Unified Deep Learning

Wei Wang*

Abstract

With the increase in depth and complexity, deep learning networks have made significant progress in various directions. However, the theoretical understanding of deep learning is still incomplete. Early research successfully proved the universal approximation theorem for linear networks, but these proofs were limited to linear networks. Subsequent studies attempted to prove the approximation properties of convolutions and Transformers, but the proof processes often relied on complex assumptions or were very intricate. This paper aims to propose a unified approach to demonstrate that multi-layer networks composed of convolutions and Transformers are specific realizations of the universal approximation theorem. This approach does not require any assumptions and proves that most networks composed of convolutions and Transformers can be mathematically written in the same form as the fully proven universal approximation theorem, thus establishing them as specific implementations of the universal approximation theorem. This bridges the gap between deep learning practice and theoretical understanding. The method of unifying them is to represent these network architectures (linear, convolutional, and Transformer) in matrix-vector form, hence this unified approach is called the matrix-vector method. This paper takes an important step towards unifying the entire field of deep learning. It deepens our theoretical understanding and reveals the fundamental principles behind the exceptional performance of these networks. It also paves the way for exploring new research directions and optimizing the learning process in various deep learning applications.

Keywords: Universal Approximation, Explainable Neural Network

1. Introduction

This article proposes a unified research approach for the theoretical study and model development of deep learning, providing valuable theoretical references. It demonstrates the unity of deep learning methods through a series of processes and proves the effectiveness of the universal approximation theory.

Firstly, the article aims to address the fundamental theoretical issues in deep learning. As is known to all, the initial theory of deep learning is the universal approximation theorem, which proves that linear networks comply with the universal approximation theorem. However, with the development of practices and application demands, convolutional and Transformer networks were proposed. Due to their complexity, the universal approximation theorem has not been extended to convolutional and Transformer networks. Although deep learning is developing rapidly and the network capabilities are becoming stronger, there have been some problems: parameter redundancy, generalization, interpretability, performance bottlenecks, and questions about the future development of deep learning. This article first uses the matrix-vector method to prove that the current popular linear, convolutional, and Transformer networks are specific implementations of the universal approximation theorem. Then, through the mathematical form of the universal approximation theorem, it provides a reasonable answer to the above problems.

*Corresponding author.
Email address: weiwangnnn@yeah.net (Wei Wang)

The article provides a new perspective on the relationship between convolutional, Transformer, and matrix-vector multiplication. This implies that both convolutional and Transformer networks can be represented using the same form of matrix-vector multiplication. This unity provides a fresh perspective to understand the connections and transformations between different models. In addition, this matrix-vector form is used to prove that most deep networks comply with the "universal approximation theorem," which is a crucial theoretical foundation of deep learning. The "universal approximation theorem" states that under certain conditions, a deep network can approximate any function to arbitrary precision. This result is essential for understanding the representational and learning capabilities of deep learning models, providing a solid foundation for the application and theoretical research of deep learning.

In summary, this article's contributions can be highlighted in three main aspects:

- I have introduced a matrix-vector approach, which serves as a unified method for various fundamental deep learning modules, such as Linear, Convolution, and Transformer. Through the utilization of this approach, we are able to represent these modules using a common mathematical framework. This signifies that we can intuitively compare the distinctions among these modules from a mathematical perspective, facilitating an analysis of their strengths and weaknesses. Consequently, this approach lays the foundation for effective model design. Furthermore, by leveraging this approach, I can demonstrate that the majority of deep learning models are specific implementations of the Universal Approximation Theorem.
- When using the matrix-vector method, I discover that the differences between linear, Transformer, and convolutional modules primarily lie in their receptive fields. Generally, the receptive fields of these three modules gradually decrease, although this is naturally influenced by the specific definitions of each model.
- I address some common issues in deep learning, including parameter redundancy, generalization performance, interpretability, performance bottlenecks, and the future development of deep learning, based on the Universal Approximation Theorem. I provide a more reliable mathematical explanation for these problems.

2. Matrix-Vector Method and Deep Learning

In the field of deep learning, we encounter a landscape dominated by three main network architectures: linear, transformer, and convolutional networks. Although these networks are touted for their capabilities, it is important to note that their advantages in this regard often rely on intuitive assumptions. To this end, in this section, I propose a new approach that goes beyond these intuitive boundaries. By leveraging the transformative power of matrix-vector methods, we seek to unify these different architectures under a comprehensive framework. Our approach promises to go beyond intuition and provide a rigorous framework to examine and understand the true advantages and limitations of each method. With the matrix-vector method as the cornerstone, we begin to explore systematically revealing the strengths and weaknesses of each network prototype. Our aim is to provide a deeper understanding of these networks, rather than relying on superficial assumptions. By adopting this unified approach, we hope that the field of deep learning will move towards a new era of evidence-based decision-making. Our aim is to provide a comprehensive perspective for researchers and practitioners, enabling them to make informed choices when selecting network architectures for specific challenges.

2.1. The Format and Advantages of Matrix-Vector method

The basis of the Matrix-Vector method is matrix multiplication. It is well-known that Linear, convolution and Transformer can be transformed into matrix multiplication. But they are corresponding different formats of formula. While the purpose of the Matrix-Vector method is to unify all of them. There are two steps of the Matrix-Vector method.

Reorganizing Input Data: In the context of data manipulation, input data is often encountered in the forms of 1D, 2D, or 3D structures. The objective is to restructure them into uniform 1D column vectors. For 1D data, it inherently aligns with the format of a 1D column vector, requiring no additional modifications. However, when dealing with 2D and 3D data, a methodical approach is employed. This entails systematically extracting elements from each row, following their sequential order, and assembling them in sequence to form a 1D column vector. To elaborate further, this process involves traversing the rows and extracting elements in the order they appear, subsequently arranging these elements vertically to establish the desired 1D column vector structure. It's akin to converting each row into a discrete column vector and then concatenating these vectors in sequential fashion. This procedure is replicated in a similar manner for output data reorganization.

Reordering and Crafting Parameter Matrices: Parameter matrices constitute a pivotal element within the gamut of deep learning, emanating from diverse foundational modules like convolutions and Transformers. The essence of this phase lies in the reconfiguration of parameters intrinsic to these elemental modules, subsequently giving rise to the birth of corresponding generated parameter matrices. The overarching aim is to engender a nexus between these generated matrices and the realigned input data. This nexus is ingeniously designed such that the multiplication of the generated parameter matrix by the restructured input data engenders the emergence of the reorganized output data. This intricate synergy between parameter matrices, input data transformation, and output data generation forms the bedrock of this comprehensive endeavor.

The Matrix-Vector method is a groundbreaking approach designed to harmonize fundamental building blocks such as Linear, Transformer, and convolutional networks, all within a unified framework:

$$T(\mathbf{x}) \rightarrow \mathbf{W}\mathbf{x}' \tag{1}$$

In this formulation, T signifies the foundational block in the realm of deep learning. $\mathbf{W}$ is the parameters generated from T. $\mathbf{x}$ represents the input of T, while $\mathbf{x}'$ corresponds to the rearranged data of $\mathbf{x}$ along the column direction to accommodate the matrix-vector approach. Allow a slight abuse of notation, and subsequently, for the sake of convenience, we will refer to both as $\mathbf{x}$ for ease of representation.

The elegance of the Matrix-Vector method lies in its ability to encapsulate diverse network architectures into a singular, comprehensible format. This uniform representation not only streamlines the understanding of these building blocks but also provides a pivotal foundation for comparative analysis. By transforming intricate models into a common language, the method facilitates a nuanced exploration of their shared attributes and distinctions. Further, we prove that almost all multi-layer networks based on the basic blocks in deep learning can be transformed into a matrix multiplying a vector by the Matrix-Vector method are the variation of the Universal Approximation theory. The proof is shown in 3.

2.2. Matrix-Vector Method for Convolution

Convolution stands as one of the foundational cornerstones within the realm of deep learning, serving as a fundamental unit of paramount significance. Consequently, this article embarks on elucidating the intricate nexus existing between convolution and matrix multiplication. The scope

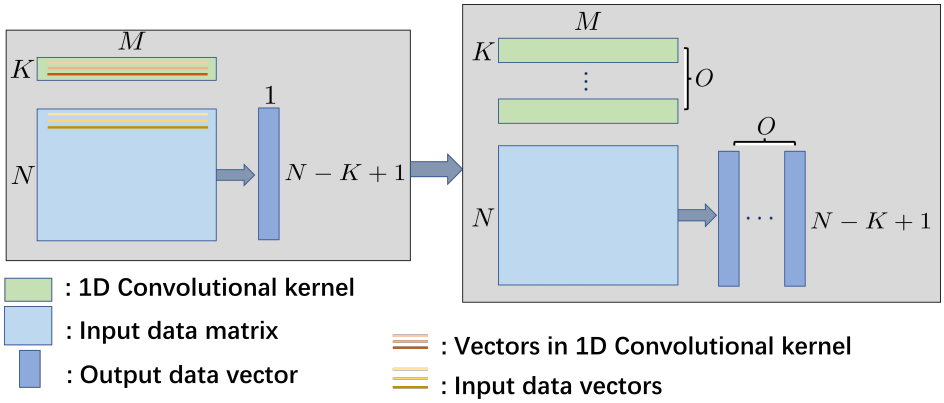

Figure 1: The diagram illustrates 1D convolution, with single-kernel and single-channel output on the left, and multiple-kernel and multiple-channel output on the right.

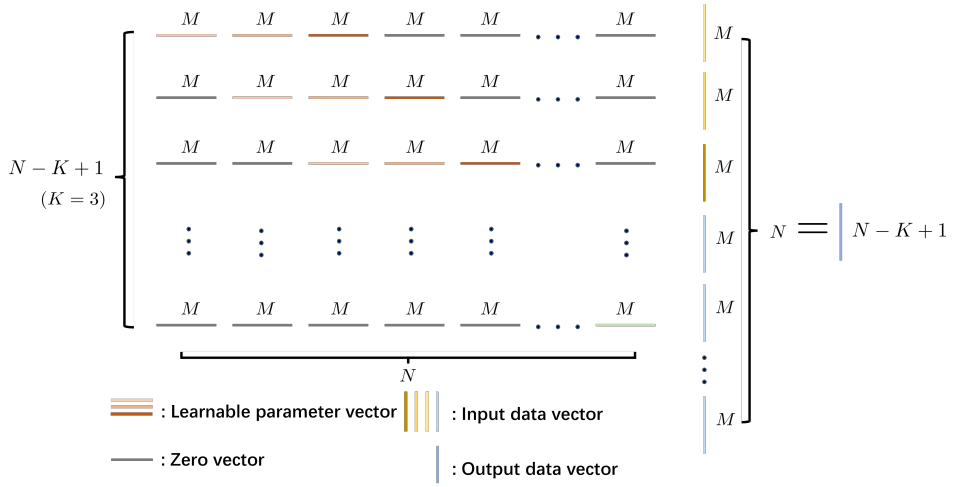

Figure 2: The matrix-vector representation of 1D convolution involves treating each row of the $N * M$ input matrix as a column vector. These individual column vectors are then concatenated in the column direction to form a new, extended column vector. Simultaneously, each row of the convolutional kernel is treated as a row vector, and these three vectors are concatenated in the row direction. Following this, the vectors and matrices are organized according to the layout depicted in the diagram, with gray vectors filling the designated positions as zeros. This method yields the output result of the convolution. The colors of the vectors in this illustration correspond to those in Figure 1.

of this elucidation primarily revolves around 1D convolution, as the methodologies employed for substantiating 2D, 3D, or higher-dimensional convolutions draw from analogous principles. To initiate this exploration, we commence by portraying the quintessence of 1D convolution through Figure 1, an illustrative representation that encapsulates the essence of this operation.

Drawing inspiration from Figure 1, we proceed to formulate the matrix multiplication congruent with the left-hand side output of single-channel convolution, as depicted in Figure 2. Moreover, we extrapolate this conceptual framework to encompass the matrix multiplication associated with the right-hand side outcome of multi-channel convolution, as showcased in Figure 3. Building upon the bedrock established by these visualizations, we adeptly formulate 1D convolution as a profound marriage between matrix multiplication and a vector, aptly summarized by the formula:

$$\mathbf{x}^{i+1} = \mathbf{W}\mathbf{x}^{i} \tag{2}$$

Here, $\mathbf{x}^{i} \in \mathbb{R}^{MN}$ represents the input of the i-th layer, whereas $\mathbf{x}^{i+1} \in \mathbb{R}^{O(N-k+1)}$ encapsulates

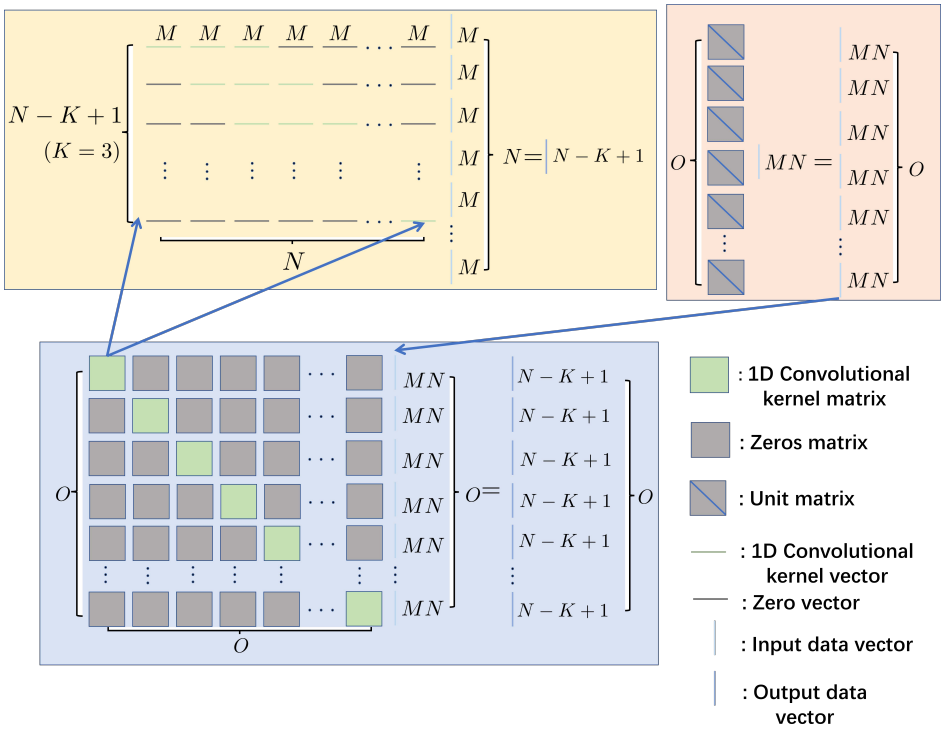

Figure 3: This is the matrix representation of convolution with multiple channel outputs. Firstly, the concatenated $M * N$ column vectors are copied O times in the column direction using the identity matrix. Then, the convolution is written as a matrix multiplied by a column vector using the method described in Figure 2.

the output from the same layer. The matrix $\mathbf{W} \in \mathbb{R}^{O(N-k+1)\times MN}$ exhibits a pronounced level of sparsity.

Importantly, it should be acknowledged that the approach maintains its inherent analogy when extended to the investigation of 2D and 3D networks. This effectively obviates the requirement for exhaustive elucidation. Consequently, convolution can be succinctly represented through the Matrix-Vector method as a matrix multiplied by a vector.

2.3. Matrix-Vector Method for Transformer

In section 2.2, we effectively conveyed the mechanism of convolution through the Matrix-Vector method. In this section, we further extend this approach to illustrate the Transformer's inner workings using a similar matrix-based representation. The Transformer architecture comprises two fundamental components: the Feed-Forward Network (FFN) and the Multi-Head Attention structure. FFT involves convolution, which we already discussed in Section 2.2. Our focus in this section, however, is on the multi-head attention mechanism. Therefore, moving forward, when we mention the Transformer, we are referring to the multi-head attention module within the Transformer. The formula for this mechanism is provided by the following equation:

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}\left(\mathbf{H}'_1, \ldots, \mathbf{H}'_h\right) \mathbf{W}_O \tag{3}$$

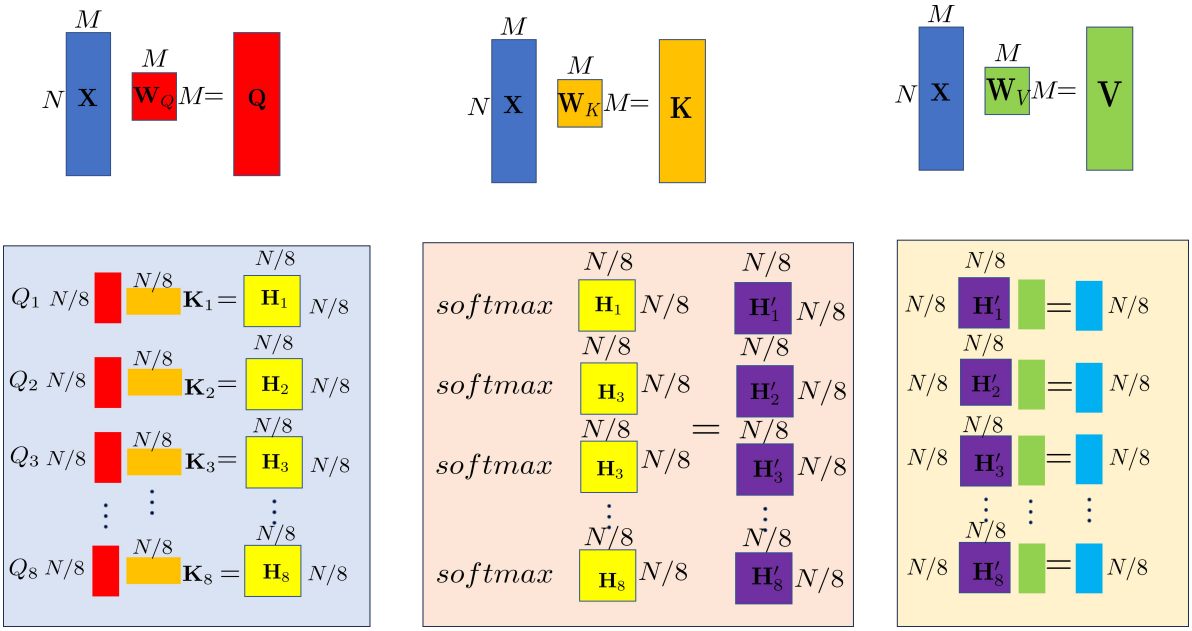

Figure 4: The image depicts the basic module of the Transformer, which is multi-head attention. The boxes are matrices.

where

$$\begin{aligned}\mathbf{H}_i' &= \text{Attention}\left(\mathbf{x}_i\mathbf{W}_{Qi} = \mathbf{Q}_i, \mathbf{x}_i\mathbf{W}_{Ki} = \mathbf{K}_i, \mathbf{x}_i\mathbf{W}_{Vi} = \mathbf{V}_i\right) \\ &= \text{softmax}\left(\frac{\mathbf{Q}_i\mathbf{K}_i^T}{\sqrt{M}}\right)\mathbf{V}_i \\ &= \text{softmax}\left(\frac{\mathbf{H}_i}{\sqrt{M}}\right)\mathbf{V}_i\end{aligned} \tag{4}$$

Here, h denotes the number of attention heads, and the input $\mathbf{x} \in \mathbb{R}^{N*M}$ is divided into $\mathbf{x}_1, ..., \mathbf{x}_h$ based on h. The parameters $\mathbf{W}_{Qi}, \mathbf{W}_{Ki}$, and $\mathbf{W}_{Vi}$ correspond to $\mathbf{x}_i$.

A pivotal operation involves both left and right matrix multiplications. This section follows the approach outlined in Section 2.2. First, we provide an overarching depiction of the multi-head attention, illustrated in Figure 4. It illustrates the multi-head attention mechanism. The intricate architecture of the Transformer revolves around this crucial module. To clarify the exposition of the multi-head attention mechanism and represent it in the form of matrix-vector multiplication, we will depict its transformation process using imagery. Symbolically, we will omit the indexing of heads in the multi-head attention, as the computational mechanisms across different heads are identical. The conversion of the multi-head attention mechanism into matrix-vector multiplication involves two key points: left multiplication by matrices and right multiplication by matrices. We will illustrate these principles using the intermediate steps of the multi-head attention mechanism, specifically $\mathbf{x}\mathbf{W}_Q$ and $\mathbf{H}'\mathbf{V}$.

First, we introduce Figure 5. Figure 5 elucidates the concept of $\mathbf{x}\mathbf{W}_Q$ within the context of the multi-head attention mechanism. Different colors highlight distinct components of vectors in $\mathbf{x}$ and $\mathbf{W}_Q$, establishing a visual correlation between the two. In Figure 6, we reiterate the concept of $\mathbf{x}\mathbf{W}_Q$ from Figure 5 as a matrix-vector multiplication. It is crucial to note that we meticulously maintain color alignment of vector components between the two images. Thus far, we have presented the method of converting matrix right multiplication into matrix-vector multiplication.

Next, we will utilize $\mathbf{H}'\mathbf{V}$ from the multi-head attention mechanism to illustrate how to express

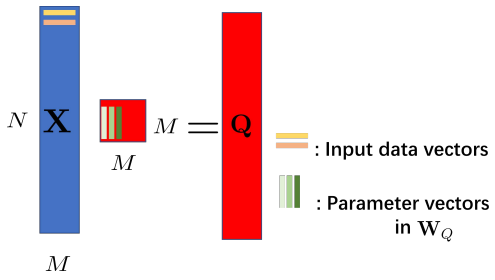

Figure 5: The image illustrates the concept of $\mathbf{x}\mathbf{W}_Q$ in the attention. We represent different parts of the vectors in $\mathbf{x}$ and $\mathbf{W}_Q$ using different colors.

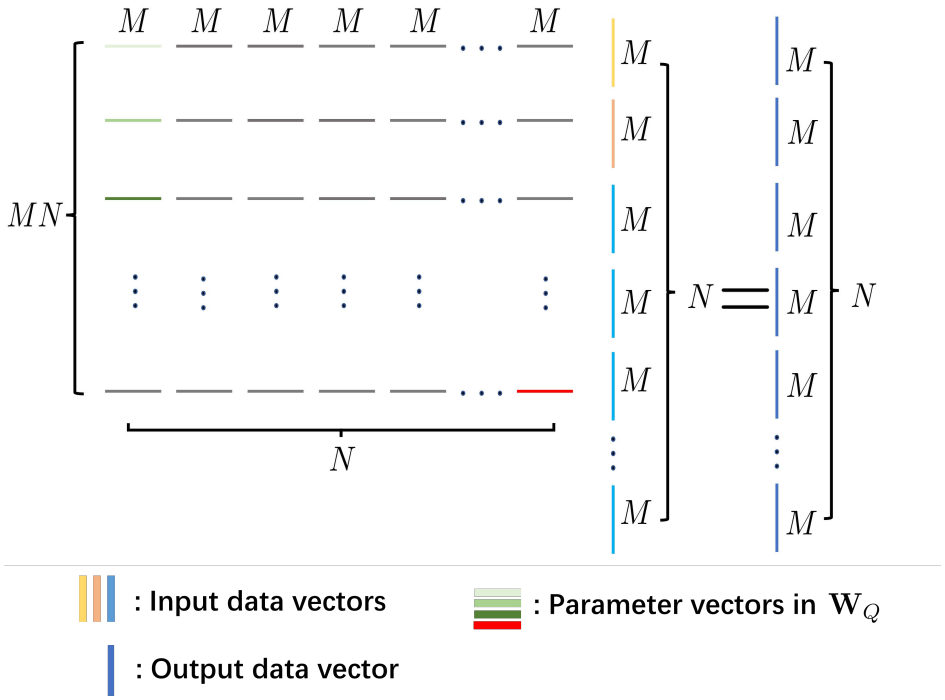

Figure 6: The representation of $\mathbf{x}\mathbf{W}_Q$ in the multi-head attention by a matrix multiplying a vector.

matrix left multiplication in the form of matrix-vector multiplication. First, we introduce Figure 7. Figure 7 provides a visual representation of $\mathbf{H}'\mathbf{V}$, where segments of vectors are depicted using differently colored lines, aimed at elucidating the transformation process clearly. The depiction in Figure 7 can be expressed in the matrix-vector multiplication form shown in Figure 8. The color correspondence between vectors in the two figures allows us to observe the transformation through vector operations. With this, we have demonstrated how matrix left multiplication can be transformed into the matrix-vector multiplication form.

Next, we are going to simplify the multi-head attention mechanism. First, we can directly consider the $\mathbf{H}'_1...\mathbf{H}'_h$ within the multi-head attention mechanism as parameters, as they are derived from the calculations involving $\mathbf{W}_{Q1}...\mathbf{W}_{Qh}$ and $\mathbf{W}_{K1}...\mathbf{W}_{Kh}$. Therefore, we only need to focus on $\mathbf{H}'_1\mathbf{W}_{V1}...\mathbf{H}'_h\mathbf{W}_{Vh}$. This implies that the three components shown at the bottom of Figure 4 can be transformed into the format depicted in Figure 9. This means that we can directly represent the multi-head attention mechanism as a matrix multiplication, as illustrated in the formula:

$$\mathbf{x}^{i+1} = \mathbf{W}_{H'VO}\mathbf{x}^i \tag{5}$$

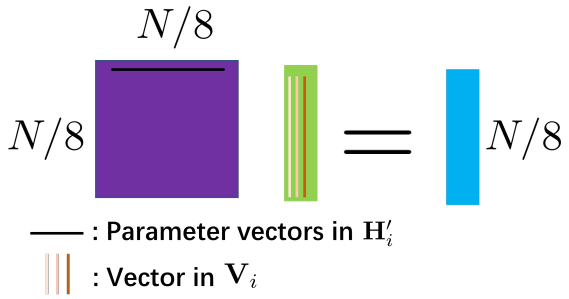

Figure 7: The representation of $\mathbf{H}'\mathbf{V}$. We use different color to represent different vectors in $\mathbf{H}'_i$ and $\mathbf{V}_i$.

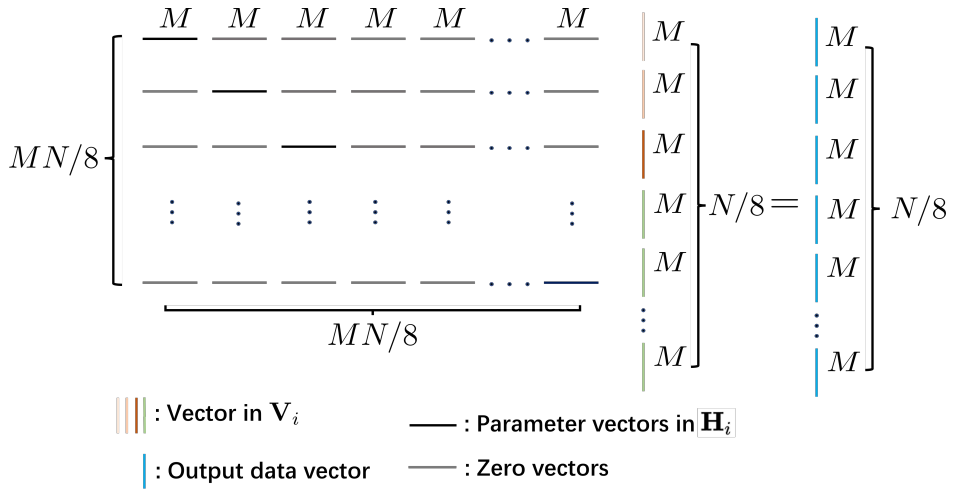

Figure 8: The matrix multiplication representation of $\mathbf{H}'\mathbf{V}$ in the form of a matrix multiplying a vector. Please pay attention to the correspondence between the vector colors in this figure and the vector colors in Figure 7.

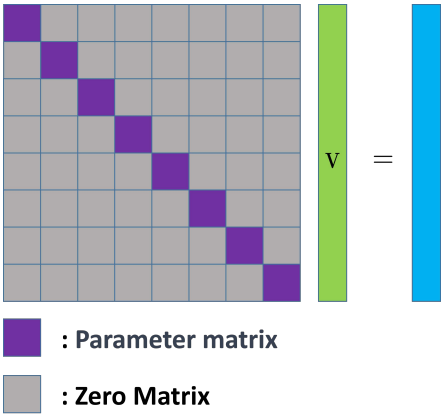

Figure 9: Representing $\mathbf{H}'\mathbf{V}$ within the multi-head attention mechanism using sparse matrix multiplication.

Here, $\mathbf{x}^i \in \mathbb{R}^{NM}$ and $x^{i+1} \in \mathbb{R}^{NM}$ are the input and output of ith layer, while $\mathbf{W}_{H'VO} \in \mathbb{R}^{NM*NM}$ is matrices generated in accordance with $\mathbf{H}'$, $\mathbf{W}_V$ and $\mathbf{W}_O$.

2.4. Summary

In summary, the derivations presented above reveal that by employing the matrix-vector method, we can represent Linear, Convolutional, and Transformer models as the product of parameter ma-

trices and input data vectors. Consequently, these fundamental modules all fall under the same mathematical framework—parameter matrix-vector multiplication. The key distinction lies in the sparsity and arrangement of matrix parameters, which vary across different fundamental modules. Specifically, Linear models typically involve learning dense parameter matrices, while Convolutional models emphasize sparse parameter matrices. In contrast, Transformer models occupy an intermediate position in terms of matrix sparsity. It is important to note that this classification is not absolute and depends on the specific parameter design of each fundamental module. When we assemble these fundamental modules, such as through the incorporation of activation functions, into multi-layer networks like ResNet, we discover that most of these composite models belong to the same mathematical model category: universal approximation theory. Hence, they share almost identical mathematical theories and conclusions. As a result, we can observe that the theoretical foundations and conclusions drawn from these models are largely interchangeable. In practice, these models often exhibit similar performance trends or alternate improvements. This practical unity further solidifies the understanding that Linear, Convolutional, and Transformer models are indeed manifestations of a common mathematical framework.

3. The Universal Approximation Theory and Matrix-Vector Method

The universal approximation theorem is a fundamental principle in deep learning and the basis for the feasibility of deep learning. It was originally introduced by Cybenko [1] and further developed by Hornik [2]. In recent years, various conclusions and proofs have been developed around this theorem. This article uses the universal approximation theorem conclusion provided in Cybenko's [1] article as an example to prove that linear, convolutional, and Transformer are all concrete implementations of this theorem. Because I have converted these transformations into the matrix-vector form required by the universal approximation function in Cybenko's article, other subsequent derivations and conclusions based on this article are applicable to networks composed of multilayer linear, convolutional, and Transformer.

Theorem 2 from [1] states that if σ is any Borel measurable function, then finite sums of the form:

$$G(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \sigma\left(\mathbf{W}_j^{\mathrm{T}}\mathbf{x} + \theta_j\right) \tag{6}$$

are dense in $C\left(I_n\right)$. Here, $\mathbf{W}_j \in \mathbb{R}^n$ and $\alpha_j, \theta \in \mathbb{R}$ are fixed. For any $f \in C\left(\mathbf{I}_n\right)$ and $\varepsilon > 0$, there exists a sum $G(\mathbf{x})$ of the above form for which:

$$|G(\mathbf{x}) - f(\mathbf{x})| < \varepsilon \quad \text{for all} \quad \mathbf{x} \in \mathbf{I}_n. \tag{7}$$

This implies that, with a sufficiently large value of N, a single-layer neural network can approximate any function. We have demonstrated that Linear, Convolutional, and Transformer architectures can all be expressed as matrix-vector multiplication forms:

$$T(\mathbf{x}) \rightarrow \mathbf{W}\mathbf{x}' \tag{8}$$

When networks consist of multiple layers, architectures like ResNet [3] are commonly employed. For instance, a two-layer network is shown in equation (9).

$$\begin{aligned} T_1(\mathbf{x}) &= \mathbf{W}_1\mathbf{x} + \sigma\left(\mathbf{W}_1'\mathbf{x}\right) \\ T_2(\mathbf{x}) &= \mathbf{W}_2 T_1(\mathbf{x}) + \sigma\left[\mathbf{W}_2' T_1(\mathbf{x})\right] \\ &= \mathbf{W}_2\left[\mathbf{W}_1\mathbf{x} + \sigma\left(\mathbf{W}_1'\mathbf{x}\right)\right] + \sigma\{\mathbf{W}_2'\left[\mathbf{W}_1\mathbf{x} + \sigma\left(\mathbf{W}_1'\mathbf{x}\right)\right]\} \\ &= \mathbf{W}_2\mathbf{W}_1\mathbf{x} + \mathbf{W}_2\sigma\left(\mathbf{W}_1'\mathbf{x}\right) + \sigma\left[\mathbf{W}_2'\mathbf{W}_1\mathbf{x} + \mathbf{W}_2'\sigma\left(\mathbf{W}_1'\mathbf{x}\right)\right] \end{aligned} \tag{9}$$

By analyzing the network expansion of ResNet, E.q. 9, and the universal approximation theorem, E.q. 6, it can be seen that E.q. 9 can be regarded as a different mathematical representation of E.q. 6. The difference lies in the definition of the parameters α_j,$\mathbf{W}_j$ and θ_j. In E.q. 6, these parameters are initialized randomly, while in E.q. 9, these parameters are related to the weight parameters of the previous layer. Therefore, all networks with multi-layer transformations of similar form are specific implementations of the universal approximation theorem. It can be seen that the most important parts in the implementation of deep learning are the activation function and the summation. In addition, the concept of multi-scale corresponds to increasing the depth of the network, which corresponds to the value of N in the universal approximation theorem.

4. Discussion

Now we can provide explanations for the previously mentioned questions regarding parameter redundancy, generalization, interpretability, performance bottlenecks, and the future development of deep learning.

Firstly, the existence of parameter redundancy in deep learning is due to the excessively low weights of certain layers. We can analyze this directly from the formula of the universal approximation theorem:

$$|\sum_{j=1}^{N} \alpha_j \sigma\left(\mathbf{W}_j^{\mathrm{T}}\mathbf{x}+\theta_j\right)-f(\mathbf{x})|<\varepsilon \quad \text{for all} \quad \mathbf{x}\in\mathbf{I}_n. \tag{10}$$

Let's assume $N_1+N_2=N$ and $|\sum_{j=1}^{N_2}\alpha_j\sigma\left(\mathbf{W}_j^{\mathrm{T}}\mathbf{x}+\theta_j\right)|\to 0$. Then we have:

$$|\sum_{j=1}^{N_1} \alpha_j \sigma\left(\mathbf{W}j^{\mathrm{T}}\mathbf{x}+\theta_j\right)+\sum_{j=1}^{N_2} \alpha_j \sigma\left(\mathbf{W}_j^{\mathrm{T}}\mathbf{x}+\theta_j\right)-f(\mathbf{x})|<\varepsilon \tag{11}$$

Since $|\sum_{j=1}^{N_2}\alpha_j\sigma\left(\mathbf{W}_j^{\mathrm{T}}\mathbf{x}+\theta_j\right)|\to 0$, we have the following inequality:

$$\begin{aligned}&\left||\sum_{j=1}^{N_1} \alpha_j \sigma\left(\mathbf{W}_j^{\mathrm{T}}\mathbf{x}+\theta_j\right)-f(\mathbf{x})|-|\sum_{j=1}^{N_2} \alpha_j \sigma\left(\mathbf{W}j^{\mathrm{T}}\mathbf{x}+\theta_j\right)|\right| \\ \leqslant &|\sum_{j=1}^{N_1} \alpha_j \sigma\left(\mathbf{W}j^{\mathrm{T}}\mathbf{x}+\theta_j\right)+\sum_{j=1}^{N_2} \alpha_j \sigma\left(\mathbf{W}_j^{\mathrm{T}}\mathbf{x}+\theta_j\right)-f(\mathbf{x})|<\varepsilon\end{aligned} \tag{12}$$

Therefore, we have:

$$|\sum_{j=1}^{N_1} \alpha_j \sigma\left(\mathbf{W}j^{\mathrm{T}}\mathbf{x}+\theta_j\right)-f(\mathbf{x})|<\varepsilon+|\sum_{j=1}^{N_2} \alpha_j \sigma\left(\mathbf{W}_j^{\mathrm{T}}\mathbf{x}+\theta_j\right)| \tag{13}$$

Hence, when parameters in certain layers are small, we can directly remove those layers since their impact on the final result is minimal.

The second question regarding the generalization of deep learning is related to the approximation conditions of the universal approximation theorem. The universal approximation theorem states that any continuous function on a closed interval can be approximated. However, when there are multiple continuous functions within a closed interval, the network cannot approximate them all. This is the fundamental reason why deep learning encounters performance bottlenecks on certain publicly available datasets.

Regarding the interpretability of deep learning, it is unnecessary to provide explanations such as visualizing intermediate layers in the network. The universal approximation theorem can be understood as solving a function equation, and we are simply finding the parameters that correspond to the solution of the equation. The number of network layers and operations such as multi-scale processing essentially increase the value of N in the universal approximation theorem. Overfitting and underfitting are errors in function approximation, and a large amount of data allows the network to approximate high-dimensional functions that are more suitable for most of the data.

Regarding the future development of deep learning, I believe more attention should be given to the data itself, specifically the closed interval continuous functions. Since the universal approximation theorem has already shown that it can approximate any function within a closed interval, we should attempt to classify the data and assign different closed interval continuous functions to different categories, using different networks for fitting. Alternatively, we can design algorithms to transform all the data into a continuous function within a closed interval.

5. Conclusion

In this paper, we delve into the foundational issues of deep learning theory. Specifically, I introduce the Matrix-Vector Method, a technique that unifies Linear, Transformer, and Convolutional modules under a common mathematical framework. Leveraging this method, the current deep learning models can be unified within the framework of the Universal Approximation Theory. Integrating the Universal Approximation Theory offers a solution to some of the existing challenges in current deep learning.

Comments

This article represents the culmination of five years of work for me, completed during my doctoral application period. It embodies what I aspired to achieve since the beginning of my graduate studies in 2018. At that time, I genuinely believed that deep learning was a magical and essential element for the future of humanity. However, when I attempted to implement my own models, they didn't perform as well as others', leaving me consistently perplexed. At times, I felt that deep learning was akin to building with blocks, and I envied those who published numerous papers. So, I set out to uncover the fundamental principles of deep learning and achieve success step by step.

Subsequently, I delved into researching and exploring the relationships among deep learning parameters. I also reviewed existing theoretical or explanatory papers on deep learning, but I found them lacking compared to my research (a belief I still hold today). Their studies quickly become obsolete, whereas what I did 3-4 years ago remains relevant and aligns with the principles of universal approximation theory. However, when I revisit the articles I wrote earlier, I feel nauseated; they are poorly written.

Amidst my confusion, I persisted in learning. Finally, one day, seemingly in my third year of graduate studies, I had a revelation that the fundamental solution to deep learning lies in matrix multiplication. This idea lingered, and I proceeded to provide some proofs. The more I proved, the more convinced I became. Although some of the proofs may have flaws in hindsight, the overall direction was correct. Later, I introduced the matrix-vector method, unifying everything under the framework of the universal approximation theorem. I knew I was on the right track.

However, this article remains unpublished. Perhaps it's due to my poor narrative skills. When I read articles published by others, I am genuinely perplexed. I feel lost and unfair. Some of what they do seems essentially worthless. They may not even use the theories they propose. Reviewing

my previous research, it essentially adheres to the fundamental principles of deep learning: the universal approximation theory, but it still cannot be published. I do not understand the significance of academic research. Is a unified deep learning article like mine not worth publishing? I don't know if it will ever be recognized and brought up by other scholars. Perhaps, my paper will fade into the river of time, producing no ripples.

References

[1] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals, and Systems*, p. 303–314, Jan 2007. [Online]. Available: http://dx.doi.org/10.1007/bf02551274

[2] K. Hornik, M. B. Stinchcombe, and H. L. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, 1989. [Online]. Available: https://api.semanticscholar.org/CorpusID:2757547

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