Not merely Memorization in Deep Networks: Universal Fitting And Specific Generalization

Xiuyi Yang

***Abstract*—We reinterpret the training of convolutional neural nets(CNNs) with universal classification theorem(UCT). This theory implies any disjoint datasets can be classified by two or more layers of CNNs based on ReLUs and rigid transformation switch units(RTSUs) we propose here, this explains why CNNs could memorize noise and real data. Subsequently, we present another fresh new hypothesis that CNN is insensitive to some variant from input training data example, this variant relates to original training input by generating functions. This hypothesis means CNNs can generalize well even for randomly generated training data and illuminates the paradox Why CNNs fit real and noise data and fail drastically when making predictions for noise data. Our findings suggest the study about generalization theory of CNNs should turn to generating functions instead of traditional statistics machine learning theory based on assumption that the training data and testing data are independent and identically distributed(IID), and apparently IID assumption contradicts our experiments in this paper.We experimentally verify these ideas correspondingly.**

***Index Terms*—generalization, RTSU, generating functions.**

I. Introduction

With the dramatically growth in power of computer, data generated by electronic devices and the invention of efficient training techniques of several hidden layer neural nets[1], deep learning has proven to be powerful for a wide range of problems, including fields of computer vision, such as image classification [2], [3] and object detection [4], [5], fields of natural language processing, for instance, speech recognition[6], [7] and Speech Synthesis[8], [9], reinforcement learning [10], [9], [11], even in predicting DNA-protein binding[12], organic chemistry reactions[13],drug discovery[14], Fluid Simulation[15].

Following the strong capability of deep networks, there has been some theoretical work eager to find the source of its powerful generalization ability. Some theoretical works have focused on the loss surface of neural nets [16], [17], [18]. Some other theoretical research have placed emphasis on universal approximation power of neural networks, earlier works [19] prove universal approximation for functions . Lately, in[20] by counting the number of regions of linearity to tells us how well the nets can approximate arbitrary curved shapes. In contrast, [21] proposed a new measure, based on Betti numbers, to show that the expressive power of deep nets is superior to its shallow counterparts. [22] has shown that threshold network may require an exponentially larger number of hidden unit so it can capture the decision boundary of a two-layer ReLU network . Similarly, [23] specifically study the topology of classification regions created by deep nets, as well as their associated decision boundary.[24] propose a new measure based on an interrelated set of measures of expressivity to the neural network expressivity, they find that the complexity of the neural nets grows exponentially with depth. However, all these theoretical justification still can not convincingly bridge the gap between the empirical success of neural nets and understanding of its excellent generalization performance. why methods such as SVM and logistic classification based on mature theory not work so well as CNNs?

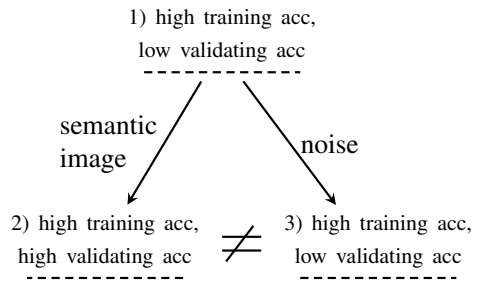


Figure 1. Inequality sign means inconsistent. 1) A typical feature when CNNs overfitting on CIFAR10 or random data. After explicit or implicit regularization, 2) CNNs could generalize well on CIFAR10, 3) but only fit random noise data with bad generalization, by [25].

In supervised learning applications the traditional view of generalization refers to the ability of the learned algorithm to fit previously unseen instance and if the model is excessively complex to memorize the training data, then it is overfitting. A model that has been overfitted has poor predictive performance, due to that it overreacts to noise fluctuations in the training data. Yet it is not well-justified that CNNs often achieve excellent generalization performance with excessively complex model. [25] conducts experiments to show that CNNs are rich enough to memorize the training data regardless of the type of training data. They argue that all kinds of regularization techniques in deep learning should not be counted that important as in traditional machine learning . Then building on this work, [26] proposes that CNNs always learn simple pattern first then fits noise, they also note that data maybe play a important role in memorization and generalization in CNNs. Though there still has a pending issue that CNNs fit random noise input datasets and pattern datasets equally, in real semantic datasets it does a good job, the other it does not, see Figure 1.

In this paper, We're going to aim at settle the generalization divergency of CNNs between benchmark data and made-up data. Except this, we discover theorem presented [27], which is named by us as UCT for ReLUs and orthogonal bidirectional

rectifier (orthogonal BReLUs) nets, could explain why CNNs fits any kinds of data. For validating the validity of UCT further, we presents a new kind of activation units that share UCT as CNNs based on ReLUs and orthogonal BReLUs.

The rest of this paper are organised as follows:

(1) Section II mainly focus on training periods of CNNs, we demonstrate that layers of CNN with units that satisfy universal classification property can classify any multiple disjoint sets ignoring its generalization ability. If no otherwise specified, "classify" means classify training dataset and ignore the performance of model on test dataset.
we present a new kind of units in subsection II.1, RTSUs and prove this units to satisfy UCT. Experimentally verify that wide residual network based on RTSUs can classify any disjoint datasets in II-C, including random datasets generated from Gaussian distribution, binary CIFAR10 datasets constructed from CIFAR10 and stardard benchmark CIFAR10/100. As a side effect, we show CNN based on RTSUs is comparable with BReLUs and Concatenated rectified Linear Units (CReLUs) [28] working on benchmark datasets CIFAR10/100.

(2) The second mainly episode lay stress on generalization of CNNs. We put forward a conjecture that Functions represented by CNNs trained with optimization method map input data and this data under some transformation invariably. This conjecture could reasonably explain performance diversity of CNNs on benchmark and made-up datasets as in [25]. And then, experiments were conducted to validate this hypothesis and resolve inconsistency in Figure 1. (Section III).

(3) Finally, Section IV concludes the paper with possible problem and future works.

II. Universal Classification

[27] attributed the excellent empirical performance of rectifier neural networks to UCT that ReLU nets could transform disjoint pattern datasets to linearly separable. Although we don't argue this theorem could explain the extremely good generalization performance, instead, it could indeed provide an illuminating insight into why DNNs could perfectly fit in arbitrary disjoint finite training datasets.

A. Universal Classification Theorem

Theorem II.1 (Universal Classification Theorem)**.** *Any two disjoint subsets $\mathcal{X}_1$ and $\mathcal{X}_2$ in $\mathbb{R}^n$ can be transformed to be linearly separable through a cascade of two ReLUs, which require $L_1(L_2+1)$ or less ReLUs if $\mathcal{X}_1$ and $\mathcal{X}_2$ have a disjoint convex hull decomposition with L_1 subsets of $\mathcal{X}_1$ and L_2 subsets of $\mathcal{X}_2$ [27].*

As in [27], a cascade of two layers of orthogonal bidirectional ReLUs possess similar theorem. In Section II-B, we will present a new kind of units that have this property. Right now, let's focus on explaining experiments by [25] by this theory.

A finite training datasets always has a trivial disjoint convex hull decomposition that each subset only includes a element of datasets[27]. In practical, every training datasets is finite, so this implies a cascade of two ReLUs can classify any two finite datasets, whether semantic datasets as image or datasets by sampling Gaussian or Uniform distribution. Replacing dataset labels with random ones regard as modification of the disjoint convex hull decomposition of training data and associated linear classifier needed to transform datasets.

Another possible outcome is that, comparing with regular datasets, random noise datasets with the same quantity and dimension need more ReLUs to transform it into being linearly separable, in another words, requiring neural nets with more parameters.

Next, we will present RTSU that satisfy UCT.

B. Rigid Transformation Switch Units

Definition II.1. RTSUs are defined as

$$\boldsymbol{f}(x)=\begin{cases}\boldsymbol{f}_1(x), & \text{for } x\le 0,\\ \boldsymbol{f}_2(x), & \text{for } x>0;\end{cases} \tag{1}$$

Denote $\boldsymbol{f}_i(x)=\boldsymbol{R}\boldsymbol{k}_i(x)+\boldsymbol{s}, i=1,2$ as a rigid transformation, $\boldsymbol{R}=\begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}$ is an rotation matrix, s is shift vector on a ray in $\mathbb{R}^2$.

For similarity, we let $\boldsymbol{k}_1(x)=\begin{bmatrix}0\\ x\end{bmatrix}$ and $\boldsymbol{k}_2(x)=\begin{bmatrix}x\\ 0\end{bmatrix}$, define

$$\boldsymbol{k}(x)=\begin{cases}\boldsymbol{k}_1(x), & \text{for } x\le 0,\\ \boldsymbol{k}_2(x), & \text{for } x>0;\end{cases} \tag{2}$$

so, $\boldsymbol{f}(x)=\boldsymbol{R}\boldsymbol{k}(x)+\boldsymbol{s}$.

We only prove convex separability to linear separability under two disjoint sets, further statements depend on this one applying similar tricks as in [27].

Firstly, we let $\boldsymbol{R}$ equal to identity matrix and $\boldsymbol{s}=0$, under this condition, $\boldsymbol{f}(x)=\boldsymbol{k}(x)$.

Lemma II.2. *If two convexly separable sets can be transformed to be linearly separable through a layer of RTSU units, then a cascade of two layers of RTSU nets could transform any two or more disjoint sets into linearly separable sets.*

Proof. It can easily be proved by the work [27] □

In order to prove the convexly separable of $\boldsymbol{f}(x)$, we start from simpler $\boldsymbol{k}(x)$.

Theorem II.3. *let $\mathcal{X}_1$ and $\mathcal{X}_2$ be two convexly separable sets with a finite number of points in $\mathbb{R}^n$, $CH(\mathcal{X}_1)\cap\mathcal{X}_2=\emptyset, \mathcal{X}_2=\bigcup_{j=1}^{L_2}\mathcal{X}_2^j$, with $CH(\mathcal{X}_1)\cap CH(\mathcal{X}_2^j)=\emptyset$; let $\boldsymbol{w}_j^T\boldsymbol{x}+b_j$ be a set of linear classifier of $CH(\mathcal{X}_2^j)$ and $CH(\mathcal{X}_1)$, such that for any $j\in[L_2]$.*
$\boldsymbol{w}_j^T\boldsymbol{x}+b_j\le 0$, for $\boldsymbol{x}\in\mathcal{X}_1$
$\boldsymbol{w}_j^T\boldsymbol{x}+b_j>0$, for $\boldsymbol{x}\in\mathcal{X}_2^j$
Denote
$\boldsymbol{W}\triangleq[\boldsymbol{w}_1,\boldsymbol{w}_2,...,\boldsymbol{w}_{L_2}]$
$\boldsymbol{b}\triangleq[b_1,b_2,...,b_{L_2}]^T$
$[\boldsymbol{k}_j(x)]\triangleq[\boldsymbol{k}_1(x),\boldsymbol{k}_2(x),...,\boldsymbol{k}_{L_2}(x)]^T, \boldsymbol{k}_j(x)=\boldsymbol{k}(\boldsymbol{w}_j^T\boldsymbol{x}+b_j)$.
$\mathcal{Z}_l\triangleq\{\boldsymbol{z}=[\boldsymbol{k}_j(x)]:\boldsymbol{x}\in\mathcal{X}_l\}, l=1,2$.
Thus, $\mathcal{Z}_1$ and $\mathcal{Z}_2$ are linearly separable.

Proof. it could be easily seen that $CH(\mathcal{Z}_1)$ is a subset of convex set $\{[\mathbf{0}, \boldsymbol{a}], \boldsymbol{a} \leq \mathbf{0}\}$ and $\mathcal{Z}_2^j$ is in the form of $\{[\boldsymbol{b}, \boldsymbol{c}], b_j > 0, c_j = 0\}$, $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c} \in \mathbb{R}^{L_2}$. For $\boldsymbol{a} \leq \mathbf{0}$, every entries in vector $\boldsymbol{a}$ is less than or equal to vector $\mathbf{0}$. According the definition of convex hull, we can easily prove $CH(\cup_j^{L_2} \mathcal{Z}_2^j) \cap \{[\mathbf{0}, \boldsymbol{a}], \boldsymbol{a} < \mathbf{0}\} = \emptyset$, due to the fact that there exists $b_j > 0$.

□

Theorem II.4. *Let $\mathcal{X}_1$ and $\mathcal{X}_2$ be two convexly separable sets with a finite number of points in $\mathbb{R}^n$, $CH(\mathcal{X}_1) \cap \mathcal{X}_2 = \emptyset$, $\mathcal{X}_2 = \bigcup_{j=1}^{L_2} \mathcal{X}_2^j$, with $CH(\mathcal{X}_1) \cap CH(\mathcal{X}_2^j) = \emptyset$; let $\boldsymbol{w}_j^T \boldsymbol{x} + b_j$ be a set of linear classifier of $CH(\mathcal{X}_2^j)$ and $CH(\mathcal{X}_1)$, such that for any $j \in [L_2]$.*

$\boldsymbol{w}_j^T \boldsymbol{x} + b_j \leq 0$, for $\boldsymbol{x} \in \mathcal{X}_1$

$\boldsymbol{w}_j^T \boldsymbol{x} + b_j > 0$, for $\boldsymbol{x} \in \mathcal{X}_2^j$

Denote

$\boldsymbol{W} \triangleq [\boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_{L_2}]$

$\boldsymbol{b} \triangleq [b_1, b_2, ..., b_{L_2}]^T$

$[\boldsymbol{f}_j(x)] \triangleq [\boldsymbol{f}_1(x), \boldsymbol{f}_2(x), ..., \boldsymbol{f}_{L_2}(x)]^T$, $\boldsymbol{f}_j(x) = \boldsymbol{f}(\boldsymbol{w}_j^T \boldsymbol{x} + b_j)$, $\boldsymbol{f}$ is RTSU.

$\mathcal{Z}_l \triangleq \{\boldsymbol{z} = [\boldsymbol{f}_j(x)] : \boldsymbol{x} \in \mathcal{X}_l\}, l = 1, 2.$

Then, $\mathcal{Z}_1$ and $\mathcal{Z}_2$ are linearly separable.

Proof. Let RTSUs with $\boldsymbol{s} = 0$, points after RTSUs transformation could be represented as in $\mathbb{R}^{2L_2}$,

$$\begin{bmatrix} \boldsymbol{f}_1(x) \\ \boldsymbol{f}_2(x) \\ \vdots \\ \boldsymbol{f}_{L_2}(x) \end{bmatrix} = \begin{bmatrix} \boldsymbol{R} & \mathbf{0} & \dots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{R} & \dots & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \boldsymbol{R} \end{bmatrix} \begin{bmatrix} \boldsymbol{k}_1(x) \\ \boldsymbol{k}_2(x) \\ \vdots \\ \boldsymbol{k}_{L_2}(x) \end{bmatrix}$$

$\boldsymbol{R}$ is orthogonal transformation, $\boldsymbol{k}_i$ is as Definition II-B. We rewrite this as, $\boldsymbol{f} = \boldsymbol{Q}\boldsymbol{k}$. Thus, $\boldsymbol{Q}$ is orthogonal transformation in $\mathbb{R}^{2L_2}$.

So,

$$\mathcal{Z}_1 \subset \boldsymbol{Q}\{[\mathbf{0}, \boldsymbol{a}], \boldsymbol{a} \leq \mathbf{0}\}$$
$$\mathcal{Z}_2 \subset \boldsymbol{Q}\{[\boldsymbol{b}, \boldsymbol{c}], b_j > 0, c_j = 0\}$$

$\boldsymbol{Q}\{[\mathbf{0}, \boldsymbol{a}], \boldsymbol{a} \leq \mathbf{0}\}$ and $\boldsymbol{Q}\{[\boldsymbol{b}, \boldsymbol{c}], b_j > 0, c_j = 0\}$ means $\boldsymbol{Q}$ acts on every element in sets. We have already proved convex sets $\{[\mathbf{0}, \boldsymbol{a}], \boldsymbol{a} < \mathbf{0}\}$ and $\{[\boldsymbol{b}, \boldsymbol{c}], b_j > 0, c_j = 0\}$ are linearly separable, $\boldsymbol{Q}$ is orthogonal transformation in $\mathbb{R}^{2L_2}$, thus $\mathcal{Z}_1$ and $\mathcal{Z}_2$ are linearly separable.

The theorem still holds under $\boldsymbol{s} \neq \mathbf{0}$, because shift does not affect linear separability of two pattern sets.

□

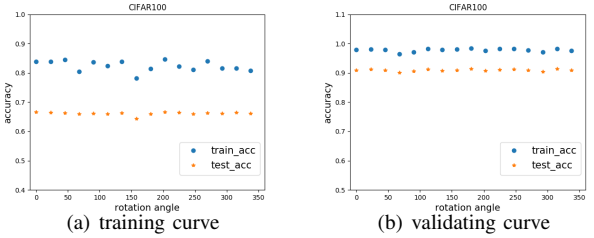
Theorem II.5. *Any two disjoint sets can be transformed to be linearly separable through a cascade of two RTSU layers.*

Proof. From lemma II.2 and theorem II.4, this theorem holds.

□

Concatenated rectified Linear Units (CReLUs) [28] and Bidirectional rectifier (BReLUs) [27] both are defined as

$$\boldsymbol{f}(x) = \begin{bmatrix} \max(0, \quad x) \\ \max(0, -x) \end{bmatrix} = \begin{bmatrix} \text{ReLU}(x) \\ \text{ReLU}(-x) \end{bmatrix} \tag{5}$$



(a) training curve (b) validating curve

Figure 2. verify RTSUs could classify standard benchmark

It is easy to see these units belong to RTSUs, and thus similar universal classification theorem works for these units.

C. Experiments

This section will cover experimental verification of universal classification power of CNNs based on RTSUs. Due to RTSUs are a new kind of units, we will firstly show its performance in benchmark datasets CIFAR10/100 in a standard way. We will cover testing on random noise datasets and nonstandard CIFAR10 later.

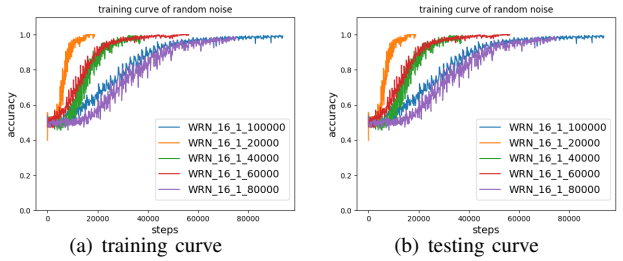
1) RTSU could classify benchmark datasets: We use wide residual network (WRN) W-16-1 with RTSU to compare with BReLU\ CReLU counterparts. The dataset was preprocessed with featurewise zero center and featurewise standard normalization to conduce to optimize and was augmented with random flip left or right and padding 4 pixel around so that we can crop to keep the image size still 32×32. Use Momentum for optimization. The mini-batch size was 64. The momentum term learning rate was fixed to 0.9. The initial learning rate was set to 0.1 and decreased by a factor of 10 after 60 epochs iterations.

In order to cover rotation and shift as thoroughly as possible, for RTSU, we set rotation angle evenly increased by 22.5 degree and added a uniform distributed angle between 1 and -1 degree and then shifted x-axis and y-axis uniform random between -1.5 and 1.5 for numerical stability.

Result is illustrated in Figure 2. That shows nets based on RTSUs could classify benchmark datasets as BReLU\ CReLU with the orthodox way. This is the first step towards demonstrating universal classification power of nets based on RTSUs and we will move on.

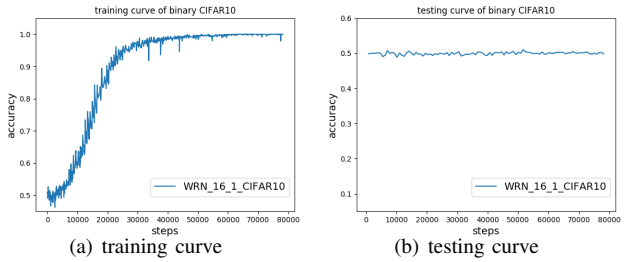
2) RTSU Could Classify Any Datasets: This section, we will show the validity of that RTSU nets could classify any data sets including non semantic pattern dataset or visually identical category with different label, "classify" here means that training data accuracy close to absolutely correct. We want to demonstrate the dilemma created by that traditional statistics machine theory encourage heavy reliance on training samples and testing samples coming from the same distribution, but obviously, these experiments below, all samples came from same distribution actually or assume it as in statistics machine learning theory. This predicament as well as work by [25] trigger our idea about neural nets generalization ability and its discussion are deferred to the next section **??**.

We conduct two sets of experiments exploiting W-16-1 as backbone, first one correspond to random data generated



(a) training curve (b) testing curve

Figure 3. verify classification of Gaussian



(a) training curve (b) testing curve

Figure 4. verify classification of binary CIFAR10

by standard Gaussian distribution and divide the data into two subsets equally, then mark it with binary labels, we carry out five tests by increasing training data number from 20000 to 100000 by 20000, meanwhile, generating test data number in the same way, fixing its number as 5000, setting learning rate as 0.1 changelessly, no data augmentation or preprocessing. The other experiment employ which we name it as binary CIFAF10, by halving each category in CIFAR10 dataset and stamp them with 0 or 1 labels similarly as in first experiment. But in order to facilitate the optimization of network, we standardize the data by featurewise centering and normalization and configure learning rate as 0.03. Both sets of experiments have vanishing weight decay, finish training until 100 epochs with batch size 64, apply with RTSU with rotation angle uniform between 0 and 360 degree, random shift x axis (-1.5, 1.5), random shift y axis (-1.5, 1.5) as in II-C1.

Results are presented in 3, 4. These two experiments demonstrate that RTSU based neural nets could classify any datasets as ReLU based nets, though test accuracy is no better than random guess and suggest that we need to seek out another maybe novel theory to interpret this phenomenon of huge discrepancy between training and test accuracy, which orthodox machine learning theory blames it on overfitting, though, this theory is inconsistent with previous experimental results by [25], the model owning the same capacity/parameters can fit semantic clustering data like CIFAR10 or random noise even with different kinds of regularization techniques. These techniques uses to be thought as relief or cure of overfitting.

III. Specific Generalization

The previous section focused on the universal classification power of DNNs and the validity of UCT is verified by our experiments. However, UCT is insufficient for explanation discrepancy of generalization behaviors of DNNs action on different kinds of datasets. So, in this section we aims at illustration on what are being actually learned by DNNs after training? We do not presents any theoretical stuff to prove some bounds or visualization technique to weights or activation of DNNs as most previous work but adopt a novel hypothesis. This hypothesis let us resolve the contradiction as in 1 explained by a overfitting regularization paradigm prevalence in traditional statistics machine learning theory.

We presents several definitions. Decision domain(DD) created by functions for a DNN function f after successful training on a training set $(\mathcal{X}, \mathcal{Y})$ is define by

Definition III.1. $\mathrm{DD}(\mathcal{X}, g) = \cup_{\mathcal{X},g}\{g(x) \mid f(x) = f(g(x)) = y,\ (x, y) \in (\mathcal{X}, \mathcal{Y}),\ \mathrm{g}\ \text{is a function}\ \}$

For a specific class label $y = k$, $\mathcal{X}_k$ means all training samples with label k, we define

Definition III.2. $\mathrm{DD}(\mathcal{X}_k, g) = \cup_{\mathcal{X}_k,g}\{g(x) \mid f(x) = f(g(x)) = k,\ (x, k) \in (\mathcal{X}, \mathcal{Y}),\ \mathrm{g}\ \text{is a function}\ \}$

Similarly, for a certain training samples $(x, y) \in (\mathcal{X}, \mathcal{Y})$, we define

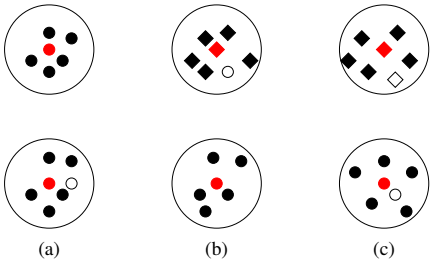
Definition III.3. $\mathrm{DD}(x, g) = \cup_{g}\{g(x) \mid f(x) = f(g(x)) = y,\ (x, y) \in (\mathcal{X}, \mathcal{Y}),\ \mathrm{g}\ \text{is a function}\ \}$

We name function g as **generating function**, and denote all such functions g as set $\mathcal{G}$, After the neural network were trained, the set $\mathcal{G}$ is changeless. Broadly, the scope of function g is subtle and intricate, decide What kind of test sample can be generalized? The identity function is a trivial element of $\mathcal{G}$, we want more than that, but not too more. In this paper, We're not attempting to figure out all of elements of $\mathcal{G}$, instead, we will authenticate that the elements of $\mathcal{G}$ consist of more than trivial identity function. We make a hypothesis that test samples in benchmark datasets such as CIFAR10/100, Imagenet live in this space $\mathrm{DD}(\mathcal{X})$, the success of DNNs in these benchmark could be explained with this hypothesis. [25] conducted several experiments with the modifications of the labels and input images, these experiments share the analogous behavior as we did in Section II. We attribute the extreme disparity between training accuracy and test accuracy to that test samples in these experiments are beyond of the space $\mathrm{DD}(\mathcal{X})$ of trained networks.

We will chase down some typical functions g. Inspiration from DNNs trained on real semantic image should performance well in affine transformation, Gaussian smooth, flip left right, rotation, adding small norm noise of images, we will adopt these as experimental functions. Besides these functions, we will adopt random shuffle training image as function g. From these functions and different types of training samples, test samples are constructed, then the performance of DNN are verified on these test examples.

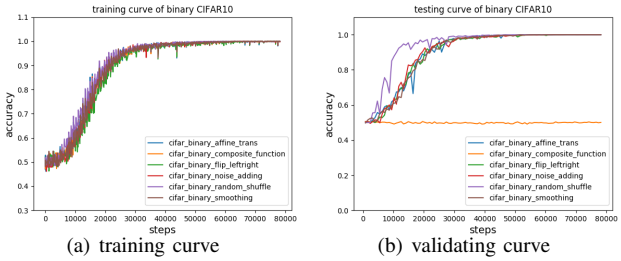
A. Experiments

We adopt W-16-1 as previously, all the input image size is $32 \times 32 \times 3$. To verify the performance of different type of input training data, some of them use similar settings as in [25]: **partially corrupted labels**, **random labels**, **random shuffled**



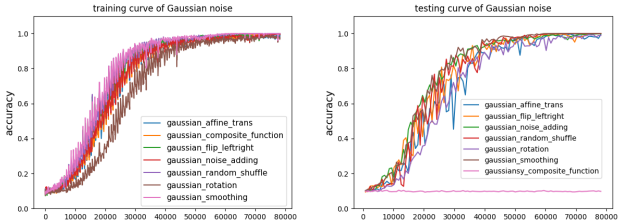
(a) (b) (c)

Figure 5. For convenience, red solid points represent training bird image. Black solid points which can be ignored mean images was enclosed by decision domain created by generating functions, this domain is represented by circle. Solid diamond points means bird was labeled by other category such as cat. Hollow points mean test bird image. (a) Standard classification on benchmark, correctly labeled, so test bird image could goes into either of decision domain; (b) Random label test images, test bird image may stay in decision domain of training bird image wrongly labeled as other category; (c) Test image created by generating function, so it can be classify correctly.



(a) training curve (b) validating curve

Figure 6. verify the generating functions of binary CIFAR10



(a) training curve (b) validating curve

Figure 7. verify the generating functions of Gaussian

pixels, **Gaussian**, but note that we only use them as training example and for partially corrupted labels, the label of each image is independently corrupted by a uniform distribution with parameter 0.3 as typical. For shuffled pixels situation, we think it is simpler than random shuffled pixels because it is the same permutation is applied to all the training images and random shuffled pixels employ random permutation for each image, so we dismiss it. Other than these, we will experiment on **binary CIFAR10** and will see miraculous DNNs could distinguish bird and bird, automobile and automobile.

Test data generated from training data above by transformations as follows:

- **Affine transformation**: we let the shape of output image of affine transformation equals to input image, scale the data by matrix $\text{diag}((1+\text{Gaussian}(0,0.3),1+\text{Gaussian}(0,0.3)))$, then rotate uniformly and apply affine_transorm function in Scipy python package on each channel and combine them together.
- **Flip left right**: mirror every training data.
- **Rotation**: although rotation is a special case of affine transformation, we experiment on it in order to increase credibility of our hypothesis.
- **Gaussian smoothing**: we use gaussian_filter function in Scipy package, with standard deviation $(0.5, 0.5, 0)$ for Gaussian kernel.
- **Random shuffle**: for each image in the input, a different random permutation of range(3072) is applied to it.
- **Adding noise**: add a random uniform(-0.02, 0.02) noise to each image.
- **Composite function**: in order to validate not all functions could perform well, we construct this function. This function is a composite function of random order of Gaussian smoothing, Random shuffle, Affine transformation, and add a uniform(-0.01, 0.01) noise.

Other settings is the same as experiments in subsection II-C2.

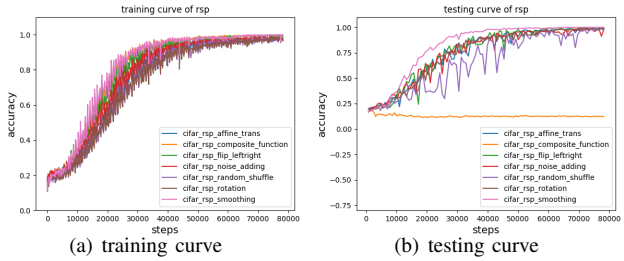
Except composite function experiments, all the Figures as 6, 7, 8, 9, 10, 11 exhibit that test examples construct from these functions could generalize well. Composite function experiments validate that not all functions could work well as a test example generating function. These results imply that:

(1) First and most important, CNNs based on stochastic optimization method could mystically learn generating function rather than data distribution. At, the same time, different CNNs trained on different data have the same generating functions. And the past, Statistician and statistics machine learning researcher concentrate on How to drive models learns data distribution.

(2) After learn the generating function, we will harvest a space generating by its generating functions, this space is that we define as decision domain previously. For a single training example x, whether a training data is random noise input or real semantic image, all the points in the space $\text{DD}(x, g)$ share the same label as x. For a category with label k, points in the space $\text{DD}(\mathcal{X}_k, g)$ share the same label k. Similarly, it can be conclude that test examples in the space $\text{DD}(\mathcal{X}, g)$ could be recognized as its marked label. In another words, after CNNs was trained on a image, it can recognize affine transformation of it, smoothing of it, noise corruption of it, etc, that is why there have "not merely memorization" in title.

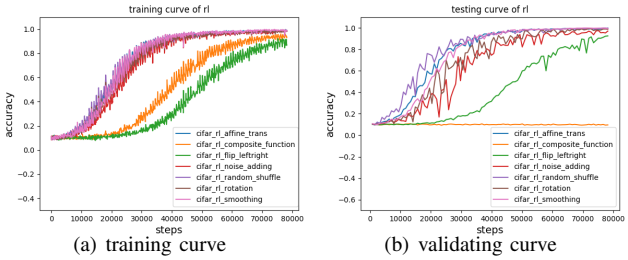
(3) It is noteworthy that the same input data but different label strategy as standard label, random label, test data created from input by generating functions inheriting input label. Our hypothesis could explain this paradox that CNNs have different performances with different label strategy, as Figure III.

(4) From the definition of $\text{DD}(\mathcal{X}, g)$, we conclude that the more training data, more likely the space $\text{DD}(\mathcal{X}, g)$ is bigger, the better generalization performance of CNNs, so we can anticipate the performance on vision tasks increases based on volume of training data as [29] observed.



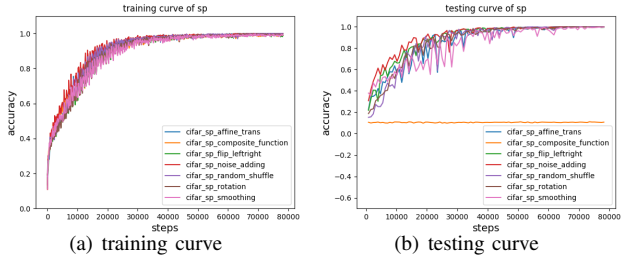
(a) training curve (b) testing curve

Figure 8. verify the generating functions of random (shuffled) pixels



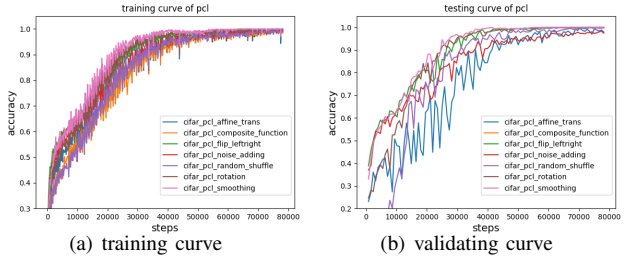
(a) training curve (b) validating curve

Figure 9. verify the generating functions of random labels



(a) training curve (b) testing curve

Figure 10. verify the generating functions of shuffle pixels



(a) training curve (b) validating curve

Figure 11. verify the generating functions of partially corrupted labels

IV. Discussion and Future Work

In the first half of this paper, we use universal classification theorem introduced by [27] to presents new unit-RTSUs, this let us retrospectively justify rationality and correctness of universal classification theorem. Also, we conducts experiments on networks based on RTSUs to verify its universal classification power on different kinds of training data sets including benchmark dataset such CIFAR10 and made-up random noise dataset. Although this provides great clarity of internal mechanism of CNNs training, there is still a gap in understanding the generalization of CNNs. So, we presents another part of works of this paper to comprehend generalization in CNNs. We demonstrate our hypothesis which CNNs learn decision domain related to generating functions other than input data distribution, through a series of experiments. All these experiments can not be explained by traditional statistics machine learning theory and are clearly explained by this hypothesis. In statistics machine learning theory, a lot of theories is based on training data and test data sampled from the same distribution, this hypothesis contradict with experiments by ours and [25]. At the same time, our work is consistent with works by [26], that memorization and generalization in CNNs depend on data.

The remaining issue is: does all the decision domain can be created by generating functions? Which kind of conditions are generating functions satisfied with? How CNNs trained with SGD-variants to produce such generating functions? Why generating functions stay fixed disregarding the training data? Further work is needed to investigate the property of generating functions of CNNs.

References

[1] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," pp. 1097–1105, 2012.

[3] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *arXiv preprint arXiv:1709.01507*, 2017.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[6] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[7] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 4845–4849.

[8] N. Perraudin, P. Balazs, and P. L. Sondergaard, "A fast griffin-lim algorithm," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*. IEEE, 2013, pp. 1–4.

[9] S. O. Arik, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, J. Raiman, S. Sengupta *et al.*, "Deep voice: Real-time neural text-to-speech," *arXiv preprint arXiv:1702.07825*, 2017.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[11] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[12] H. Zeng, M. D. Edwards, G. Liu, and D. K. Gifford, "Convolutional neural network architectures for predicting dna–protein binding," *Bioinformatics*, vol. 32, no. 12, pp. i121–i127, 2016.

[13] J. N. Wei, D. Duvenaud, and A. Aspuru-Guzik, "Neural networks for the prediction of organic chemistry reactions," *ACS central science*, vol. 2, no. 10, pp. 725–732, 2016.

[14] I. Wallach, M. Dzamba, and A. Heifets, "Atomnet: a deep convolutional neural network for bioactivity prediction in structure-based drug discovery," *arXiv preprint arXiv:1510.02855*, 2015.

[15] J. Tompson, K. Schlachter, P. Sprechmann, and K. Perlin, "Accelerating eulerian fluid simulation with convolutional networks," *arXiv preprint arXiv:1607.03597*, 2016.

[16] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, "The loss surfaces of multilayer networks," in *Artificial Intelligence and Statistics*, 2015, pp. 192–204.

[17] K. Kawaguchi, "Deep learning without poor local minima," in *Advances in Neural Information Processing Systems*, 2016, pp. 586–594.

[18] Q. Nguyen and M. Hein, "The loss surface of deep and wide neural networks," *arXiv preprint arXiv:1704.08045*, 2017.

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

[20] 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.

[21] 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.

[22] X. Pan and V. Srikumar, "Expressiveness of rectifier networks," in *International Conference on Machine Learning*, 2016, pp. 2427–2435.

[23] A. Fawzi, S.-M. Moosavi-Dezfooli, P. Frossard, and S. Soatto, "Classification regions of deep neural networks," *arXiv preprint arXiv:1705.09552*, 2017.

[24] M. Raghu, B. Poole, J. M. Kleinberg, S. Ganguli, and J. Sohldickstein, "On the expressive power of deep neural networks," *international conference on machine learning*, pp. 2847–2854, 2016.

[25] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016.

[26] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio *et al.*, "A closer look at memorization in deep networks," *arXiv preprint arXiv:1706.05394*, 2017.

[27] S. An, F. Boussaid, and M. Bennamoun, "How can deep rectifier networks achieve linear separability and preserve distances," pp. 514–523, 2015.

[28] W. Shang, K. Sohn, D. Almeida, and H. Lee, "Understanding and improving convolutional neural networks via concatenated rectified linear units," pp. 2217–2225, 2016.

[29] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," *arXiv preprint arXiv:1707.02968*, 2017.

REFERENCES

[1] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," pp. 1097–1105, 2012.

[3] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *arXiv preprint arXiv:1709.01507*, 2017.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[6] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[7] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 4845–4849.

[8] N. Perraudin, P. Balazs, and P. L. Sondergaard, "A fast griffin-lim algorithm," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*. IEEE, 2013, pp. 1–4.

[9] S. O. Arik, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, J. Raiman, S. Sengupta *et al.*, "Deep voice: Real-time neural text-to-speech," *arXiv preprint arXiv:1702.07825*, 2017.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[11] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[12] H. Zeng, M. D. Edwards, G. Liu, and D. K. Gifford, "Convolutional neural network architectures for predicting dna–protein binding," *Bioinformatics*, vol. 32, no. 12, pp. i121–i127, 2016.

[13] J. N. Wei, D. Duvenaud, and A. Aspuru-Guzik, "Neural networks for the prediction of organic chemistry reactions," *ACS central science*, vol. 2, no. 10, pp. 725–732, 2016.

[14] I. Wallach, M. Dzamba, and A. Heifets, "Atomnet: a deep convolutional neural network for bioactivity prediction in structure-based drug discovery," *arXiv preprint arXiv:1510.02855*, 2015.

[15] J. Tompson, K. Schlachter, P. Sprechmann, and K. Perlin, "Accelerating eulerian fluid simulation with convolutional networks," *arXiv preprint arXiv:1607.03597*, 2016.

[16] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun, "The loss surfaces of multilayer networks," in *Artificial Intelligence and Statistics*, 2015, pp. 192–204.

[17] K. Kawaguchi, "Deep learning without poor local minima," in *Advances in Neural Information Processing Systems*, 2016, pp. 586–594.

[18] Q. Nguyen and M. Hein, "The loss surface of deep and wide neural networks," *arXiv preprint arXiv:1704.08045*, 2017.

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

[20] 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.

[21] 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.

[22] X. Pan and V. Srikumar, "Expressiveness of rectifier networks," in *International Conference on Machine Learning*, 2016, pp. 2427–2435.

[23] A. Fawzi, S.-M. Moosavi-Dezfooli, P. Frossard, and S. Soatto, "Classification regions of deep neural networks," *arXiv preprint arXiv:1705.09552*, 2017.

[24] M. Raghu, B. Poole, J. M. Kleinberg, S. Ganguli, and J. Sohldickstein, "On the expressive power of deep neural networks," *international conference on machine learning*, pp. 2847–2854, 2016.

[25] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," *arXiv preprint arXiv:1611.03530*, 2016.

[26] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio *et al.*, "A closer look at memorization in deep networks," *arXiv preprint arXiv:1706.05394*, 2017.

[27] S. An, F. Boussaid, and M. Bennamoun, "How can deep rectifier networks achieve linear separability and preserve distances," pp. 514–523, 2015.

[28] W. Shang, K. Sohn, D. Almeida, and H. Lee, "Understanding and improving convolutional neural networks via concatenated rectified linear units," pp. 2217–2225, 2016.

[29] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," *arXiv preprint arXiv:1707.02968*, 2017.