THE ESSENCE OF DEEP LEARNING

This paper was downloaded from TechRxiv (<https://www.techrxiv.org>).

LICENSE

CC BY 4.0

SUBMISSION DATE / POSTED DATE

25-11-2021 / 17-12-2021

CITATION

wang, wei (2021): THE ESSENCE OF DEEP LEARNING. TechRxiv. Preprint.
<https://doi.org/10.36227/techrxiv.17078699.v2>

DOI

[10.36227/techrxiv.17078699.v2](https://doi.org/10.36227/techrxiv.17078699.v2)

THE ESSENCE OF DEEP LEARNING

Wei Wang
xiaokeaiww888@yeah.net

ABSTRACT

Currently, we use a huge amount of parameters and extremely complicated Deep Neural Networks to assure strong performances on datasets. Indeed, most of them can achieve great results on some tasks, but they are still black boxes and we do not know the details of their processes. I start from a single sample to introduce a novel theory for deep learning networks. I prove that the essence of deep learning networks is a large amount of matrixes' multiplication. In particular, each sample has its matrix which could enable sample matching the result of it, and deep learning networks memory some of their corresponding matrixes. The process of training deep learning is seeking a set of parameters that could generate as many as possible matrixes which match the samples. If a model could memory many samples' matrix, this model could get a good result on a test set called generalization. Otherwise, the model is called overfitting or not fitting. I present empirical results showing that my theory is right on Cifar-10 Krizhevsky [2012].

1 Introduction

Deep learning networks on many tasks can get excellent performances, and more and more people are getting into the research and improvement work of deep learning networks. In which appeared the following questions.

1. How deep learning works: we all know deep learning is a combination of many simple nonlinear functions, because of its nonlinear characteristic, cause it is hard to study what happened in it. Another reason is that the dimensions of the input data tend to be multidimensional beyond people's understanding, although people adopted a variety of ways, such as showing the middle feature maps, due to the large dimensions and feature maps were too abstract, so people can only give some subjective interpretations, it is difficult to persuade the readers Zeiler and Fergus [2013], Li et al. [2015], Karpathy et al. [2015], Bau et al. [2017], Alvarez-Melis and Jaakkola [2018].
2. The generalization: This is the ultimate goal of all the people who work in deep learning. We want to get a model that can perform well in a variety of data sets, eventually applying the model in real life with the same effects. Sometimes there will be a phenomenon called overfitting in many experiments, so what is the nature of overfitting? Why some models can get good performance on the test set, some bad? Many people do their best to find the model that could have a better generalization Zhang et al. [2016], Wang et al. [2019].
3. What is the theory of deep learning: This is the problem for all people engaged in research on deep learning. Do not really have a way to explain deep learning? Although people have made efforts in this area for a long time and made all kinds of deductions. But, they are far from our reality Sun et al. [2021], Mittelstadt et al. [2018], Lipton [2016], Du et al. [2018].

This paper points out that deep learning is by no means an illusory and groundless thing. Next, this paper will give an explanation of all the above problems and a mathematical representation of the nature of deep learning.

2 Theory

2.1 What is the essence of Deep learning

First of all, I will give an intuitive explanation. We all know that deep learning has huge parameters and many nonlinear functions, so it is difficult to study. This paper argues that deep learning is essentially a memory of some

matching matrixes of corresponding samples in data set, which stores countless matrices that can fit input samples to corresponding results or feature maps.

Firstly, considering a model has converged. When the model has converged, inputting a sample into a model, then its corresponding deep learning network can be thought of as a large matrix, which are some linear combinations of the input sample, because all the states of Relu have fixed. The proof is given below.

I donate scalars as a , vectors as $\mathbf{a}$, matrices as $\mathbf{A}$, set as $\mathcal{A}$ and equality by definition as $\equiv$. Given a training dataset $\mathcal{S} \equiv \bigcup_i^n \{(x_i, y_i)\}$.

2.2 The theory of full connected neural networks

When the model is fixed, the whole process of deep learning can be written as:

$$\text{Theorem1} : \hat{y}_i = x_i^T \mathbb{W} \equiv x_i^T \mathbf{W}_1 \mathbf{\Lambda}_1 \mathbf{W}_2 \mathbf{\Lambda}_2 \mathbf{W}_3 \mathbf{\Lambda}_3 \dots \quad (1)$$

$x_i \subseteq \mathbb{R}^{n \times 1}$, $\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3 \dots$ are the matrixes of the model. $\mathbf{\Lambda}_1, \mathbf{\Lambda}_2, \mathbf{\Lambda}_3$ are diagonal matrix, if the corresponding values are less than 0, the values in the diagonal line are 0, otherwise they are 1. $\mathbb{W} = \mathbf{W}_1 \mathbf{\Lambda}_1 \mathbf{W}_2 \mathbf{\Lambda}_2 \mathbf{W}_3 \mathbf{\Lambda}_3 \dots$. So we find that the product of $\mathbf{W}_1 \mathbf{\Lambda}_1 \mathbf{W}_2 \mathbf{\Lambda}_2 \mathbf{W}_3 \mathbf{\Lambda}_3 \dots$ is the matching $\mathbb{W}$ of x_i and it can be calculated, when we give a sample.

2.3 The theory of convolutional neural networks

$x_i \subseteq \mathbb{R}^{m \times n}$. Firstly, we reshape x_i in to $\hat{x}_i(m * n, 1)$ and the convolutional kernels are $3 * 3$, and strides are 1.

The whole process of convolution can be written as:

$$\hat{y}_i = x_i \otimes \mathbf{W} \equiv \mathbf{Z}_1 \mathbf{E} \hat{x}_i^T \mathbf{T} \hat{\mathbf{K}} \mathbf{Z}_2 \mathbf{\Lambda} \quad (2)$$

$\hat{x}_i^T \subseteq \mathbb{R}^{1 \times (m * n)}$: The transpose of $\hat{x}_i$.

$\mathbf{E} \subseteq \mathbb{R}^{(m-2)(n-2) \times 1}$: Copying the $\hat{x}_i$ in the direction of row. Its size depends on how many times the convolution.

$\mathbf{T} \subseteq \mathbb{R}^{(m * n) \times 9(m-2)(n-2)}$: Putting $\hat{x}_i$ in a form that we can deal with matrixes.

$\hat{\mathbf{K}} \subseteq \mathbb{R}^{9(m-2)(n-2) \times 1}$: $\mathbf{K}$ is the original kernel. $\hat{\mathbf{K}}$ is a column vector copied in the row direction.

$\mathbf{Z}_1, \mathbf{Z}_2 \subseteq \mathbb{R}^{1 \times (m-2)(n-2)}$: They are responsible for the transpose of feature map.

$\mathbf{\Lambda}$: diagonal matrix, if the corresponding values are less than 0, the values in the diagonal line are 0, otherwise they are 1

The details of all matrixes:

$$\begin{aligned} \hat{x}_i^T &= (x_{1,1}, x_{1,2}, x_{1,3} \dots x_{m,n}) \\ \mathbf{E}^T &= (1, 1, 1 \dots 1) \end{aligned} \quad (3)$$

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \dots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 1 \end{pmatrix} \quad (4)$$

$$E\hat{x}_i^T T = \begin{pmatrix} x_{11} & x_{12} & x_{13} & x_{21} & x_{22} & x_{23} & x_{31} & x_{32} & x_{33} & 0 & 0 & 0 & \dots 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & x_{12} & x_{13} & x_{14} & \dots 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \dots 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \dots 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots 0 \\ \cdot & \cdot & \cdot & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots 0 \\ \cdot & \cdot & \cdot & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dots x_{m,n} \end{pmatrix} \quad (5)$$

$$\hat{K}^T = (k_{11}, k_{12}, k_{13}, k_{21}, k_{22}, k_{23}, k_{31}, k_{32}, k_{33} \dots k_{31}, k_{32}, k_{33}) \quad (6)$$

$$E\hat{x}_i^T T \hat{K} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ \cdot \\ y_n \\ \cdot \\ y(m-2)(n-2) \end{pmatrix} \quad (7)$$

$$Z_1 = (1, 1, 1 \dots 1) \quad (8)$$

$$\begin{aligned} z &= E\hat{x}_i^T T \hat{K} \\ z^T &= \{z_1 \dots z_{(m-2)(n-2)}\} \end{aligned} \quad (9)$$

$$s = Z_1 E\hat{x}_i^T T \hat{K} = z_1 + z_2 + z_3 \dots + z_{(m-2)(n-2)} \quad (10)$$

$$Z_2 = \left(\frac{z_1}{s}, \frac{z_2}{s}, \frac{z_3}{s} \dots \frac{z_{(m-2)(n-2)}}{s} \right) \quad (11)$$

According to the above proof, the whole convolution process in a network can be written as:

$$\textit{Theorem2} : \hat{y}_i = \hat{x}_i^T \mathbb{W}$$

. $\mathbb{W}$ is a representation of all the corresponding convolution and Relu. It also can be calculated.

2.4 The essence of fully connected and fully convolutional neural network

Based on the above Theorem 1 and Theorem 2, we could conclude that they just generate their corresponding matrixes according to different samples. There are similarities between their matrixes, otherwise it is hard for them to fulfill the tasks with some combination of their parameters. The similarities of matrixes are shown in the next section.

3 The reasons behind the feasibility of neural networks

3.1 Test the theorem 2

We know the best solution for the whole dataset is a set of matrixes that could match as many as possible samples. It is hard for us to find the corresponding matrixes, but we could let deep learning networks generate the matrixes.

The first experiment was conducted on Cifar-10. Firstly, It is hard for me to find the corresponding matrixes, so I trained a Resnet-20-W (Altering the structure of ResNet He et al. [2015]) on the dataset, and the batch size was 128. The process is as follows:

Figure 1: Setting of Resnet-20-W .

The results of model 1.

Table 1: Cifar-10 Result on Resnet-20-W

dataset	acc	loss
train set	1.0	0.0261
test set	0.7693	0.7738

The loss value in the train set is very small because the Resnet-20-W could generate all the matching matrixes of all samples in the train set but only could generate about 0.76% corresponding matrixes of the test set.

In figures 2 3, I show the images and their corresponding matrixes which are reshaped into (240,128).

We could find that there is some similarity between the $\mathbb{W}$, but it seems that the differences do not depend on classes. So I contend that Resnet-20-W will generate a similar weight according to the dataset.

3.2 Rewrite the theorem 2

According to function 13, we know that if the inputs' and outputs' shapes are large, we will need to do a huge amount of matrixes' calculation. So we could according to the process in section 2.3 unfold the input images' channels in the directions of the row. The whole process could be written as $x \otimes W = Z_1 S E \hat{x}^T T \hat{k} Z_2 \Lambda$. S is responsible for summing the values of different channels after convolution. The whole process could be written as:

$$\text{Theorem 3 : } \hat{y} = \mathbb{W}' \hat{x}^T \mathbb{W}'' \quad (12)$$

Theorem 3 could be used in studying the middle process if we have a pre-trained model. Besides, it also could be used in tasks with high dimensions.

Firstly, I trained ResNet-20 on Cifar-10. I used the output of layer block 3 as the label. Then I trained the model of ResNet-20-WW. The setting of ResNet-20-WW is shown in 4:

Figure 2: Resnet-20-W: Classes of airplanes. It shows the images and their corresponding matrixes.

Figure 3: Resnet-20-W: Classes of cars. It shows the images and their corresponding matrixes.

Figure 4: Setting of ResNet-20-WW.

The results are as follows:

Table 2: Cifar-10 Result on ResNet-20-WW

dataset		model teacher	model student
train set	acc	0.9993	0.9972
	loss		0.0046
test set	acc	0.9199	0.8440
	loss		0.0269

In figures 5, 6, I show the images and their corresponding matrixes.

We could find that theorem 3 could be used in studying the middle layers or complex tasks. Besides, we can find that it is easier for a deep learning network to find a better solution for the whole dataset.

Figure 5: Resnet-20-WW: Classes of airplane. It shows the images and their corresponding matrixes.

Figure 6: Resnet-20-WW: Classes of cars. It shows the images and their corresponding matrixes.

3.3 The analysis of ResNet

According to figures 5 and 6, we could find there some similities between their matrixes in different samples.

In the equation 13, $Z_1 E$ is a unit matrix. $T\hat{k}Z_2\Lambda$ could write as a matrix W . So the residual process can be simply written as:

$$\begin{aligned}\hat{x}_i^{l+2} &= \hat{x}_i^l W^l W^{l+1} + \hat{x}_i^l \\ &= \hat{x}_i^l (W^l W^{l+1} + I)\end{aligned}\quad (13)$$

$\hat{x}_i^l, \hat{x}_i^{l+2}$ are the input of the l th and $l+2$ th layers for the simple x_i . W^l, W^{l+1} are corresponding matrixes of the l th and $l+1$ th layers. So all the middle processes can be written as $\hat{x}_i^L = \hat{x}_i^l \mathbb{W}_i$. $\hat{x}_i^L$ is the output before the linear operation. $\mathbb{W}_i$ is responsible for simplifying the input data (including pooling).

The last process of Cifar-10 on Resnet is $\hat{y}_i = \hat{x}_i^L W^L$. All data share the same W^L .

So the whole processes before the last linear function are just trying to simplify the input data and W^L is the corresponding matrix which can fulfill all the simplified data. So the ResNet could find the relationship of all samples and make them can be calculated in the same matrix. Besides, we could find that compared with full convolutional neural networks, the matrixes $\mathbb{W}$ is more varied. It could help to provide information of original inputs.

4 Rethinking neural networks

For most of fully connected neural networks or full convolutional neural networks, they just generate their corresponding matching matrixes before the last operations which can fulfill the tasks. But it is too difficult to meet the requirements of all samples, we will reach the bottleneck. So we introduce some structures which can simplify the data and make it easier to deal with to achieve a state-of-the-art result, like ResNet. Many people are engaging in creating a new model. But there is a crucial problem, whether the works are meaningful like trying to achieve great results on some datasets or whether the models have an upper bound.

We take a ten classification problem on ResNet as an example. The real question that deep learning networks are solving is as follows.

$$x_1^T = \hat{x}_1^T \mathbb{W}_1; x_2^T = \hat{x}_2^T \mathbb{W}_2; x_3^T = \hat{x}_3^T \mathbb{W}_3 \cdots x_n^T = \hat{x}_n^T \mathbb{W}_n \quad (14)$$

$\hat{x}_1^T; \hat{x}_2^T, \hat{x}_3^T, \cdots \hat{x}_n^T$ are the input data. $\mathbb{W}_1, \mathbb{W}_2, \mathbb{W}_3 \cdots \mathbb{W}_n$ are the operation before the last linear operation.

Those are responsible for decomposing the data and making them can be calculated by one matrix.

$$\begin{pmatrix} x_1^T \\ x_2^T \\ x_3^T \\ \vdots \\ x_n^T \end{pmatrix} (k_1 \ k_2 \ k_3 \ \cdot \ \cdot \ k_{10}) \equiv \begin{pmatrix} \hat{y}_{1,1} & \hat{y}_{1,2} & \cdot & \cdot & \hat{y}_{1,10} \\ \hat{y}_{2,1} & \hat{y}_{2,2} & \cdot & \cdot & \hat{y}_{2,10} \\ \hat{y}_{3,1} & \hat{y}_{3,2} & \cdot & \cdot & \hat{y}_{3,10} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \hat{y}_{n,1} & \hat{y}_{n,2} & \cdot & \cdot & \hat{y}_{n,10} \end{pmatrix} \quad (15)$$

$x_i^T \subseteq \mathbb{R}^{1 \times m}$ are simplified inputs for last linear operation of different samples. $k_i \subseteq \mathbb{R}^{m,1}$ is the corresponding vector for ten classes. Only one class is right for one sample, so the values of the corresponding classes need to exceed other values. Most of the time $n \gg m$, therefore we need to find a group of parameters that match as many as possible samples.

This is a first order inequality equations with multivariate, but the equations do not have strict targets. It just has some corresponding states. (If the targets are right, the corresponding values are larger than other values.) So we could call it multivariate first-order state equations(MFSE). When we know labels, we could write the equation as:

$$\begin{pmatrix} x_{1,1} & x_{1,2} & \cdot & \cdot & x_{1,m} \\ x_{2,1} & x_{2,2} & \cdot & \cdot & x_{2,m} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ x_{n,1} & x_{n,2} & \cdot & \cdot & x_{n,m} \end{pmatrix} \begin{pmatrix} w_{1,1} & w_{1,2} & \cdot & \cdot & w_{1,10} \\ w_{2,1} & w_{2,2} & \cdot & \cdot & w_{2,10} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ w_{m,1} & w_{m,2} & \cdot & \cdot & w_{m,10} \end{pmatrix} \\
\preceq \begin{pmatrix} x_{1,1} & x_{1,2} & \cdot & \cdot & x_{1,m} \\ x_{2,1} & x_{2,2} & \cdot & \cdot & x_{2,m} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ x_{n,1} & x_{n,2} & \cdot & \cdot & x_{n,m} \end{pmatrix} \odot \begin{pmatrix} w_{k^1,1} & w_{k^1,2} & \cdot & \cdot & w_{k^1,m} \\ w_{k^2,1} & w_{k^2,2} & \cdot & \cdot & w_{k^2,m} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ w_{k^n,1} & w_{k^n,2} & \cdot & \cdot & w_{k^n,m} \end{pmatrix} \quad (16)$$

$\preceq$ means all the elements of fore matrix are less than or equal to the back matrix's elements in all rows. $\odot$ means that the two matrices multiply each other's elements and do summation in row direction.

We could simply write the 16 as:

$$\begin{pmatrix} x_1^T \\ x_2^T \\ x_3^T \\ \cdot \\ \cdot \\ x_n^T \end{pmatrix} (k_1 \ k_2 \ k_3 \ \cdot \ \cdot \ k_{10}) \preceq \begin{pmatrix} x_1 k^1 \\ x_2 k^2 \\ x_3 k^3 \\ \cdot \\ \cdot \\ x_n k^n \end{pmatrix} \quad (17)$$

k^j is the corresponding the k_j for x_i and j is the number of class. We need to find the the a combination of middle process and the best matrix for the linear operation. But considering that the $n \gg m$, so sometimes this equation perhaps do not have a solution. So perhaps most of time we are trying to solve a problem without a solution. We just try the best to find a solution could satisfy most of the samples. So there's a occasionality in solving this problem, sometimes we could get a better result.

5 Summary and Conclusion

Based on the the theorems, we know that the full connected neural networks and convolutional neural networks can be written as $\hat{y} = x^T \mathbb{W}$ in simple. For most of tasks, the essence for a single sample inputting a deep learning network is some underdetermined equations for input vectors: the $\hat{y} = x^T \mathbb{W}$. It has to have a solution. When we consider the whole dataset, the whole function can be written as $\hat{y}_1 = \hat{x}_1^T \mathbb{W}_1, \hat{y}_2 = \hat{x}_2^T \mathbb{W}_2 \dots$. According to samples, the combinations of all parameters in networks will generate the corresponding matrixes.

Now, we could answer the questions in the Introduction.

1. How deep learning works:

- Full connected and full convolutional neural networks: Deep learning networks will generate the corresponding matrixes of a sample to match the targets.
- ResNet: It will decompose the inputs, then finish the tasks by the same matrix.

2. The generalization: The generalization depends on whether a network can memory as many as possible of samples' matrixes. Perhaps, some of their matrixes have similarities, and one of the tasks of a network is to find the similarity.

- Full connected and full convolutional neural networks: If the deep learning networks could generate most of the matching matrixes of samples in the dataset, the model will show the property of generalization, otherwise, it will show overfitting or not fitting.
- ResNet: To some extent, it depends on the results of decomposition. If the results after decomposition is good enough, this problem have one or some solutions. Otherwise, this is an equation without a solution.

3. What is the theory of deep learning: The essence of deep learning is matrix multiplication. Deep learning is doing its best to find some combination of parameters that can fulfill as many as possible samples.

To sum up, the deep learning networks are machines that can solve complex equations. They can convert complex datas into simple feature which can be solved by multivariate equation of first order with the same parameters.

References

- Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.
- Matthew D Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. *arXiv e-prints*, art. arXiv:1311.2901, November 2013.
- Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and Understanding Neural Models in NLP. *arXiv e-prints*, art. arXiv:1506.01066, June 2015.
- Andrej Karpathy, Justin Johnson, and Li Fei-Fei. Visualizing and Understanding Recurrent Networks. *arXiv e-prints*, art. arXiv:1506.02078, June 2015.
- David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network Dissection: Quantifying Interpretability of Deep Visual Representations. *arXiv e-prints*, art. arXiv:1704.05796, April 2017.
- David Alvarez-Melis and Tommi S. Jaakkola. On the Robustness of Interpretability Methods. *arXiv e-prints*, art. arXiv:1806.08049, June 2018.
- Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv e-prints*, art. arXiv:1611.03530, November 2016.
- Haohan Wang, Xindi Wu, Zeyi Huang, and Eric P. Xing. High Frequency Component Helps Explain the Generalization of Convolutional Neural Networks. *arXiv e-prints*, art. arXiv:1905.13545, May 2019.
- Xiaofei Sun, Diyi Yang, Xiaoya Li, Tianwei Zhang, Yuxian Meng, Han Qiu, Guoyin Wang, Eduard Hovy, and Jiwei Li. Interpreting Deep Learning Models in Natural Language Processing: A Review. *arXiv e-prints*, art. arXiv:2110.10470, October 2021.
- Brent Mittelstadt, Chris Russell, and Sandra Wachter. Explaining Explanations in AI. *arXiv e-prints*, art. arXiv:1811.01439, November 2018.
- Zachary C. Lipton. The Mythos of Model Interpretability. *arXiv e-prints*, art. arXiv:1606.03490, June 2016.
- Simon S. Du, Jason D. Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient Descent Finds Global Minima of Deep Neural Networks. *arXiv e-prints*, art. arXiv:1811.03804, November 2018.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv e-prints*, art. arXiv:1512.03385, December 2015.