Understanding Convolutional Neural Networks with A Mathematical Model

C.-C. Jay Kuo

Ming-Hsieh Department of Electrical Engineering
University of Southern California, Los Angeles, CA 90089-2564, USA

Abstract

This work attempts to address two fundamental questions about the structure of the convolutional neural networks (CNN): 1) why a non-linear activation function is essential at the filter output of every convolutional layer? 2) what is the advantage of the two-layer cascade system over the one-layer system? A mathematical model called the "REctified-COrrelations on a Sphere" (RECOS) is proposed to answer these two questions. After the CNN training process, the converged filter weights define a set of anchor vectors in the RECOS model. Anchor vectors represent the frequently occurring patterns (or the spectral components). The necessity of rectification is explained using the RECOS model. Then, the behavior of a two-layer RECOS system is analyzed and compared with its one-layer counterpart. The LeNet-5 and the MNIST dataset are used to illustrate discussion points. Finally, the RECOS model is generalized to a multi-layer system with the AlexNet as an example.

Keywords: Convolutional Neural Network (CNN), Nonlinear Activation, Mathematical Modeling, MNIST Dataset.

1. Introduction

There is a strong resurging interest in the neural-network-based learning because of its superior performance in many speech and image/video understanding applications nowadays. The recent success of deep neural networks (DNN) is due to the availability of a large amount labeled training data (e.g. the ImageNet) and more efficient computing hardware. It is called deep learning since the performance improves as the number of neural network

layers increases. The resulting networks and extracted features are called deep networks and deep features, respectively. There are two common neural network architectures: the convolutional neural networks (CNNs) [1] and the recurrent neural networks (RNNs). CNNs are often used to recognize visual patterns directly from pixel images with variability. RNNs are designed to recognize patterns in time series composed by symbols or audio/speech waveforms. Both CNNs and RNNs are special types of multi-layer neural networks. They are trained with the back-propagation algorithm. We will focus on CNNs in this work.

Although deep learning often outperforms classical pattern recognition methods experimentally, a mathematical theory to explain its behavior and performance is nevertheless lacking. Without a solid understanding of deep learning, we can only have a set of empirical rules and intuitions, which is not sufficient to advance the scientific knowledge profoundly. There has been a large amount of efforts devoted to the understanding of CNNs from various angles. Examples include scattering networks [2, 3, 4], tensor analysis [5], generative modeling [6], relevance propagation [7], Taylor decomposition [8], etc. Another popular topic along this line is on the visualization of filter responses at various layers [9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26].

This research attempts to address two fundamental questions about CNNs: 1) Why a non-linear activation operation is needed at the filter output of every convolutional layer? 2) What is the adavantage of the cascade of two layers in comparison with a single layer? These two questions are related to each other. The convolutional operation is a linear one. If the non-linear operation between every two convolutional layers is removed, the cascade of two linear systems is equivalent to a single linear system. Then, we can simply go with one linear system and the necessity of a multi-layer network architecture is not obvious. Although one may argue that a multi-layer network has a multi-resolution representation capability, this is a well known fact and has been extensively studied before. Examples include the Gaussian and the wavelet pyramids. There must be something deeper than the multi-resolution property in the CNN architecture due to the adoption of the non-linear activation unit.

The adoption of non-linear activation is motivated by a biological system. That is, a neuron cannot represent a negative output. However, this does not prevent us from using negative responses in an artificial neural network. The non-linear property makes the analysis of CNNs extremely challenging.

To tackle this problem, we propose a mathematical model to understand the behavior of CNNs. We view a CNN as a network formed by basic operational units that conducts "REctified COrrelations on a Sphere (RECOS)". Thus, it is called the RECOS model. A set of anchor vectors is selected for each RECOS model to capture and represent frequently occurring patterns. For an input vector, we compute its correlation with each anchor vector to measure their similarity. All negative correlations are rectified to zero in the RECOS module, and the necessity of rectification is explained. Anchor vectors are called weights in the CNN literature. In the CNN training, weights are first initialized and then adjusted by backpropagation to minimize a cost function. Here, we adopt a different name to emphasize its role in representing clustered input data in the RECOS model. After neuron modeling, we examine two-layer neural networks, where the first layer consists of either one or multiple neurons while the second layer contains only one neuron. We conduct a mathematical analysis on the behavior of the cascaded RECOS systems. This analysis sheds light on the advantage of deeper networks. The study concludes by analyzing the Alex Net which is an exemplary multi-layer CNN.

To illustrate several discussion points, this paper will use the MNIST dataset and the LeNet-5 as examples. The MNIST dataset[1] is formed by ten handwritten digits (0, 1, ..., 9). All digits are size-normalized and centered in an image of size 32 by 32. The dataset has a training set of 60,000 examples and a test set of 10,000 examples. The LeNet-5 is the latest CNN designed by LeCun et al. [27] for handwritten and machine-printed character recognition. Its architecture is shown in Fig. 1. The input image is an 8-bit image of size 32 by 32. The LeNet-5 has two pairs of convolutional/pooling layers, denoted by C1/S2 and C3/S4 in the figure, respectively. C1 has 6 filters of size 5 by 5. C3 has 16 filters of size 5 by 5. Each of them is followed by a nonlinear activation function (e.g. the sigmoid function). Furthermore, there are two fully connected layers, denoted by C5 and F6, after the two pairs of cascaded convolutional/pooling/clipping operations and before the output layer. The LeNet-5 has a strong impact on the design of deeper networks in recent years. For example, the AlexNet proposed by Krizhevsky *et al.* in [28] is a generalization of the LeNet-5 from two compound convolutional/pooling/activation layers to five.
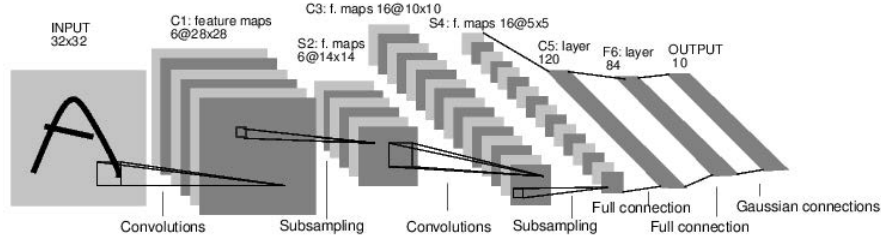
---

[1]http://yann.lecun.com/exdb/mnist/
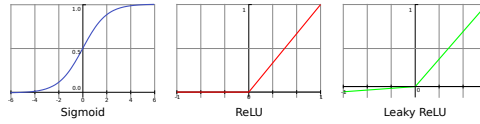
Figure 1: The LeNet-5 architecture [27].

Figure 2: Three nonlinear activation functions adopted by CNNs: the sigmoid function (left), the ReLU (middle) and the Leaky ReLU (right).

2. Why Nonlinear Activation?

Generally speaking, CNNs attempt to learn the relationship between the input and the output and store the learned experience in their filter weights. One challenge to understand CNNs is the role played by the nonlinear activation unit after the convolutional operation. We will drop the pooling operation in the discussion below since it mainly provides a spatial dimension reduction technique and its role is not as critical.

There are three activation functions adopted by CNNs: the sigmoid function, the rectified linear unit (ReLU) and the parameterized ReLU (PReLU). The PReLU is also known as the leaky ReLU. They are shown in Fig. 2. They play a clipping role. The sigmoid clips the input into an interval between 0 and 1. The ReLU clips negative values to zero while keeping positve values unchanged. The leaky ReLU has a role similar to the ReLU but it maps larger negative values to smaller ones by reducing the slope of the mapping function. It is observed experimentally that, if the nonlinear operation is removed, the system performance drops by a large margin.

Each convolutional layer is specified by its filter weights which are determined in the training stage by an iterative update process. That is, they are first initialized and then adjusted by backpropagation to minimize a cost function. All weights are then fixed in the testing stage. These weights play the role of "system memory". In this work, we adopt a different name for

4

filter weights to emphasize their role in the testing stage. We call them "anchor vectors" since they serve as reference signals (or visual patterns) for each input patch of test images. It is well known that signal convolution can also be viewed as signal correlation or projection. For an input image patch, we compute its correlation with each anchor vector to measure their similarity. Clearly, the projection onto a set of anchor vectors offers a spectral decomposition of an input.

Anchor vectors are usually not orthogonal and under-complete. Consider the LeNet-5. For the first convolutional layer (C1), the input patch is of size $5 \times 5 = 25$. It has 6 filters (or anchor vectors) of the same size. Thus, the dimension and the number of anchor vectors at C1 are 25 and 6, respectively. For the second convolutional layer (C3), its input is a hybrid spatial/spectral representation of dimension $(5 \times 5) \times 6 = 150$. Then, the dimension and the number of anchor vectors in C3 are 150 and 16, respectively.

Here, we interpret the compound operation of "convolution followed by nonliear activation" as a mechanism to conduct "REctified COrrelations on a Sphere (RECOS)". Without loss of generality, we adopt the ReLU activation function in the following discussion. All negative correlations are rectified to zero by the ReLU operation in the RECOS module. The necessity of rectification is explained below. To begin with, we consider a unit sphere centered at the origin.

Origin-Centered Unit Sphere. Let $\mathbf{x} = (x_1, \cdots, x_N)^T$ be an arbitrary vector on a unit sphere centered at the origin in the N-dimensional space, denoted by

$$S = \left\{ \mathbf{x} \middle| \|\mathbf{x}\| = (\sum_{n=1}^{N} x_n^2)^{1/2} = 1 \right\}. \tag{1}$$

We are interested in clustering $\mathbf{x}$'s with the geodesic distances over S. The geodesic distance between vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ in S is proportional to the magnitude of their angle, which can be computed by

$$\theta(\mathbf{x}_i, \mathbf{x}_j) = \cos^{-1}(\mathbf{x}_i^T \mathbf{x}_j). \tag{2}$$

Since $\cos \theta$ is a monotonically decreasing function for $0^o \leqslant |\theta| \leqslant 90^o$, we can use the correlation $0 \leqslant \mathbf{x}_i \mathbf{x}_j^T = \cos \theta \leqslant 1$ as a similarity measure between two vectors, and cluster vectors in S accordingly. However, when $90^o \leqslant |\theta| \leqslant 180^o$, the correlation, $\mathbf{x}_i^T \mathbf{x}_j = \cos \theta$, is a negative value. The correlation is no more a good measure for the geodesic distance.
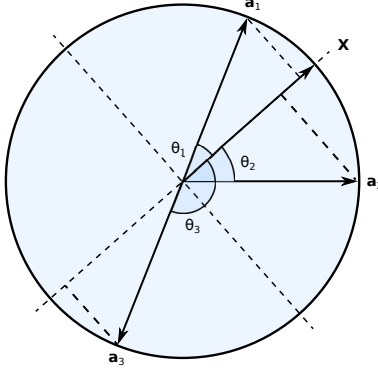
Figure 3: An example to illustrate the need of correlation rectification in the unit circle.

To show the necessity of rectification, a 2D example is illustrated in Fig. 3, where $\mathbf{x}$ and $\mathbf{a}_k$ ($k = 1, 2, 3$) denote an input and three anchor vectors on the unit circle, respectively, and θ_i is their respective angle. Since θ_1 and θ_2 are less than 90 degrees, $\mathbf{a}_1^T \mathbf{x}$ and $\mathbf{a}_2^T \mathbf{x}$ are positive. The correlation can be viewed as a projection from an anchor vector to the input (and vice versa). For positive correlations, the geodesic distance is a monotonically decreasing function of the projection value. The larger the correlation, the shorter the distance.

The angle, θ_3, is larger than 90 degrees and correlation $\mathbf{a}_3^T \mathbf{x}$ is negative. The two vectors, $\mathbf{x}$ and $\mathbf{a}_3$, are far apart in terms of the geodesic distance, yet their correlation is strong (although a negative one). Consider the extreme case. If $\mathbf{a}_3 = -\mathbf{x}$, $\mathbf{x}$ and $\mathbf{a}_3$ have the farthest geodesic distance on the unit circle, yet they are fully correlated but in the negative sense (see the example in Fig. 4). For this reason, when the correlation value is negative, it does not serve as a good indicator of the geodesic distance.

One may argue that the negative sign could be used to signal a farther geodesic distance. This is however not the case in a multi-layer RECOS system if it does not have the nonlinear clipping operation. When two RECOS units are in cascade, the filter weights of the 2nd RECOS unit can take either positive or negative values. If the response of the 1st RECOS unit is negative, the product of a negative response and a negative filter weight will produce a positive value. Yet, the product of a positive response and a positive filter weight will also produce a positive value. As a result, the system cannot differentiate these two cases. Similarly, a system without rectification cannot

Figure 4: A gray-scale cat image and its negative image. They are negatively correlated after mean removal. Their distance should be large (a white cat versus a black cat).
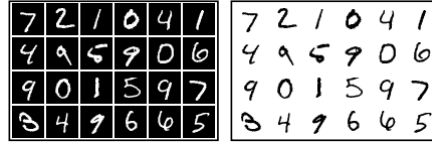

Figure 5: Samples from the MNIST dataset: the orignal one (left) and the gray-scale-reversed one (right).

differentiate the following two cases: 1) a positive response at the first layer followed by a negative filter weight at the second layer; and 2) a negative response at the first layer followed by a positive filter weight at the second layer. For this reason, it is essential to set a negative correlation value (i.e. the response) at each layer to zero (or almost zero) to avoid confusion in a multi-layer RECOS system.

We conducted an experiment to verify the importance of rectification. We trained the LeNet-5 using the MNIST training dataset, and obtained a correct recognition rate of 98.94% for the MNIST test dataset. Then, we applied the same network to gray-scale-reversed test images as shown in Fig. 5. The accuracy drops to 37.36%. Next, we changed all filter weights in C1 to their negative values while keeping the rest of the network the same. The slightly modified LeNet-5 gives a correct recognition rate of 98.94% for the gray-scale-reversed test dataset but 37.36% for the original test dataset. We can design a new network to provide a high recognition rate to both test data in Fig. 5 by doubling the number of anchor vectors in the first layer.

The above discussion can be written formally below. Consider the case where there are K anchor vectors in the N-dimensional unit sphere, denoted

7

by $\mathbf{a}_k \in R^N$, $k = 1, \cdots, K$. For given $\mathbf{x}$, its K rectified correlations with $\mathbf{a}_k$, $k = 1, \cdots, K$, defines a nonlinear transformation from $\mathbf{x}$ to an output vector

$$\mathbf{y} = (y_1, \cdots, y_k, \cdots, y_K)^T, \tag{3}$$

where

$$y_k(\mathbf{x}, \mathbf{a}_k) = \max(0, \mathbf{a}_k^T \mathbf{x}) \equiv \mathrm{Rec}(\mathbf{a}_k^T \mathbf{x}). \tag{4}$$

The form in Eq. (4) is ReLU. Other variants such as the sigmoid function and the leaky ReLU are acceptable. As long as the negative correlation values remain to be small, these vectors are weakly correlated and they will not have a major impact on the final result.

We can further generalize the RECOS model to a translated unit sphere

$$S_\mu = \left\{ \mathbf{x} \middle| \|\mathbf{x} - \mu\mathbf{1}\| = \left[\sum_{n=1}^{N} (x_n - \mu)^2 \right]^{1/2} = 1 \right\}. \tag{5}$$

where $\mu = \frac{1}{N} \sum_{n=1}^{N} x_n$ is the mean of all x_n's and $\mathbf{1} = (1, \cdots, 1, \cdots 1)^T \in R^N$ is a constant vector with all elements equal to one. Sphere S_μ is a translation of S with a new center at $\mu\mathbf{1}^T$. This generalization is desired for the following reason.

For vision problems, elements x_n, $n = 1, \cdots, N$, of $\mathbf{x}$ denote N pixel values of an input image, and μ is the mean of all pixels. If the input is a full image, its mean is the global mean that has no impact on image understanding. It can be removed before the processing. Thus, we set $\mu = 0$. However, if an input image is large, we can partition it into smaller patches, and process all patches in parallel. In this case, the mean of each patch is a local mean. It should not be removed since an integration of local means provides a coarse view of the full image. This corresponds to the general case in Eq. (5).

Based on Eq. (4), the output with respect to S_μ can be written as $\mathbf{y} = (y_1, \cdots, y_K)$, where

$$y_k(\mathbf{x} - \mu\mathbf{1}, \mathbf{a}_k) = \mathrm{Rec}(\mathbf{a}_k^T \mathbf{x} + \mu a_{k,0}), \tag{6}$$

and where $a_{k,0} = -\sum_{n=1}^{N} a_{k,n}$. By augmenting $\mathbf{x}$ and $\mathbf{a}_k$, with one additional element

$$\mathbf{x}' = (\mu, x_1, \cdots, x_N)^T, \quad \mathbf{a}'_k = (a_{k,0}, a_{k,1}, \cdots, a_{k,N})^T, \tag{7}$$

8

we can re-write Eq. (6) as

$$y_k(\mathbf{x}', \mathbf{a}'_k) = \mathrm{Rec}(\mathbf{a}'^T_k \mathbf{x}'), \quad k = 1, \cdots, K. \tag{8}$$

Although $\mathbf{x}', \mathbf{a}'_k \in R^{(N+1)}$, they only have N independent elements since their first elements are computed from the remaining N elements.

Furthermore, the length of the input and anchor vectors may not be one. We use $\mathbf{x}''$ and $\mathbf{a}''_k$ to denote the general case. Then, we can set

$$\mathbf{x}' \equiv \frac{\mathbf{x}''}{||\mathbf{x}''||}, \quad \mathbf{a}'_k \equiv \frac{\mathbf{a}''_k}{||\mathbf{a}''_k||}. \tag{9}$$

Then, Eq. (8) can be re-written as

$$y_k(\mathbf{x}'', \mathbf{a}''_k) = ||\mathbf{x}''||\,||\mathbf{a}''_k||\mathrm{Rec}(\mathbf{a}'^T_k \mathbf{x}'). \tag{10}$$

If there are K frequently occuring patterns in input data to form K clusters, we can assign anchor vector, $\mathbf{a}_k$, to the centroid of the kth cluster, $k = 1, \cdots, K$. Then, data points in this cluster will generate a strong correlation with $\mathbf{a}_k$. They will generate weaker correlations with other anchor vectors. Along this line, it is worthwhile to mention that it was observed in [29] that the K-means clustering is effective in a single-layer network. A CNN consists of multiple RECOS units working cooperatively, and they can be organized into multiple layers. The advantage of multi-layer CNNs will be explained below.

3. Advantages of Cascaded Layers?

The LeNet-5 is essentially a neural network with two convolutional layers since the compound operations of convolution/sampling/nonlinear clipping are viewed as one complete layer in the modern neural network literatures. The input to the first layer of the LeNet-5 is a purely spatial signal while the input to the second layer of the LeNet-5 is a hybrid spectral-spatial signal consisting of spatial signals from 6 spectral bands. The spatial coverage of a filter with respect to the source image is called its receptive field. The receptive fields of the first and the second layers of the LeNet-5 are 5×5 and 13×13, respectively. For each spatial location in the 13×13 receptive field, it may be covered by one, two or four layer-one filters as shown in Fig. 6. In the following, we conduct a mathematical analysis on the behavior of
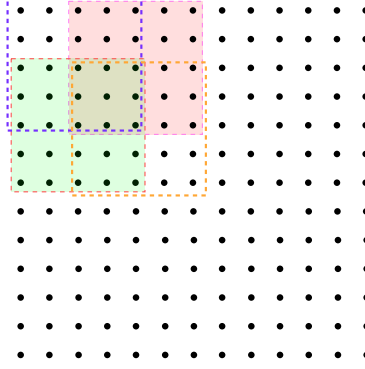
Figure 6: The receptive fields of the first- and the second-layer filters of the LeNet-5, where each dot denotes a pixel in the input image, the 5×5 window indicates the receptive field of the first-layer filter and the whole 13×13 window indicates the receptive field of the second-layer filter. The second-layer filter accepts the outputs from $5 \times 5 = 25$ first-layer filters. Only four of them are shown in the figure for simplicity.

the cascaded systems. This analysis sheds light on the advantage of deeper networks. In the following discussion, we begin with the cascade of one layer-1 RECOS unit and one layer-2 RECOS unit, and then generalize it to the cascade of multiple layer-1 RECOS units to one layer-2 RECOS unit. For simplicity, the means of all inputs are assumed to be zero.

One-to-One Cascade. We define two anchor matrices:

$$\mathbf{A} = [\mathbf{a}_1, \cdots, \mathbf{a}_k \cdots, \mathbf{a}_K], \ \mathbf{B} = [\mathbf{b}_1, \cdots, \mathbf{b}_l \cdots, \mathbf{b}_L], \tag{11}$$

whose column are anchor vectors $\mathbf{a}_k$ and $\mathbf{b}_l$ of the two individual RECOS units. Clearly, $\mathbf{A} \in R^{N \times K}$ and $\mathbf{B} \in R^{K \times L}$. To make the analysis tractable, we begin with the correlation analysis and will take the nonlinear rectification effect into account at the end. For the correlation part, let $\mathbf{y} = \mathbf{A}^T\mathbf{x}$ and $\mathbf{z} = \mathbf{B}^T\mathbf{y}$. Then, we have

$$\mathbf{z} = \mathbf{B}^T\mathbf{A}^T\mathbf{x} = \mathbf{C}^T\mathbf{x}, \ \mathbf{C} \equiv \mathbf{A}\mathbf{B}. \tag{12}$$

Clearly, $\mathbf{C} \in R^{N \times L}$ with its (n, l)th element equal to
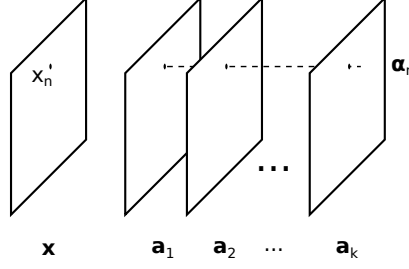
$$c_{n,l} = \boldsymbol{\alpha}_n^T\mathbf{b}_l, \tag{13}$$

Figure 7: Visualization of anchor-position vector $\boldsymbol{\alpha}_n$.

where $\boldsymbol{\alpha}_n^T \in R^K$ is the nth row vector of $\mathbf{A}$. The meaning of $\boldsymbol{\alpha}_n$ can be visualized in Fig. 7. Mathematically, we decompose

$$\mathbf{x} = \sum_{n=1}^{N} x_n \mathbf{e}_n, \tag{14}$$

where $\mathbf{e}_n \in R^N$ is the nth coordinate-unit-vector. Then,

$$\boldsymbol{\alpha}_n^T = \mathbf{A}\mathbf{e}_n. \tag{15}$$

Since $\boldsymbol{\alpha}_n$ captures the position information of anchor vectors in $\mathbf{A}$, it is called the anchor-position vector. Finally, we apply the rectification to all negative elements in $\mathbf{C}$ to obtain an anchor matrix $\mathbf{C}'$ from $\mathbf{x}$ to $\mathbf{z}'$:

$$\mathbf{z}' = \mathbf{C}'^T \mathbf{x}, \quad \mathbf{C}' = [c'_{n,l}]_{N \times L}, \tag{16}$$

where

$$c'_{n,l} = \text{Rec}(c_{n,l}) = \text{Rec}(\boldsymbol{\alpha}_n \mathbf{b}_l). \tag{17}$$

Rigorously speaking, $\mathbf{z}$ and $\mathbf{z}'$ are not the same. The former has no rectification operation while the latter applies the rectification operation to the matrix product. Since the actual system applies the rectification operations to the output of both layers and its final result, denoted by $\mathbf{z}''$, can be different from $\mathbf{z}$ and $\mathbf{z}'$. However, we are most interested in areas where strong positive correlations exist and no rectification is needed in both layers. In other words, we have $\mathbf{z} \approx \mathbf{z}' \approx \mathbf{z}''$ in these areas. Our following analysis applies to this case only.

Many-to-One Cascade. It is straightforward to extend generalize one-to-one cascaded case to the many-to-one cascaded case. The correlation of the first-layer RECOS units can be written as

$$\mathbf{Y} = \mathbf{A}^T\mathbf{X}, \tag{18}$$

where

$$\mathbf{Y} = [\mathbf{y}_1, \cdots \mathbf{y}_P], \ \mathbf{X} = [\mathbf{x}_1, \cdots \mathbf{x}_P], \tag{19}$$

There are P RECOS units working in parallel in the first layer. They cover spatially adjacent regions yet share one common anchor matrix. They are used to extract common representative patterns in different regions. The correlation of the second-layer RECOS can be expressed as

$$\mathbf{z} = \mathbf{B}^T\tilde{\mathbf{y}}, \tag{20}$$

where $\mathbf{z} \in R^L$, $\mathbf{B} \in R^{PK \times L}$ and $\tilde{\mathbf{y}} = (\mathbf{y}_1^T, \cdots, \mathbf{y}_P^T)^T \in R^{PK}$ is formed by the cascade of P output vectors of the first-layer RECOS units. Finally, the correlation of the compound effect of Eqs. (18) and (20) can be written as

$$\mathbf{z} = \mathbf{C}^T\mathbf{x}, \ \mathbf{C} \equiv \mathbf{AB}, \tag{21}$$

where $\mathbf{C} \in R^{PN \times L}$ and input vectors, $\mathbf{x}_1, \cdots \mathbf{x}_P$, have both overlapping and non-overlapping regions as shown in Fig. 6. Anchor matrix $\mathbf{A}$ extracts representative patterns in different regions while anchor matrix $\mathbf{B}$ is used to stitch these spatially-dependent representative patterns to form larger representative patterns. For example, consider a large lawn composed by multiple grass patches. Suppose that the grass patterns can be captured by an anchor vector in $\mathbf{A}$. Then, an anchor vector in $\mathbf{B}$ will provide a superposition rule to stitch these spatially distributed anchor vectors of grass to form a larger lawn.

Comparison of One-Layer and Two-Layer Systems. To explain the advantage of deeper networks, we compare the two-layer system in Eq. (16) with the following one-layer system

$$\mathbf{z} = \mathbf{D}^T\mathbf{x}, \tag{22}$$

where $\mathbf{D} = [\mathbf{d}_1, \cdots, \mathbf{d}_L] \in R^{N \times L}$ is an anchor matrix with $\mathbf{d}_l$ as its anchor vector. Anchor matrices $\mathbf{A}$ and $\mathbf{D}$ essentially play the same role in capturing global frequently occurring patterns in $\mathbf{x}$. However, the two-stage system has

additional anchor matrix, $\mathbf{B}$, in capturing representative patterns of $\mathbf{y}$. It is best to examine anchor vectors of C' to understand the compound effect of $\mathbf{A}$ and $\mathbf{B}$ fully. Based on Eq.(16), anchor vectors of $\mathbf{C}'$ are

$$\mathbf{c}'_l = (c'_{1,l}, \cdots, c'_{n,l})^T, \; l = 1, \cdots, L, \tag{23}$$

where $c'_{n,l}$ is the rectified inner product of $\boldsymbol{\alpha}_n$ and $\mathbf{b}_l$ as given in Eq. (17). Anchor vectors $\mathbf{a}_k$ capture representative global patterns, but they are weak in capturing position sensitive information. This shortcoming can be compensated by modulating $\mathbf{b}_l$ with elements of anchor-position vector $\boldsymbol{\alpha}_n$.

We use an example to illustrate this point. First, we enlarge the MNIST training and testing dataset by adding ten different background scenes to the original handwritten digits in the MNIST dataset [27]. They are shown in the top two rows in Fig. 8. For the bottom three rows, we show three input digital images in the leftmost column, the six spectral output images from the convolutional layer and the ReLU layer in the middle column and the 16 spectral output images in the right two columns. It is difficult to find a good anchor matrix of the first layer due to background diversity. However, background scenes in these images are not consistent in the spatial domain while foreground digits are. As a result, they can be filtered out more easily by modulating anchor vectors $\mathbf{b}_l$ in the second layer using the anchor-position vector, $\boldsymbol{\alpha}_n$, in the first layer.

Role of Fully Connected Layers. A CNN can be decomposed into two subnetworks (or subnets): the feature extraction subnet and the decision subnet. For the LeNet-5, the feature extraction subnet consists of C1, S2, C3 and S4 while the decision subnet consists of C5, F6 and Output as shown in Fig. 9. The decision subnet has the following three roles: 1) converting the spectral-spatial feature map from the output of S4 into one feature vector of dimension 120 in C5; 2) adjusting anchor vectors so that they are aligned with the coordinate-unit-vector with correct feature/digit pairing in F6; and 3) making the final digit classification decision in Output.

The functions of C5 and F6 are illustrated in Fig. 9(a) and (b), respectively. F5 assigns anchor vectors to feature clusters as shown in Fig. 9(a). There are 120 anchor vectors in the 400-D space to be assigned (or trained) in the LeNet-5. Then, an anchor matrix is used in F6, to rotate and rescale anchor vectors in C5 to their new positions. The objective is to ensure the feature cluster of an object class to be aligned with the coordinate-unit-vector of the same object class for decision making in the Output. This is illustrated
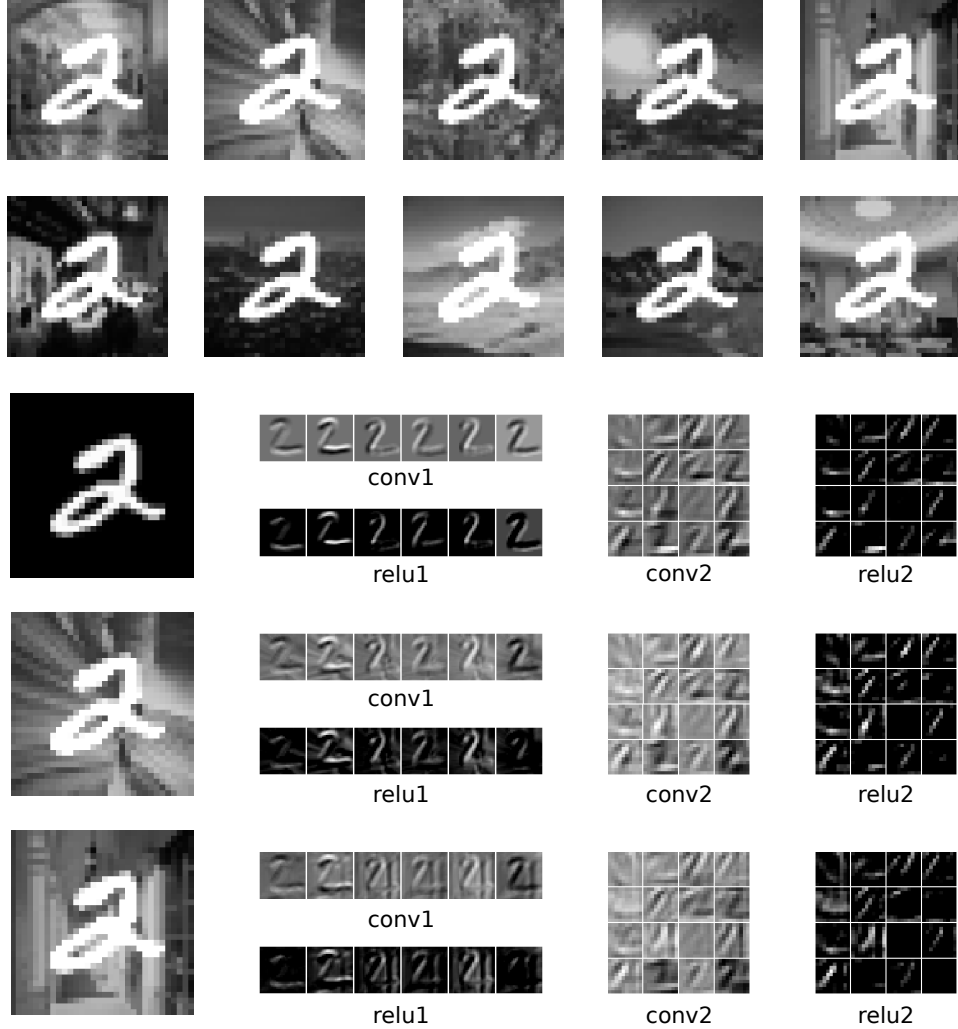
13

Figure 8: The MINST dataset with 10 different background scenes are shown in the top two rows while the output images in 6 spectral channels and 16 spectral channels of the first-layer and the second-layers with respect to the input images given in the leftmost column are shown in the bottom three rows. The structured background has an impact on the 6 channel responses at the first layer yet their impact on the 16 channel responses at the second layer diminishes. This phenomenon can be explained by the analysis given in Section 3.
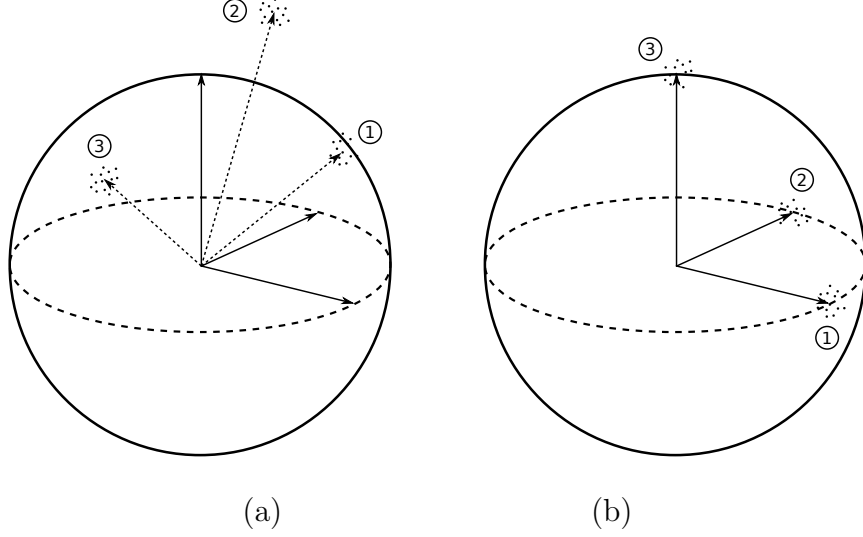
Figure 9: Illustration of functions of (a) C5 and (b) F6.

in Fig. 9 (b). Every coordinate-unit-vector in the Output is an anchor vector, and each of them represents a digit class. The softmax rule is widely used in the Output for final decision making.

Multi-Layer CNN. We list the traditional layer names, RECOS notations, their input and the output vector dimensions of the LeNet-5 in Table 1. The output vector dimension is the same as the number of anchor vectors in the same layer. Vector augmentation is needed in S^1 since their local mean could be non-zero. However, it is not needed in S^2, S^3 S^4 and S^5 since the global mean is removed.
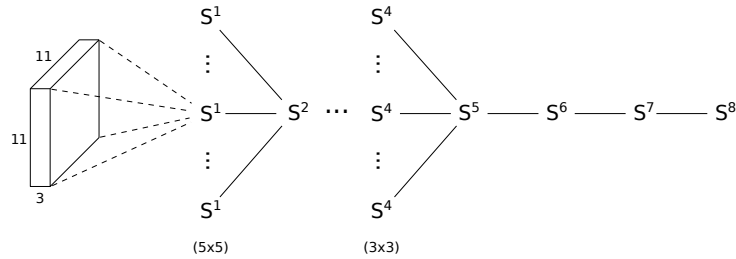

Figure 10: The organization of the AlexNet using tree-structured RECOS units.

Table 1: The specification of RCS units used in the LeNet-5, where the third column (N) shows the dimension of the input and the fourth column (K) shows the dimension of the output of the corresponding layer. Note that K is also the number of anchor vectors.

LeNet-5	RECOS	N	K
C1/S2	S^1	$(5 \times 5) + 1$	6
C3/S4	S^2	$(6 \times 5 \times 5)$	16
C5	S^3	$16 \times 5 \times 5$	120
F6	S^4	$120 \times 1 \times 1$	84
Output	S^5	$84 \times 1 \times 1$	10

Table 2: The specification of RCS units used in the AlexNet, where the third column (N) shows the dimension of the input and the fourth column (K) shows the dimension of the output of S^l, where $l = 1, \cdots, 8$. K is also the number of anchor vectors.

AlexNet	RECOS	N	K
$Conv\_1$	S^1	$(3 \times 11 \times 11) + 1$	96
$Conv\_2$	S^2	$(96 \times 5 \times 5) + 1$	256
$Conv\_3$	S^3	$(256 \times 3 \times 3) + 1$	384
$Conv\_4$	S^4	$(384 \times 3 \times 3) + 1$	384
$Conv\_5$	S^5	$(384 \times 3 \times 3) + 1$	256
FC_6	S^6	$256 \times 1 \times 1$	4096
FC_7	S^7	$4096 \times 1 \times 1$	4096
FC_8	S^8	$4096 \times 1 \times 1$	1000

Next, we use the AlexNet proposed by Krizhevsky *et al.* [28] as another example to illustrate a multi-layer CNN. Its RECOS representation is shown in Fig. 10 and Table 2. We denote the RECOS units at the lth level by S^l ($l = 1, \cdots 8$). The input to S^1 is a color image patch of size 11×11 with R, G and B three channels. The covered regions become larger as we proceed from S^1 to S^5. They are used to capture representative visual patterns of various sizes and at different spatial locations.

4. Conclusion and Open Problems

In this work, a RECOS model was adopted to explain the role of the nonlinear clipping function in CNNs, and a simple matrix analysis was used to explain the advantage of the two-layer RECOS model over the single-layer RECOS model. The proposed RECOS mathematical model is centered on the selection of anchor vectors. CNNs do offer a very powerful tool for image processing and understanding. There are however a few open problems remaining in CNN interpretability and wider applications. Some of them are listed below.

1. **Application-Specific CNN Architecture**
   In CNN training, the CNN architecture (including the layer number and the filter number at each layer, etc.) has to be specified in advance. Given a fixed architecture, the filter weights are optimized by an end-to-end optimization framework. Generally speaking, simple tasks can be well handled by smaller CNNs. However, there is still no clear guideline in the CNN architecture design for a class of applications. The anchor-vector viewpoint encourages us to examine the properties of source data carefully. A good understanding of source data distribution contributes to the design of more efficient CNN architectures and more effective training.

2. **Robustness to Input Variations**
   The LeNet-5 was shown to be robust with respect to a wide range of variations in [27]. Yet, the robustness of CNNs is challenged by recent studies, e.g. [30]. It is an interesting topic to understand the causes of these problems so as to design more error-resilient CNNs.

3. **Weakly Supervised Learning**
   The training of CNNs demand a large amount of labeled data. It is expensive to collect labeled data. Furthermore, the labeling rules could be different from one dataset from another even for the same applications. It is important to reduce the labeling burden and allow CNN training using partially and flexibly labeled data. In other words, we need to move from the heavily supervised learning to weakly supervised learning to make CNNs widely applicable.

References

[1] Y. LeCun, Y. Bengio, G. E. Hinton, Deep learning, Nature 521 (2015) 436–444.

[2] S. Mallat, Group invariant scattering, Communications on Pure and Applied Mathematics 65 (10) (2012) 1331–1398.

[3] J. Bruna, S. Mallat, Invariant scattering convolution networks, IEEE transactions on pattern analysis and machine intelligence 35 (8) (2013) 1872–1886.

[4] T. Wiatowski, H. Bölcskei, A mathematical theory of deep convolutional neural networks for feature extraction, arXiv preprint arXiv:1512.06293.

[5] N. Cohen, O. Sharir, A. Shashua, On the expressive power of deep learning: a tensor analysis, arXiv preprint arXiv:1509.05009 556.

[6] J. Dai, Y. Lu, Y.-N. Wu, Generative modeling of convolutional neural networks, arXiv preprint arXiv:1412.6296.

[7] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, W. Samek, On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation, PloS one 10 (7) (2015) e0130140.

[8] G. Montavon, S. Bach, A. Binder, W. Samek, K.-R. Müller, Explaining nonlinear classification decisions with deep taylor decomposition, arXiv preprint arXiv:1512.02479.

[9] L. Arras, F. Horn, G. Montavon, K.-R. Müller, W. Samek, Explaining predictions of non-linear classifiers in nlp, arXiv preprint arXiv:1606.07298.

[10] D. Bruckner, Ml-o-scope: a diagnostic visualization system for deep machine learning pipelines, Tech. rep., DTIC Document (2014).

[11] M. Denil, A. Demiraj, N. Kalchbrenner, P. Blunsom, N. de Freitas, Modelling, visualising and summarising documents with a single convolutional neural network, arXiv preprint arXiv:1406.3830.

[12] A. Dosovitskiy, T. Brox, Inverting visual representations with convolutional networks, arXiv preprint arXiv:1506.02753.

[13] D. Erhan, Y. Bengio, A. Courville, P. Vincent, Visualizing higher-layer features of a deep network, University of Montreal 1341.

[14] A. W. Harley, An interactive node-link visualization of convolutional neural networks, in: International Symposium on Visual Computing, Springer, 2015, pp. 867–877.

[15] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, S. Liu, Towards better analysis of deep convolutional neural networks, arXiv preprint arXiv:1604.07043.

[16] A. Mahendran, A. Vedaldi, Understanding deep image representations by inverting them, in: 2015 IEEE conference on computer vision and pattern recognition (CVPR), IEEE, 2015, pp. 5188–5196.

[17] A. Mordvintsev, C. Olah, M. Tyka, Inceptionism: Going deeper into neural networks, Google Research Blog. Retrieved June 20.

[18] A. Nguyen, J. Yosinski, J. Clune, Multifaceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks, arXiv preprint arXiv:1602.03616.

[19] W. Samek, A. Binder, G. Montavon, S. Bach, K.-R. Müller, Evaluating the visualization of what a deep neural network has learned, arXiv preprint arXiv:1509.06321.

[20] K. Simonyan, A. Vedaldi, A. Zisserman, Deep inside convolutional networks: Visualising image classification models and saliency maps, arXiv preprint arXiv:1312.6034.

[21] C. Vondrick, A. Khosla, T. Malisiewicz, A. Torralba, Hoggles: Visualizing object detection features, in: Proceedings of the IEEE International Conference on Computer Vision, 2013, pp. 1–8.

[22] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, H. Lipson, Understanding neural networks through deep visualization, arXiv preprint arXiv:1506.06579.

[23] W. Yu, K. Yang, Y. Bai, H. Yao, Y. Rui, Visualizing and comparing convolutional neural networks, arXiv preprint arXiv:1412.6631.

[24] M. D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European Conference on Computer Vision, Springer, 2014, pp. 818–833.

[25] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, A. Torralba, Object detectors emerge in deep scene cnns, arXiv preprint arXiv:1412.6856.

[26] L. M. Zintgraf, T. S. Cohen, M. Welling, A new method to visualize deep neural networks, arXiv preprint arXiv:1603.02518.

[27] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.

[28] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems 25, Curran Associates, Inc., 2012, pp. 1097–1105.

[29] A. Coates, H. Lee, A. Y. Ng, An analysis of single-layer networks in unsupervised feature learning, in: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS), Vol. 15, JMLR:W&CP 15, 2011, pp. 215–223.

[30] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, R. Fergus, Intriguing properties of neural networks.
URL http://arxiv.org/abs/1312.6199