illustrations

In this section, we provide a unit-level analysis of the neuron network. To be specific, we compare the ability of networks with different activations in approximating an objective function. Given a network with 1d input, 1 hidden layer with 10 neurons, and 1d output, consider an objective function:

$$y = 2\cos(\pi x) + x(x-1) + 1. \tag{11}$$

To investigate how weights affect the training of the model, we initialize the weights and biases of the network with different settings. Denote ReLU(w, b) as a network with ReLU activation function of which the weight and bias are initialized from a random uniform distribution $[-w, w]$ and $[-b, b]$, respectively, and so is the Sigmoid(w,b). During the training, an epoch of sample is set as data pair $\{x_i, f(x_i)\}$ where x_i is an arithmetic sequence in $[-1, 1]$. In Figure 2, we present the contributed value of each neuron, predicted value of the network, and ground-truth value of the function in upper graphs. The predicted value and ground truth value are shown in blue and red as illustrated in the figure, while other curves are the contributed value of each neuron to the output. That is, the sum of other curves equals the predicted value. The lower graphs show the gradient value of each neuron at the y-axis in response to input at the x-axis. All of the models are trained for 1000 epochs with a learning rate of 0.001 at which time the models already reach their convergence.

Among three of the Sigmoid models, Sigmoid(1, 1) shown in 2(a) has the worst performance where almost all of the neurons yield a constant regardless of the input. However, the gradients of the model regarding an epoch of inputs are non-zero. Because of the symmetry and limited range of sigmoid activation function, the gradient of different samples are cancelling each other out, therefore instead of finding the global optimum, the training merely adjusts each neuron as a constant c that $\int (f_i'(x) - c)dx \approx 0$, where f_i' is the gradient of neuron i. The only neuron that can provide useful information is coloured purple due to the relatively high gradients around 1. Apart from that, as the initialization of weight is centred with less diversity, for each of the inputs, the backpropagation process fails to find the steepest direction. Comparing with Sigmoid(1, 1), Sigmoid(15, 0) and Sigmoid(15, 15) have better performances. However, the *dying neuron* issue still exists. For the Sigmoid(15, 0) due to the small initialized bias, most of the post-activation functions are centred around zero, resulting in an underfitting of the non-zero region.

Figure 2(d)-2(f) is the results for ReLU(1, 1), ReLU(15, 1) and ReLU(15, 0). Notice that due to the different forms of ReLU and Sigmoid, we choose different parameter set so that the contributed value of each pair of models are comparable around 0. We find that the three models have similar performance regardless of the initialization. This can be explained by the weight gradients graphs. As the ReLU is known as asymmetric, the ReLU only responds to the given input x at half of its domain, therefore, the gradients of ReLU are not neutralized by the evenly distributed inputs within a defined domain.

To conclude, we find a *dying neuron* issue that occurs when the model is trapped in the local optimum. Similar to that of the gradient vanishing issue, the *dying neuron* issue also prevents the model from updating its weights. However, the gradient vanishing issue is raised due to the over-saturated gradient, while the *dying neuron* issue is because the gradient of the model regarding different samples can cancel each other out so that the model can hardly update. Moreover, once a neuron is trapped into the local optimum, it outputs similar

results regardless of the input. Due to the symmetry of Sigmoid activation, the issue is worse for the Sigmoid net which results in the performance gap between the ReLU net and Sigmoid net.

IV. Generalized Activation Regions and Basic Properties

The analysis of the toy model provides some insights into the differences between the Sigmoid net and the ReLU net. However, we are interested in a more general and practical result for detecting the *dying neuron* issue and comparing the expressivity of networks.

In this section, we aim to provide a toolbox for analyzing the complexity of models with arbitrary activation functions. In particular, we first generalize the definition of activation pattern from piece-wise linear activation to any activation. Then we discuss the basic properties of activation regions locally. At last, we focus on the global information that activation patterns can provide in evaluating model performance.

A. Generalized Activation Pattern / Regions

Consider a network N with a set of parameters θ and the activation function σ. If σ is a piece-wise linear function, it is natural the mapping function from input domain R^{in} to output range R^{out} has piece-wise linearity. The activation pattern and linear region for a network with piece-wise linear activation are defined based on the above observation [35]. To be specific, an activation pattern is an assignment to each neuron of a sign that determines which linear part the neuron is for given inputs.

Here we generalize the definition by specifying the breakpoint of the activation function to provide a way to separate the input domain for a network with an arbitrary activation function.

Definition 1 (Generalized Activation Region)**.** *Denote N as a feedforward network with activation σ and input dimension f_{in}. Given a set of breakpoints $\Gamma = \{\gamma_0, \gamma_1, \ldots, \gamma_{n-1}, \gamma_n\}$, the activation pattern $\mathcal{A}$ regarding the parameters θ is defined as:*

$$\mathcal{A} := \{a_z : z \text{ is a neuron in } N\} \in \{1, \ldots, n-1\}^{\#neurons}$$

For a fixed model with a set of parameters θ and any input $x \in R^{in}$, denote $z_{in}(x;\theta)$, w_z and b_z as input, weight vector and bias for neuron z, respectively. The activation region then can be defined according to the activation function σ, activation pattern $\mathcal{A}$, breakpoints Γ and model parameters θ:

$$\mathcal{R}(\sigma, \mathcal{A}, \theta, \gamma) := \left\{x \in R^{in} | \gamma_{a_z - 1} < z(x;\theta) <= \gamma_{a_z}\right\}, \quad (12)$$

where γ_0 is $-\infty$ and γ_n is ∞.

*For any $x \in R^{in}$, denote the activation pattern of x as $\mathcal{A}(x)$. Moreover, given σ and a set of breakpoints Γ, σ is continues within every interval $(\gamma_0, \gamma_1), (\gamma_1, \gamma_2) \ldots (\gamma_{n-1}, \gamma_n)$, we say the Γ is a **continuous separation** of net N.*

In the following part of this section, for a fixed model with parameters θ and activation σ, given an activation pattern $\mathcal{A}$, breakpoints Γ and an input x within the corresponding activation region. We denote $z^{i,j}_{in}(x; \mathcal{A}, \Gamma, \theta)$ and $z^{i,j}_{out}(x; \mathcal{A}, \Gamma, \theta)$ as the input and output of the jth neuron in layer i. $z^{i}_{in}(x; \mathcal{A}, \Gamma, \theta)$ and $z^{i}_{out}(x; \mathcal{A}, \Gamma, \theta)$ as the input and output vector in layer i. Therefore,

$$z^{i}_{out}(x; \mathcal{A}, \Gamma, \theta) = \sigma(z^{i}_{in}(x; \mathcal{A}, \Gamma, \theta)^T w_z + b_z).$$

For reading convenience, we short them as $z^{i,j}_{in}(x)$ and $z^{i,j}_{out}(x)$.

The generalized activation pattern is an extension of linear activation regions. For instance, given ReLU activation and breakpoints $\Gamma = \{0\}$, it is equivalent to the definition from previous works. On the other hand, it provides a way to separate the input domain into different subspaces where each of the subspaces has a different reaction regarding the input. The generalized activation pattern can be viewed as a toolbox to further investigate the model performance with different activations. Figure 3 shows the activation regions of a Sigmoid net trained on the MNIST dataset. Notice that the activation regions themselves can hardly provide any information since it is just a separation of the input space according to the model. To make it useful for our analysis, it is necessary to illustrate some basic properties.

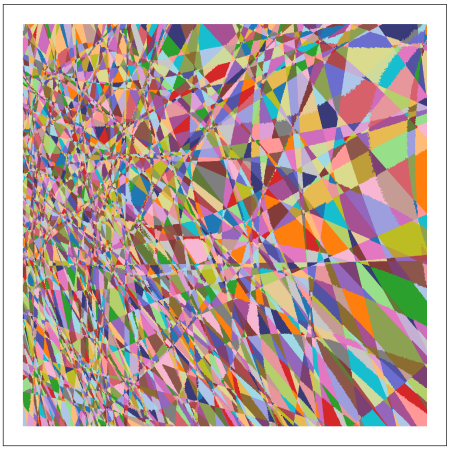


Fig. 3. An illustration of generalized activation region for Sigmoid net with 3 layers, with 64 neurons within each layer. The continuous separation is set as $\Gamma = \{-1, 1\}$.

B. Properties In Single Region

In this section, we first illustrate the properties of a single activation region, including the convexity, continuity and Lipschitz continuity.

Previous works suggest that for a network with piece-wise linear activation and measure zero parameters set θ with respect to Lebesgue measure, the linear activation regions are convex [35]. Here we claim that the convexity of activation regions holds for any monotonous activation function.

Lemma 1. *Denote N as a net with activation σ, if σ is monotonous and Γ is a continuous separation of net N, any activation pattern $\mathcal{R}$ and parameter set θ for N, the activation region $\mathcal{R}(\sigma, \mathcal{A}, \theta, \Gamma)$ is convex.*

With convexity, we know that for any $x_1, x_2 \in R_i$, the segment connecting x_1 and x_2 are completely included in the region. Further, if all the activation regions are convex, the straight line connecting any two points in R^{in} can only cross each activation region once. This property is useful in building the connection between model performance and similarity of activation patterns, which we will discuss in the next section.

Starting with the continuity of the local mapping function, we next consider the Lipschitz continuity in an activation region.

Lemma 2. *Given N as a feed forward network with non-linearity σ and a set of breakpoints Γ, for any activation region defined by $\mathcal{R}(\sigma, \mathcal{A}, \theta, \Gamma)$, the mapping from $\mathcal{R}(\sigma, \mathcal{A}, \theta, \Gamma) \to N(\mathcal{R}(\sigma, \mathcal{A}, \theta, \Gamma))$ is continuous.*

The continuity of the activation region the analysis of Lipschitz property of an activation region. Notice that given a neuron z, its pattern a_z and the activation function σ, the post-activation and of its output with respect to the input vector are bounded, formally:

Lemma 3. *Denote $\mathcal{R}(\sigma, \mathcal{A}, \theta, \Gamma)$ as an activation region defined on network N. For any $x_1, x_2 \in \mathcal{R}(\sigma, \mathcal{A}, \theta, \Gamma)$, there is a Lipschitz constant ρ_i for layer i that:*

$$\|z^i_{in}(x_1) - z^i_{in}(x_2)\|_2 \leq \rho_i \|z^i_{out}(x_1) - z^i_{out}(x_2)\|_2. \quad (13)$$

The Lipschitz constant ρ_i is:

$$\rho_i = \max_z \sup_{x \in R^z_{in}} \frac{\partial \sigma(x)}{\partial x} \cdot \mathit{Norm}(W_i), \quad (14)$$

where z is neuron in layer i, R^z_{in} is the input domain of activation function defined by the pattern of neuron z, i.e., $(\gamma_{a_z} - 1, \gamma_{a_z})$, and Norm$(w_i)$ is the spectral norm of weights in layer i.

By generalizing the above Lemma to the network, we then yield the Lipschitz continuity of the network N within an activation region.

Lemma 4. *Given an activation region $\mathcal{R}(\sigma, \mathcal{A}, \theta, \Gamma)$ where Γ is a continuous separation of net N with depth n, denote the local mapping function defined on R as F_R, then for any $x_1, x_2 \in R$, there exist an upper Lipschitz constant ρ that:*

$$\|F_R(x_1) - F_R(x_2)\| \leq \rho \|x_1 - x_2\|, \quad (15)$$

where

$$\rho = \prod_i^n \rho_i, \quad (16)$$

and ρ_i is the Lipschitz constant of layer i defined in Equation 14.

Empirically, the ρ can also be estimated by [38]:

$$\rho = \sup_{\epsilon, x \in R,} \frac{\| \bigtriangledown f(x)\epsilon\|_2}{\|\epsilon\|_2} = \sup_{x \in R} \|J_f(x)\|_2, \quad (17)$$

$\|J_f(x)\|_2$ is the Jacobian matrix of function f at x.

Above theorem bounds the difference between predictions of data within an activation region given its activation pattern.

C. Properties Across Regions

In this section, we discuss the distance between predictions of data in different activation regions. For a network, N with activation σ, the continuous separation Γ splits the input space into many connected components. Formally, if θ and Γ are measure zero set with respect to Lebesgue measure, the separation is given by hyperplane arrangements in R^{in} where each of the hyperplanes is defined as:

$$H_z(\gamma, \theta) := \{x \in R^{in} | z_{in}(x)^T w_z + b_z = \gamma\} \quad (18)$$

If two activation regions R_1 and R_2 are partition by a single hyperplane, then R_1 and R_2 are adjacent regions. Naturally, the activation pattern for $\mathcal{R}_1$ and $\mathcal{R}_2$ are identical to expect for the pattern of neuron z on which the hyperplane is defined, formally:

Definition 2 (Adjacent Activation Regions)**.** *Denote N is a deep network with activation σ and parameters θ. Γ is a continuous separation of N. $\mathcal{R}_1$ and $\mathcal{R}_2$ are two activation regions, $H_{z_s}(\gamma_s, \theta)$ is a hyperplane defined by neuron z_s and breakpoint γ_s. For any $x_1 \in \mathcal{R}_1$ and $x_2 \in \mathcal{R}_2$, if:*

$$h(x_1, z, \gamma) \cdot h(x_2, z, \gamma) \begin{cases} <0, z = z_s, \gamma = \gamma_s; \\ >0, \textit{otherwise}, \end{cases} \quad (19)$$

where $h(x, z, \gamma) = z_{in}(x)^T w_z + b_z - \gamma$, then $\mathcal{R}_1$ and $\mathcal{R}_2$ are adjacent activation regions separated by hyperplane $H_{z_s}(\gamma_s, \theta)$.

The $h(x, z, \gamma)$ above is introduced to indicate which side of a data is located regarding a hyperplane. Geometrically, two adjacent activation regions can be merged into one by removing the hyperplane that separates them. Therefore, every $x_1 \in \mathcal{R}_1$ and $x_2 \in \mathcal{R}_2$ are at the same side of other hyperplanes except for the one that separates them.

Lemma 5. *If $\mathcal{R}_1$ and $\mathcal{R}_2$ are adjacent activation regions separated by $H_{1,2}$, then for an arbitrary small $\epsilon > 0$, there exist $x_1 \in \mathcal{R}_1$ and $x_2 \in \mathcal{R}_2$ s.t.*

$$D(x_1, x_2) < \epsilon \quad (20)$$

where $D(\cdot, \cdot)$ is a distance metric defined on input space.

Here we remark that the choice of distance metric does not affect our results. In the following of this work, we use $D(\cdot, \cdot)$ as the distance metric on the corresponding space unless specifically stated.

This observation is useful for the analysis of model performance on multi-class classification tasks. Now we start to build a connection between the activation region and model prediction. In particular, we are interested in the bound of prediction differences between data in the same region and data in different regions.

Let N be a network with activation σ and parameters θ that computes a function $F : R^{in} \to R^n$. Γ is a continuous separation of N. Given data $x_1 \in \mathcal{R}_1$, we consider the data inside $\mathcal{R}_1$ and outside $\mathcal{R}_1$ with same distance to x_1. That is, $x_3 \in \mathcal{R}_1$ and $x_2 \notin \mathcal{R}_1$ with $D(x_1, x_2) = D(x_1, x_3)$. Denote l_1 and l_2 are the lines connecting x_1, x_2 and x_1, x_3 respectively.

To simplify our analysis here, we assume that x_2 located in an adjacent region of $\mathcal{R}_1$ which we denote as $\mathcal{R}_2$. $\mathcal{R}_1$ and $\mathcal{R}_2$ are separated by hyperplane $H_{12}(\gamma, \theta)$, and l does not cross any other activation regions. Notice that the above assumption can be removed in the formal statement of the theorem. Denote the piece-wise function on each region as $F_i : R_i \to R^{out}$. From Lemma 4, there is a Lipschitz constant ρ_i for each of the F_i. From Lemma 5, given an arbitrary small $\epsilon > 0$, we can find $m_1 \in R_1$ and $m_2 \in R_2$ on l, so that

$$\begin{aligned} D(m_1, H_{12}(\gamma, \theta)) < \epsilon, \\ D(m_2, H_{12}(\gamma, \theta)) < \epsilon. \end{aligned} \tag{21}$$

where $D(m_i, H_{12}(\gamma, \theta))$ is distance between m_i to the hyperplane $H_{12}(\gamma, \theta)$.

According to Lemma 4 and triangle inequality of norms, when $\epsilon \to 0$ we have:

$$\begin{aligned} \|F(x_1) - F(x_2)\|_2 \leq \rho_1 \|x_1 - m_1\|_2 + \rho_2 \|x_2 - m_2\|_2 \\ \|F(x_1) - F(x_3)\|_2 \leq \rho_1 \|x_1 - x_3\|_2 \end{aligned} \tag{22}$$

For a network with high complexity, almost surely there exists an adjacent activation region of $\mathcal{R}_1$, denoted as $\mathcal{R}_h$, that satisfies $\rho_h > \rho_1$, where ρ_h is the Lipschitz constant of activation region $\mathcal{R}_h$. Therefore, the upper bound of $\|F(x_1) - F(x_3)\|_2$ is lower than that of $\|F(x_1) - F(x_2)\|_2$. By nature, the mapping of network N is equivalent to the combination of all the piece-wise functions. The above equation can be generalized to the network:

Theorem 2. *Let $N : R^{in} \to R^n$ be a network with parameters θ and activation σ. Γ is a continuous separation of N. For any $x \in R^{in}$, denote the activation region on which x is located within as $\mathcal{R}$. Given $r > 0$, almost surely:*

$$\sup_{\substack{x' \in \mathcal{R}, \\ D(x,x')=r}} \|F(x) - F(x')\|_2 \leq \sup_{\substack{x' \notin \mathcal{R}, \\ D(x,x')=r}} \|F(x) - F(x')\|_2. \tag{23}$$

In other words, for network N, the bounds of the spread between the prediction of data in the same activation region are lower than that of data in different activation regions. Similar results can also be obtained for the lower bound. When the number of activation regions is large enough, the volume of each region is bounded [33] and the above theorem can be generalized globally.

Intuitively, for data x and x', if there exist a continuous separation and activation pattern $\mathcal{A}$ that both $x, y \in \mathcal{R}(\sigma, \mathcal{A}, \theta, \Gamma)$, then the differences between predicted result of x and x' has a tighter bound. On the other hand, for any data two x and x', if they are located in activation regions that far apart, then the bound of difference between their prediction is much looser. Therefore, the similarity between activation patterns can be used as a measure of the topological distance between data on the network.

V. Pattern Similiarity and Model Performance

In this section, we aim to utilize the activation pattern to evaluate model expressivity. In particular, we propose a *pattern similarity* metric to evaluate neuron networks based on the neuron level response toward the data. We then illustrate the relationship between our metrics and the performance of a deep network. To build the connection, we use the transition density proposed in earlier work as a bridge. To be specific, we first show that for a dataset with large enough capacity, a high transition density of trajectories connecting pairs of data implies high prediction differences. Then we show that our metric has an inverse relationship with the aggregated transition density.

A. Pattern Similiarity

We start with review the transition density of trajectoreis proposed in earlier work [9]. Given two close points x_1 and x_2 in the input domain, if the activation patterns are different for x_1 and x_2, we say there is a transition between x_1 and x_2. For a one-dimensional trajectory connecting x_0 and x_k in the input space, if we sample $k-1$ equidistant points on the trajectory, the transition density is defined as the number of transitions for the set $\{x_0, x_1, x_2, \ldots, x_k\}$:

$$TD(l_{x_1,x_2}) = \lim_{k \to \inf} \sum_{i=0}^{k-1} \mathbf{Tra}(x_i, x_{i+1}), \tag{24}$$

where

$$\mathbf{Tra}(x_i, x_{i+1}) = \begin{cases} 1 & \mathcal{A}(x_i) \neq \mathcal{A}(x_{i+1}) \\ 0 & \text{elsewise} \end{cases} \tag{25}$$

It was introduced to evaluate the sensitivity of a network by measuring the transition density of a trajectory around real data. However, the metric is not suitable for evaluating model performance on the dataset due to the following reason. First, the average volume of each activation region is relatively small for a network with high complexity, which means empirically the transition density of a trajectory can be imprecise. Second, computing the transition density of numerous trajectories connecting data pairs is a computational task. To address the issue, we introduce our pattern similarity metric.

Definition 3 (Pattern Similiarity)**.** *Denote N as a feedforward network with non-linearity σ and parameters θ. Γ is a continuous separation of net N. θ and Γ are measure zero set with respect to Lebesgue measure. Denote the activation pattern for any $x \in R^{in}$ as:*

$$AP(x; \sigma, \theta, \Gamma) = \{(z, a_z) | z \textit{is a neuron in} N\},$$

shortened as $AP(x)$ if other settings are fixed. The pattern similarity between two data $x_1, x_2 \in R^{in}$ is then defined as:

$$PS(x_1, x_2; \sigma, \theta, \Gamma) = \frac{\#(AP(x_1) \bigcap AP(x_2))}{\#\textit{number of neurons in} N}.$$

Additionally, given a dataset distribution X, the pattern similiarity of dataset X :

$$PS_E(X) = \mathop{\mathbf{E}}_{x_i, x_j \sim X} [\![PS(x_i, x_j; \sigma, \theta, \Gamma)]\!] \tag{26}$$

The pattern similiarity distribution of X for a given $\lambda > 0$ is:

$$PS_D(X, \lambda) = \mathbf{P}(PS(x_i, x_j; \sigma, \theta, \Gamma) > \lambda | x_i, x_j \sim X) \tag{27}$$

Pattern simliarity illustrates how the model responses to single data as well as a dataset. The provided information

is useful in analyzing the model performance. Next, we aim to describe the relationship between pattern similiarity and transition density of trajectories. Lemma 1 suggests that activation regions of a network with monotonous activation function are convex. Here we show that, with convexity, the trnasition density of a straight line can only accross each activation region once.

Lemma 6. *Let N be a network with monotonous activtion σ. Given a continuous seperation Γ of net N, denote: $H_z(\theta, \gamma_z) := \{x \in R^{in} | W_z z_{in} + b_z = \gamma_z\}$ as the hyperplanes defined by neuron z. Then for any two points $x, y \in R^{in}$, the line l that connecting x and y intersects each $H_z(\sigma, \theta, \gamma_z)$ at most once.*

In the following, we denote the set of regions crossed by line l as $S(l)$. As two adjacent activation regions are separated by a hyperplane defined by the state of a neuron, their activation patterns have limited differences. Lemma 6 suggests that, given a continuous separation under certain conditions, a straight line crosses each region only once, which implies a lower transition density always comes with a higher pattern similarity. However, for any two points located in regions far apart, it is hard to determine the qualitative relationship between their pattern similarity and the transition density of the line connecting them. In particular, for a network with a complex structure, there exist *closed* patterns whose regions are measure zero set in the input space. Therefore,

$$TD(l_{x_1,x_2}) + \#(AP(x_1) \bigcap AP(x_2)) = \#\text{neurons in N} \tag{28}$$

does not always hold.

To describe the relationship between transition density and pattern similarity formally, we have to add additional constraints.

Theorem 3. *Let N be a net with monotonous non-linearity σ, measure zero parameter set θ and Γ is continuous speration of net N. Given dataset distribution X_1 and X_2, almost surely following statements are equivalent:*

$$\begin{aligned} \mathop{\mathbf{E}}_{x_i,x_j \sim X_1} [\![TD(l_{x_i,x_j})]\!] \leq \mathop{\mathbf{E}}_{x_i,x_j \sim X_2} [\![TD(l_{x_i,x_j})]\!], \\ PS_E(X_1) \geq PS_E(X_2). \end{aligned} \tag{29}$$

Theorem 3 builds a connection between the pattern similarity metric and trajectory transition density. For any two data in the input space, the lower transition density of the segment connecting them implies higher pattern similarity between the data. The result also holds for dataset distributions. Notice that for the data distribution case, the assumption is satisfied with probability 1 therefore can be removed.

B. Pattern Similiarity and Model Performance

This section aims to build a connection between pattern similarity and model expressive ability as well as generalization.

Given any $x \in R^{in}$, consider $X_1, X_2 \in R^{in}$ with $D(x, X_1) = D(x, X_2)$ and $S(l_1) \subseteq S(l_2)$ where l_1 and l_2 are the lines x, X_1 and x, X_2 respecitvely. With Lemma 6 we know that l_1 and l_2 cross each activation region only once. According to the definition of adjacent activation region, a set of adjacent actviation regions can be merged into one region by removing the hyperplanes seprates them. Therefore, we denote the R_0 as the union of all activation regions crossed by l_1:

$$R_0 = \bigcup_{R_i \in S(l_1)} R_i \tag{30}$$

Then l_1 is within the region R_0. Denote the Lipschitz constant of R_0 as ρ_0, which equals to:

$$\rho_0 = \max_i \rho_i \tag{31}$$

where ρ_i is the Lipschitz constant of $R_i \in S(l_1)$.

Now we consider l_2. Since $S(l_1) \subset S(l_2)$, there are other regions crossed by l_2, which we denote as $R_1, \ldots, R_n$ with $x_2 \in R_n$. Now the comparison between $D(x, x_1)$ and $D(x, x_2)$ can be conducted by Theorem 2:

$$\sup_{x_1} D(F(x), F(x_1)) \leq \sup_{x_2} D(F(x), F(x_2)) \tag{32}$$

By loosing the restraint of input we have:

$$\sup_{x_1} R(x, x_1) \leq \sup_{x_1} R(x, x_2) \tag{33}$$

where $R(x, y) = D(F(x), F(y))/D(x, y)$ is the distance of prediction to distance between data ratio. Formally, we have the following theorem:

Theorem 4. *Let N be a net with monotonous non-linearity σ, measure zero parameter set θ and a continuous speration Γ. Given x, for any $x_1, x_2 \in R^{in}$, if $S(l_{x_1,x_2}) \subset S(l_{x_3,x_4})$, then following statements are equivalent:*

$$\begin{aligned} TD(l_{x,x_1}) &\leq TD(l_{x,x_2}) \\ \sup_{x_1} R(x, x_1) &\leq \sup_{x_2} R(x, x_2) \end{aligned} \tag{34}$$

In particular, given dataset distribution X_1 and X_2, if:

$$\mathop{\mathbf{E}}_{x_i,x_j \sim X_1} [\![TD(l_{x_i,x_j})]\!] \leq \mathop{\mathbf{E}}_{x_i,x_j \sim X_2} [\![TD(l_{x_i,x_j})]\!], \tag{35}$$

then

$$\mathop{\mathbf{E}}_{x_i,x_j \sim X_1} [\![\sup_{x_1} R(x_i, x_j)]\!] \leq \mathop{\mathbf{E}}_{x_i,x_j \sim X_2} [\![\sup_{x_1} R(x_i, x_j)]\!] \tag{36}$$

Combining Theorem 3 and Theorem 4, the connection between activation pattern, transition density and bound of prediction difference can be obtained. To be specific, given data x and x', lower transition density of line l connecting x and x' indicates a tighter upper bound and lower bound of distance between $F(x)$ and $F(x')$, as well as a high pattern similarity between x and x'. In practice, when evaluating dataset capacity is large enough, the statement still holds empirically as shown in Figure 5. Experiment detail will be discussed in section VI.

For any continuous separation, a well-behaved model should satisfy that the difference between predictions is small for data with the same label but large for data with different labels. To illustrate the statement, image a worst case of a deep model. Suppose that there are n classes of data and N different connected components provided by hyperplanes, where $N \gg n$. Given a model with fixed parameters θ, if

there exist a continuous separation Γ satisfies, for any sample (x, y), where y is the label of x:

$$x \begin{cases} \in R_1, & y \neq 1 \\ \notin R_1, & \text{elsewise} \end{cases} \tag{37}$$

Then for any data x_1, x_2 from class $\{2, 3, \ldots, N\}$, we have:

$$\|F_1(x_1) - F_1(x_2)\|_2 \leq \rho_1 \|x_1 - x_2\|_2, \tag{38}$$

where ρ_1 is the Lipschitz constant of R_1.

For a set of data with different labels drawn from region R_i, the above equation constraints the distance of their prediction. In other words, the model fails to distinguish the differences between samples with different labels. Conversely, data with label 1 are distributed within $N-1$ different regions, therefore having lower pattern similarity and looser bound of the prediction distance. By such means, we are able to use pattern similarity as a metric to evaluate the model expressivity and generalization towards different datasets.

One of the difficulties that always encountered during understanding deep learning models is that we hardly know how the neurons act in the black box. The *pattern similarity* fills the gap and can be used for evaluating the model expressivity and generalization. Moreover, it is able to provide a glimpse of the inference mechanism of the model. In the next section, we use experiments to illustrate the usage of our metrics.

VI. Experiments and Discussion

In this section, we first illustrate properties of the pattern similarity metric following our previous discussion, then we return to the objective of this work and evaluate the model with different activations using our metric.

A. Re-examine Over-Saturation

We first re-examine the over-saturation issue for network with batch normalization and weight initialization.

Experiment Settings. We use a stacked fully connected neuron network (FCNN) to examine the over-saturation issue in coincidence with the experiments which discovered the issue. The Sigmoid net and ReLU net both consist of 9 hidden layers, with 256 neurons within each layer, and a softmax logistic regression for the output layer. The cost function we use is the cross-entropy loss which was widely adopted in the classification task. Both of the nets are trained on the MNIST dataset for 24000 iterations with a batch size of 128 using the SGD optimizer and a linear rate scheduler decaying from 0.1 to 0.0001

Post Activation. Figure 1 shows the mean and variance of the post-activation value for 128 fixed test samples and 64 fixed neurons for each layer. For the Sigmoid net, the post-activation of all the layers have a mean around 0.5 and a variance around 0.05. Layer 8 and 0 have the highest variance while the others have a relatively lower variance. For the ReLU net, the mean of post-activation values is around 0.1, and the variance decreases as the model converges. Notice that for any arbitrary neuron set and test sample set, the results are similar.

Weights and Gradients. Next, we compare the weights, gradients and gradient to weight ratio during the training. The objectives are: (1) to further examine the gradient vanishing issue from gradient perspective, and (2) to illustrate Theorem 1 empirically. The experiment setting is same as above.

Figure 4(b) and Figure 4(c) present the absolute value of unit-wise gradient and weights. Generally, during the early and middle of the training, the gradients and weights of both of the nets have similar performance, which means that the differences of unit-wise variance of the gradient to weight ratio between the two nets are not caused by the difference between weights and gradients alone.

Combining the result from Figure 1, Figure 4(b) and Figure 4(c), we find that the neurons in both of the networks are activated, implying the gradient facing the issue is not the major cause of the gap between their performance.

Figure 4(a) compares the $|\frac{\partial L/\partial W}{W}|$ ratio. At each step, after the backward propagation, we compute $|\frac{\partial L/\partial W}{W}|$ for each neuron and average the value every 200 steps. We use the lines to represent the mean and the shaded area to represent the variance of the ratio of neurons at each layer. The solid lines and dashed lines show the ratio of Sigmoid net and ReLU net, respectively. Due to the high ratio variance of the Sigmoid net, we scale the variance to 0.25 for both of the nets for aesthetic reasons.

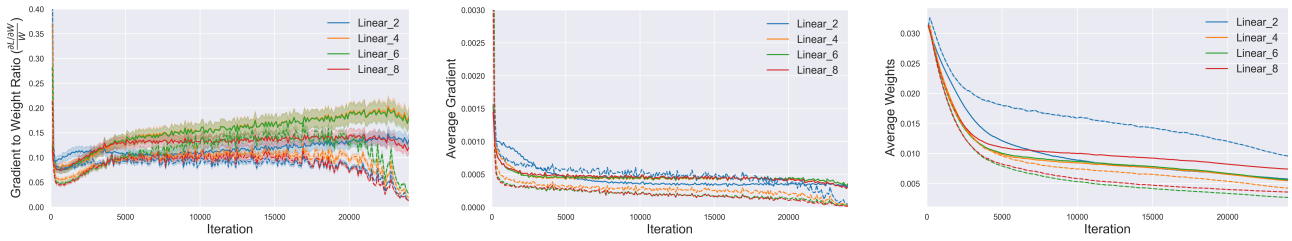
The variance of the gradient to weight ratio can be viewed as an indicator of training stableness. For the ReLU net, the lower variance of the ratio suggests that the gradient are proportional to their current weights at neuron level as Theorem 1 suggested, and the current value of the weights can be viewed as a scale factor of the learning rate. In particular, the variance gradually decreases as the training proceeds. However, for the Sigmoid net, the gradient to weight ratio of neurons high diverge through the training. Equation 6 and 10 provide an explanation of such performance. As the model goes deeper, the weight update ΔW_i is more likely to be affected by its current weight W_i other than the training samples, therefore the training is less stable. The experiment results support our theoretical analysis in Section III.

B. Evaluating Pattern Similarity

We present Figure 5 to illustrate the relationships between pattern similarity, transition density and prediction distance of model with Sigmiod, Tanh, ReLU and GeLU activations.

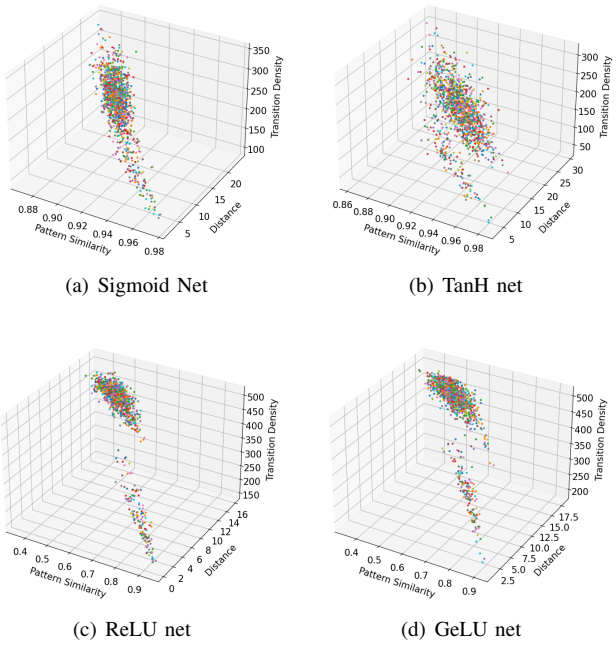
Experiment Settings The model structure and training schedules are the same as that in Section VI-A. Since the choice of distance metric does not affect our result, we use the Euclidean distance to evaluate the prediction difference. The transition density is calculated using 512 points on the line connecting two samples.

Choice of Γ. As the activation pattern is proposed to describe the regional mapping relationship of a network, the Γ for each model is set according to the activation properties. For the Sigmoid-net and Tanh-net, we set $\Gamma_1 = \{-0.5, 0, 0.5\}$, which partition the activation functions into two deactivated regions and one semi-linear region. For the ReLU net and GeLU net, we set $\Gamma_2 = \{0\}$, which partition the activation function into one deactivated region and one activated region.



(a) Gradients to Weights Ratio (b) Unit-wise Average Gradients (c) Average Weights

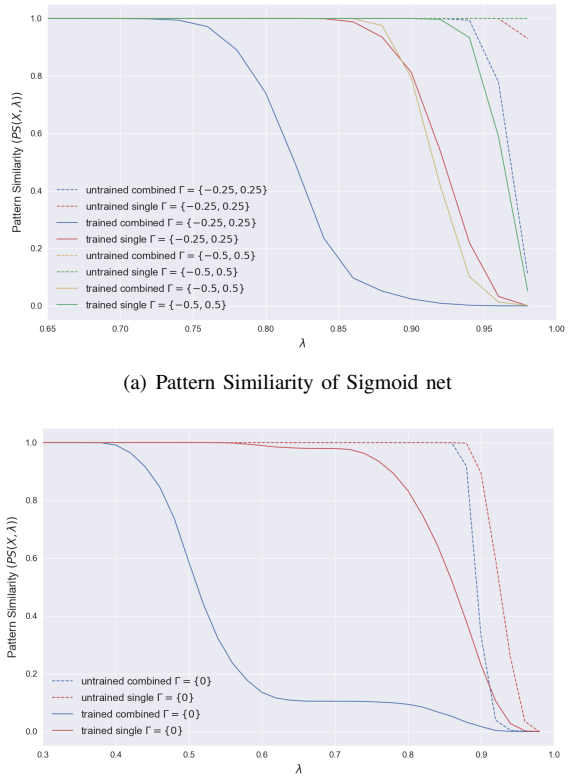
Fig. 4. We present (a) gradient to weight ratio ($\frac{\partial L/\partial W}{W}$), (b) average gradient and (c) average weight for activations across different layers. We use solid lines and dotted lines to represent the Sigmoid net and the ReLU net.



(a) Sigmoid Net (b) TanH net

(c) ReLU net (d) GeLU net

Fig. 5. Distance of predictions, transition density and pattern similiarity of 1225 pair of data.



(a) Pattern Similiarity of Sigmoid net

(b) Pattern Similiarity of ReLU net

Fig. 6. Pattern Similiarity of ReLU net and Sigmoid net calculated on given datasets.

Generally, the samples are clustered into two strips facing the upper left. The points in the upper space mostly consist of pair data with different labels, which have higher transition density, prediction distance and lower pattern similarity, while the points in the lower space are contrary. The results are coincidence with Theorem 3 and 4.

The two clustered strips are separated clearly for ReLU net and GeLU net, suggesting that data with different labels and data with the same labels have distinguishable different metric features. However, for that Sigmoid net and Tanh-net, the data scatters are interspersed together, which means that those networks fail to provide a clear separation for different data.

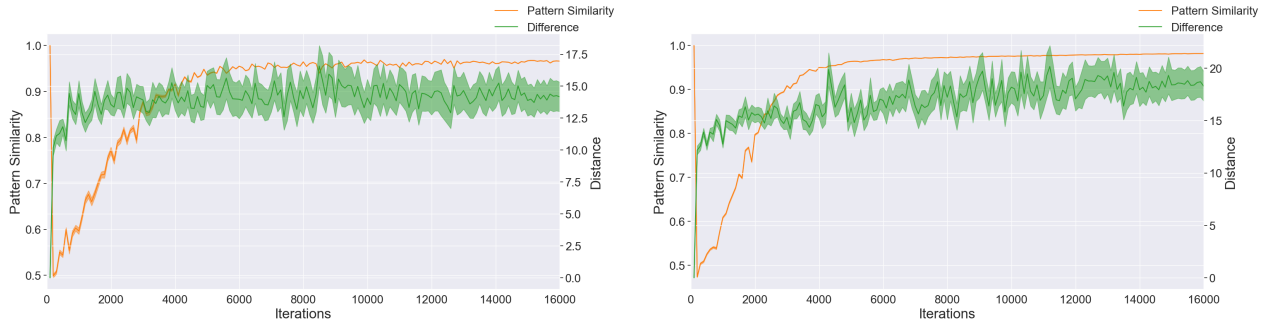
Additionally, we find that the pattern similarity of Sigmoid net and Tanh-net have limited range are relatively higher, suggesting there are fewer activation regions for those networks. As the number of activation regions relates to the expressive power of the network, the Sigmoid-net and Tanh-net have the worse ability in approximating complex functions.

C. *Insights from Pattern Similarity*

Here we further discuss the insights provided by pattern similarity. In particular, we show that the pattern similarity can be viewed as an indicator of the severity of *dying neuron* issue.

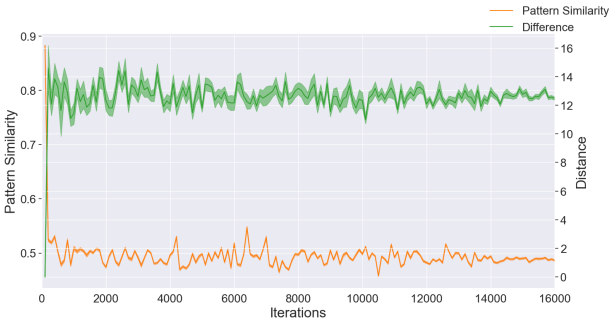
We construct 2 datasets to evaluate the model performance. The *single* dataset is constructed by selecting 1000 samples with label same, while the *combined* dataset is 1000 samples randomly selected from the test dataset. We compare the pattern similarity of the two datasets on the trained and untrained models.

Experiment settings. The model structure and training scheduler is same as previous. We evaluate the pattern similarity using $\Gamma_{s_1} = \{-0.25, 0.25\}$ and $\Gamma_{s_2} = \{-0.5, 0.5\}$ for

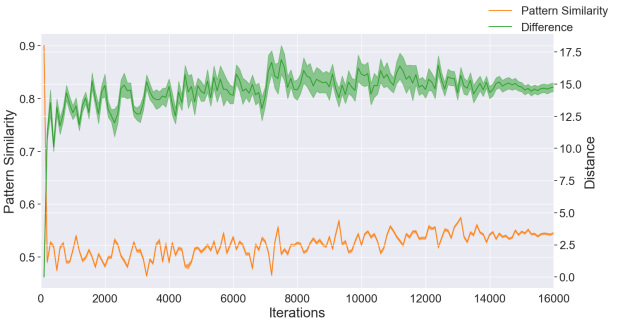


(a) Full training of Sigmoid net

(b) Full training of Tanh net



(c) Full training of ReLU net



(d) Full training of GeLU net

Fig. 7. Pattern similarity, prediction distance of fully connected network with different activations during the training. The network consists of 9 layers fully connected network with 256 neurons within each layer.

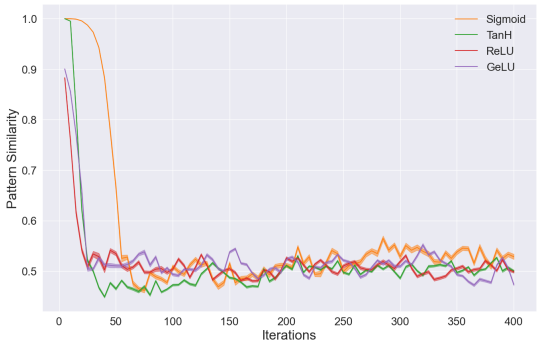


Fig. 8. Pattern Similiarity of fully connected network with different activations at early stage of training. The network consists of 9 layers fully connected network with 256 neurons within each layer.

Sigmoid net and $\Gamma_r = \{0\}$ for ReLU net.

Figure 6 presents the experiment results for Sigmoid net and ReLU net. The gap between pattern similarity of *single* dataset and *combined* dataset can be viewed as an indicator of the practical model expressivity. To be specific, since the pattern similarity measures the neuron level response towards a dataset, a model with high expressivity should be able to distinguish the difference between data with different labels, therefore has low pattern similarity for *combined* dataset and the opposite for the *single* dataset.

The dashed lines show the pattern similarity of the untrained models on different datasets. For both of the models, the dashed lines start to decline at around $\lambda = 0.90$, suggesting the activation pattern of the dataset are around 90 % the same for untrained models.

The blue solid line of 6(a) shows that, after the model is trained, the pattern similarity for the *combined* dataset decreases dramatically compared with the blue dashed line. This means that the model is able to capture the essential features of data with different labels. Meanwhile, for the red solid line, the pattern similarity remains at a high level till λ reaches 0.7, indicating that for data with the same label, around 70% of activation patterns are the same.

On the contrary, 6(a) shows that the expressivity of the Sigmoid net is relatively worse. For both of the separations, the gap between *combined* and *single* is less than 10%. In particular, solid lines remains at 1 until λ reaches around 0.8, which means that for any input data, there are 80% of neurons yield similarity post-activation values. In other words, neurons fail to learn the determinate features of data with different labels. This result in a novel *dying neuron* issue that even most of the neurons are activated, their post-activation value remains in the same region for data with different labels, therefore failing to provide useful information.

D. Expressivity during Training

In this section, we investigate the *dying neuron* issue during the training of neuron networks to compare the expressivity of network with different activations.

Experiment Settings. We consider both stacked fully connected neuron networks (FCNN) and convolutional neuron networks (CNN). The structure and training scheduler of FCNN is the same as we introduced in Section VI-A. The

	MNIST	CIFAR10
Model	FCNN	VGG16
Sigmoid	98.28	82.45
Tanh	98.25	84.30
ReLU	98.45	92.27
GeLU	**99.17**	**92.73**

TABLE I
TEST ACCURACY OF NETWORK DIFFERENT ACTIVATIONS.

CNN we use in this work is the VGG16 network [39] that trained on the CIFAR10 dataset for 120 epochs with a batch size of 128 and SGD optimized with linear decay learning rate from 0.1 to 0.0001. The pattern similarity and prediction distance are computed using fixed 1000 pairs of test data with different labels.

Early Stage. We first discuss the change of pattern similarity in the early stage of the training. For both the FCNN and VGG16 networks, the pattern similarity is recorded every 5 iterations. Figure 8 presents the result of the FCNN network. For all of the models, the pattern similarity gradually decreases to around 0.5. However, the Sigmoid net showed to converge slower.

This phenomenon is even more severe when it comes to a larger network. Figure 9(a) presents the change of pattern similarity in early stage of training for the VGG network. The pattern similarity of the Sigmoid net remains at 100% at the very beginning of training, which means the network fails to learn valid information at the beginning of the training.

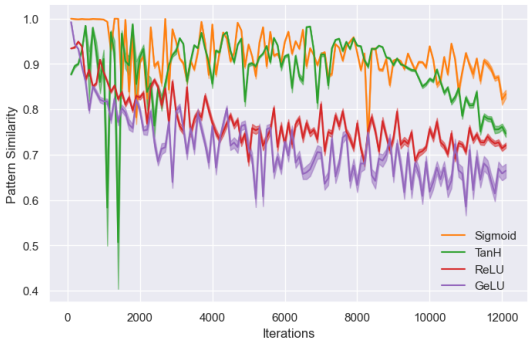
General Performance. We then illustrate the general performance of the models. The metrics are recorded every 100 iterations for the FCNN and every epoch while for the VGG16 network.

Figure 7 shows the change of pattern similarity and prediction distance for the FCNN. For the Sigmoid net and Tanh-net, the pattern similarity first decreases, and then gradually increases to around 0.95, while that of ReLU net and GeLU-net remains at the same level. Additionally, the prediction distance of the Sigmoid net has a higher variance, which means the model performance of the Sigmoid net is less stable.

Figure 9(b) shows the change of pattern similarity of the VGG16 network with different activations. During the training, the pattern similarity of the GeLU-network and ReLU network is relatively more stable. At the end of the training, GeLU-network has the lowest pattern similarity for data with different labels, implying that the GeLU-network has better expressivity.

The expressivity provided by our matrices is directly related to the model performance. Table I shows the test accuracy of different networks. The GeLU activation has the highest test accuracy for both of the networks, with 99.17% on MNIST and 92.73% on CIFAR10, followed by ReLU net with slightly lower accuracy. On the other hand, the Sigmoid net and Tanh net have has lower test accuracy. In particular, combing with the result from Figure 9(b), we find that the accuracy ranking is the same as the pattern similarity ranking. As the pattern similarity is an indicator of the severity *dying neuron* issue, the performance gap between models then can be explained by the *dying neuron*.

(a) Early Stage



(b) Full training

Fig. 9. Pattern Similiarity of VGG16 network with different activations.

VII. CONCLUSION AND FUTURE WORK

This work documents a novel *dying neuron* issue that explains the performance gap caused by the choice of activations. To be specific, the issue refers to the phenomenon that most of the neurons yield similar results for any data, which resulted in a loss of practical expressivity. Different from the gradient vanishing issue, the gradient of *dying neuron* is non-zero, but still fails to provide useful information.

In Section III, we first show that with state-of-art techniques, the gradient vanishing issue is no longer the major reason that causes the poor performance of Sigmoid nets. By investing in the learning dynamic of a deep network, we find that the training of certain networks is less stable. In particular, once a neuron is trapped into the local optimum region, any update of weights are cancelling each other out and the neuron can hardly escape the region.

To better understand the issue, we introduce the generalized activation pattern as a toolbox and discuss some basic properties of the activation pattern in Section IV. Based on our discussion, we propose a metrics named pattern similarity to evaluate the expressivity of neuron networks in Section V. The pattern similarity measures the neuron level response toward the input. Given a model, high pattern similarity of data with different labels implies that most of the neurons in the model fails to distinguish the difference between inputs. In other words, the network suffers a severe *dying neuron* issue.

In Section VI we illustrate the properties of the proposed metric and investigate the *dying neuron* issue. We find that the *dying neuron* issue widely exists, which is the reason that network has practically less expressivity than theoretical. Apart from that, the severity of the *dying neuron* issue can explain the difference between the model performance.

The objective of this work is to analyze the expressivity of neuron network expressivity from an activation perspective. In particular, we provide some insights on explaining the difference in model performance caused by activation functions with proposed tools and metrics. Interesting directions of future works include investigating the regional performance and illustrating the mapping relationship of the network using the activation patterns.

VIII. Acknowledgement

This work is supported in part by the Huawei Technologies Co., Ltd under Grant HIRP2019041002010, the UK EPSRC under Grant EP/P009727/1, and the Leverhulme Trust under Grant RF-2019-492.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[2] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[4] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[5] S. Park, C. Yun, J. Lee, and J. Shin, "Minimum width for universal approximation," *arXiv preprint arXiv:2006.08859*, 2020.

[6] 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*, pp. 249–256, 2010.

[7] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[8] 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*, pp. 1026–1034, 2015.

[9] R. Novak, Y. Bahri, D. A. Abolafia, J. Pennington, and J. Sohl-Dickstein, "Sensitivity and generalization in neural networks: an empirical study," *arXiv preprint arXiv:1802.08760*, 2018.

[10] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315–323, 2011.

[11] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Icml*, 2010.

[12] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[13] L. Lu, Y. Shin, Y. Su, and G. E. Karniadakis, "Dying relu and initialization: Theory and numerical examples," *arXiv preprint arXiv:1903.06733*, 2019.

[14] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, "Learning activation functions to improve deep neural networks," *arXiv preprint arXiv:1412.6830*, 2014.

[15] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[16] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," *arXiv preprint arXiv:1511.00363*, 2015.

[17] B. Xu, R. Huang, and M. Li, "Revise saturated activation functions," *arXiv preprint arXiv:1602.05980*, 2016.

[18] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[19] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[20] P. Ramachandran, B. Zoph, and Q. V. Le, "Searching for activation functions," *arXiv preprint arXiv:1710.05941*, 2017.

[21] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*, pp. 1310–1318, 2013.

[22] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, *et al.*, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," 2001.

[23] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[24] S. Santurkar, D. Tsipras, A. Ilyas, and A. Mkadry, "How does batch normalization help optimization?," in *Proceedings of the 32nd international conference on neural information processing systems*, pp. 2488–2498, 2018.

[25] G. Yang, J. Pennington, V. Rao, J. Sohl-Dickstein, and S. S. Schoenholz, "A mean field theory of batch normalization," *arXiv preprint arXiv:1902.08129*, 2019.

[26] J. Kohler, H. Daneshmand, A. Lucchi, T. Hofmann, M. Zhou, and K. Neymeyr, "Exponential convergence rates for batch normalization: The power of length-direction decoupling in non-convex optimization," in *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 806–815, PMLR, 2019.

[27] H. W. Lin, M. Tegmark, and D. Rolnick, "Why does deep and cheap learning work so well?," *Journal of Statistical Physics*, vol. 168, no. 6, pp. 1223–1247, 2017.

[28] J. Lee, L. Xiao, S. Schoenholz, Y. Bahri, R. Novak, J. Sohl-Dickstein, and J. Pennington, "Wide neural networks of any depth evolve as linear models under gradient descent," *Advances in neural information processing systems*, vol. 32, pp. 8572–8583, 2019.

[29] P. L. Bartlett, "The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network," *IEEE transactions on Information Theory*, vol. 44, no. 2, pp. 525–536, 1998.

[30] M. Telgarsky, "Benefits of depth in neural networks," in *Conference on learning theory*, pp. 1517–1539, PMLR, 2016.

[31] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, "The expressive power of neural networks: A view from the width," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 6232–6240, 2017.

[32] R. Pascanu, G. Montufar, and Y. Bengio, "On the number of response regions of deep feed forward networks with piece-wise linear activations," *arXiv preprint arXiv:1312.6098*, 2013.

[33] B. Hanin and D. Rolnick, "Complexity of linear regions in deep networks," in *International Conference on Machine Learning*, pp. 2596–2604, 2019.

[34] G. F. Montufar, R. Pascanu, K. Cho, and Y. Bengio, "On the number of linear regions of deep neural networks," in *Advances in neural information processing systems*, pp. 2924–2932, 2014.

[35] B. Hanin and D. Rolnick, "Deep relu networks have surprisingly few activation patterns," in *Advances in Neural Information Processing Systems*, pp. 361–370, 2019.

[36] P. Bartlett, D. J. Foster, and M. Telgarsky, "Spectrally-normalized margin bounds for neural networks," *arXiv preprint arXiv:1706.08498*, 2017.

[37] B. Neyshabur, S. Bhojanapalli, and N. Srebro, "A pac-bayesian approach to spectrally-normalized margin bounds for neural networks," *arXiv preprint arXiv:1707.09564*, 2017.

[38] H. Federer, *Geometric measure theory*. Springer, 2014.

[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.