Spectral Complexity-scaled Generalisation Bound of Complex-valued Neural Networks

Haowen Chen[a,b,c], Fengxiang He[d,b,*], Shiye Lei[e], Dacheng Tao[e,b]

[a]*Department of Mathematics, ETH Zürich, 8092 Zürich, Switzerland*
[b]*JD Explore Academy, JD.com, Inc., Beijing, 100176, China*
[c]*Department of Mathematics, Faculty of Science, The University of Hong Kong, Hong Kong SAR*
[d]*Artificial Intelligence and its Applications Institute, School of Informatics, University of Edinburgh, Edinburgh EH8 9AB, United Kingdom*
[e]*School of Computer Science, Faculty of Engineering, The University of Sydney, Darlington NSW 2008, Australia*

Abstract

Complex-valued neural networks (CVNNs) have been widely applied in various fields, primarily in signal processing and image recognition. Few studies have focused on the generalisation of CVNNs, although it is vital to ensure the performance of CVNNs on unseen data. This study is the first to prove a generalisation bound for complex-valued neural networks. The bounds increase as the spectral complexity increases, with the dominant factor being the product of the spectral norms of the weight matrices. Furthermore, this work provides a generalisation bound for CVNNs trained on sequential data, which is also affected by the spectral complexity. Theoretically, these bounds are derived using the Maurey Sparsification Lemma and Dudley entropy integral. We conducted empirical experiments on various datasets including MNIST, ashionMNIST, CIFAR-10, CIFAR-100, Tiny ImageNet, and IMDB by training complex-valued convolutional neural networks. The Spearman rank-order correlation coefficient and the corresponding p-values on these datasets provide strong proof of the statistically significant correlation between the spectral complexity of a network and its generalisation ability, as measured by the spectral norm product of the weight matrices. The code is available at https://github.com/LeavesLei/cvnn_generalization.

Keywords: Complex-valued neural networks, Generalisation, Spectral complexity

[*] Corresponding author
Email address: F.He@ed.ac.uk (Fengxiang He)

1. Introduction

Complex-valued neural networks (CVNNs) have garnered significant attention in various fields, such as signal processing [1, 2], voice processing [3], and image reconstruction [4]. To reduce complex operations, it is natural to link CVNNs to two-dimensional real-valued neural networks with fewer degrees of freedom [5, 6]. A complex number consists of a real part and imaginary part, which can alternatively be expressed as amplitude and phase. When performing computations using complex numbers, distinct arithmetic operations are applied separately to the real and imaginary parts.

Several recent studies endeavoured to investigate the different properties of CVNNs and built basic algorithms for their implementation. For example, Nitta [7, 8, 9, 10] proved the orthogonality of the decision boundary of complex-valued neurones, addressed the redundancy problem of the parameters of CVNNs, extended the backpropagation algorithm to complex numbers, and Trabelsi et al. [11] organised the essential components of complex-valued deep neural networks, such as complex convolutions, complex batch normalisation, and complex weight initialisation. Empirical studies were conducted to examine the experimental performance of CVNNs. Hirose and Yoshida [5] used different neural networks, including CVNNs, to process signals of different coherence and Nitta [7] found that for the same computational cost, CVNNs display a higher learning speed than real-valued neural networks.

Previous studies have shown satisfactory experimental performance for CVNNs. However, there is still a lack of theoretical analysis of their generalisation ability. This gap in understanding has motivated us to derive a generalisation bound for CVNNs.

This is the first study to provide theoretical evidence for the generalisation performance of CVNNs. We propose novel upper bounds which positively correlate with the spectral complexity of CVNNs trained on both independent identically distributed (i.i.d.) and sequential data. The spectral complexity-scaled upper bounds suggest a direct correlation between the generalisation ability of CVNNs and the spectral norm product of their complex-valued weight matrices.

From an empirical perspective, the experiments were conducted to investigate the influence of spectral complexity on the generalisation ability. Specifically, we trained CVNNs using stochastic gradient descent (SGD) on six standard datasets: CIFAR-10, CIFAR-100, MNIST, FashionMNIST, Tiny ImageNet, and IMDB. Excess risks were collected for analysis. When the training error is almost zero across all datasets, the excess risk equals the test accuracy and is informative in expressing generalisation ability. In addition, because the change in the spectral norm product of the weight matrices primarily contributes to the change in spectral complexity, it is used to simulate spectral complexity. Our experimental results demonstrate a strong correlation between the spectral-norm product and excess risk, which is consistent with our theoretical analysis. The code is available at https://github.com/LeavesLei/cvnn_generalization.

The remainder of this paper is organised as follows. Section 2 presents the motivation behind the research and provides a review of related work. Section 3 provides an introduction to the preliminaries of complex-valued neural networks. Section 4 presents the theoretical results, while Section 5 presents the experimental results. In Section 6, a comparison is made between CVNNs and real-valued neural networks to explore the novelties and advantages of the proposed bound. In Section 7, the practical applications of the proposed theorems in spectral normalisation algorithms are discussed in detail.

2. Motivation and Related Works

Complex values are widely adopted in different neural networks for their biological [12], computational [7, 13], and representational advantages [14, 15].

From a biological perspective, Reichert and Serre [12] proposed that the complex-valued neuronal unit is a more appropriate abstraction in modelling the activity of neurones in the brain than a real-valued unit. To better process cortical information, the modelling mechanism must consider both firing rate and spike timing. In incorporating these two elements into deep neural networks, the amplitude of a complex-valued neuron represents the firing rate and the phase represents the spike timing. When two inputs of an excitatory complex-valued neuron have similar or dissimilar phase information, the magnitude of the net input may increase or decrease depending on whether the phases are similar, which correspond to synchronous and asynchronous situations, respectively. The incorporation of complex values into deep neural networks helps construct richer and more versatile representations.

Regarding the computational aspect, Danihelka et al. [13] combined long short-term memory (LSTM) with the concept of holographic reduced representations and used complex values to increase the efficiency of information retrieval. Experiments showed that this method achieves a faster learning speed on multiple memorisation tasks. Nitta [7] extended the back-propagation algorithm to complex values, preserving the basic idea of real-valued back-propagation, with updates conducted on both real and imaginary parts. Through experiments, it was demonstrated that under the same time complexity, the learning speed of complex backpropagation is definitely faster than the real speed when the learning rate is low, that is, less than 0.5.

Complex-valued neural networks also provide advantages over real-valued neural networks in terms of representational ability. Arjovsky et al. [14] proposed a unitary recurrent neural network (RNN) with unitary matrices as the weight matrix, to circumvent the well-studied gradient vanishing and gradient exploding issues. The unitary matrix is the generalised form of the orthogonal matrices in the complex field, and the absolute value of its eigenvalue is 1. Compared to an orthogonal matrix, a complex-valued matrix has a richer representation, particularly in applications of the discrete Fourier Transform. Wisdom et al. [15] further proposed full-capacity unitary RNNs, thereby improving the performance over unitary evolution RNN (uRNN).

Given these advantages and applications of CVNNs, an increasing number of researchers have been investigating the properties of complex-valued neural networks to provide a basic framework for the implementation of CVNNs. Nitta [8] demonstrated that the decision boundary of a two-layered complex-valued network is orthogonal, and for a three-layered network, the decision boundary is nearly orthogonal. This reflects the computational power and versatality of complex values. In their work, Trabelsi et al. [11] provided the building blocks for complex-valued deep neural networks, including complex batch normalisation and complex weight initialisation strategies. They also compared the performances of different activation functions on three datasets: CIFAR-

10, CIFAR-100, and SVHN.

While there are studies presenting empirical results on the generalisation performance of complex-valued neural networks [7, 5], there is still a lack of theoretical evidence to support these findings. Therefore, our study aims to present the first upper bound for the generalisation error of CVNNs.

Various complexity measures have been proposed to derive an upper bound for the generalisation error of real-valued neural networks, such as VC-dimension and Rademacher complexity [16]. Bartlett et al. [17] proved a margin-based multi-class generalisation bound based on covering number and Rademacher complexity. These two tools are also used in our work. Compared with Bartlett et al. [17]'s work, our work defines the spectral complexity of CVNNs by defining the spectral norm of a complex-valued matrix and providing generalisation bounds for complex-valued neural networks when processing regression tasks.

3. Preliminaries

This section introduces complex-valued neural networks (CVNNs) and presents the notations used in the theoretical analysis.

3.1. Model Construction

Each layer of CVNN consists of several complex-valued neurones, as described below. The input signals, weight parameters, threshold values, and output signals are all complex numbers in complex-valued neurones. Assuming that the complex-valued neurone n is linked with m neurones in the previous layer, the net input to this neurone n is described as follows:

$$T^n_{\text{input}} = \sum_{i=1}^{m} W_{in} X_{in} + H_n.$$

Here, T^n_{input} denotes the complex-valued network input of neurone n and W_{in} denotes the weight connecting the n and i neurones from the previous layer. X_{in} denotes the complex-valued input signal from neurone i to neurone n and H_n denotes the threshold value of neurone n. If we denote $Re(T^n_{\text{input}})$ and $Im(T^n_{\text{input}})$ as the real and imaginary parts of T^n_{input}, respectively, and $\left|T^n_{input}\right|$ and θ^n_{input} as the amplitude and phase of T^n_{input}, respectively, then the output of neurone n can be described as follows:

$$T^n_{\text{output}} = f_r\left(Re(T^n_{\text{input}})\right) + f_i\left(Re(T^n_{\text{input}})\right) \tag{1}$$

or

$$T^n_{\text{output}} = e^{i f_p\left(\theta^n_{\text{input}}\right)} f_a\left(\left|T^n_{input}\right|\right)_. \tag{2}$$

Equation (1) describes the output derived by applying the activation function separately to the real and imaginary parts, whereas Equation (2) describes the situation when the activation function is applied to the amplitude and phase, where f_r is the activation function applied to the real part; f_i is the activation function applied to the imaginary part; f_p is the activation function applied to the phase; and f_a is the activation function applied to the amplitude.

3.2. Complex-valued Activation Functions

Several forms of complex-valued activation functions corresponding to real-valued functions have been proposed.

Arjovsky et al. [14] has proposed a modReLU activation function, which preserves the phase information and applies the real-valued ReLU function to the amplitude. This function is described as follows:

$$\text{modReLU}(z) = \text{ReLU}(|z| + b)e^{i\theta_z} = \begin{cases} (|z| + b)\frac{z}{|z|} & \text{if } |z| + b \geq 0 \\ 0 & \text{otherwise} \end{cases}.$$

In this formula, $|z|$ denotes the amplitude of the complex number z and $b \in \mathbb{R}$ denotes the threshold for the amplitude of z.

Nitta [10] applied the hyperbolic tangent function to the real and imaginary parts of the input complex number to propose the following activation function:

$$\sigma(z) = \tanh(\text{Re}(z)) + i \tanh(\text{Im}(z)),$$

where $i = \sqrt{-1}$, $\tanh(u) \overset{\text{def}}{=} (\exp(u) - \exp(-u))/(\exp(u) + \exp(-u)), u \in \mathbb{R}$.

These two functions represent two main types of complex-valued activation functions with one applied to the real and imaginary parts, and the other applied to the amplitude and phase values. There are other variations of activation functions, such as zReLU and $\mathbb{C}$ReLU [18]. These different activation functions have different properties in terms of satisfying the Cauchy-Riemann equations. Therefore, it is important to carefully select activation functions based on the specific task at hand and the characteristics of the data.

3.3. Basic Notations and Definitions

Suppose $S = \{(z_1, y_1), (z_2, y_2), (z_3, y_3), ..., (z_n, y_n) | z_i \in \mathcal{Z} \subset \mathbb{C}^{d_Z}, y_i \in \mathcal{Y} \subset \mathbb{C}^{d_Y}\}$ is the training sample set, where y_i is the corresponding label of z_i, d_Z and d_Y are the dimensions of the z and y separately. We define $\mathcal{D}$ as a distribution following (z_i, y_i).

Assume that the network has L layers, and in the ith layer, an ρ_i-Lipschitz activation function $\sigma_i : \mathbb{C}^{d_i} \to \mathbb{C}^{d_i}$ (activation functions such as the $\mathbb{C}$ReLU function and hyperbolic tangent function can be used here. Their Lipschitz properties are proven in Appendix A) and a weight matrix $A_i \in \mathbb{C}^{d_{i-1} \times d_i}$ are applied to the input matrix passed from the previous layer. Let $\sigma_i(0) = 0$, $\mathcal{A} = (A_1, A_2, ..., A_L)$, and $F_{\mathcal{A}}$ be the function computed by CVNNs:

$$F_{\mathcal{A}}(z) := \sigma_L \left(A_L \sigma_{L-1} \left(A_{L-1} \cdots \sigma_1 \left(A_1 z\right)\right)\right).$$

with output $F_{\mathcal{A}}(z) \in \mathbb{C}^{d_L}$ $\big($it is assumed that $d_0 = d_Z = d, d_L = d_Y$, and $W = max\{d_0, d_2, ..., d_L\}\big)$. For input data $\{z_1, z_2, ..., z_n\}$, a matrix $Z \in \mathbb{C}^{n \times d}$ can be formed by collecting each z_i in the ith row. Therefore, the output of this neural network can be expressed as $F_{\mathcal{A}}(Z^T)$, the ith column of which is $F_{\mathcal{A}}(z_i)$.

To avoid ambiguity, it is necessary to clarify the definition of a complex-valued matrix norm. The norm of any complex matrix $[A_{i,j}] \in \mathbb{C}^{d \times k}$ is defined as the norm of a corresponding real-valued matrix as follows:

$$\left\| [A_{i,j}] \right\|_p \triangleq \left\| [|A_{i,j}|] \right\|_p,$$

where $[A_{i,j}]$ denotes the matrix whose i, jth entry is $A_{i,j}$. In this paper, the L_2 norm is calculated entrywise, which means that the L_2 matrix norm is defined to be the

Frobenius norm, i.e.,

$$\|A\|_2 \triangleq \sqrt{\sum_i \sum_j |A_{i,j}|^2}.$$

Moreover, $\|\cdot\|_\sigma$ denotes the spectral norm:

$$\|A\|_\sigma := \sup_{\|v\|_2=1} \|Av\|_2 = \sqrt{\lambda_{max}(A^* A)},$$

where A^* denotes the Hermitian transpose of A, and λ_{max} denotes the largest absolute value of eigenvalues of A. Meanwhile, $\|A\|_{p,q}$ is defined as:

$$\|A\|_{p,q} : \triangleq \left\| \left(\|A_{:;1}\|_p , \|A_{:;2}\|_p , ..., \|A_{:;m}\|_p \right) \right\|_q$$

for $A \in \mathbb{C}^{d \times m}$.

To investigate generalisation ability, it suffices to derive a high probability bound for the generalisation error:

$$\underset{(z,y)\sim\mathcal{D}}{E}[l(F_{\mathcal{A}}(z), y)] - \frac{1}{n}\sum_{i=1}^{n} l(F_{\mathcal{A}}(z_i), y_i),$$

where $l(F_{\mathcal{A}}(z), y) : \mathcal{Z} \times \mathcal{Y} \to \mathbb{R}$ denotes the loss function, which is usually set as

$$l(F_{\mathcal{A}}(z), y) = ||F_{\mathcal{A}}(z) - y||_2.$$

Finally, the spectral complexity $R_{\mathcal{A}}$ of neural network $F_{\mathcal{A}}$ is defined as follows [17]:

$$R_{\mathcal{A}} := \left(\prod_{i=1}^{L} \rho_i \|A_i\|_\sigma \right) \left(\sum_{i=1}^{L} \frac{\|A_i^\top\|_{2,1}^{2/3}}{\|A_i\|_\sigma^{2/3}} \right)^{3/2} .$$

This complexity measure plays a crucial role in the generalisation bound presented in the next section.

4. Main Theorems and Proof Sketch

4.1. Generalisation Bound

In this section, the main theorems of this study are presented.

Theorem 1. *(independent and identically distributed data)*

Let $S = \{(z_1, y_1), (z_2, y_2), (z_3, y_3), ..., (z_n, y_n)\}$ be a sample dataset of size n with elements drawn independently and identically from distribution $\mathcal{D}$. Given activation functions σ_i (σ_i is ρ_i-Lipschitz and $\sigma_i(0) = 0$) and weight matrices $\mathcal{A} = (A_1, A_2, ..., A_L)$, as stated in Section 3.3, then with probability of at least $1 - \delta$, the corresponding complex-valued neural network must satisfy the following:

$$\begin{aligned}&\mathbb{E}_{(z,y)\sim\mathcal{D}}[l(F_{\mathcal{A}}(z), y)] - \frac{1}{n}\sum_{i=1}^{n} l(F_{\mathcal{A}}(z_i), y_i) \\ &\qquad\leq \frac{8M}{n^{\frac{3}{2}}} + \frac{36\,\|Z\|_2\sqrt{2\ln(2W)}\ln(n)R_{\mathcal{A}}}{n} + 3M\sqrt{\frac{\ln\frac{2}{\delta}}{2n}},\end{aligned}$$

where $l(F_{\mathcal{A}}(z), y) = ||F_{\mathcal{A}}(z) - y||_2$ denotes the loss function, and $l(F_{\mathcal{A}}(z), y) \leq M$ for any (z, y).

It is observed that there is no explicit occurrence of any combinatorial parameters, such as L (the depth of neural networks). However, this upper bound depends on L implicitly, as $R_{\mathcal{A}}$ is formed by each layer's weight matrix norm and the Lipschitz constant of each layer's activation function.

The full proof is provided in Appendix B, and a proof sketch is presented in Section 4.2.

Theorem 2. *(sequential data) Consider $S = \{(z_1, y_1), (z_2, y_2), (z_3, y_3), ..., (z_n, y_n)\}$ as a sample dataset, where $(z_t)_{t\geq 1}$ is a sequence of random data adopted to filter $(\mathcal{A}_t)_{t\geq 1}$. Given the activation functions σ_i (σ_i is ρ_i-Lipschitz and $\sigma_i(0) = 0$), and weight matrices $\mathcal{A} = (A_1, A_2, ..., A_L)$, as stated in Section 3.3, then with a probability of at least $1 - \delta$, the corresponding complex-valued neural network must satisfy the following:*

$$\begin{aligned}&\frac{1}{n}\sum_{t=1}^{n}\left(\mathbb{E}\left[l\left(z_t, y_t\right) \mid \mathcal{A}_{t-1}\right] - l\left(z_t, y_t\right)\right) \\ &\leq \frac{8M}{n} + \frac{24\,\|Z\|_2\sqrt{2\ln(2W)}\ln(n)R_{\mathcal{A}}}{n} + M\sqrt{\frac{\ln\frac{2}{\delta}}{2n}},\end{aligned}$$

where $l(z, y) = ||F_{\mathcal{A}}(z) - y||_2$ denotes the loss function, and $l(z, y) \leq M$ for any (z, y).

Theorem 2 illustrates the generalisation capability of complex-valued neural networks when dealing with sequential data. The proof sketch of this theorem is omitted from the main text because there exists some overlap with Theorem 1; however, the full proof is shown in Appendix D. In the Appendix, we also present the definitions of sequential Rademacher complexity, sequential covering number, and sequential Dudley entropy integral, which were proposed in the work of Rakhlin et al. [19].

4.2. Proof Sketch

In this section, we provide the proof of Theorem 1, using the following lemmas:

The proof is presented in three steps: **I)** An upper bound for the covering number is obtained: $\mathcal{N}(\{ZA : A \in \mathbb{C}^{d\times m}, \|A\|_{q,s} \leq a, \epsilon\})$, as in Lemma 1. **II)** Starting with a single layer and applying the induction method, an upper bound for the covering number of the entire network is derived. This result is illustrated in Lemma 2. **III)** The upper bound of Rademacher complexity is derived via Dudley entropy integral and the above covering number bound, further combining this bound with Theorem 3.

In preparation for the proof, we first state Theorem 3, which is a crucial tool for step **III**. This theorem derives the generalisation bound for regression in the case of L_p loss function through Rademacher complexity. Let us recall the theorem presented by Mohri et al. [16].

Theorem 3 ([16])**.** *Let $l : \mathcal{Z} \times \mathcal{Y} \to R$ be an L_p loss function bounded by $M > 0$; $\mathcal{F}$ be the hypothesis set; family $\mathcal{G} = \{(x, y) \mapsto l(F_{\mathcal{A}}(x), y) : F_{\mathcal{A}} \in \mathcal{F}\}$, then for any δ, with probability at least $1 - \delta$, the following inequality holds:*

$$\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}[l(x, y)] \leq \frac{1}{m}\sum_{i=1}^{m} l(x_i, y_i) + 2\hat{\mathfrak{R}}_S(\mathcal{G}) + 3M\sqrt{\frac{\log\frac{2}{\delta}}{2m}}$$

where $\hat{\mathfrak{R}}_S(\mathcal{G})$ denotes the empirical Rademacher complexity of family $\mathcal{G}$.

Obviously, to bound the generalisation error, it suffices to derive an upper bound for the Rademacher complexity of the loss function family $\mathcal{G} = \{(x, y) \mapsto l(F_{\mathcal{A}}(x), y) : F_{\mathcal{A}} \in \mathcal{F}\}$, which is realised through the first and second steps. In the following paragraphs, we illustrate the proof in detail, in three steps.

Step I In this step, we obtain a matrix covering for the set of matrix products ZA (Z represents the data matrix passed to the present layer, and A is instantiated as the weight matrix) under L_2 norm. Lemma 1 is derived as follows:

Lemma 1. *(p, q) and (r, s) are two conjugate exponents with $p \leq 2$. Let a, b and ϵ be three positive real numbers, and let d and m be two positive integers. Impose a constraint on the norm of Z such that $\|Z\|_p \leq b$. Therefore, we have*

$$\ln \mathcal{N}\left(\left\{ZA : A \in \mathbb{C}^{d\times m}, \|A\|_{q,s} \leq a\right\}, \epsilon, \|\cdot\|_2\right) \leq \left\lceil \frac{a^2 b^2 m^{2/r}}{\epsilon^2} \right\rceil \ln(4dm).$$

The proof of Lemma 1 is based on the Maurey sparcification lemma. This lemma inspired us to cover the targeting set using a sparsifying convex hull of complex-valued matrices, which is constructed using the product of the rescaled data matrix Z [20] and some ”standard matrices” such as $\mathbf{e}_i\mathbf{e}_j^T$. Moreover, to prove Theorem 1, constraints are imposed on $||A||_{2,1}$ (i.e. $q = 2$, $s = 1$), instead of $||A||_2$, which helps avoid the occurrence of combinatorial numbers such as L and W outside the log term in the upper bound [17].

Step II After obtaining the matrix covering upper bound in **Step I**, the idea is extended to the proof of the entire network covering number upper bound, thereby obtaining Lemma 2. The proof of Lemma 2 relies on induction and on Lemma 1.

Lemma 2. *$(\sigma_1, \sigma_2, \sigma_3, ..., \sigma_L)$ are fixed activation functions with each σ_i being $\rho_i - Lipschitz$. The spectral norm bound of matrix A_i is denoted by s_i and the matrix (2,1) norm is denoted by b_i ($i \in \{1, 2, ..., L\}$). Given Z as the fixed data matrix, where $Z \in \mathbb{C}^{n\times d}$, and each row denotes a group of data points, for any ϵ, we have*

$$\ln \mathcal{N}\left(\mathcal{F}, \epsilon, \|\cdot\|_2\right) \leq \frac{\|Z\|_2^2 \ln\left(4W^2\right)}{\epsilon^2} \left(\prod_{j=1}^{L} s_j^2 \rho_j^2\right) \left(\sum_{i=1}^{L} \left(\frac{b_i}{s_i}\right)^{2/3}\right)^3,$$

where $\mathcal{F} := \left\{F_{\mathcal{A}}\left(Z^T\right) : \mathcal{A} = (A_1, \ldots, A_L), \|A_i\|_\sigma \leq s_i, \left\|A_i^\top\right\|_{2,1} \leq b_i\right\}$ is the family of outputs generated by feasible choices of complex-valued neural networks $\mathcal{F}_{\mathcal{A}}$, and W denotes the maximum of $\{d_0, d_1, ..., d_L\}$.

In general, we separate the proof of Lemma 2 into two parts: The first part determines the relationship between the entire network upper bound and the matrix covering bounds of the previous L layers, which is addressed in Appendix B.2 Lemma 6. The second part combines Lemmas 1 and 6, which together provide Lemma 2 through the induction technique.

Step III Since we are only deriving a bound for the covering number of the whole network $\mathcal{N}(\mathcal{F}, \epsilon, \|\cdot\|_2)$ is not sufficient. We still have to derive an upper bound for the empirical Rademacher complexity of the loss function family $\hat{\mathfrak{R}}_S(\mathcal{G})$. It is natural to connect these two concepts using the Dudley entropy integral. However, some preparation work is required to satisfy the condition for using the standard Dudley entropy integral.

The standard Dudley entropy integral only illustrates the relationship between $\mathcal{N}(\mathcal{G}, \epsilon, \|\cdot\|_2)$ and $\hat{\mathfrak{R}}_S(\mathcal{G})$. Hence, Lemma 3 has upper bounds $\mathcal{N}(\mathcal{G}, \epsilon, \|\cdot\|_2)$ by $\mathcal{N}(\mathcal{F}, \epsilon, \|\cdot\|_2)$

Lemma 3. *Given family* $\mathcal{F} := \{F_{\mathcal{A}}(Z) : \mathcal{A} \in \mathcal{B}_1 \times \cdots \times \mathcal{B}_L\}$ *and family* $\mathcal{G} := \{(z, y) \mapsto l\left(F_{\mathcal{A}}(z), y\right) : F_{\mathcal{A}} \in \mathcal{F}\}$*, the covering number of these two families satisfy*

$$\mathcal{N}(\mathcal{F}, \epsilon, \|\cdot\|_2) \geq \mathcal{N}(\mathcal{G}, \epsilon, \|\cdot\|_2), \tag{3}$$

Let $l\left(F_{\mathcal{A}}(z), y\right) = \|\mathcal{F}_{\mathcal{A}}(z) - y\|_2$.

Moreover, because the range of the loss function that we adopt does not lie in $[0, 1]$, Lemma 4 calculates the covering number after rescaling.

Lemma 4. *If a coefficient, say* $\alpha > 0$*, is multiplied by the targeting set* $\mathcal{G}$ *and distance constant* ϵ*, then the covering number remains unchanged, that is,*

$$\mathcal{N}(\mathcal{G}, \epsilon, \|\cdot\|_2) = \mathcal{N}(\alpha\mathcal{G}, \alpha \cdot \epsilon, \|\cdot\|_2).$$

Here, $\alpha\mathcal{G}$ *represents a set obtained by scaling* α *to each element in* $\mathcal{G}$*.*

Using Lemmas 3 and 4, the Rademacher complexity of $\mathcal{G}$ can be bounded through the Dudley entropy integral. We prove Theorem 1 by substituting $\hat{\mathfrak{R}}_S(\mathcal{G})$ into Theorem 3 for the value of the upper bound obtained. A detailed proof is provided in Appendix B.3.

5. Results of Experiments

In this section, we present the experimental results of training complex-valued neural networks using SGD on six datasets: MNIST, FashionMNIST, CIFAR-10, CIFAR-100, Tiny ImageNet, and IMDB.

Before presenting the experimental results, a summary of the two upper bounds is derived in Theorems 1 and 2. Both for the independent identically distributed and sequential data cases, we showed that the derived upper bound scales with the spectral complexity of the complex-valued neural network:

$$R_{\mathcal{A}} := \left(\prod_{i=1}^{L} \rho_i \left\|A_i\right\|_{\sigma}\right) \left(\sum_{i=1}^{L} \frac{\left\|A_i^{\top}\right\|_{2,1}^{2/3}}{\left\|A_i\right\|_{\sigma}^{2/3}}\right)^{3/2}.$$

The formula for the spectral norm $R_{\mathcal{A}}$ consists of two parts: the Lipschitz constant of this neural network $\left(\prod_{i=1}^{L} \rho_i \left\|A_i\right\|_{\sigma}\right)$ and another factor related to the sum of quotients of weight matrix norms $\left(\left(\sum_{i=1}^{L} \frac{\left\|A_i^{\top}\right\|_{2,1}^{2/3}}{\left\|A_i\right\|_{\sigma}^{2/3}}\right)^{3/2}\right)$. Since the two norms are equivalent, the factor $\left(\left(\sum_{i=1}^{L} \frac{\left\|A_i^{\top}\right\|_{2,1}^{2/3}}{\left\|A_i\right\|_{\sigma}^{2/3}}\right)^{3/2}\right)$ remains in the interval $[C_1, C_2]$ for some constants C_1 and C_2. Therefore, we assume that the change in $\left(\left(\sum_{i=1}^{L} \frac{\left\|A_i^{\top}\right\|_{2,1}^{2/3}}{\left\|A_i\right\|_{\sigma}^{2/3}}\right)^{3/2}\right)$ is minor. Then, in the training process, the part which dominates the change in $R_{\mathcal{A}}$ is $\left(\prod_{i=1}^{L} \rho_i \left\|A_i\right\|_{\sigma}\right)$ for the Lipschitz constant of the neural network. This is because the Lipschitz constants of activation functions (ρ_i) remain unchanged. Therefore, we use the change in the spectral norm product ($\prod_{i=1}^{L} \left\|A_i\right\|_{\sigma}$) to simulate the changing trend in $R_{\mathcal{A}}$.

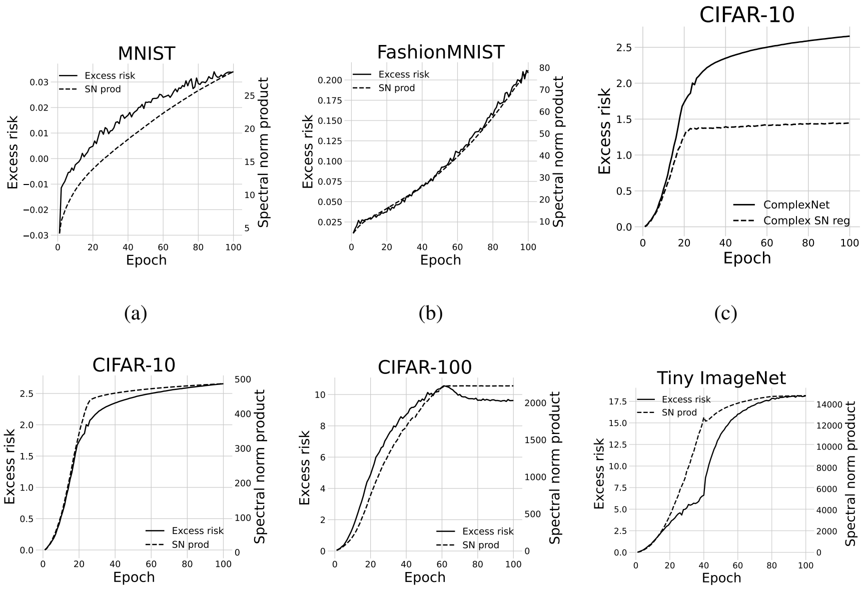
5.1. Spectral Norm of the Weight Matrix

First, the calculation of the spectral norm of the complex weight matrix in each convolutional layer is demonstrated.

Considering the complex-valued kernel $W = X + Yi$ in each layer, where X and Y are real-valued kernels, because each convolutional kernel corresponds to a matrix transformation [21], the real-valued weight matrices of kernels X and Y can be derived, which are denoted by C and D. Hence, the complex-valued weight matrix A of each layer can be expressed as $C + Di$. Then, by definition, the spectral norm of the complex-valued matrix A is:

$$\begin{aligned}\|A\|_\sigma := \sup_{\|v\|_2=1} \|Av\|_2 &= \sqrt{\lambda_{\max}\left(A^* A\right)} \\ &= \sqrt{\lambda_{\max}\left(C^T C + D^T D + (C^T D - C D^T) i\right)}.\end{aligned}$$

Here, because A^*A is a Hermitian matrix, it has only real eigenvalues.



(a) (b) (c)

(d) (e) (f)

Figure 1: Plots of excess risk and spectral norm product (SN prod) as functions of the epoch. The right y-axis denotes the spectral norm product, and the left y-axis denotes the excess risk.

5.2. *Complex-valued Convolutional Neural Networks*

In our experiments, we trained complex-valued convolutional neural networks (CVCNN) and complex-valued multi-layer perceptrons (CVMLP) on different datasets. The IMDb dataset was used for training the CVMLP. The proposed complex-valued neural network architectures are described in detail in Appendix E.2.

The CVCNN architecture consists of three types of layers: convolutional, maxpooling, and fully connected layers. The last layer is a CVMLP. The convolutional and maxpooling layers in CVCNN are analogous to the hidden layers in CVMLPs, as is the case in real-valued neural networks [22]. The operations on the convolutional and maxpooling layers can be expressed as matrix-vector multiplication, as shown in [23]. To analyze these results, we considered the convolutional and maxpooling layers separately.

5.2.1. Matrix Multiplication Interpretation of Convolution Operations

First, we interpret the convolution process using matrix multiplication.

The convolutional layer usually receives as input a matrix of dimensions $[h_1{*}w_1{*}d_1]$, which is a 3D matrix with length h_1, width w_1, and height d_1. Further, the kernels are defined as follows. Assuming that the number of output channels is d_2, we have d_2 kernels, which can be written as a matrix with dimensions $[h_2 * w_2 * d_1]$. Finally, we assume that the output matrix has dimensions $[h_3 * w_3 * d_2]$. Figure 2 shows the input, kernel, and output matrices of the convolution process.

The relationship between the output and input matrix dimensions can be expressed as follows:

$$w_3 = (w_1 - w_2 + 2p + 1)/s$$
$$h_3 = (h_1 - h_2 + 2p + 1)/s$$

where p denotes the padding number, and s indicates the stride number. Without loss of generality, we assume that $p = 0$ and $s = 1$. The next step is to flatten the input matrix into a vector and write these d_2 3D kernel matrices into a 2D matrix.

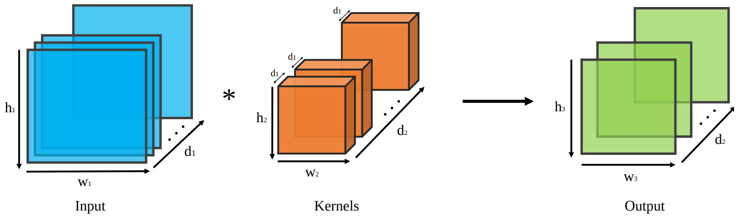


Figure 2: The input matrix, kernel, and output matrix in a convolution process.

Because the input matrix is of dimension $[h_1 * w_1 * d_1]$, it can be flattened into a vector $\boldsymbol{x} \in \mathbb{R}^{h_1 w_1 d_1}$ such that

$$\boldsymbol{x} = \left(x^1_{(1,1)}, x^1_{(1,2)}, \ldots, x^d_{(h,w)}, \ldots, x^{d_1}_{(1,1)}, \ldots, x^{d_1}_{(h_1,w_1)}\right)^T .$$

Here, $x^d_{(h,w)}$ denotes the (h, w, d)th entry of the input matrix ($0 < h \leq h_1, 0 < w \leq w_1$, $0 < d \leq d_1$).

In terms of constructing the kernel matrix, we first consider the convolutional matrix $\boldsymbol{K}^i_{(h,d)}$ formed by the vector $\boldsymbol{k}^i_{(h,d)} = \left(k^i_{(h,1,d)}, \ldots, k^i_{(h,w_2,d)}\right) \in \mathbb{R}^{w_2}$, where $k^i_{h,w,d}$ denotes the (h, w, d)th entry of the ith kernel ($0 < h \leq h_2$, $0 < w \leq w_2$, $0 < d \leq d_1$. $0 < i \leq d_2$). The convolutional matrix $\boldsymbol{K}^i_{(h,d)}$ induced by the vector $\boldsymbol{k}^i_{(h,d)} = \left(k^i_{(h,1,d)}, \ldots, k^i_{(h,w_2,d)}\right) \in \mathbb{R}^{w_2}$ is formed as follows:

$$\begin{bmatrix} k^i_{(h,1,d)} & k^i_{(h,2,d)} & k^i_{(h,3,d)} & \cdots & k^i_{(h,w_2,d)} & 0 & \cdots & 0 \\ 0 & k^i_{(h,1,d)} & k^i_{(h,2,d)} & k^i_{(h,3,d)} & \cdots & k^i_{(h,w_2,d)} & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & k^i_{(h,1,d)} & k^i_{(h,2,d)} & k^i_{(h,3,d)} & \cdots & k^i_{(h,w_2,d)} \end{bmatrix},$$

and $\boldsymbol{K}^i_{(h,d)}$ is the Toeplitz matrix in $\mathbb{R}^{(w_1-w_2+1)\times w_1}$.

The second step is to construct the convolutional matrix $\boldsymbol{K}^i_d$ induced by matrix $\boldsymbol{K}^i_{(h,d)}$. $\boldsymbol{K}^i_d$ is a Toeplitz block matrix in $\mathbb{R}^{(h_1-h_2+1)(w_1-w_2+1)\times h_1 w_1}$, which is constructed by considering $\boldsymbol{K}^i_{(h,d)}$ as its components. We can write $\boldsymbol{K}^i_d$ as follows:

$$\begin{bmatrix} \boldsymbol{K}^i_{(1,d)} & \boldsymbol{K}^i_{(2,d)} & \boldsymbol{K}^i_{(3,d)} & \cdots & \boldsymbol{K}^i_{(h_2,d)} & 0 & \cdots & 0 \\ 0 & \boldsymbol{K}^i_{(1,d)} & \boldsymbol{K}^i_{(2,d)} & \boldsymbol{K}^i_{(3,d)} & \cdots & \boldsymbol{K}^i_{(h_2,d)} & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \boldsymbol{K}^i_{(1,d)} & \boldsymbol{K}^i_{(2,d)} & \boldsymbol{K}^i_{(3,d)} & \cdots & \boldsymbol{K}^i_{(h_2,d)} \end{bmatrix}$$

Therefore, the entire convolutional matrix $\boldsymbol{K} \in \mathbb{R}^{d_2(h_1-h_2+1)(w_1-w_2+1)\times h_1 w_1 d_1}$ induced by the d_2 kernels, can be viewed as a block matrix by taking $\boldsymbol{K}^i_d$ as its components. It is constructed as follows:

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{K}_1^1 & \boldsymbol{K}_2^1 & \dots & \boldsymbol{K}_{d_1}^1 \\ \boldsymbol{K}_1^2 & \boldsymbol{K}_2^2 & \dots & \boldsymbol{K}_{d_1}^2 \\ \vdots & \ddots & \ddots & \vdots \\ \boldsymbol{K}_1^{d_2} & \boldsymbol{K}_2^{d_2} & \dots & \boldsymbol{K}_{d_1}^{d_2} \end{bmatrix}$$

Consequently, we regard $\boldsymbol{K}$ as the weight matrix of the convolutional layer and apply its spectral norm to calculate the generalisation bound.

For the maxpooling layer, there exists a weight matrix with entries of either 0 or 1, depending on which entry $\boldsymbol{x}$ is reserved for the maxpooling process. For instance, if each $x_{(1,1)}^d$ $(1 \leq d \leq d_1)$ is reserved, then the weight matrix $\boldsymbol{M} \in \mathbb{R}^{d_1 \times h_1 w_1 d_1}$ of the maxpooling layer is

$$\begin{bmatrix} 1 & 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 1 & 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & 1 & 0 & \dots & 0 \end{bmatrix}.$$

$\boldsymbol{M}$ is a sparse matrix and its (i, ih_1w_1) entry equals 1.

In conclusion, CVCNNs are a type of complex-valued neural network that still fits the theoretical analysis presented in Section 4.

5.3. Results

The architectures of the complex-valued neural networks of this study are described in Appendix E.2. The datasets used were MNIST, FashionMNIST, CIFAR-10, CIFAR-100, Tiny ImageNet, and IMDB. Descriptions of these datasets are presented in Appendix E.1. We trained CVCNNs using SGD on MNIST, FashionMNIST, CIFAR-10, CIFAR-100, and Tiny ImageNet to investigate the generalisation bound derived in Theorem 1 and the CVNN trained on IMDB to investigate the generalisation bound derived in Theorem 2 for sequential training data. The results are presented in Figure 1.

Figure 1 displays excess risk and spectral norm product as a function of epoch. In addition, we performed Spearman rank-order correlation tests on all excess risks and spectral norm products of MNIST, FashionMNIST, CIFAR-10, CIFAR-100, IMDB, and Tiny ImageNet. Spearman's rank-order correlation coefficients (SCCs) and p-values show that the correlation between the spectral norm product and generalisation ability is statistically significant ($p < 0.005$[1]), as listed in Table 1. These results strongly support our theoretical findings.

Table 1: SCC and p values of the spectral norm product and excess risk

CIFAR-10		CIFAR-100		MNIST	
SCC	p	SCC	p	SCC	p
0.99	3.703×10^{-228}	0.80	4.124×10^{-23}	0.99	9.044×10^{-142}
IMDB		FashionMNIST		Tiny ImageNet	
SCC	p	SCC	p	SCC	p
0.99	6.118×10^{-194}	0.99	3.703×10^{-142}	0.99	4.060×10^{-125}

6. Comparison with Real-valued Neural Networks

6.1. Theoretical Analysis

In previous studies, Bartlett et al. [17] proved that the generalisation bound of real-valued neural networks (RVNNs) is positively correlated with spectral complexity. In his work, spectral complexity is defined as the product of real-valued weight matrix spectral norms, as defined in the following formula, where A_i denotes the weight matrices of real-valued neural networks, and M_i denotes the reference matrices.

$$R_{\mathcal{A}} := \left(\prod_{i=1}^{L} \rho_i \left\| A_i \right\|_\sigma \right) \left(\sum_{i=1}^{L} \frac{\left\| A_i^\top - M_i^\top \right\|_{2,1}^{2/3}}{\left\| A_i \right\|_\sigma^{2/3}} \right)^{3/2}$$

The generalisation bound we presented for CVNNs is similar to the one for real-valued neural networks (RVNNs) presented in [17]. Theorem 4 illustrates the relationship between the generalisation bounds of RVNNs and their spectral complexity.

[1] The definition of "statistically significant" has various versions, such as $p < 0.05$ and $p < 0.01$. This paper used a more rigorous approach ($p < 0.005$).

Theorem 4 ([17]). *Let nonlinearities* $(\sigma_1, \ldots, \sigma_L)$ *and reference matrices* $\big(M_1, \ldots, M_L\big)$ *be given as above (that is,* σ_i *is* ρ_i*-Lipschitz and* $\sigma_i(0) = 0$*). Then for* (x, y) *,*(x_1, y_1)*,* $\ldots, (x_n, y_n)$ *drawn i.i.d. from any probability distribution over* $\mathbb{R}^d \times \{1, \ldots, k\}$*, with probability at least* $1 - \delta$ *over* $((x_i, y_i))_{i=1}^n$*, every margin* $\gamma > 0$ *and network* $F_{\mathcal{A}} : \mathbb{R}^d \to \mathbb{R}^k$ *with weight matrices* $\mathcal{A} = (A_1, \ldots, A_L)$ *satisfy*

$$\Pr\left[\arg\max_j F_{\mathcal{A}}(x)_j \neq y\right] \leq \widehat{\mathcal{R}}_\gamma\left(F_{\mathcal{A}}\right) + \widetilde{\mathcal{O}}\left(\frac{\|X\|_2 R_{\mathcal{A}}}{\gamma n}\ln(W) + \sqrt{\frac{\ln(1/\delta)}{n}}\right) \quad (4)$$

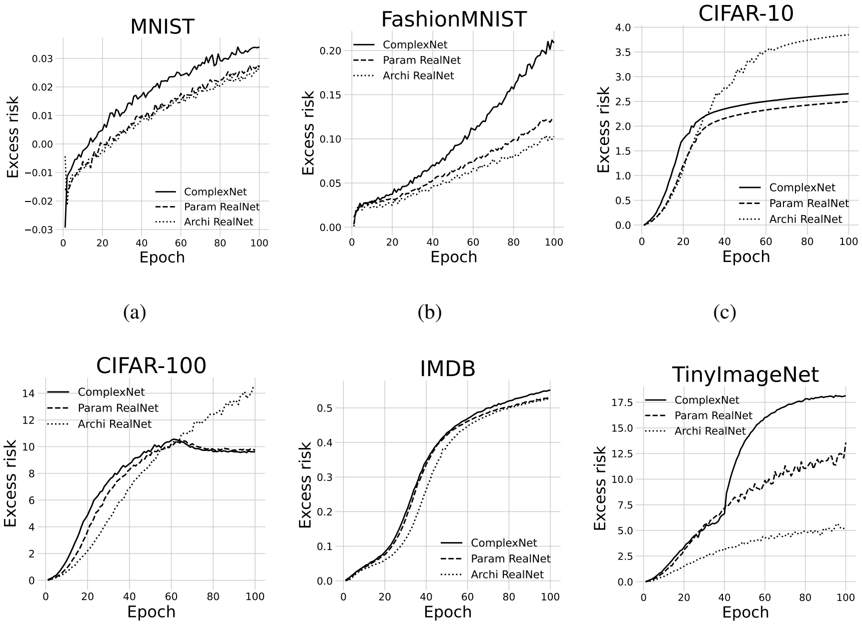
where $\widehat{\mathcal{R}}_\gamma(f) \leq n^{-1}\sum_i \mathbb{1}_{\left[f(x_i)_{y_i} \leq \gamma + \max_{j \neq y_i} f(x_i)_j\right]}$ *and* $\|X\|_2 = \sqrt{\sum_i \|x_i\|_2^2}$.

The main difference between the generalisation bounds of CVNNs and RVNNs lies in how the spectral complexity and spectral norm of the weight matrix are defined. In our theorem, we redefine the spectral norm of a complex-valued matrix, instead of reformulating the complex-valued matrix into a constrained real-valued matrix to calculate its spectral norm. By doing so, we preserve the unique structure of complex-valued matrices and avoid increasing computational costs. Reformulating a complex-valued matrix into a real-valued matrix can result in a considerable increase in the matrix dimensions, making the calculation of its spectral norm more complicated. Additionally, our contribution lies in proving that the generalisation bound is positively correlated with the defined spectral complexity, which inspired us to consider the effectiveness of spectral normalisation in CVNNs, as discussed in Section 7.

6.2. Empirical Analysis

Empirical comparisons of CVNNs and RVNNs have been conducted in various forms in the literature. For instance, Trabelsi et al. [11] controlled the number of parameters of both RVNNs and CVNNs with a trade-off on width and depth, while Nitta [9] compared the learning speed of CVNNs and RVNNs under the same architecture or with a controlled number of parameters. However, it is challenging to impose a fair constraint on CVNNs and RVNNs during comparison due to the abundance of studies on real-valued neural networks compared to the limited work on investigating the best architecture for complex-valued neural networks. Although this study presents two methods and roughly compares the generalisation ability of CVNNs and RVNNs, the results cannot be used as conclusive evidence to indicate which type of neural network is better. A fair comparison of their performances remains an open problem.

In this study, we compared the performance of CVNNs with those of different RVNNs. The first type of RVNN has the same number of parameters as CVNN, while the second type has the same architecture as CVNN. The results are presented in Figure 3.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 3: Change in excess risk with the number of epochs when complex-valued and real-valued neural networks are trained on different datasets. The solid line represents the performance of complex-valued neural networks. The dashed line represents the performance of real-valued neural networks with the same number of parameters. The dotted line represents the performance of real-valued neural networks with the same architecture.

As shown in Figure 3, it is difficult to draw conclusions when comparing the generalisation performance of complex-valued and real-valued neural networks. Controlling for the architecture, the generalisation ability of real-valued neural networks is superior to that of complex-valued neural networks. However, when controlling for the number of parameters, the performance of CVNNs is better when trained on CIFAR-10 and

Algorithm 1: SGD with complex-valued spectral regularisation

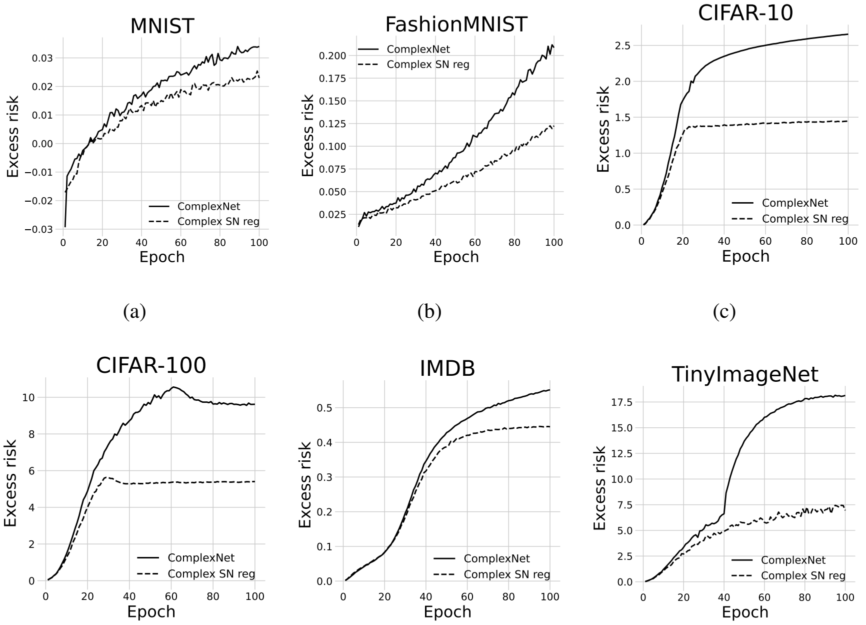
```
Input: Initialised network parameter $\boldsymbol{\theta}_0$, # training epoch $N$, spectral
       regularisation factor $\lambda$, learning rate $\alpha$
Output: Optimised network parameter $\boldsymbol{\theta}_N$

for $t = 0$ to $N - 1$ do
    Sample a minibatch $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{k}$ from the training data.
    for $l = 1$ to $L$ do
        Express the $l$-th layer weight matrix $A_l$ as $C_l + D_l i$ where $C_l$ and $D_l$
          are real-valued matrices
        Compute the layer-wise complex-valued spectral norm,
          $\|A_l\|_\sigma = \sqrt{\lambda_{\max}\left(C_l^T C_l + D_l^T D_l + \left(C_l^T D_l - C_l D_l^T\right) i\right)}$
    end
    Compute $\mathcal{L}_t = \frac{1}{k}\sum_{i=1}^{k} \ell(f_{\boldsymbol{\theta}_t}(\boldsymbol{x}_i), y_i) + \lambda \sum_{i=1}^{L} \|A_i\|_\sigma$
    Update $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \alpha \nabla_{\boldsymbol{\theta}_t} \mathcal{L}_t$
end
```

CIFAR-100.

7. Applications

In this section, we demonstrate that the **spectral regularisation algorithm** is a practical application of our theory. Because the generalisation bound of complex-valued neural networks correlates positively with the product of the spectral norms of the weight matrices, it is natural to investigate whether adding a regularisation term of the spectral norm to the loss function decreases excess risk in real-life training of complex-valued neural networks. We conducted experiments on MNIST, FashionMNIST, CIFAR-10, CIFAR-100, IMDB, and TingImageNet. The empirical results show that applying the spectral regularisation algorithm decreases excess risk significantly, fully supporting our idea. The spectral regularisation algorithm is presented in Algorithm 1. The empirical results are presented in Figure 4.



(a) (b) (c)

(d) (e) (f)

Figure 4: Plots of excess risk as a function of epoch. The right y-axis denotes the spectral norm product, and the left y-axis denotes the excess risk. The solid line denotes CVNNs without spectral regularisation, whereas the dashed line denotes the results after adding spectral regularisation.

8. Conclusions

In this study, we propose two generalisation bounds for complex-valued neural networks for i.i.d. and sequential data. Our analysis shows that the bounds scale with spectral complexity, which includes the spectral norm product of weight matrices as a factor. We also provide empirical evidence to support our theoretical findings. Our work contributes to the understanding of the generalisation ability of complex-valued neural networks and encourages further exploration of their unique properties.

Acknowledgements

Mr Shiye Lei was supported in part by Australian Research Council Projects FL170100117 and IH180100002.

References

[1] S. L. Goh, D. P. Mandic, Nonlinear adaptive prediction of complex-valued signals by complex-valued prnn, IEEE Transactions on Signal Processing 53 (2005) 1827–1836.

[2] A. Hirose, R. Eckmiller, Behavior control of coherent-type neural networks by carrier-frequency modulation, IEEE transactions on neural networks 7 (1996) 1032–1034.

[3] H. Sawada, R. Mukai, S. Araki, S. Makino, Polar coordinate based nonlinear function for frequency-domain blind source separation, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences 86 (2003) 590–596.

[4] E. K. Cole, J. Y. Cheng, J. M. Pauly, S. S. Vasanawala, Analysis of deep complex-valued convolutional neural networks for mri reconstruction, arXiv preprint arXiv:2004.01738 (2020).

[5] A. Hirose, S. Yoshida, Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence, IEEE Transactions on Neural Networks and learning systems 23 (2012) 541–551.

[6] A. Hirose, Complex-valued neural networks, volume 400, Springer Science & Business Media, 2012.

[7] T. Nitta, An extension of the back-propagation algorithm to complex numbers, Neural Networks 10 (1997) 1391–1415.

[8] T. Nitta, Orthogonality of decision boundaries in complex-valued neural networks, Neural computation 16 (2004) 73–97.

[9] T. Nitta, On the inherent property of the decision boundary in complex-valued neural networks, Neurocomputing 50 (2003) 291–303.

[10] T. Nitta, Redundancy of the parameters of the complex-valued neural network, Neurocomputing 49 (2002) 423–428.

[11] C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, C. J. Pal, Deep complex networks, arXiv preprint arXiv:1705.09792 (2017).

[12] D. P. Reichert, T. Serre, Neuronal synchrony in complex-valued deep networks, arXiv preprint arXiv:1312.6115 (2013).

[13] I. Danihelka, G. Wayne, B. Uria, N. Kalchbrenner, A. Graves, Associative long short-term memory, in: International Conference on Machine Learning, PMLR, 2016, pp. 1986–1994.

[14] M. Arjovsky, A. Shah, Y. Bengio, Unitary evolution recurrent neural networks, in: International Conference on Machine Learning, PMLR, 2016, pp. 1120–1128.

[15] S. Wisdom, T. Powers, J. Hershey, J. Le Roux, L. Atlas, Full-capacity unitary recurrent neural networks, Advances in neural information processing systems 29 (2016) 4880–4888.

[16] M. Mohri, A. Rostamizadeh, A. Talwalkar, Foundations of machine learning, MIT press, 2018.

[17] P. Bartlett, D. J. Foster, M. Telgarsky, Spectrally-normalized margin bounds for neural networks, arXiv preprint arXiv:1706.08498 (2017).

[18] N. Guberman, On complex valued convolutional neural networks, arXiv preprint arXiv:1602.09046 (2016).

[19] A. Rakhlin, K. Sridharan, A. Tewari, Sequential complexities and uniform martingale laws of large numbers, Probability Theory and Related Fields 161 (2015) 111–153.

[20] T. Zhang, Statistical analysis of some multi-category large margin classification methods, Journal of Machine Learning Research 5 (2004) 1225–1251.

[21] P.-C. Guo, A frobenius norm regularization method for convolutional kernels to avoid unstable gradient problem, arXiv preprint arXiv:1907.11235 (2019).

[22] A. Botalb, M. Moinuddin, U. Al-Saggaf, S. S. Ali, Contrasting convolutional neural network (cnn) with multi-layer perceptron (mlp) for big data analysis, in: 2018 International conference on intelligent and advanced system (ICIAS), IEEE, 2018, pp. 1–5.

[23] Y. H. Geum, A. K. Rathie, H. Kim, Matrix expression of convolution and its generalized continuous form, Symmetry 12 (2020) 1791.

[24] G. Pisier, Remarques sur un résultat non publié de b. maurey, Séminaire Analyse fonctionnelle (dit (1981) 1–12.

[25] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (1998) 2278–2324.

[26] H. Xiao, K. Rasul, R. Vollgraf, Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, arXiv preprint arXiv:1708.07747 (2017).

[27] A. Krizhevsky, G. Hinton, Learning multiple layers of features from tiny images, Technical Report, Citeseer, 2009.

[28] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, C. Potts, Learning word vectors for sentiment analysis, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, Portland, Oregon, USA, 2011, pp. 142–150. URL: http://www.aclweb.org/anthology/P11-1015.

[29] Y. Le, X. Yang, Tiny imagenet visual recognition challenge, CS 231N 7 (2015) 7.

[30] M. W. Matthès, Y. Bromberg, J. de Rosny, S. M. Popoff, Learning and avoiding disorder in multimode fibers, Phys. Rev. X 11 (2021) 021060. URL: https://link.aps.org/doi/10.1103/PhysRevX.11.021060. doi:10.1103/PhysRevX.11.021060.

Appendix A. Lipschitz Properties of Several Activation Function

In this section, our goal is to prove three types of activation functions that are widely used in complex-valued neural networks being Lipschitz continuous.

The first one is introduced in [10]:

$$\sigma_1(z) = tanh(Re(z)) + itanh(Im(z)).$$

This activation function applies the hyperbolic tangent function on both the real part and the imaginary part. Since the derivative of the hyperbolic tangent function is upper bounded by 1, hence we can see that σ_1 is 1-Lipschitz in each coordinate, if we view the real part and imaginary part as different coordinates. Then we have

$$\begin{aligned} & \left\|\sigma_1\left(z_1\right)-\sigma_1\left(z_1^{\prime}\right)\right\|_p \\ = & {\left[\left[\tanh \left(\operatorname{Re}\left(z_1\right)\right)-\tanh \left(\operatorname{Re}\left(z_1^{\prime}\right)\right)\right]^p+\left[\tanh \left(\operatorname{Im}\left(z_1\right)\right)-\tanh \left(\operatorname{Im}\left(z_1^{\prime}\right)\right)\right]^p\right]^{\frac{1}{p}} } \\ \leq & {\left[\left(\operatorname{Re}\left(z_1\right)-\operatorname{Re}\left(z_1^{\prime}\right)\right)^p+\left(\operatorname{Im}\left(z_1\right)-\operatorname{Im}\left(z_1^{\prime}\right)\right)^p\right]^{\frac{1}{p}} } \\ = & \left\|z_i-z_i^{\prime}\right\|_p . \end{aligned}$$

The first inequality holds because the hyperbolic tangent function is 1-Lipschitz.

The second type of activation function is the $\mathbb{C}ReLU$ function introduced in [11]:

$$\sigma_2(z) = ReLU(Re(z)) + iReLU(Im(z)).$$

This function also operates separately on both the real part and the imaginary part. The proof process is quite similar to the first one since the ReLU function is also 1-Lipschitz.

$$\begin{aligned} & \left\|\sigma_1\left(z_1\right)-\sigma_1\left(z_1^{\prime}\right)\right\|_p \\ = & {\left[\left[\tanh \left(\operatorname{Re}\left(z_1\right)\right)-\tanh \left(\operatorname{Re}\left(z_1^{\prime}\right)\right)\right]^p+\left[\tanh \left(\operatorname{Im}\left(z_1\right)\right)-\tanh \left(\operatorname{Im}\left(z_1^{\prime}\right)\right)\right]^p\right]^{\frac{1}{p}} } \\ \leq & {\left[\left(\operatorname{Re}\left(z_1\right)-\operatorname{Re}\left(z_1^{\prime}\right)\right)^p+\left(\operatorname{Im}\left(z_1\right)-\operatorname{Im}\left(z_1^{\prime}\right)\right)^p\right]^{\frac{1}{p}} } \\ = & \left\|z_i-z_i^{\prime}\right\|_p . \end{aligned}$$

The third type of activation function is

$$\sigma_3(z) = tanh(|z|)\exp(i\theta),$$

where $\theta = arg(z)$. If we write $z = x + yi$ and use vector notation to represent the real part and imaginary part of the operation, we will get

$$\begin{bmatrix} Re(\sigma_3(z)) \\ Im(\sigma_3(z)) \end{bmatrix} = \begin{bmatrix} tanh[(x^2+y^2)^{\frac{1}{2}}]\frac{x}{(x^2+y^2)^{\frac{1}{2}}} \\ tanh[(x^2+y^2)^{\frac{1}{2}}]\frac{y}{(x^2+y^2)^{\frac{1}{2}}} \end{bmatrix}.$$

Notice that

$$\left|tanh[(x^2+y^2)^{\frac{1}{2}}]\frac{1}{(x^2+y^2)^{\frac{1}{2}}}\right| \leq 1.$$

Hence, we have the following inequality

$$\begin{aligned}
&\left|tanh[(x_1^2+y_1^2)^{\frac{1}{2}}]\frac{x_1}{(x_1^2+y_1^2)^{\frac{1}{2}}} - tanh[(x_2^2+y_2^2)^{\frac{1}{2}}]\frac{x_2}{(x_2^2+y_2^2)^{\frac{1}{2}}}\right| \\
\leq &\left|tanh[(x_1^2+y_1^2)^{\frac{1}{2}}]\frac{x_1}{(x_1^2+y_1^2)^{\frac{1}{2}}} - tanh[(x_1^2+y_1^2)^{\frac{1}{2}}]\frac{x_2}{(x_1^2+y_1^2)^{\frac{1}{2}}}\right| \\
+ &\left|tanh[(x_1^2+y_1^2)^{\frac{1}{2}}]\frac{x_2}{(x_1^2+y_1^2)^{\frac{1}{2}}} - tanh[(x_2^2+y_2^2)^{\frac{1}{2}}]\frac{x_2}{(x_2^2+y_2^2)^{\frac{1}{2}}}\right| \\
= &\left|\frac{tanh[(x_1^2+y_1^2)^{\frac{1}{2}}]}{(x_1^2+y_1^2)^{\frac{1}{2}}}\right| |x_1-x_2| + \left|\frac{tanh[(x_1^2+y_1^2)^{\frac{1}{2}}]}{(x_1^2+y_1^2)^{\frac{1}{2}}} - \frac{tanh[(x_2^2+y_2^2)^{\frac{1}{2}}]}{(x_2^2+y_2^2)^{\frac{1}{2}}}\right| |x_2|.
\end{aligned}$$

Noted that, assume $g(x) = \frac{tanh(x)}{x}$, then through calculation, we have $|g'(x)|$ bounded by 1. Hence, $g(x)$ is 1-Lipschitz.

Therefore, we have

$$\begin{aligned}
&\left|\frac{tanh[(x_1^2+y_1^2)^{\frac{1}{2}}]}{(x_1^2+y_1^2)^{\frac{1}{2}}} - \frac{tanh[(x_2^2+y_2^2)^{\frac{1}{2}}]}{(x_2^2+y_2^2)^{\frac{1}{2}}}\right| |x_2| \\
\leq &\left|(x_1^2+y_1^2)^{\frac{1}{2}} - (x_2^2+y_2^2)^{\frac{1}{2}}\right| |x_2| \\
\leq &\left|\frac{x_1^2-x_2^2+y_1^2-y_2^2}{(x_1^2+y_1^2)^{\frac{1}{2}}+(x_2^2+y_2^2)^{\frac{1}{2}}}\right| |x_2| \\
\leq &|x_2|\left|\frac{(x_1+x_2)}{(x_1^2+y_1^2)^{\frac{1}{2}}+(x_2^2+y_2^2)^{\frac{1}{2}}}\right| |x_1-x_2| \\
+ &|x_2|\left|\frac{(y_1+y_2)}{(x_1^2+y_1^2)^{\frac{1}{2}}+(x_2^2+y_2^2)^{\frac{1}{2}}}\right| |y_1-y_2| \\
\leq &\alpha|x_1-x_2| + \alpha|y_1-y_1|
\end{aligned}$$

for some constant α such that $|x_2| \leq \alpha$.

Then, we can bound the first coordinate by

$$\begin{aligned}
&\left|tanh[(x_1^2+y_1^2)^{\frac{1}{2}}]\frac{x_1}{(x_1^2+y_1^2)^{\frac{1}{2}}} - tanh[(x_2^2+y_2^2)^{\frac{1}{2}}]\frac{x_2}{(x_2^2+y_2^2)^{\frac{1}{2}}}\right| \\
\leq &(\alpha+1)|x_1-x_2| + \alpha|y_1-y_1|.
\end{aligned}$$

Without loss of generality, the second coordinate is bounded by $\alpha|x_1 - x_2| + (\alpha + 1)|y_1 - y_1|$.

Finally, we have

$$\begin{aligned}
&\|\sigma_3(z_1) - \sigma_3(z_2)\|_p \\
\leq &\left(((\alpha+1)|x_1 - x_2| + \alpha|y_1 - y_2|)^p + (\alpha|x_1 - x_2| + (\alpha+1)|y_1 - y_2|)^p\right)^{\frac{1}{p}} \\
\leq &\left(M|x_1 - x_2|^p + M|y_1 - y_2|^p\right)^{\frac{1}{p}} = (2\alpha + 1)\,\|z_1 - z_2\|_p\,,
\end{aligned}$$

where $M = (2\alpha + 1)^p$, $z_1 = x_1 + iy_1$ and $z_2 = x_2 + iy_2$.

Hence, we have proved that $\sigma_3(z) = tanh(|z|)exp(i\theta)$ is Lipschitz continuous.

Appendix B. Proof of Theorem 1

Appendix B.1. Proof of Lemma 1

Before proving lemma 1, we first introduce Maurey's sparsification lemma [24, 17].

Lemma 5 (Maurey's sparsification lemma [24])**.** *In a Hilbert space $\mathcal{H}$ equipped with norm $||\cdot||$, consider $f \in \mathcal{H}$ such that $f = \sum_{i=1}^{n} \alpha_i g_i$ where $g_i \in \mathcal{H}$, α_i are positive real numbers, and $\alpha = \sum_{i=1}^{n} \alpha_i \neq 0$. Then, for any positive integer k, there always exist non negative integers $k_1, k_2, ..., k_n$ such that $\sum_{i=1}^{n} k_i = k$ such that*

$$\left\| f - \frac{\alpha}{k}\sum_{i=1}^{n} k_i g_i \right\|^2 \leq \frac{\alpha}{k}\sum_{i=1}^{n} \alpha_i \|g_i\|^2 \leq \frac{\alpha^2}{k}\max_i \|g_i\|^2$$

i.e.

$$\left\| \sum_{i=1}^{n} \frac{\alpha_i}{\alpha} g_i - \sum_{i=1}^{n} \frac{k_i}{k} g_i \right\|^2 \leq \sum_{i=1}^{n} \frac{\alpha_i}{k} \|g_i\|^2 \leq \frac{\alpha}{k}\max_i \|g_i\|^2 .$$

Proof. Define k i.i.d random variable $W_1, W_2, ..., W_k$ such that $P(W_1 = \alpha g_i) = \frac{\alpha_i}{\alpha}$. Let $W = \frac{\sum_{i=1}^{k} W_i}{k}$. Therefore,

$$E[W] = E[W_1] = f.$$

Hence, we have

$$
\begin{aligned}
E[\|f-W\|^2] &= \frac{1}{k^2}E[\langle\sum_{i=1}^{k}(f-W_i),\sum_{i=1}^{k}(f-W_i)\rangle] \\
&= \frac{1}{k^2}\,E[\sum_{i=1}^{k}\|f-W_i\|^2] \\
&= \frac{1}{k}\,E[\|f-W_1\|^2] \\
&= \frac{1}{k}(E[\|W_1\|^2]-\|f\|^2) \\
&\leq \frac{1}{k}\,E[\|W_1\|^2] \\
&= \sum_{i=1}^{n}\frac{\alpha_i}{k\alpha}\cdot\alpha^2\,\|g_i\|^2 \\
&\leq \frac{\alpha^2}{k}\,\max_i\|g_i\|^2\,.
\end{aligned}
$$

Since for a random variable, the minimal value it can take is at most the value of expectation, hence, there must exist a sequence of k numbers $(l_1, l_2, ..., l_k) \in \{1, 2, ..., n\}^k$, such that $W_i = \alpha g_{l_i}$, $W = \sum_{i=i}^{k} W_i$, and

$$\|W-f\|^2 \leq \frac{\alpha^2}{k}\,\max_i\|g_i\|^2\,.$$

To finish the proof, we assign integer k_i mentioned in the lemma to be

$$k_i = \sum_{j=1}^{k}\mathbb{1}_{g_{l_j}=i}.$$

□

As Bartlett et al. [17] indicated, the Maurey sparsification lemma only discussed the L_1 norm case. Zhang [20] generalized this lemma to create bounds for non-L_1 norm cases, which is also applicable in our proof of lemma 1.

Proof. (lemma 1)
Given the data matrix $Z \in \mathbb{C}^{n\times d}$, re-scaling each column of the matrix Z and get a matrix $Y \in \mathbb{C}^{n\times d}$, where

$$Y_{:;j} = Z_{:;j}/\,\|Z_{:;j}\|\,.$$

Set $N = 4dm, k = \lceil a^2 b^2 m^{2/r}/\epsilon^2 \rceil$, and $\alpha = am^{1/r}\|X\|_p$. To construct an appropriate convex hull, we define

$$\{V_1, V_2, ..., V_N\} = \left\{\sigma Y \mathbf{e}_i \mathbf{e}_j^T : \sigma \in \{-1, +1\}, i \in \{1, 2, ..., d\}, j \in \{i, 2, ..., m\}\right\}$$
$$\cup \left\{\sigma Y \mathbf{c}_i \mathbf{e}_j^T : \sigma \in \{-1, +1\}, i \in \{1, 2, ..., d\}, j \in \{i, 2, ..., m\}\right\}$$

and

$$\begin{aligned} C &= \left\{ \frac{\alpha}{k} \sum_{i=1}^{N} k_i V_i : k_i \geq 0, \sum_{i=1}^{N} k_i = k \right\} \\ &= \left\{ \frac{\alpha}{k} \sum_{m=1}^{k} V_{l_m} : (l_1, \ldots, l_m) \in [N]^k \right\}, \end{aligned}$$

where $k_i \triangleq \sum_{m=1}^{k} \mathbb{1}_{l_m = i}$.

Here, $\mathbf{e}_i$ defines the d-dimensional standard vector, $\mathbf{e}_j$ defines the m-dimensional standard vector, and $\mathbf{c}_i$ defines the d-dimensional vector in which only the ith entry equals $\sqrt{-1}$, and other entries equal 0.

Because of the way V_i defined and $p \leq 2$, we have

$$\max_i \|V_i\|_2 \leq \max_i \{\|Y\mathbf{e}_i\|_2, \|Y\mathbf{c}_i\|_2\} = \max_i \{\|Y\mathbf{e}_i\|_2\} = \max_i \frac{\|X\mathbf{e}_i\|_2}{\|X\mathbf{e}_i\|_p} \leq 1.$$

The first equality is due to the definition of complex-valued vector norms, and the second equality holds because of the monotonicity of matrix norm in terms of p.

Next, it suffices to prove that C is a cover of $\left\{ZA : A \in \mathbb{C}^{d \times m}, \|A\|_{q,s} \leq a\right\}$. To prove this, we desire to bound $\left\|ZA - \frac{\alpha}{k}\sum_{i=1}^{N} k_i V_i\right\|_2$ by ϵ for some $(k_1, ..., k_N)$.

Define $M \in \mathbb{R}^{d \times m}$ where the element of each row j equals $\|Z_{:,j}\|_p$, hence we have

$$ZA = Y(M \odot A),$$

where $\odot$ represents the Hadamard product.

$$\begin{aligned}\|M\|_{p,r} &= \left\|\left(\left\|\left(\|Z_{:,1}\|_p, \ldots, \|Z_{:,d}\|_p\right)\right\|_p, \ldots, \left\|\left(\|Z_{:,1}\|_p, \ldots, \|Z_{:,d}\|_p\right)\right\|_p\right)\right\|_r \\ &= m^{1/r}\left\|\left(\|Z_{:,1}\|_p, \ldots, \|Z_{:,d}\|_p\right)\right\|_p = m^{1/r}\left(\sum_{j=1}^{d}\|Z_{:,j}\|_p^p\right)^{1/p} \\ &= m^{1/r}\left(\sum_{j=1}^{d}\sum_{i=1}^{n} Z_{i,j}^p\right)^{1/p} \\ &= m^{1/r}\|Z\|_p.\end{aligned}$$

Hence, if we denote $S = M \odot A$, we have

$$\|S\|_1 \leq \langle M, |A|\rangle \leq \|M\|_{p,r}\|A\|_{q,s} \leq m^{1/r}\|Z\|_p a = \alpha.$$

We can see that ZA indeed lies in a convex hull related with $\{V_1, V_2, ..., V_N\}$:

$$\begin{aligned}ZA &= YM \\ &= Y\sum_{i=1}^{d}\sum_{j=1}^{m}\left(\operatorname{Re}(M_{i,j})\,\mathbf{e}_i\mathbf{e}_j^\top + \operatorname{Im}(M_{i,j})\,\mathbf{c}_i\mathbf{e}_j^T\right) \\ &= \|M\|_1\sum_{i=1}^{d}\sum_{j=1}^{m}\left(\frac{\operatorname{Re}(M_{ij})}{\|M\|_1}\left(Y\mathbf{e}_i\mathbf{e}_j^\top\right) + \frac{\operatorname{Im}(M_{ij})}{\|M\|_1}\left(Y\mathbf{c}_i\mathbf{e}_j^\top\right)\right) \\ &\in \alpha \cdot \operatorname{conv}\left(\{V_1, \ldots, V_N\}\right).\end{aligned}$$

where $\operatorname{conv}\left(\{V_1, \ldots, V_N\}\right)$ denotes the convex hull formed by $\{V_1, V_2, ..., V_N\}$.

Finally, by Lemma 5, there exist non-negative integers $(k_1, k_2, ..., k_N)$ such that

$$\begin{aligned}\left\|ZA - \frac{\alpha}{k}\sum_{i=1}^{N} k_i V_i\right\|_2^2 = \left\|YM - \frac{\alpha}{k}\sum_{i=1}^{N} k_i V_i\right\|_2^2 &\leq \frac{\alpha^2}{k}\max_i \|V_i\|_2 \\ &\leq \frac{a^2 m^{2/r}\|Z\|_p^2}{k} \\ &\leq \epsilon^2.\end{aligned}$$

Hence, C is a covering of the desire set. Since the cardinality of set C equals N^k, we have the target inequality:

$$\ln \mathcal{N}\left(\left\{ZA : A \in \mathbb{C}^{d\times m}, \|A\|_{q,s} \leq a\right\}, \epsilon, \|\cdot\|_2\right) \leq \left\lceil \frac{a^2 b^2 m^{2/r}}{\epsilon^2} \right\rceil \ln(4dm).$$

□

Appendix B.2. Proof of Lemma 2

As stated in the third section, this lemma shall be proved by mathematical induction. The basic idea is as follows. Denotes Z_i to be the data set passing from the i-1th layer to the ith layer ($Z_0 = Z^T$). According to Lemma 1, assume that fixed specific layer i, there exists a sequence of covering matrices $(\widehat{A}_0, \widehat{A}_1, ..., \widehat{A}_{i-1})$ for i-1 previous layers, and a covering matrix $\widehat{A}_i$ such that $\left\|A_i\widehat{Z_i} - \widehat{A}_i\widehat{Z}_i\right\|_2 \leq \epsilon$ for some $\epsilon > 0$. As a consequence, the input data for the i+1th layer shall be $Z_{i+1} = \sigma_{i+1}(A_i Z_i)$, and $\widehat{Z}_{i+1} = \sigma_{i+1}(\widehat{A}_i\widehat{Z}_i)$.

$$\begin{aligned}
\left\|Z_{i+1} - \widehat{Z}_{i+1}\right\|_2 &\leq \rho_i \left\|A_i Z_i - \widehat{A}_i\widehat{Z}_i\right\|_2 \\
&\leq \rho_i \left(\left\|A_i Z_i - A_i\widehat{Z}_i\right\|_2 + \left\|A_i\widehat{Z}_i - \widehat{A}_i\widehat{Z}_i\right\|_2\right) \\
&\leq \rho_i \left\|A_i\right\|_\sigma \left\|Z_i - \widehat{Z}_i\right\|_2 + \rho_i\epsilon_i.
\end{aligned}$$

Since the first term of the right-hand side part depends on the inductive hypothesis, hence intuitively, we can see that the covering number upper bound depends on the product of spectral norms of all covering matrices. The detailed proof is illustrated as follows.

We first define two sequences of vector space $\{V_1, V_2, ..., V_L\}$, and $\{W_2, W_3, ..., W_{L+1}\}$. The first sequence of vector spaces are equipped with $\|\cdot\|_V$, and the second sequences are equipped with $\|\cdot\|_W$. For each layer's input matrix, $Z_i \in V_i$, and the first layer's input $Z \in V_1$ have the constraint:$\|Z\|_V \leq B$.

Moreover, under our assumptions, A_i can be viewed as a linear operator: $V_i \to W_{i+1}$, and the norm of each linear operation is defined as:

$$\|A_i\|_{V\to W} = \sup_{|Z|_V \leq 1} \|A_i Z\|_W = c_i.$$

σ_i can be treated as a mapping from $W_i \to V_i$, and the ρ_i-lipschitz property means

$$\|\sigma_i(z) - \sigma_i(z')\|_V \leq \rho_i \|z - z'\|_W .$$

With these preparations, we claim the following lemma which is based on a similar lemma raised by Bartlett et al. [17].

Lemma 6 ([17]). *Assume that a sequence of positive numbers* $(\epsilon_1, \ldots, \epsilon_L)$*, along with Lipschitz non-linear mappings* $(\sigma_1, \ldots, \sigma_L)$ *(where* σ_i *is* ρ_i *- Lipschitz), and linear operator norm bounds* $(c_1, \ldots, c_L)$ *as described above are given. Suppose the sequence of matrices* $\mathcal{A} = (A_1, \ldots, A_L)$ *lies within* $\mathcal{B}_1 \times \cdots \times \mathcal{B}_L$ *where* $\mathcal{B}_i$ *are classes satisfying the property that each* $A_i \in \mathcal{B}_i$ *has* $\|A_i\|_{V \to W} \leq c_i$*. Let data* Z *be given with* $\|Z\|_V \leq B$*. Then, define* $\tau := \sum_{j \leq L} \epsilon_j \rho_j \prod_{l=j+1}^{L} \rho_l c_l$*, the complex-valued neural network images* $\mathcal{F} := \{F_{\mathcal{A}}(Z) : \mathcal{A} \in \mathcal{B}_1 \times \cdots \times \mathcal{B}_L, \|Z\|_V \leq B\}$ *has the following covering number bound*

$$\mathcal{N}\left(\mathcal{F}, \tau, \|\cdot\|_V\right) \leq \prod_{i=1}^{L} \sup_{\substack{(A_1, \ldots, A_{i-1}) \\ \forall j < i. A_j \in \mathcal{B}_j}} \mathcal{N}\left(\left\{A_i F_{(A_1, \ldots, A_{i-1})}(Z) : A_i \in \mathcal{B}_i\right\}, \epsilon_i, \|\cdot\|_W\right).$$

Proof. The lemma is proved by Mathematical induction.

A sequence of covering set $\{\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_L\}$ is constructed where $\mathcal{F}_i$ covers W_i.

Base case: When i=1, we have $\mathcal{F}_1$ to be constructed according to Lemma 1, and

$$|\mathcal{F}_1| \leq \mathcal{N}\left(\{A_1 Z : A_1 \in \mathcal{B}_1\}, \epsilon_1, \|\cdot\|_W\right) =: N_1.$$

Inductive Hypothesis: Assume that for i=n, we can find a ϵ_n-covering $\mathcal{F}_n$ for set $\left\{A_n \mathcal{F}_{\mathcal{A}_1, \ldots, \mathcal{A}_{n-1}}(Z) : A_n \in \mathcal{B}_n\right\}$ such that:

$$|\mathcal{F}_n| \leq \prod_{l=1}^{n} N_l.$$

Induction Step: For every element $F \in \mathcal{F}_n$, construct an ϵ_{n+1}-cover $\mathcal{G}_{n+1}(F)$ of

$$\{A_{n+1} \sigma_n(F) : A_{n+1} \in \mathcal{B}_{n+1}\}.$$

Since these covers are proper, meaning $F = A_{n+1} F_{(A_1, \ldots, A_n)}(Z)$ for some matrices $(A_1, \ldots, A_n) \in \mathcal{B}_1 \times \cdots \times \mathcal{B}_n$, it follows that

$$\begin{aligned} |\mathcal{G}_{n+1}(F)| \leq & \sup_{\substack{(A_1, \ldots, A_n) \\ \forall j \leq i. A_j \in \mathcal{B}_j}} \mathcal{N}\left(\{A_{n+1} F_{A_1, \ldots, A_n}(Z) : A_{n+1} \in \mathcal{B}_{n+1}\}, \epsilon_{n+1}, \|\cdot\|_W\right) \\ & =: N_{n+1}. \end{aligned}$$

Lastly, we can form the cover

$$\mathcal{F}_{n+1} := \bigcup_{F \in \mathcal{F}_n} \mathcal{G}_{n+1}(F),$$

whose cardinality satisfies

$$|\mathcal{F}_{n+1}| \leq |\mathcal{F}_n| \cdot N_{n+1} \leq \prod_{l=1}^{n+1} N_l.$$

Define $\mathcal{H} := \{\sigma_L(F) : F \in \mathcal{F}_L\}$. It's trivial to see that the cardinality of $\mathcal{H}$ is the same as $\mathcal{F}_L$. Then, it suffices to show that $\mathcal{H}$ is indeed a covering of $\mathcal{F}$. If we fix any $(A_1, \ldots, A_L)$ satisfying the constraints, then recursively, we denote

$$F_1 = A_1 Z \in W_2, \quad G_i = \sigma_i\left(F_i\right) \in V_{i+1} \quad F_{i+1} = A_{i+1} G_i \in W_{i+2}.$$

In other words, we need to prove that there exist $\widehat{G}_L \in \mathcal{H}$ such that $\left\|G_L - \widehat{G}_L\right\|_V \leq \tau$.

Base case: Set $\widehat{G}_0 = Z$.

Inductive hypothesis: Choose $\widehat{F}_i \in \mathcal{F}_i$ with $\left\|A_i \widehat{G}_{i-1} - \widehat{F}_i\right\|_W \leq \epsilon_i$, and set $\widehat{G}_i := \sigma_i\left(\widehat{F}_i\right)$.

Induction Step:

$$\begin{aligned}
\left\|G_{i+1} - \widehat{G}_{i+1}\right\|_V &\leq \rho_{i+1} \left\|F_{i+1} - \widehat{F}_{i+1}\right\|_W \\
&\leq \rho_{i+1} \left\|F_{i+1} - A_{i+1}\widehat{G}_i\right\|_W + \rho_{i+1} \left\|A_{i+1}\widehat{G}_i - \widehat{F}_{i+1} \mid\right\|_W \\
&\leq \rho_{i+1} \|A_{i+1}\|_{V \to W} \left\|G_i - \widehat{G}_i\right\|_V + \rho_{i+1}\epsilon_{i+1} \\
&\leq \rho_{i+1} c_{i+1} \left(\sum_{j \leq i} \epsilon_j \rho_j \prod_{l=j+1}^{i} \rho_l c_l\right) + \rho_{i+1}\epsilon_{i+1} \\
&= \sum_{j \leq i+1} \epsilon_j \rho_j \prod_{l=j+1}^{i+1} \rho_l c_l \\
&= \gamma.
\end{aligned}$$

Hence, we get proved. □

To prove Lemma 2, the key idea is to apply the result of Lemma 1 and Lemma 6.

Proof. (Lemma 2)

To begin with, we assume the same setting as above. However, to prove Lemma 2, $\|\cdot\|_V = \|\cdot\|_W = \|\cdot\|_2$, and the operator norm is set to the spectral norm, i.e. $\|A_i\|_{V\to W} = \|A_i\|_\sigma$. Also, the sequence of number $\{\epsilon_1, \epsilon_2, ..., \epsilon_L\}$ are defined as

$$\epsilon_i := \frac{\alpha_i \epsilon}{\rho_i \prod_{j>i} \rho_j s_j} \quad \text{where} \quad \alpha_i := \frac{1}{\bar{\alpha}} \left(\frac{b_i}{s_i}\right)^{2/3}, \quad \bar{\alpha} := \sum_{j=1}^{L} \left(\frac{b_j}{s_j}\right)^{2/3}.$$

By this setting, we find that the γ defined in Lemma 6 satisfies

$$\tau \leq \sum_{j\leq L} \epsilon_j \rho_j \prod_{l=j+1}^{L} \rho_l s_l = \sum_{j\leq L} \alpha_j \epsilon = \epsilon.$$

Then

$$\begin{aligned}
&\ln \mathcal{N}\left(\mathcal{F}_{|S}, \epsilon, \|\cdot\|_2\right) \\
&\leq \sum_{i=1}^{L} \sup_{\substack{(A_1,\ldots,A_{i-1}) \\ \forall j<i.A_j\in\mathcal{B}_j}} \ln \mathcal{N}\left(\left\{A_i F_{(A_1,\ldots,A_{i-1})}\left(Z^\top\right) : A_i \in \mathcal{B}_i\right\}, \epsilon_i, \|\cdot\|_2\right) \\
&\leq \sum_{i=1}^{L} \sup_{\substack{(A_1,\ldots,A_{i-1}) \\ \forall j<i,A_j\in\mathcal{B}_j}} \ln \mathcal{N}\left(\left\{F_{(A_1,\ldots,A_{i-1})}\left(Z^\top\right)^\top (A_i)^\top : \left\|A_i^\top\right\|_{2,1} \leq b_i\right\}, \epsilon_i, \|\cdot\|_2\right) \\
&\leq \sum_{i=1}^{L} \sup_{\substack{(A_1,\ldots,A_{i-1}) \\ \forall j<i.A_j\in\mathcal{B}_j}} \frac{b_i^2 \left\|F_{(A_1,\ldots,A_{i-1})}\left(Z^\top\right)^\top\right\|_2^2}{\epsilon_i^2} \ln\left(4W^2\right).
\end{aligned}$$

The first equality holds because we use L_2 norms here. Hence the covering number for a matrix and its transpose are the same. To further simplify the formula, we can upper bound $\left\|F_{(A_1,\ldots,A_{i-1})}\left(Z^\top\right)^\top\right\|_2^2$ by

$$\begin{aligned}
\left\|F_{(A_1,\ldots,A_{i-1})}\left(Z^\top\right)^\top\right\|_2 &= \left\|F_{(A_1,\ldots,A_{i-1})}\left(Z^\top\right)\right\|_2 \\
&= \left\|\sigma_{i-1}\left(A_{i-1}F_{(A_1,\ldots,A_{i-2})}\left(Z^\top\right) - \sigma_{i-1}(0)\right\|_2 \\
&\leq \rho_{i-1}\left\|A_{i-1}F_{(A_1,\ldots,A_{i-2})}\left(X^\top\right) - 0\right\|_2 \\
&\leq \rho_{i-1}\left\|A_{i-1}\right\|_\sigma \left\|F_{(A_1,\ldots,A_{i-2})}\left(Z^\top\right)\right\|_2.
\end{aligned}$$

Inductively, we have

$$\max_j \left\|F_{(A_1,\ldots,A_{i-1})}\left(Z^\top\right)^\top \mathbf{e}_j\right\|_2 \leq \|Z\|_2 \prod_{j=1}^{i-1} \rho_j \left\|A_j\right\|_\sigma.$$

Finally, we obtain

$$\begin{aligned}
\ln \mathcal{N}\left(\mathcal{F}_{|S}, \epsilon, \|\cdot\|_2\right) &\leq \sum_{i=1}^{L} \sup_{\substack{(A_1,\ldots,A_{i-1}) \\ \forall j<i.\, A_j \in \mathcal{B}_j}} \frac{b_i^2 \|Z\|_2^2 \prod_{j<i} \rho_j^2 \left\|A_j\right\|_\sigma^2}{\epsilon_i^2} \ln\left(4W^2\right) \\
&\leq \sum_{i=1}^{L} \frac{b_i^2 B^2 \prod_{j<i} \rho_j^2 s_j^2}{\epsilon_i^2} \ln\left(4W^2\right) \\
&= \frac{B^2 \ln\left(4W^2\right) \prod_{j=1}^{L} \rho_j^2 s_j^2}{\epsilon^2} \sum_{i=1}^{L} \frac{b_i^2}{\alpha_i^2 s_i^2} \\
&= \frac{B^2 \ln\left(4W^2\right) \prod_{j=1}^{L} \rho_j^2 s_j^2}{\epsilon^2} \left(\bar{\alpha}^3\right).
\end{aligned}$$

□

Appendix B.3. Proof of Theorem 1

As stated in the third section, the main theorem we used to prove Theorem 1 is the Dudley Entropy Integral. The standard Dudley Entropy Integral introduces a method to obtain Rademacher complexity bound via covering number [16].

Theorem 5 ([16])**.** *Let $\mathcal{F}$ be a real-valued function class taking values in $[0,1]$, and assume that $\mathbf{0} \in \mathcal{F}$. Then*

$$\mathfrak{R}\left(\mathcal{F}_{|S}\right) \leq \inf_{\alpha>0}\left(\frac{4\alpha}{\sqrt{n}} + \frac{12}{n}\int_{\alpha}^{\sqrt{n}} \sqrt{\log \mathcal{N}\left(\mathcal{F}_{|S}, \varepsilon, \|\cdot\|_2\right)} d\varepsilon.\right)$$

Proof. [17] Let $N \in \mathbb{N}$ be arbitrary and let $\varepsilon_i = \sqrt{n} 2^{-(i-1)}$ for each $i \in [N]$. For each i, let V_i denote the cover achieving $\mathcal{N}\left(\mathcal{F}_{|S}, \varepsilon_i, \|\cdot\|_2\right)$, so that

$$\forall f \in \mathcal{F} \quad \exists v \in V_i \quad \left(\sum_{t=1}^{n} \left(f\left(x_t\right) - v_t\right)^2\right)^{1/2} \leq \varepsilon_i,$$

and $|V_i| = \mathcal{N}\left(\mathcal{F}_{|S}, \varepsilon_i, \|\cdot\|_2\right)$. For a fixed $f \in \mathcal{F}$, let $v^i[f]$ denote the nearest element in V_i. Then

$$\begin{aligned}
&\mathbb{E}\left(\sup_{f\in\mathcal{F}}\sum_{t=1}^{n}\varepsilon_i f\left(x_t\right)\right)\\
&=\mathbb{E}\left(\sup_{f\in\mathcal{F}}\left[\sum_{t=1}^{n}\epsilon_t\left(f\left(x_t\right)-v_t^N[f]\right)+\sum_{i=1}^{N-1}\sum_{t=1}^{n}\epsilon_t\left(v_t^i[f]-v_t^{i+1}[f]\right)-\sum_{t=1}^{n}\epsilon_t v_t^1[f]\right]\right)\\
&\leq\mathbb{E}\left(\sup_{f\in\mathcal{F}}\left[\sum_{t=1}^{n}\epsilon_t\left(f\left(x_t\right)-v_t^N[f]\right)\right]\right)+\sum_{i=1}^{N-1}\mathbb{E}\left(\sup_{f\in\mathcal{F}}\left[\sum_{t=1}^{n}\epsilon_t\left(v_t^i[f]-v_t^{i+1}[f]\right)\right]\right)\\
&+\mathbb{E}\left(\sup_{f\in\mathcal{F}}\left[\sum_{t=1}^{n}\epsilon_t v_t^1[f]\right]\right).
\end{aligned}$$

For the third term, observe that it suffices to take $V_1 = \{\mathbf{0}\}$, which implies

$$\mathbb{E}\sup_{\epsilon}\left[\sum_{f\in\mathcal{F}}\left[\sum_{t=1}^{n}\epsilon_t v_t^1[f]\right]=0\right].$$

The first term may be handled using Cauchy-Schwarz as follows:

$$\begin{aligned}
&\mathbb{E}\sup_{\epsilon}\left[\sum_{f\in\mathcal{F}}\left[\sum_{t=1}^{n}\epsilon_t\left(f\left(x_t\right)-v_t^N[f]\right)\right]\right]\\
\leq&\sqrt{\mathbb{E}\sum_{t=1}^{n}\left(\epsilon_t\right)^2}\sqrt{\sup_{f\in\mathcal{F}}\sum_{t=1}^{n}\left(f\left(x_t\right)-v_t^N[f]\right)^2}\\
\leq&\sqrt{n}\varepsilon_N.
\end{aligned}$$

Last to take care of are the terms of the form

$$\mathbb{E}\sup_{\epsilon}\left[\sum_{t=1}^{n}\epsilon_t\left(v_t^i[f]-v_t^{i+1}[f]\right)\right].$$

For each i, let $W_i = \left\{v^i[f]-v^{i+1}[f] \mid f\in\mathcal{F}\right\}$. Then $|W_i| \leq |V_i|\,|V_{i+1}| \leq |V_{i+1}|^2$,

$$\mathbb{E}\sup_{\epsilon}\left[\sum_{f\in\mathcal{F}}\left[\sum_{t=1}^{n}\epsilon_t\left(v_t^i[f]-v_t^{i+1}[f]\right)\right]\right]\leq\mathbb{E}\sup_{w\in W_i}\left[\sum_{t=1}^{n}\epsilon_t w_t\right],$$

and furthermore

$$
\begin{aligned}
\sup_{w \in W_i} \sqrt{\sum_{t=1}^{n} w_t^2} &= \sup_{f \in \mathcal{F}} \left\| v^i[f] - v^{i+1}[f] \right\|_2 \\
&\leq \sup_{f \in \mathcal{F}} \left\| v^i[f] - \left(f\left(x_1\right), \ldots, f\left(x_n\right)\right) \right\|_2 \\
&+ \sup_{f \in \mathcal{F}} \left\| \left(f\left(x_1\right), \ldots, f\left(x_n\right)\right) - v^{i+1}[f] \right\|_2 \\
&\leq \varepsilon_i + \varepsilon_{i+1} \\
&= 3\varepsilon_{i+1}.
\end{aligned}
$$

With this observation, the standard Massart finite class lemma [16] implies

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{\epsilon} \sup_{w \in W_i} \left[\sum_{t=1}^{n} \epsilon_t w_t \right] \\
\leq &\sqrt{2 \sup_{w \in W_i} \sum_{t=1}^{n} \left(w_t\right)^2 \log |W_i|} \leq 3\sqrt{2 \log |W_i|}\varepsilon_{i+1} \leq 6\sqrt{\log |V_{i+1}|}\varepsilon_{i+1}.
\end{aligned}
$$

Collecting all terms, establishes

$$
\begin{aligned}
\mathbb{E}_{\epsilon} \sup_{f \in \mathcal{F}} \sum_{t=1}^{n} \epsilon_t f\left(x_t\right) &\leq \varepsilon_N \sqrt{n} + 6 \sum_{i=1}^{N-1} \varepsilon_{i+1} \sqrt{\log \mathcal{N}\left(\mathcal{F}_{|S}, \varepsilon_{i+1}, \|\cdot\|_2\right)} \\
&\leq \varepsilon_N \sqrt{n} + 12 \sum_{i=1}^{N} \left(\varepsilon_i - \varepsilon_{i+1}\right) \sqrt{\log \mathcal{N}\left(\mathcal{F}_{|S}, \varepsilon_i, \|\cdot\|_2\right)} \\
&\leq \varepsilon_N \sqrt{n} + 12 \int_{\varepsilon_{N+1}}^{\sqrt{n}} \sqrt{\log \mathcal{N}\left(\mathcal{F}_{|S}, \varepsilon, \|\cdot\|_2\right)} d\varepsilon.
\end{aligned}
$$

Finally, select any $\alpha > 0$ and take N be the largest integer with $\varepsilon_{N+1} > \alpha$. Then $\varepsilon_N = 4\varepsilon_{N+2} < 4\alpha$, and so

$$
\begin{aligned}
&\varepsilon_N \sqrt{n} + 12 \int_{\varepsilon_{N+1}}^{\sqrt{n}} \sqrt{\log \mathcal{N}\left(\mathcal{F}_{|S}, \varepsilon, \|\cdot\|_2\right)} d\varepsilon \\
\leq &4\alpha\sqrt{n} + 12 \int_{\alpha}^{\sqrt{n}} \sqrt{\log \mathcal{N}\left(\mathcal{F}_{|S}, \varepsilon, \|\cdot\|_2\right)} d\varepsilon.
\end{aligned}
$$

□

However, it's worth noticing that $\mathcal{N}(\mathcal{F}, \epsilon, \|\cdot\|_2)$ can not be directly used in Theorem 3 to obtain the upper bound of $\hat{\mathfrak{R}}_S(\mathcal{G})$. Hence we raise Lemma 3 and Lemma 4 to make it applicable. We shall first prove these two lemmas.

Proof. (Lemma 3) Consider $\mathcal{H} \subset \mathcal{F}$ is a cover of family $\mathcal{F}$ which satisfies that the cardinality of $\mathcal{H}$ equals the covering number of $\mathcal{F}$. Then for any $\mathcal{F}_{\mathcal{A}} \in \mathcal{F}$, we have a corresponding $h \in \mathcal{H}$ such that

$$\left\|\mathcal{F}_{\mathcal{A}}(Z) - h(Z)\right\|_2 \leq \epsilon.$$

Then consider $\left\|\mathcal{F}_{\mathcal{A}}(Z) - Y\right\|_2 \in \mathcal{G}$, we have

$$\begin{aligned} \left|\left\|\mathcal{F}_{\mathcal{A}}(Z) - Y\right\|_2 - \left\|h(Z) - Y\right\|_2\right| &\leq \left|\left\|\mathcal{F}_{\mathcal{A}}(Z) - Y - h(Z) + Y\right\|_2\right| \\ &= \left|\left\|\mathcal{F}_{\mathcal{A}}(Z) - h(Z)\right\|_2\right| \\ &\leq \epsilon. \end{aligned}$$

Therefore, it's trivial that $\bar{\mathcal{H}} = \{\left\|h(Z) - Y\right\|_2 : h \in \mathcal{H}\}$ is a cover of $\mathcal{G}$, and the cardinality of $\mathcal{H}$ equals that of $\bar{\mathcal{H}}$.

Hence, the covering number of $\mathcal{G}$ is less than the covering number of $\mathcal{F}$. □

Proof. (Lemma 4) Consider $\mathcal{H} \subset \mathcal{G}$ is a cover of family $\mathcal{G}$ which satisfies that the cardinality of $\mathcal{H}$ equals the covering number of $\mathcal{G}$. For any $g \in \mathcal{G}$, there exist $h \in \mathcal{H}$ such that

$$\left\|\alpha g - \alpha h\right\|_2 = \alpha \left\|g - h\right\|_2 \leq \alpha\epsilon.$$

Therefore, $\alpha\mathcal{H}$ is a cover of $\alpha\mathcal{G}$.

Vice versa, if $\alpha\mathcal{H}$ is a cover of $\alpha\mathcal{G}$, then $\mathcal{H}$ is a cover of $\mathcal{G}$.

Hence, Lemma 4 is get proved. □

After all preparations have been done, the proof of Theorem 1 is given as follows:

Proof. (Theorem 1)

Consider family $\mathcal{F} = \left\{F_{\mathcal{A}}(z) : \mathcal{A} = (A_1, \ldots, A_L), \|A_i\|_\sigma \leq s_i, \left\|A_i^\top\right\|_{2,1} \leq b_i\right\}$, and family

$$\mathcal{G} = \left\{(z, y) \mapsto l\left(F_{\mathcal{A}}(z), y\right) : F_{\mathcal{A}} \in \mathcal{F}\right\}.$$

As a consequence of Lemma 3,

$$\mathcal{N}\left(\mathcal{F}_{|S}, \epsilon, \|\cdot\|_2\right) \geq \mathcal{N}\left(\mathcal{G}_{|S}, \epsilon, \|\cdot\|_2\right)$$

when the loss function is set to be $l(\mathcal{F}_{\mathcal{A}}(z), y) = \|\mathcal{F}_{\mathcal{A}}(z) - y\|_2$. Since in the standard Dudley Entropy Integral, it requires the value of loss function to be always located in the interval $[0, 1]$, and we make the assumption that $l(\mathcal{F}_{\mathcal{A}}(z), y) \leq M$ always holds for the given data set, hence, we can rescale the loss function by $\frac{1}{M}$.

Define $\bar{\mathcal{G}} = \left\{(z, y) \mapsto \frac{1}{M} l\left(F_{\mathcal{A}}(z), y\right) : F_{\mathcal{A}} \in \mathcal{F}\right\}$, then Lemma 4 indicates

$$\mathcal{N}\left(\mathcal{G}, M\epsilon, \|\cdot\|_2\right) = \mathcal{N}\left(\bar{\mathcal{G}}, \epsilon, \|\cdot\|_2\right).$$

Therefore

$$\begin{aligned}
&\mathcal{N}\left(\bar{\mathcal{G}}_{|S}, \epsilon, \|\cdot\|_2\right) \leq \mathcal{N}\left(\mathcal{F}_{|S}, M\epsilon, \|\cdot\|_2\right), \\
&\ln \mathcal{N}\left(\bar{\mathcal{G}}_{|S}, \epsilon, \|\cdot\|_2\right) \leq \ln \mathcal{N}\left(\mathcal{F}_{|S}, M\epsilon, \|\cdot\|_2\right) \\
&\leq \frac{\|Z\|_2^2 \ln\left(4W^2\right)}{M^2\epsilon^2} \left(\prod_{j=1}^{L} s_j^2 \rho_j^2\right) \left(\sum_{i=1}^{L} \left(\frac{b_i}{s_i}\right)^{2/3}\right)^3.
\end{aligned}$$

If we denote $R = \|Z\|_2^2 \ln\left(4W^2\right) \left(\prod_{j=1}^{L} s_j^2 \rho_j^2\right) \left(\sum_{i=1}^{L} \left(\frac{b_i}{s_i}\right)^{2/3}\right)^3$, then we have $\ln \mathcal{N}\left(\bar{\mathcal{G}}_{|S}, \epsilon, \|\cdot\|_2\right) \leq \frac{R}{M^2\epsilon^2}$.

As stated in Theorem 3,

$$\begin{aligned}
\hat{\mathfrak{R}}_S(\bar{\mathcal{G}}) &= \underset{\sigma}{E}[\sup_{\bar{g} \in \bar{\mathcal{G}}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i \bar{g}(z_i)] \\
&= \underset{\sigma}{E}[\sup_{g \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i \frac{1}{M} g(z_i)] \\
&= \frac{1}{M} \hat{\mathfrak{R}}_S(\mathcal{G}) \\
&\leq \inf_{\alpha > 0} \left(\frac{4\alpha}{\sqrt{n}} + \frac{12}{n} \int_{\alpha}^{\sqrt{n}} \sqrt{\ln \mathcal{N}\left(\bar{\mathcal{G}}_{|S}, \varepsilon, \|\cdot\|_2\right)} d\varepsilon\right) \\
&\leq \inf_{\alpha > 0} \left(\frac{4\alpha}{\sqrt{n}} + \frac{12}{n} \int_{\alpha}^{\sqrt{n}} \sqrt{\frac{R}{M^2\epsilon^2}} d\varepsilon\right) \\
&= \inf_{\alpha > 0} \left(\frac{4\alpha}{\sqrt{n}} + \frac{12\sqrt{R}}{Mn} \ln \frac{\sqrt{n}}{\alpha}\right).
\end{aligned}$$

To make the upper bound neater, we make a simple choice at $\alpha = \frac{1}{n}$, hence,

$$\hat{\mathfrak{R}}_S(\mathcal{G}) \leq \frac{4M}{n^{3/2}} + \frac{18\,\|Z\|_2 \sqrt{2\ln(2W)} \ln n R_{\mathcal{A}}}{n}.$$

Plugging this upper bound into Theorem 2, the desired result can be obtained.

□

Appendix C. PAC Learnability of Complex-valued Neural Networks

In this section, we desire to present the proof which shows that complex-valued neural networks are PAC-learnable.

We denote f_S to be the empirical error minimizer, i.e., $f_S = \underset{f \in \mathcal{F}}{arg \min} \frac{1}{n} \sum\limits_{i=1}^{n} l\left(f(z_i), y_i\right)$. Similarly, f is the expected error minimizer: $f = \underset{f \in \mathcal{F}}{arg \min} \mathbb{E}\left[\frac{1}{n} \sum\limits_{i=1}^{n} l\left(f(z_i), y_i\right)\right]$. $R(f)$ and $\widehat{R}(f)$ respectively represents the expected error and the empirical error.

The concept of PAC-learnable is defined as follows.

Definition 1. *(PAC-learnable) Let $\mathcal{F}$ be a hypothesis set. $\mathcal{A}$ is a PAC-learnable algorithm if there exists a polynomial function poly $(\cdot,\cdot,\cdot,\cdot)$ such that for any $\epsilon > 0$ and $\delta > 0$, for all distributions $\mathcal{D}$ over Z, the following holds for any sample size $m \geq \operatorname{poly}(1/\epsilon, 1/\delta, n,$ size $(c))$:*

$$\underset{S \sim \mathcal{D}^m}{\mathbb{P}}\left[R\left(f_S\right) - R(f) \leq \epsilon\right] \geq 1 - \delta.$$

Here f and f_S are defined above.

Corollary 1. *Define the loss function to be $l(F_{\mathcal{A}}(z), y) = ||F_{\mathcal{A}}(z) - y||_2$, and is upper-bounded by a constant M. For a complex-valued neural network: $F_{\mathcal{A}}(z) := \sigma_L\left(A_L \sigma_{L-1}\left(A_{L-1} \cdots \sigma_1\left(A_1 z\right)\right)\right)$, where activation functions σ_i are ρ_i-Lipschitz, it is PAC-learnable.*

Proof. It suffices to prove that $R(f_S) - R(f) \leq \epsilon$ via the generalization upper bound under high probability.

Since

$$\begin{aligned} R(f_S) - R(f) &= R(f_S) - \widehat{R}(f_S) + \widehat{R}(f_S) - R(f) \\ &\leq R(f_S) - \widehat{R}(f_S) + \widehat{R}(f) - R(f), \end{aligned}$$

the last inequality holds because f_S is the empirical error minimizer, therefore, we have

$$\begin{aligned}|R(f_S) - R(f)| &\leq \left|R(f_S) - \widehat{R}(f_S) + \widehat{R}(f) - R(f)\right| \\ &\leq \left|R(f_S) - \widehat{R}(f_S)\right| + \left|\widehat{R}(f) - R(f)\right| \\ &\leq 2 \sup_{f\in\mathcal{F}} \left|\widehat{R}(f) - R(f)\right|.\end{aligned}$$

Hence

$$P\left(|R(f_S) - R(f)| \leq \epsilon\right) \geq P\left(\sup_{f\in\mathcal{F}} \left|\widehat{R}(f) - R(f)\right| \leq \frac{\epsilon}{2}\right)$$

.

As in Theorem 1, we have for any $f \in \mathcal{F}$

$$P\left(\left|R(f) - \widehat{R}(f)\right| \leq \frac{8M}{n^{\frac{3}{2}}} + \frac{36||Z||_2\sqrt{2ln(2W)}ln(n)R_{\mathcal{A}}}{n} + 3M\sqrt{\frac{ln\frac{2}{\delta}}{2n}}\right)$$
$$\geq 1 - \delta.$$

Notice that, these two statments are equivalent:

$$\begin{aligned}&\sup_{f\in\mathcal{F}} \left|\widehat{R}(f) - R(f)\right| \leq \frac{\epsilon}{2} \\ \Leftrightarrow &\forall f \in \mathcal{F}, \left|R(f) - \widehat{R}(f)\right| \leq \frac{\epsilon}{2}.\end{aligned}$$

Hence, we can claim that if

$$\frac{\epsilon}{2} \geq \frac{8M}{n^{\frac{3}{2}}} + \frac{36||Z||_2\sqrt{2ln(2W)}ln(n)R_{\mathcal{A}}}{n} + 3M\sqrt{\frac{ln\frac{2}{\delta}}{2n}},$$

then

$$P\left(\sup_{f\in\mathcal{F}} \left|\widehat{R}(f) - R(f)\right| \leq \frac{\epsilon}{2}\right) \geq 1 - \delta$$

i.e.

$$P\left(|R(f_S) - R(f)| \leq \epsilon\right) \geq 1 - \delta.$$

Hence, we can get the conclusion that if

$$n \geq \frac{8}{\epsilon^3}\left(8M + 36\left\|Z\right\|_2\sqrt{2\ln\left(2W\right)}R_{\mathcal{A}} + 3M\sqrt{\frac{\ln\frac{2}{\delta}}{2}}\right)^3,$$

then

$$P\left(|R(f_S) - R(f)| \leq \epsilon\right) \geq 1 - \delta.$$

Therefore, PAC-learnability of complex-valued neural networks get proved. □

Appendix D. Generalization of Sequential Data

In this section, we aim at proving Theorem 2. Theorem 2 shows an extension of generalization to sequential data case. Therefore, sequential analogues of complexities [19] are presented in this section to complete the proof.

Appendix D.1. Sequential Rademacher Complexity

In the case of classical complexity measure, we use the expectation of the supremum of Rademacher process to define the Rademacher complexity. In the sequential Rademacher case, the intuition is quite similar. Rakhlin et al. [19] illustrated a binary tree process to be the analogue of Rademacher process, which coincides with Rademacher process under i.i.d assumption, but behaves differently in general. The notion of a tree is defined as following:

"A $\mathcal{Z}$-valued tree $\mathbf{z}$ of depth n is a rooted complete binary tree with nodes labeled by elements of $\mathcal{Z}$. We identify the tree $\mathbf{z}$ with the sequence $(\mathbf{z}_1, \ldots, \mathbf{z}_n)$ of labeling functions $\mathbf{z}_i : \{\pm 1\}^{i-1} \mapsto \mathcal{Z}$ which provide the labels for each node. Here, $\mathbf{z}_1 \in \mathcal{Z}$ is the label for the root of the tree, while $\mathbf{z}_i$ for $i > 1$ is the label of the node obtained by following the path of length $i - 1$ from the root, with $+1$ indicating 'right' and -1 indicating 'left'. A path of length n is given by the sequence $\epsilon = (\epsilon_1, \ldots, \epsilon_n) \in \{\pm 1\}^n$. For brevity, we shall often write $\mathbf{z}_t(\epsilon)$, but it is understood that $\mathbf{z}_t$ only depends only on the prefix $(\epsilon_1, \ldots, \epsilon_{t-1})$ of ϵ. Given a tree $\mathbf{z}$ and a function $f : \mathcal{Z} \mapsto \mathbb{R}$, we define the composition $f \circ \mathbf{z}$ as a real-valued tree given by the labeling functions $(f \circ \mathbf{z}_1, \ldots, f \circ \mathbf{z}_n)$." [19]

Therefore, the definition of sequential Rademacher complexity is stated in Definition 2.

Definition 2 ([19]). *For a $\mathcal{Z}$-valued tree $\mathbf{z}$ with depth n, then the sequential Rademacher complexity of a function class*

$\mathcal{G}_{sq} := \{(z_t, y_t) \mapsto l\left(F_{\mathcal{A}}(z), y\right) : F_{\mathcal{A}} \in \mathcal{F}\}$ is defined as follows:

$$\mathfrak{R}_n^{sq}(\mathcal{G}_{sq}, \mathbf{z}) = \mathbb{E}\left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{t=1}^{n} \epsilon_t l\left(f\left(z_t(\epsilon)\right), y_t\right)\right],$$

and

$$\mathfrak{R}_n^{sq}(\mathcal{G}_{sq}) = \sup_{\mathbf{z}} \mathfrak{R}_n^{sq}(\mathcal{G}_{sq}, \mathbf{z}).$$

Here ϵ_t is the Rademacher variables taking value from $\{+1, -1\}$ with equal probability.

Appendix D.2. Sequential Rademacher Complexity Generalization Bound

When investigating the relation between generalization error and Rademacher complexity, we have the following theorem.

Theorem 6. *Given function class $\mathcal{F}$, sample $S = \{(z_1, y_1), (z_2, y_2), ..., (z_n, y_n)\}$ where (z_i, y_i) are i.i.d data points, we have*

$$\sup_{f\in\mathcal{F}} \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} f(z_i, y_i) - \mathbb{E}\left[f\right]\right] \leq \mathbb{E}\left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} f(z_i, y_i) - \mathbb{E}\left[f\right]\right] \leq 2\mathfrak{R}(\mathcal{F})$$

where $\mathfrak{R}(\mathcal{F}) = \mathbb{E}[\widehat{\mathfrak{R}}_S(\mathcal{F})]$.

For sequential Rademacher complexity, Rakhlin et al. [19] proved a similar theorem.

Theorem 7. *Given function class $\mathcal{F}$, sample $S = \{(z_1, y_1), (z_2, y_2), ..., (z_n, y_n)\}$ where (z_1, y_1) are sequential data points, then the following inequality holds:*

$$\mathbb{E}\left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{t=1}^{n} \left(\mathbb{E}\left[f\left(z_t, y_t\right) \mid \mathcal{A}_{t-1}\right] - f\left(z_t, y_t\right)\right)\right] \leq 2\mathfrak{R}_n^{sq}(\mathcal{F})$$

where $\mathfrak{R}_n^{sq}(\mathcal{F})$ denotes the sequential Rademacher complexity.

If the function class $\mathcal{F}$ is bounded, i.e. for any $f \in \mathcal{F}$, $\|f\|_\infty \leq M$, then the generalization error $\frac{1}{n}\sum_{t=1}^{n}\left(\mathbb{E}\left[f\left(z_t, y_t\right) \mid \mathcal{A}_{t-1}\right] - f\left(z_t, y_t\right)\right)$ is sharply concentrated around its expectation. which leads to Corollary 2.

Corollary 2. *Assume that for the target function class, any $f \in \mathcal{F}$, we have$\|f\|_\infty \leq M$. Given sample $S = \{(z_1, y_1), (z_2, y_2), ..., (z_n, y_n)\}$ where (z_1, y_1) are sequential data points, then under probability at least $1 - \delta$ the following inequality holds:*

$$\frac{1}{n}\sum_{t=1}^{n}\left(\mathbb{E}\left[f\left(z_t, y_t\right) \mid \mathcal{A}_{t-1}\right] - f\left(z_t, y_t\right)\right) \leq 2\mathfrak{R}_n^{sq}(\mathcal{F}) + M\sqrt{\frac{\log\frac{2}{\delta}}{2n}}.$$

Proof. This corollary is a consequence of McDiarmid's Inequality and Theorem 6. By McDiarmid's Inequality, since $\|f\|_\infty \leq M$, we have

$$\mathbb{P}(|\Delta(\mathcal{F}) - \mathbb{E}[\Delta(\mathcal{F})]| \geq t) \leq 2\exp\left(-2nt^2/M^2\right)$$

where $\Delta(\mathcal{F}) = \frac{1}{n}\sum_{t=1}^{n}\left(\mathbb{E}\left[f\left(z_t, y_t\right) \mid \mathcal{A}_{t-1}\right] - f\left(z_t, y_t\right)\right)$. Then by Theorem 6, we can get the sequential Rademacher complexity upper bound. □

As a consequence of Corollary 1, it's necessary to bound the sequential Rademacher complexity if we want to prove the generalization upper bound. This leads to the introduction of sequential Dudley Entropy Integral.

Appendix D.3. Sequential Dudley Entropy Integral

Before stating the sequential Dudley Entropy Integral, we first present the definition of sequential covering number [19]

Definition 3. *(Sequential Covering Number) A set C is a sequential α-cover (with respect to ℓ_p-norm) of $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{Z}}$ on a tree $\mathbf{z}$ of depth n if*

$$\forall f \in \mathcal{F}, \forall \epsilon \in \{\pm 1\}^n, \exists \mathbf{c} \in C \quad \textit{s.t.} \quad \left(\frac{1}{n}\sum_{t=1}^{n}\left|\mathbf{c}_t(\epsilon) - f\left(\mathbf{z}_t(\epsilon)\right)\right|^p\right)^{1/p} \leq \alpha.$$

The sequential covering number of a function class $\mathcal{F}$ on a given tree $\mathbf{z}$ is defined as

$$\mathcal{N}_p^{sq}(\alpha, \mathcal{F}, \mathbf{z}) = \min\left\{|C| : C \textit{ is an } \alpha\textit{-cover w.r.t. } \ell_p\textit{-norm of } \mathcal{F} \textit{ on } \mathbf{z}\right\},$$

and $\mathcal{N}_p^{sq}(\alpha, \mathcal{F}, n) := \sup_{\mathbf{z}} \mathcal{N}_p^{sq}(\alpha, \mathcal{F}, \mathbf{z})$.

Rakhlin et al. [19] provides the sequential version Dudley Entropy Integral as follows:

Theorem 8. *(Sequential Dudley Entropy Integral) For $p \geq 2$, the sequential Rademacher complexity of a function class $\mathcal{F} \subseteq [-1, 1]^{\mathcal{Z}}$ on a $\mathcal{Z}$-valued tree of depth n satisfies*

$$\mathfrak{R}_n^{sq}(\mathcal{F}) \leq \inf_{\alpha}\left\{4\alpha + \frac{12}{\sqrt{n}}\int_{\alpha}^{1}\sqrt{\log \mathcal{N}_2^{sq}(\delta, \mathcal{F}, n)}d\delta\right\}.$$

Notice that for the classical α-cover of $\mathcal{F}$ with regard to l_2 norm, denote it by V, we have for any given data matrix Z, and for any $\mathcal{F}_{\mathcal{A}} \in \mathcal{F}$, there exist $v \in V$ such that

$$\|\mathcal{F}_{\mathcal{A}}(Z) - v(Z)\|_2 \le \alpha.$$

Since given a set of sequential data, $\|\mathcal{F}_{\mathcal{A}}(Z) - v(Z)\|_2 = \sqrt{\sum\limits_{t=1}^{n} \left(\mathcal{F}_{\mathcal{A}}(z_t) - v(z_t)\right)^2}$.

Hence, if

$$\|\mathcal{F}_{\mathcal{A}}(Z) - v(Z)\|_2 \le \alpha,$$

then we have

$$\sqrt{\frac{1}{n}\sum_{t=1}^{n} \left(\mathcal{F}_{\mathcal{A}}(z_t) - v(z_t)\right)^2} \le \frac{\alpha}{\sqrt{n}}.$$

Hence, as a consequence of Lemma 2,

$$\ln \mathcal{N}_2^{sq}(\frac{\alpha}{\sqrt{n}}, \mathcal{G}_{sq}, n) \le \ln \mathcal{N}(\mathcal{G}, \alpha, \|\cdot\|_2) \le \frac{\|Z\|_2 \ln 4W^2}{\alpha^2} R_{\mathcal{A}}^{sq},$$

where $\mathcal{G}_{sq}$ denotes the loss function family of the sequential data set, $R_{\mathcal{A}}^{sq}$ denotes the spectral complexity of the CVNNs under the case of sequential data set, and $\mathcal{G}$ denotes the loss function family of the i.i.d data set.

Hence, we have

$$\ln \mathcal{N}_2^{sq}\left(\alpha, \mathcal{G}_{sq}, n\right) \le \frac{\|Z\|_2 \ln 4W^2}{n\alpha^2} R_{\mathcal{A}}^{sq}.$$

Appendix D.4. Proof of Theorem 2

As a consequence of the previous Sections D.1-D.3, we have

$$\ln \mathcal{N}_2^{sq}\left(\epsilon, \mathcal{G}_{sq}, n\right) \le \frac{\|Z\|_2 \ln 4W^2}{n\epsilon^2} R_{\mathcal{A}}^{sq}$$

for any $\epsilon \in \mathbb{R}^+$.

Therefore, by Lemma 4 and the sequential Dudley Entropy Integral, we can derive the following bound for the sequential Rademacher complexity:

$$\mathfrak{R}_n^{sq}(\mathcal{G}_{sq}) \le \frac{4M}{n} + 12\frac{\ln n\sqrt{R_{\mathcal{A}}}}{Mn}$$

where M denotes the upper bound for the loss function.

After plugging the above inequality into Corollary 2, we can get the desired bound stated in Theorem 2.

Appendix E. Additional Experiments Details

The section provides all the additional details of our experiments. The code is available at https://github.com/LeavesLei/cvnn_generalization.

Appendix E.1. Datasets

Our experiments are conducted on six datasets: MNIST [25], FashionMNIST [26], CIFAR-10, CIFAR-100, [27], IMDB [28], and Tiny ImageNet [29]. The details of these datasets are shown as follows.

- **MNIST** consists of $60,000$ training images and $10,000$ test images from 10 different classes. It can be downloaded from http://yann.lecun.com/exdb/mnist/.
- **FashionMNIST** consists of $60,000$ training images and $10,000$ test images from 10 different classes. It can be downloaded from https://github.com/zalandoresearch/fashion-mnist.
- **CIFAR-10** consists of $50,000$ training images and $10,000$ test images from 10 different classes, and **CIFAR-100** has the same data as CIFAR-10 while images in CIFAR-100 belong to 100 classes. CIFAR-10 and CIFAR-100 can be downloaded from https://www.cs.toronto.edu/~kriz/cifar.html.
- **IMDB** is a movie reviews sentiment classification dataset, in which each of training and test sets consists of $25,000$ movie reviews from 2 different classes. It can be downloaded from http://ai.stanford.edu/~amaas/data/sentiment/.
- **Tiny ImageNet** consists of $100,000$ training images and $10,000$ test images from 200 different classes. It can be downloaded from http://cs231n.stanford.edu/tiny-imagenet-200.zip.

For the image datasets, *i.e.*, MNIST, FashionMNIST, CIFAR-10, CIFAR-100, and Tiny ImageNet, we normalize each pixel of the images from the datasets to the range of $[0, 1]$ before feeding them into the neural network. For the IMDB dataset, we perform data pre-processing following https://github.com/manavgakhar/imdbsentiment/blob/master/IMdB_sentiment_analysis_project.ipynb.

Appendix E.2. Model Architectures

We employ the Python package **complexPyTorch** [30] to implement our CVNNs, which include complex-value CNNs and complex-value MLPs. The detailed architectures of CVNNs are presented in Supplementary Table E.2, and all the parameters in these network architectures are complex values except for the last layer.

Supplementary Table E.2: Detailed model architectures for different datasets.

MNIST/FashionMNIST/CIFAR-10/CIFAR-100	Tiny ImageNet	IMDB
$5 \times 5, 10$	$5 \times 5, 10$	fc-500
maxpool, 2×2	maxpool, 2×2	
$5 \times 5, 20$	$(5 \times 5, 20) \times 2$	fc-200
maxpool, 2×2	maxpool, 2×2	
fc-500	fc-500	
abs	abs	abs
fc-$10/100$, softmax	fc-200, softmax	fc-2, softmax

In Supplementary Table E.2, “$5 \times 5, 10$” denotes that the convolutional layer has 5×5 kernel size and 10 output channels. The strides for all convolutional layers are set to 1. “fc-500” denotes the fully-connected layer with the output features of 500. All convolutional layers and fully-connected layers are followed the ReLU layer except for the last layer. “abs” is the absolute layer that computes the absolute value of each element in input and can convert complex values to real values.

Appendix E.3. Implementation Details

This section provides all the additional implementation details for our experiments.

Model training. We employ SGD to optimize all the models with $\mathrm{momentum} = 0.9$.

Training strategy for MNIST and FashionMNIST. Every model is trained by SGD for 100 epochs, in which the batch size is set as 1024, and the learning rate is fixed to 0.01.

Training strategy for CIFAR-10 and CIFAR-100. Models is trained by SGD for 100 epochs, in which the batch size is set as 128, and the learning rate is fixed to 0.01.

Training strategy for IMDB. Models is trained by SGD for 100 epochs, in which the batch size is set as 512. The learning rate is initialized as 0.01 and decayed by 0.2 every 40 epoch.

Training strategy for Tiny ImageNet. Models is trained by SGD for 100 epochs, in which the batch size is set as 128. The learning rate is initialized as 0.01 and decayed by 0.2 every 40 epoch.