Received: 5 December 2019 | Accepted: 26 February 2021 | Published on: 23 March 2021

DOI: 10.1002/sam.11506

WILEY

RESEARCH ARTICLE

Approximation error of Fourier neural networks

Abylay Zhumekenov[1] | **Rustem Takhanov**[2] | **Alejandro J. Castro**[2] | **Zhenisbek Assylbekov**[2]

[1]Computer, Electrical and Mathematical Sciences and Engineering, King Abdullah University of Science and Technology, Thuwal, Kingdom of Saudi Arabia

[2]Department of Mathematics, School of Sciences and Humanities, Nazarbayev University, Nur-Sultan, Kazakhstan

Correspondence
Zhenisbek Assylbekov, 010000, Kazakhstan, Nur-Sultan, Kabanbay Batyr ave., 53, Office 7.231.
Email: zhassylbekov@nu.edu.kz

Funding information
Ministry of Education and Science of the Republic of Kazakhstan, Grant/Award Number: IRN AP05133700; Nazarbayev University, Grant/Award Number: 240919FD3921

Abstract

The paper investigates approximation error of two-layer feedforward Fourier Neural Networks (FNNs). Such networks are motivated by the approximation properties of Fourier series. Several implementations of FNNs were proposed since 1980s: by Gallant and White, Silvescu, Tan, Zuo and Cai, and Liu. The main focus of our work is Silvescu's FNN, because its activation function does not fit into the category of networks, where the linearly transformed input is exposed to activation. The latter ones were extensively described by Hornik. In regard to non-trivial Silvescu's FNN, its convergence rate is proven to be of order $O(1/n)$. The paper continues investigating classes of functions approximated by Silvescu FNN, which appeared to be from Schwartz space and space of positive definite functions.

KEYWORDS

approximation error, convergence, Fourier, neural networks

1 | INTRODUCTION

Artificial neural networks have been widely used in machine learning and acquired their popularity during 1990s. The second burst occurred in recent years due to a rapid increase of proccessable data ("Big-Data") and availability of powerful computing machinery. Modern trends in deep neural networks helped to achieve superior results in pattern recognition, text processing, and other fields of machine learning. However, this paper is focused on "shallow" 2-layer neural nets - a classic case for multilayer perceptrons. Specifically, the study assesses the convergence rate for one of a two-layer neural network with activators composed of a product of cosine functions with different frequencies. The architecture was proposed by Silvescu in 1999 [1]. Since this idea is inspired by Fourier series and Fourier transform, such networks are referred to as "Fourier Neural Networks" (FNNs).

While most of the theory behind neural networks with standard activation functions is well-established, the peculiar ones such as above leave a room for research. For this particular FNN, we derive an upper bound for the approximation error equal to $C^2 4^{d-1}/n$, where d, n, C are, respectively, input dimension size, number of neurons, and a constant dependent on the activation function. The proof of this assertion uses a technique similar to the one

used in Reference [2]. The paper proceeds with identifying classes of functions, for which the approximation error bound can be computed.

Besides theoretical findings, the paper attempts to confirm them with computational results. The experiments include implementation of the Silvescu Fourier Neural Network in Python's TensorFlow library, generating synthetic data according to classes of functions obtained previously, training the model, and assessing the approximation error.

2 | OVERVIEW

2.1 | General model

Before talking about FNN implementations, we shall give a context of feedforward neural nets in general. The name "feedforward" is derived from the concept of passing weighted value of the input vector forward, to the hidden layer. It is then subject to a non-linear transformation called "activation". This helps to catch non-linear behavior of the input data. After this, the obtained value is passed further, to the output layer. Mathematically, the general model can be given as a mapping

$$x \to v_0 + \sum_{k=1}^{n} v_k \sigma(w_k \cdot x + b_k), \tag{1}$$

where $x \in \mathbb{R}^d$ represents a multidimensional input, $w_k \in \mathbb{R}^d$ is a weight vector, $b_k \in \mathbb{R}$ is a bias, σ is called an "activation function", and $v_k \in \mathbb{R}$ are output weights.

Such model falls in the paradigm "weighted sum of input characteristics". The standard choices for the activation function are "sigmoidal" functions (sometimes reffered as "squashers"), which have a bounded output. For example, the logistic activation function has the following expression

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \tag{2}$$

where $z \in \mathbb{R}$, and the output is in the interval (0, 1). The hyperbolic tangent function is given by

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}. \tag{3}$$

In general, most of them have the shape similar to a heaviside step function

$$H(z) = \begin{cases} 0, & z < 0, \\ 1, & z \geq 0. \end{cases} \tag{4}$$

The values of these activations are stored in so-called "hidden neurons", a transitional state between input and output. There can be several such neurons placed in parallel in a feedforward network, composing a hidden layer. The anatomy of the network is illustrated in Figure 1.

Having sigmoidal activation and a sufficient number of neurons, a two-layer neural network can approximate almost any continuous function [3, 4]. Studies show that the accuracy of approximation increases with the number of hidden layers [5] and with the number of hidden neurons in general [4].

2.2 | Existing implementations

Fourier Neural Networks were inspired in some way by Fourier Series and Fourier Transform. For this reason, activations used in these nets contain cosine transformations. Several implementations with different activation functions have been proposed starting from the late 1980s.

2.3 | Gallant and White FNN

One of the earliest examples is the network of Gallant and White [6]. The suggested model uses a "cosine squasher" instead of a standard sigmoid activation. The function is monotone and bounded.

$$\sigma_{GW}(z) = \begin{cases} 0, & -\infty < z < \frac{\pi}{2}, \\ \frac{1}{2}\left[\cos\left(z + \frac{3\pi}{2}\right) + 1\right], & -\frac{\pi}{2} \leq z \leq \frac{\pi}{2}, \\ 1, & \frac{\pi}{2} < z < \infty. \end{cases} \tag{5}$$

The main characteristic of the activation is that it cuts off frequencies higher than $\frac{\pi}{2}$. The network then represents a mapping

$$x \to v_0 + \sum_{k=1}^{n} v_k \sigma_{GW}(w_k \cdot x + b_k), \tag{6}$$

with n being the number of hidden layer units. According to Gallant and White, the network possesses Fourier Series approximation properties. That is, if a periodic function to be approximated is in L_2, the network converges in L_2 sense as n grows; and if it is continuous, the network converges uniformly.

2.4 | Tan, Zuo and Cai, Liu FNN

Another implementation was proposed by several authors, by Tan [7], by Zuo and Cai [8]. In general, they can be

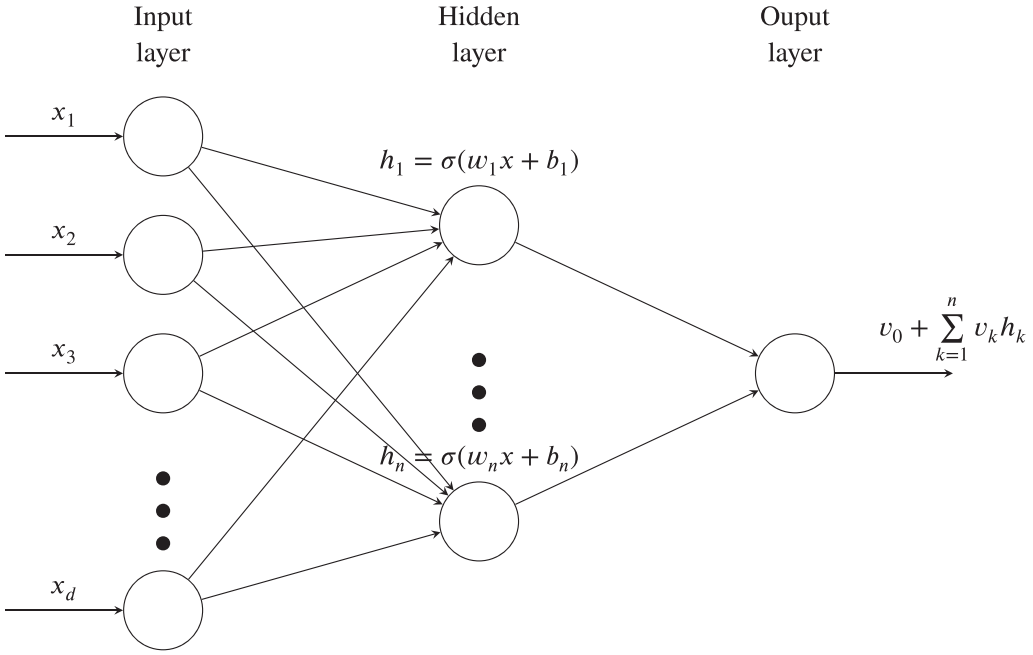


FIGURE 1 Feedforward neural network with one hidden layer and one-dimensional output

written as

$$x \to v_0 + \sum_{k=1}^{n} v_k \cos(w_k \cdot x + b_k) + u_k \sin(w_k \cdot x + b_k). \quad (7)$$

Note that we have same weights and biases in both sin and cos. A slightly different approach is to set the parameters of sin and cos independent of each other:

$$x \to v_0 + \sum_{k=1}^{n} v_k \cos(w_k \cdot x + b_k) + u_k \sin(s_k \cdot x + t_k). \quad (8)$$

This was suggested by Liu [9]. These two implementations resemble Fourier Series, with the exception that the coefficients are not fixed, as it is the case with Fourier partial sums, but rather trained on data.

2.5 | Silvescu FNN

Silvescu proposed a different implementation of an FNN [1]. It has the most exotic activation function, as a result of non-standard way of discretization of a problem. The activation is as follows

$$\sigma_{\text{Silv}}(x; w_k, b_k) = \prod_{i=1}^{d} \cos(w_{ki} x_i + b_{ki}), \quad (9)$$

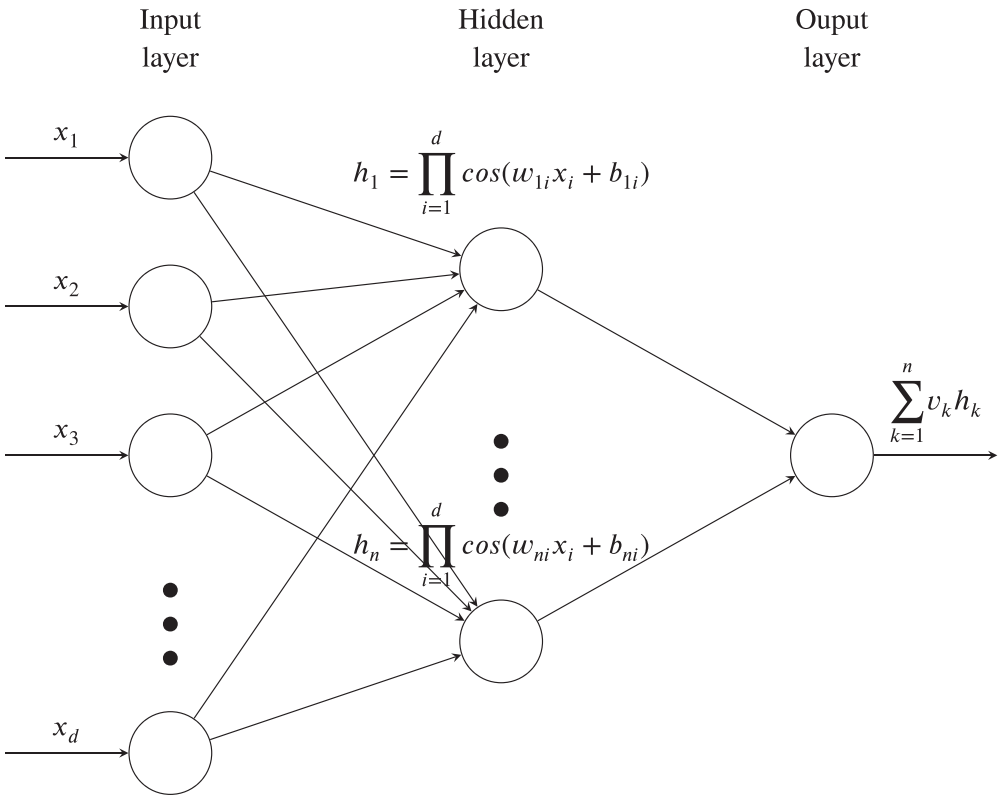
where d is the dimension size of an input. The network formula is then

$$x \to \sum_{k=1}^{n} v_k \sigma_{\text{Silv}}(x; w_k, b_k). \quad (10)$$

Instead of a nonlinear transformation of weighted sums of inputs, here we take a product of nonlinear filters for each weighted input. This makes Silvescu's implementation the most interesting and challenging to analyze among existing FNNs. Therefore, the paper is focused on the convergence properties of this particular FNN.

2.6 | Discussion

Despite having several FNN implementations, this study is concerned with Silvescu's FNN. It does not fit into the standard way of constructing two-layer feedforward neural networks, and appears to have the least trivial proof for convergence rate. As one might have noticed, the rest of FNNs (including standard neural nets) can be rewritten in the form $x \to v_0 + \sum v_k \sigma(w_k \cdot x + b_k)$. All such neural networks have already been studied by Hornik [10] in 1989, and it has been shown that networks converge at a rate of $O(1/n)$ as the hidden layer size n increases. Silvescu's activation function does not take an affine transformation of input $w_k \cdot x + b_k$ as its argument. Instead, it is nonlinear in each spatial component of an input, $\sigma_{\text{Silv}} = \prod \cos(w_{ki} x_i + b_{ki})$ and graphically it looks as follows:



Since such networks fall out of the paradigm of classic neural nets, little is known about their convergence properties. Our research, therefore, attempts to fill in some of the gaps in neural network approximation theory. In particular, we show that for a smooth enough function f there exists a network of the form

$$f_n(x) = \sum_{k=1}^{n} c_k \sigma_{\text{Silv}}(x; a_k, b_k) \tag{11}$$

which has the error bounded as

$$\int_B |f(x) - f_n(x)|^2 \mu(dx) \leq \frac{C^2 4^{d-1}}{n}, \tag{12}$$

where $B \subset \mathbb{R}^d$ is a bounded set, C is a constant, and μ is a probability measure. That is, the order of convergence for Silvescu FNN is $O(1/n)$, given fixed dimension size d and a bound C for Fourier transform. Convergence rates for all mentioned FNN implementations are also expected to be of order $O(1/n)$.

A very important method to derive the convergence rate is used by Barron in his investigation of sigmoidal activation functions [2]. The Gallant and White FNN falls under the scope of Barron's work. The study shows that for functions with an existing Fourier integral $f(x) = \int e^{iw \cdot x} \widetilde{f}(w) dw$ and a finite first moment $\int |w| |\widetilde{f}(w)| dw < C$, two-layer neural networks with sigmoidal activation have an error of $O(1/n)$. Indeed, the way σ_{GW} was defined lets us perceive it as a sigmoidal function, in which case we can apply Barron's results directly.

In the course of our paper, we utilize the main idea of Barron's error estimation and derive our own proof for Silvescu FNN. The main difference between these two results is in assumptions on the moments of Fourier transform of a function. While Barron sets the first moment $\int |w| |\widetilde{f}(w)| dw < C$, we choose functions with finite zeroth moment $\int |\widetilde{f}(w)| dw < C$. This is, in fact, equivalent to demanding Fourier integral to be in $L_1(\mathbb{R}^d)$.

Such assumption was also used by Jones [11] to show convergence rate of order $O(1/n)$ for networks with an 'affine transformation of an input' mentioned above. FNNs used in the works of Tan, Zuo and Cai, and Liu come from this category. Proper modifications to Barron's proof would also confirm the results of Jones and Hornik [10, 11]. The consequent analysis of the assumption helps to identify suitable classes of functions for which the error bound $\frac{C^2 4^{d-1}}{n}$ holds. These are the classes of positive definite functions and the class of Schwartz functions. The most common functions in both classes are Gaussians and their modifications. Experiments for such choices of functions show controversial results. Although functions can be approximated within the indicated upper error bound, errors stop decreasing at some point. Possible reason for such behavior might be an over-parametrization.

However, for smaller number of neurons, the dependence of error on the hidden layer size seems to experimentally confirm the convergence rate of $O(1/n)$.

3 | MAIN RESULTS

3.1 | Proposition

Let us consider functions on $\mathbb{R}^d$ for which the Fourier representation has the form

$$f(x) = \int_{\mathbb{R}^d} e^{iw\cdot x}\widetilde{F}(dw), \tag{13}$$

where $\widetilde{F}(dw) = e^{i\theta(w)}F(dw)$ is a unique complex-valued measure (Fourier distribution), $F(dw)$ is the magnitude distribution, and $\theta(w)$ is the phase. Define the class of functions Γ, which can be represented in the form above for some $\widetilde{F}(dw)$ and finite $\int F(dw)$:

$$\Gamma = \left\{ f : \mathbb{R}^d \to \mathbb{R} \middle| \int_{\mathbb{R}^d} F(dw) < \infty \right\}.$$

Then, for each $C > 0$ we can define the classes Γ_C

$$\Gamma_C = \left\{ f : \mathbb{R}^d \to \mathbb{R} \middle| C_f = \int_{\mathbb{R}^d} F(dw) < C \right\}.$$

Now consider a bounded set $B \subset \mathbb{R}^d$. Then define Γ_B for which the representation in Equation (13) holds for $x \in B$ for some $\widetilde{F}(dw)$ with finite magnitude measure:

$$\Gamma_B = \left\{ f : B \to \mathbb{R} \middle| \int_{\mathbb{R}^d} F(dw) < \infty \right\}.$$

Finally, for each $C > 0$, let

$$\Gamma_{C,B} = \left\{ f : B \to \mathbb{R} \middle| C_{f,B} = \int_{\mathbb{R}^d} F(dw) \le C \right\}.$$

Theorem 1. *Let*

$$\phi = \sigma_{\text{Silv}}(x; a_k, b_k) = \prod_{j=1}^{d} \cos(a_{kj}x_j + b_{kj})$$

for $a_{kj}, b_{kj} \in \mathbb{R}$ and $x, a_k, b_k \in \mathbb{R}^d$. For every function $f \in \Gamma_{B,C}$ and any probability measure μ, there exists $f_n(x)$, $n \ge 1$, of the form

$$f_n(x) = \sum_{k=1}^{n} c_k \sigma_{\text{Silv}}(x; a_k, b_k) \tag{14}$$

such that

$$\int_B |f(x) - f_n(x)|^2 \mu(dx) \le \frac{C^2 4^{d-1}}{n}. \tag{15}$$

3.2 | Proof

The proof of Theorem 1 uses several lemmas, which we specify below.

Lemma 1. *If f is in the closure of the convex hull of a set G in a Hilbert space, with $\|g\| \le b$ for each $g \in G$, then for every $n \ge 1$, and every $c' > b^2 - \|f\|^2$, there exists f_n in the convex hull of n points in G such that $\|f - f_n\|^2 \le \frac{c'}{n}$.*

Proof The proof of this lemma can be found in Barron's paper [2]. ■

Lemma 2. *For every function $f \in \Gamma_{C,B}$ and every function $g \in G_{\cos}$, the function f is in the closure of the convex hull of $G_{\cos}$, where the closure is taken in $L_2(\mu, B)$ and*

$$G_{\cos} = \{\gamma \cos(w \cdot x + b) : |\gamma| \le C\}.$$

Proof Let $\widetilde{F}(dw) = e^{i\theta(w)}F(dw)$ be the magnitude and the phase decomposition of the function f on B for which $\int F(dw) \le C$, where $w \in \mathbb{R}^d$. Since the function f we are considering is a real-valued function, from Equation (13) we deduce that for $x \in B$

$$\begin{aligned} f(x) &= \text{Re}\int_{\mathbb{R}^d} e^{iw\cdot x}\widetilde{F}(dw) = \text{Re}\int_{\mathbb{R}^d} e^{iw\cdot x}e^{i\theta(w)}F(dw) \\ &= \text{Re}\int_{\mathbb{R}^d} e^{iw\cdot x + i\theta(w)}F(dw) = \int_{\mathbb{R}^d} \cos(w\cdot x + \theta(w))F(dw) \\ &= \int_{\mathbb{R}^d} C_{f,B}\cos(w\cdot x + \theta(w))\frac{F(dw)}{C_{f,B}} = \int_{\mathbb{R}^d} g(x,w)\Lambda(dw), \end{aligned} \tag{16}$$

where $C_{f,B} = \int F(dw) \le C$ is our bound on functions from $\Gamma_{B,C}$; $g(x, w) = C_{f,B}\cos(w \cdot x + \theta(w))$; and $\Lambda(dw) = F(dw)/C_{f,B}$ is a probability measure, since

$$\int_{\mathbb{R}^d} \Lambda(dw) = \int_{\mathbb{R}^d} \frac{F(dw)}{C_{f,B}} = \frac{1}{C_{f,B}}\int_{\mathbb{R}^d} F(dw) = 1.$$

From Equation (16), it is known that $f(x) = \mathbb{E}_w[g(x, w)]$. In addition, $|g(x, w)| \le |C_{f,B}| \le C$. Most importantly, the last row of Equation (16) represents $f(x)$ as an infinite convex combination of functions in the set

$$G_{\cos} = \{\gamma \cos(w \cdot x + b) : |\gamma| \le C\}.$$

To show this, let $w_1, w_2, \ldots, w_n \in \mathbb{R}^d$ be drawn independently from the same distribution Λ. Consider now $g(x, w_1), g(x, w_2), \ldots, g(x, w_n)$. Then the sample mean $f_n(x; w_1, \ldots, w_n) = \frac{1}{n}\sum_{i=1}^{n} g(x, w_i)$ has an expected value equal to $f(x)$, that is, $\mathbb{E}_w[f_n(x)] = f(x)$, and at the same time it is a convex combination of functions from $G_{\cos}$. We will get results if we apply Fubini's Theorem to the expected

value of squared $L_2(\mu, B)$ norm.

$$\begin{aligned}\mathbb{E}_w\|f-f_n\|^2_{L_2(\mu,B)} &= \mathbb{E}_w\left(\int_B |f-f_n|^2\mu(dx)\right)\\ &= \int_B (\mathbb{E}_w[\mathbb{E}_w[f_n]-f_n]^2)\mu(dx)\\ &= \int_B \mathrm{Var}[f_n]\mu(dx) = \int_B \frac{1}{n}\mathrm{Var}[g]\mu(dx)\\ &\le \frac{1}{n}\int_B C^2\mu(dx) = \frac{C^2}{n}.\end{aligned}$$

The $L_2(\mu, B)$ norm above converges to zero in mean. By taking into account that the norm is positive, there is a sequence $w_1, \ldots, w_n$ for which the norm converges to zero. Thus, there exists a convex combination $f_n(x; w_1, \ldots, w_n)$ of functions from $G_{\cos}$ that converges to f in $L_2(\mu, B)$. Therefore, f is in the closure of the convex hull of $G_{\cos}$ in $L_2(\mu, B)$. ■

Lemma 3. *Functions in $G_{\cos}$ are in the convex hull of functions in G_{Silv}, that is, $G_{\cos} \subset coG_{\text{Silv}}$, where*

$$G_{\text{Silv}} = \left\{ 2^{d-1}\gamma\prod_{j=1}^{d}\cos(a_{kj}x_j + b_{kj}) : |\gamma| \le C \right\}$$

for $w, x \in \mathbb{R}^d$; $b_{kj} \in \mathbb{R}$ and $|\gamma| = |C_{f,B}| = |\int F(dw)| \le C$.

Proof The proof is by induction. First, we consider the base case $d = 1$. Each function $g \in G_{\cos}$ is a single variable function. Then we have

$$\begin{aligned}g(x) &= \gamma\cos(w_1x_1 + b)\\ &= \sum_{k=1}^{2^0}\frac{1}{2^0}\left[2^0\gamma\prod_{j=1}^{1}\cos(a_{kj}x_j + b_{kj})\right]\\ &= \sum_{k=1}^{2^{1-1}}\frac{1}{2^{1-1}}\left[2^{1-1}\gamma\prod_{j=1}^{1}\cos(a_{kj}x_j + b_{kj})\right],\end{aligned}$$

which is a convex combination of 2^{1-1} functions in the class G_{Silv} with $a_{kj} = w_1$. Thus, when the dimension is $d = 1$, $G_{\cos} \subset coG_{\text{Silv}}$.

Next, let us prove the general case for $d = m+1$. Assume $G_{\cos} \subset coG_{\text{Silv}}$ for $d = m$. Then

$$\begin{aligned}g(x) &= \gamma\cos(w \cdot x + b)\\ &= \sum_{k=1}^{2^{m-1}}\frac{1}{2^{m-1}}\left[2^{m-1}\gamma\prod_{j=1}^{m}\cos(a_{kj}x_j + b_{kj})\right],\end{aligned}$$

where $x, w \in \mathbb{R}^m$ and $a_{kj}, b, b_{kj} \in \mathbb{R}$.

Now consider $\widetilde{g}(\widetilde{x}) \in \widetilde{G}_{\cos}$, where $\widetilde{x} \in \mathbb{R}^{m+1}$ and $\widetilde{g}$ is a function on $\mathbb{R}^{m+1}$. We also have $\widetilde{w} \in \mathbb{R}^{m+1}$. Actually, we can write $\widetilde{x} = (x_1, x_2, \ldots, x_m, x_{m+1})$, where the first m coordinates are from x. So,

$$\begin{aligned}\widetilde{g}(\widetilde{x}) &= \gamma\cos(\widetilde{w}\cdot\widetilde{x} + b)\\ &= \gamma\cos(w_1x_1 + \cdots + w_mx_m + w_{m+1}x_{m+1} + b)\\ &= \gamma\cos(w_1x_1 + \cdots + \widehat{w}_m\widehat{x}_m + b)\end{aligned}$$

Where we define $\widehat{w}_m\widehat{x}_m \coloneqq w_mx_m + w_{m+1}x_{m+1}$. Now we can apply results for the m-dimensional case to the identity above.

$$\begin{aligned}\widetilde{g}(\widetilde{x}) &= \sum_{k=1}^{2^{m-1}}\frac{1}{2^{m-1}}\left[2^{m-1}\gamma\left(\prod_{j=1}^{m-1}\cos(a_{kj}x_j + b_{kj})\right)\right.\\ &\quad\left.\times(\cos(\widehat{w}_m\widehat{x}_m + b_{km}))\right]\\ &= \sum_{k=1}^{2^{m-1}}\left[\gamma\left(\prod_{j=1}^{m-1}\cos(a_{kj}x_j + b_{kj})\right)\right.\\ &\quad\left.\times(\cos(w_mx_m + w_{m+1}x_{m+1} + b_{km}))\right]\\ &= \sum_{k=1}^{2^{m-1}}\left[\gamma\left(\prod_{j=1}^{m-1}\cos(a_{kj}x_j + b_{kj})\right)\right.\\ &\quad\left.\times(\cos((w_mx_m + b_{km}) + (w_{m+1}x_{m+1})))\right]\\ &= \sum_{k=1}^{2^{m-1}}\gamma\left(\prod_{j=1}^{m-1}\cos(a_{kj}x_j + b_{kj})\right)\\ &\quad\times(\cos(w_mx_m + b_{km})\cos(w_{m+1}x_{m+1})\\ &\quad-\sin(w_mx_m + b_{km})\sin(w_{m+1}x_{m+1}))\\ &= \sum_{k=1}^{2^{m-1}}\left[\gamma\left(\prod_{j=1}^{m-1}\cos(a_{kj}x_j + b_{kj})\right)\right.\\ &\quad\times(\cos(w_mx_m + b_{km})\cos(w_{m+1}x_{m+1})\\ &\quad+\cos\left(w_mx_m + b_{km} - \frac{\pi}{2}\right)\\ &\quad\left.\times\cos\left(w_{m+1}x_{m+1} + \frac{\pi}{2}\right)\right)\Bigg]\\ &= \sum_{k=1}^{2^{m-1}}\left[\gamma\left[\prod_{j=1}^{m+1}\cos(a_{kj}x_j + b_{kj})\right.\right.\\ &\quad\left.\left.+\prod_{i=1}^{m+1}\cos(a_{ki}x_i + b_{ki})\right]\right]\\ &= \sum_{k=1}^{2^m}\gamma\left[\prod_{j=1}^{m+1}\cos(a_{kj}x_j + b_{kj})\right]\\ &= \sum_{k=1}^{2^m}\frac{1}{2^m}\left[2^m\gamma\prod_{j=1}^{m+1}\cos(a_{kj}x_j + b_{kj})\right].\end{aligned}$$

The function above is a convex combination of 2^m functions from G_{Silv}. Then, $G_{\cos} \subset coG_{\text{Silv}}$ when $d = m+1$, and therefore, by induction, for any dimension d. Note that expanding the cosine resulted in two cosine products, which have same frequencies. This suggests that almost every a_{kj} is repeated twice along the k-index.

Lemma 4. *For every function $f \in \Gamma_{C,B}$ and every function $g \in G_{\text{Silv}}$, the function f is in the closure of the convex hull of G_{Silv}, where the closure is taken in $L_2(\mu, B)$.*

Proof We will use Lemma 2 and Lemma 3 to show that f is in the closure of the convex hull of G_{Silv}. From Lemma 2 it follows that $f \in \Gamma_{C,B} \subset \overline{co}G_{\cos}$, while from Lemma 3 we obtain $G_{\cos} \subset coG_{\text{Silv}}$. Convex hull of $G_{\cos}$ is also a subset of G_{Silv}, that is, $coG_{\cos} \subset coG_{\text{Silv}}$. By taking limits, one can show that $\overline{co}G_{\cos} \subset \overline{co}G_{\text{Silv}}$ also holds. Thus, $f \in \overline{co}G_{\text{Silv}}$.

Proof of Theorem 1. Two important results, which we utilize are Lemma 1 and Lemma 4. From Lemma 4, we have $f(x) \in \overline{co}G_{\text{Silv}}$, where

$$G_{\text{Silv}} = \left\{ 2^{d-1}\gamma \prod_{j=1}^{d} \cos(a_{kj}x_j + b_{kj}) \,:\, |\gamma| \le C \right\}.$$

Functions in this class are measurable and bounded by $g(x) \le \ |2^{d-1}\gamma| \ \le 2^{d-1}C$. Then the $L_2(\mu, B)$ norm is

$$\begin{aligned} \|g(x)\|^2_{L_2(\mu,B)} &= \int_B |g(x)|^2 \mu(dx) \le \mu(B) \sup_{x\in B} |g(x)|^2 \\ &\le \mu(B) \sup_{g\in G_{\text{Silv}}} \sup_{x\in B} |g(x)|^2 \le \sup_{g\in G_{\text{Silv}}} \sup_{x\in B} |g(x)|^2 \\ &\le |2^{d-1}\gamma|^2 \le 4^{d-1}C^2 = b^2. \end{aligned}$$

Then, by Lemma 1, there exists f_n in the convex hull of n points of G_{Silv} such that $\|f - f_n\| \le c'/n$ for every $n \ge 1$ and every $c' > b^2 - \|f\|^2$. Let us choose $c' = b^2$. Then the desired bound holds

$$\|f - f_n\| \le \frac{c'}{n},$$

where

$$\begin{aligned} f_n(x) &= \sum_{k=1}^{n} t_k \left(2^{d-1}\gamma \prod_{j=1}^{d} \cos(a_{kj}x_j + b_{kj}) \right) \\ &= \sum_{k=1}^{n} c_k \phi(x; a_k, b_k), \end{aligned}$$

for $t_k > 0$ and $\sum t_k = 1$. We can restrict coefficients in our network to satisfy the bound $\sum |\, c_k \,| \le 2^{d-1}C$.

4 | FUNCTIONS IN Γ_C

4.1 | Functions with absolutely integrable Fourier transform

The goal is to identify functions in the class Γ_C (or $\Gamma_{C,B}$), that is, for which $\int F(dw) < C$ holds. This condition is equivalent to demanding Fourier transform to be in L_1, that is, $\int |\widetilde{F}(dw)| < C$ (or alternatively, $\int |\widetilde{f}(w)| dw < C$). We also give the formula to compute the constant C, which we use later to compare a theoretical error bound with experimental findings.

There are two main classes of functions, which satisfy the mentioned conditions for the transform:

1. Positive definite functions
2. Schwartz functions

First, we consider positive definite functions. In the definition, these are very similar to positive definite kernels, which are important tools in statistical learning theory. Let f be a positive definite function, which requires $\sum x_i x_j f(x_i - x_j)$ to be non-negative for any choice of x_i, x_j [12]. Alternatively, the matrix A, whose elements are $a_{i,j} = f(x_i - x_j)$, has to be positive semi-definite [2]. Then, if the function is continuous on $\mathbb{R}^d$, by Bochner's theorem [13, p. 144], it can be represented as $f(x) = \int e^{iw\cdot x} F(dw)$. Here, $F(dw)$ is a positive measure and corresponds to $|\widetilde{F}(dw)|$. Then the constant in the error bound can be computed as

$$C = \int_{\mathbb{R}^d} F(dw) = \int_{\mathbb{R}^d} |\widetilde{f}(w)| dw = \int_{\mathbb{R}^d} \widetilde{f}(w) dw = f(0). \quad (17)$$

We can observe that for positive definite functions, the L_1-norm of its Fourier transform is defined by the behavior of f at the origin. That is, one does not even need to know the transform in order to compute the constant C.

There exists another class of functions with absolutely integrable Fourier transform - Schwartz space $\mathcal{S}(\mathbb{R}^d)$. This is a space of rapidly decreasing functions on $\mathbb{R}^d$ (functions with derivatives decreasing faster than the inverse of any polynomial). The Schwartz space is widely used in Fourier analysis and applications can be found in the field of partial differential equations. The definition is given as follows [14, p. 134]:

$$\mathcal{S}(\mathbb{R}^d) = \{ f \in C^\infty(\mathbb{R}^d) : \| f \|_{\alpha,\beta} < \infty \quad \forall \alpha, \beta \in \mathbb{N}^d \} \quad (18)$$

where α, β are multi-indices, $C^\infty(\mathbb{R}^d)$ is the space of infinitely differentiable functions and

$$\| f \|_{\alpha,\beta} = \sup_{x\in\mathbb{R}^d} |x^\alpha D^\beta f(x)|. \quad (19)$$

It is known that Fourier transform maps functions from Schwartz space onto itself, that is, if $f \in S(\mathbb{R}^d)$, then $\widetilde{f} \in S(\mathbb{R}^d)$ [15, p. 331]. Since we also know that $S(\mathbb{R}^d) \subset L_p(\mathbb{R}^d)$ [16], the transform then satisfies $\widetilde{f} \in L_1(\mathbb{R}^d)$.

4.2 | Properties of functions in Γ_C

Of course, functions obtained from Schwartz space by translation (shift), dilation (scaling), summation, and consequently by a linear combination are also in Schwartz space. This can be seen straight from the definition. From Reference [14, p. 142], product and convolution of Schwartz functions are also in the same space. This is also true for some of such cases when we pick functions from Γ_C. To show this, let f be in the class Γ_C. Functions obtained by applying the following operations are also in Γ (particularly in $\Gamma_{C'}$, where C' is a different constant).

(a) Translation (time shifting). If we consider $f(x+a)$, its Fourier transform can be rewritten as $\widetilde{f}(x+a) = e^{-ia\cdot w}\widetilde{f}(x)$ [17, p. 92]. Then, since the resulting Fourier transform is a rotation on a complex plane, this transform also lives in L_1. Therefore, $f \in \Gamma_C$.

(b) Dilation (time scaling). For $g(x) = f(ax)$, we have $\widetilde{g}(w) = (1/2\pi)^d \int e^{-iw\cdot x} g(x)dx = (1/2\pi)^d \int e^{-iw\cdot x} f(ax)\, dx$. If we use a substitution $x = y/a$, we obtain the transform $\widetilde{g}(w) = (1/2\pi)^d \int e^{-iw/a\cdot y} f(y)dy = \widetilde{f}(w/a)/a$.

Now, by taking a change of variables as $u = w/a$ we get $\int |\widetilde{g}(w)|dw = \int |\widetilde{f}(w/a)/a|dw = \int |\widetilde{f}(u)|du < C$. Thus, the constant is invariant to time scaling and $f(ax) \in \Gamma_C$.

(c) Linear transformation (of time domain). Let $f \in \Gamma_C$ and A be an invertible matrix. Consider a function $g(x) = f(Ax)$. Then its Fourier transform will be given by the equation

$$\widetilde{g}(w) = \frac{1}{|\det A|}\widetilde{f}(A^{-T}w). \tag{20}$$

However, the determinant term disappears during the change of variables, when we try to find a bound for L_1-norm of $\widetilde{f}$. The determinant appears from Jacobian and then cancels out with the denominator. That means the transform $f(Ax) = g(x) \in \Gamma_C$. Keep in mind that Fourier transform itself is invariant only for orthogonal transformations.

(d) Summation. Let $f_1 \in \Gamma_{C_1}$ and $f_2 \in \Gamma_{C_2}$. Define $g(x) = f_1(x) + f_2(x)$. Then by the linearity of Fourier transform, $\widetilde{g}(w) = \widetilde{f}_1(w) + \widetilde{f}_2(w)$, and the L_1-norm is bounded by $C_1 + C_2$. Therefore, $g(x) \in \Gamma_{C_1+C_2}$.

(e) Linear combination. If $f_i \in \Gamma_{C_i}$ and $g = \sum \beta_i f_i$, then by the linearity of Fourier transform and the previous summation property, $\widetilde{g}(w) = \sum \beta_i \widetilde{f}_i(w)$. Consequently, $\int |\widetilde{g}(w)|dw \le \int \sum |\beta_i \widetilde{f}_i(w)|dw = \sum \beta_i \left(\int |\widetilde{f}_i(w)|dw \right)$. Therefore, $g(x) \in \Gamma_{\sum |\beta_i| C_i}$.

(f) Product. Let $f_1 \in \Gamma_{C_1}$ and $f_2 \in \Gamma_{C_2}$, that is, $\int |\widetilde{f}_1(w)|dw < C_1$ and $\int |\widetilde{f}_2(w)|dw < C_2$. If we define $g(x) = f_1(x) f_2(x)$, the by the convolution theorem [17, p. 92] we obtain

$$\begin{aligned} g(x) &= \int_{\mathbb{R}^d} e^{iw\cdot x}\widetilde{g}(w)dw \\ &= \int_{\mathbb{R}^d} e^{iw\cdot x} \left(\int_{\mathbb{R}^d} \widetilde{f}_1(z)\widetilde{f}_2(w-z)dz \right) dw. \end{aligned} \tag{21}$$

Then, $g(x) \in \Gamma_{C_1 C_2}$:

$$\begin{aligned} \int_{\mathbb{R}^d} |\widetilde{g}(w)|dw &= \int_{\mathbb{R}^d} \left| \int_{\mathbb{R}^d} \widetilde{f}_1(z)\widetilde{f}_2(w-z)dz \right| dw \\ &\le \int_{\mathbb{R}^d}\int_{\mathbb{R}^d} |\widetilde{f}_1(z)||\widetilde{f}_2(w-z)|dwdz \\ &= \int_{\mathbb{R}^d} |\widetilde{f}_1(z)| \left(\int_{\mathbb{R}^d} |\widetilde{f}_2(w-z)|dw \right) dz \\ &= \int_{\mathbb{R}^d} |\widetilde{f}_1(z)| \left(\int_{\mathbb{R}^d} |\widetilde{f}_2(w)|dw \right) dz \\ &= \left(\int_{\mathbb{R}^d} |\widetilde{f}_1(z)|dz \right) \left(\int_{\mathbb{R}^d} |\widetilde{f}_2(w)|dw \right) < C_1 C_2. \end{aligned} \tag{22}$$

(g) Composition with polynomials. We will use results for the combination and the product, mentioned above. Consider $f \in \Gamma_C$, then by the property of the product of functions, $f^k \in \Gamma_{C^k}$. Now, let $g(z)$ be a univariate polynomial function $g(f(x)) = \sum_{k=1}^n \beta_k (f(x))^k$. By the property of the linear combination of functions, we obtain $f(z) \in \Gamma_{\sum_{k=1}^n |\beta_k| C^k}$.

(h) Composition with analytic functions. Let $f(x) \in \Gamma_C$ and $g(z)$ be an analytic function $g(z) = \sum_{k=0}^\infty \beta_k z^k$, for which the radius of absolute convergence is $r > C$. By a similar approach as with polynomials, we get $g(z) \in \Gamma_{\sum_{k=0}^\infty |\beta_k| C^k}$. Here, the summation in the subscript of Γ is absolutely convergent since $C < r$.

(i) Ridge functions. Let $g(x) = f(a\cdot x)$ for some $a \in \mathbb{R}^d$ with $|a| = 1$, and $f(z)$ be a univariate function from Γ_C. Then

$$g(x) = f(a\cdot x) = \int_{\mathbb{R}} e^{ita\cdot x}\widetilde{f}(t)dt. \tag{23}$$

This can be viewed as a Fourier transform of $g(x)$, concentrated on the set of points $w \in \mathbb{R}^d$ on a line along the direction a. Then, the constant can be taken as $C_f = C_g = \int \widetilde{g}(t)dt$.

Note that sigmoidal functions of the form $f(x) = \phi(a \cdot x + b)$ do not have an integrable Fourier transform.

4.3 | Examples of functions in Γ_C

Let us now look at some examples of functions mentioned above.

4.3.1 | Gaussian function

The simplest example would be the Gaussian function $f(x) = e^{-\|x\|^2/2}$. It is in the class of both positive definite functions and Schwartz functions, and, therefore, $f(x) \in \Gamma_C$. Its Fourier transform is $\widetilde{f}(w) = (2\pi)^{-d/2} e^{-\|w\|^2/2}$ [18, p. 302]. Since the Fourier transform is a real and positive function of w,

$$\begin{aligned} C &= \int_{\mathbb{R}^d} F(dw) = \int_{\mathbb{R}^d} |\widetilde{f}(w)| dw \\ &= \int_{\mathbb{R}^d} \widetilde{f}(w) dw = \int_{\mathbb{R}^d} (2\pi)^{-d/2} e^{-\|w\|^2/2} dw \\ &= (2\pi)^{-d/2} \int_{\mathbb{R}^d} e^{-\|w\|^2/2} dw = (2\pi)^{-d/2} (2\pi)^{d/2} = 1 \end{aligned}$$

Alternatively, we can use the fact that the function is positive definite (proving this fact is hard in practice). We evaluate the Gaussian at the origin and get $C = f(0) = 1$. Shifted versions of Gaussian will also be in the class Γ_C, according to the translation property. Therefore, we include functions of the form $f(x) = e^{-\|x-\mu\|^2/2}$, where $\mu \in \mathbb{R}^d$ is a shift.

4.3.2 | Multivariate Gaussian distributions

These functions arise in almost every field of probability and statistics. Multivariate Gaussians are also in the same Schwartz space: this can be shown by a change of variables (first, use shift, then a linear transformation). However, we now want to find the constant C for this function from our observations above. The multivariate Gaussian is given by the formula [19, p. 197]:

$$f(x) = \frac{1}{(2\pi)^{d/2} (\det \Sigma)^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}, \tag{24}$$

where Σ is a symmetric positive definite matrix (covariance matrix), μ is shift in x and is the mean of distribution.

First, we can define a matrix $A = \Sigma^{-1/2} = Q^T D^{-1/2} Q$, so that $A^T A = \Sigma^{-1}$. The matrix Q is orthogonal, A is symmetric positive definite and invertible. The function now is of the form

$$\begin{aligned} f(x) &= \frac{1}{(2\pi)^{d/2} (\det \Sigma)^{1/2}} e^{-\frac{1}{2}(x-\mu)^T A^T A (x-\mu)} \\ &= \frac{\det A}{(2\pi)^{d/2}} e^{-\frac{1}{2}[A(x-\mu)]^T [A(x-\mu)]} \\ &= \frac{\det A}{(2\pi)^{d/2}} e^{-\frac{\|A(x-\mu)\|^2}{2}} = \frac{\det A}{(2\pi)^{d/2}} g(A(x-\mu)), \end{aligned} \tag{25}$$

where $g(y) = e^{-\|y\|^2/2}$, for which $\widetilde{g}(u) = (2\pi)^{-d/2} e^{-\|u\|^2/2}$. Using the result for the transformation in time domain and the translation property, the Fourier transform of the latter is

$$\begin{aligned} \widetilde{f}(w) &= \frac{\det A}{(2\pi)^{d/2}} \frac{1}{|\det A|} e^{i\mu \cdot w} \widetilde{g}(A^{-T} w) \\ &= \frac{1}{(2\pi)^{d/2}} e^{i\mu \cdot w} \widetilde{g}(A^{-T} w) \end{aligned} \tag{26}$$

L_1-norm of $\widetilde{f}$ is then bounded by

$$\begin{aligned} \int_{\mathbb{R}^d} |\widetilde{f}(w)| dw &\le \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} |\widetilde{g}(A^{-T} w)| dw \\ &= \frac{\det A}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} |\widetilde{g}(u)| du = \frac{1}{(2\pi)^{d/2} (\det \Sigma)^{1/2}}. \end{aligned} \tag{27}$$

The constant then can be taken as $C = \frac{1}{(2\pi)^{d/2} (\det \Sigma)^{1/2}}$. Multivariate Gaussian distributions are then in the class $\Gamma_{1/[(2\pi)^{d/2} (\det \Sigma)^{1/2}]}$. Note that C is a reciprocal of the normalizing constant. Similar results should be true for other probability densities.

4.3.3 | Gaussian mixture models

Another important function in probability, statistics and machine learning. Gaussian mixture can be defined as a linear combination of different Multivariate Gaussians [20, p. 425]:

$$f(x) = \sum_{k=1}^{n} a_k f_k(x; \mu_k, \Sigma_k), \tag{28}$$

where each f_k is a multivariate Gaussian with a different mean (μ_k) and a different covariance matrix (Σ_k), a_k's are corresponding weights in the mixture for which $\sum a_k = 1$ and $a_k > 0$. Using the linear combination property, we can verify that the constant $C = \sum a_k C_i$, where $C_i = 1/[(2\pi)^{d/2} (\det \Sigma)^{1/2}]$.

Many other functions can be derived from the Gaussian, using the properties we have obtained. We will use

some of them in the next section to assess the approximability of functions in Γ by a Silvescu Fourier Neural Network.

5 | EXPERIMENTS

5.1 | Generating data for functions in Γ_C

For the experimental part, we fix the dimension size $d = 3$. The reason is due to enormous error bound that we obtain by setting dimension too large. However, our goal is to assess if the convergence is of order $O(1/n)$, where n is the number of hidden neurons. Therefore, we will test functions obtained in the previous section on different values of n. Namely, $n = \{1,4,16,64,256,1024\}$. Regarding the classes of functions in Γ_C, the following functions were chosen:

1. Gaussian

   $f(x) = e^{-\frac{1}{2}\|x\|^2}$.

   $C = 1$.

2. Translated and scaled (dilated) Gaussian

   $f(x) = e^{-\frac{1}{2}\|ax+b\|^2}$.

   $C = 1$.

3. Gaussian, linearly transformed in time argument

   $f(x) = e^{-\frac{1}{2}\|Ax\|^2}$.

   $C = 1$.

4. Linear combination of Gaussians

   $f(x) = a_1 e^{-\frac{1}{2}\|x\|^2} + a_2 e^{-\frac{1}{2}\|Ax\|^2}$.

   $C = 1$.

5. Composition of Gaussian and a polynomial

   $$f(x) = a_1[g(x)]^1 + a_2[g(x)]^2 + a_3[g(x)]^4 + a_4[g(x)]^4,$$

   where $g(x) = e^{-\frac{1}{2}\|x\|^2}$. $C = \sum |a_k|$.

6. Composition of Gaussian and analytic function cosine

   $f(x) = cos\left(e^{-\frac{1}{2}\|x\|^2}\right)$.

   $C = \sum 1/(2k)! = \cosh(1)$.

7. Ridge function

   $f(x) = e^{-(a\cdot x)^2/2}$,

   $C = 1$.

8. Multivariate Gaussian (normal) distribution

   $$f(x) = \frac{1}{(2\pi)^{d/2}(\det\Sigma)^{1/2}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)} = \mathcal{N}(\mu,\Sigma).$$

   $$C = \frac{1}{(2\pi)^{d/2}(\det\Sigma)^{1/2}}.$$

9. Gaussian mixture model

   $f(x) = a_1\mathcal{N}(\mu_1,\Sigma_1) + a_2\mathcal{N}(\mu_1,\Sigma_1)$.

   $C = \sum a_k C_i$, where $C_i = 1/[(2\pi)^{d/2}(\det\Sigma)^{1/2}]$.

10. Silvescu activation function

    $f(x) = \prod \cos(w_j x_j + b_j)$

    It was also interesting to check if the network can approximate the Silvescu activation function adequately.

5.2 | Software

All parameters such as constants, coefficients, mean vectors and covariance matrices were chosen randomly. Then, the datasets with training size 10,000 and test size 5,000 were generated according to chosen functions. A small normal error $\mathcal{N}(0,0.03)$ was added on top. The code was written with the help of Python's NumPy [21] library.

Regarding the implementation for Silvescu FNN, computations were run using TensorFlow library [22]. The network used Adam Optimizer [23] for solving a minimization problem, which is a Stochastic Gradient Descent (SGD) algorithm with momentum and an adaptive learning rate. The momentum fastens the convergence by adding a portion of a previous gradient to a current one. With the momentum and the adaptive learning rate combined, the algorithm is less likely to slow down when the gradient approaches zero around a minimum. An initial learning rate was set to $\alpha = 0.01$, the momentum constants were set to default $\beta_1 = 0.99$ and $\beta_2 = 0.999$.

Since the implementation uses SGD, we needed to divide the dataset into batches. For a batch size 100 and 10,000 observations in total, we obtained 100 batches. The network training was 50 epochs long (50 iterations over the same dataset). After each epoch (=100 processed batches), the data was shuffled to avoid repetitions. Then, the minimum mean squared error (MSE) across the epochs was chosen. The reason why we use the minimum values for MSE is that our main theorem proved the existence of network with indicated MSE.

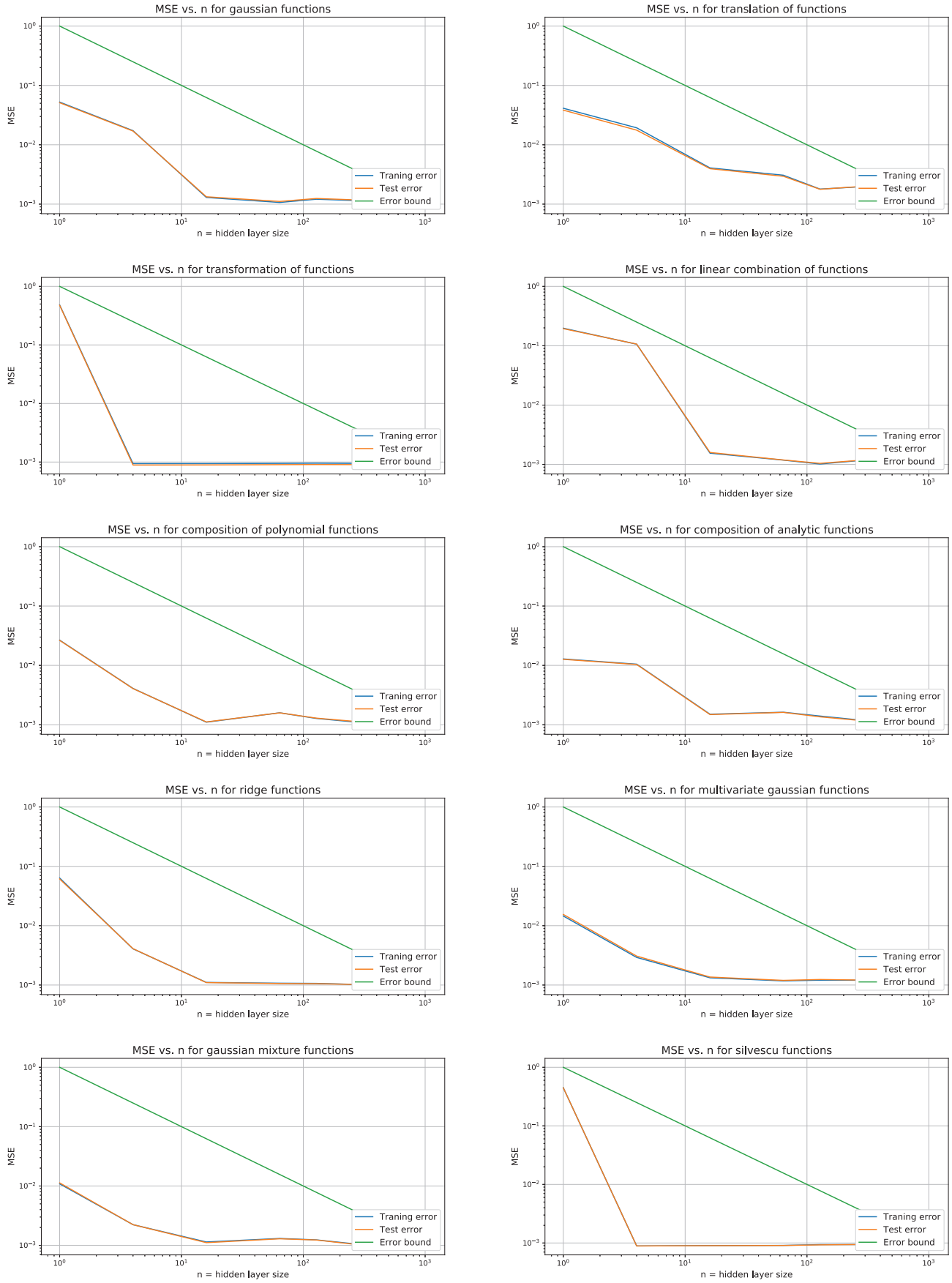


FIGURE 2 MSEs for different functions

5.3 | Hardware

The TensorFlow package enables to use Graphics Processing Unit (GPU) acceleration for training neural networks. This requires to have GPU's that support Nvidia's CUDA Toolkit for accelerated computing. For running the Python code, we used Nvidia's GeForce GTX 1050 Ti. The computations for Silvescu FNNs on average were about 30% slower than for standard feedforward neural nets with sigmoid activation function. This is due to a larger number of floating point operations in Silvescu's activation ($\approx 30d$ flops) with respect to sigmoid activation ($\approx 2d + 25$ flops). It is obvious that with the increase of dimensionality of the problem that gap is expected to rise, too.

5.4 | Results

The results of the experimental part were good for small values of n (number of neurons), in the interval $16 < n < 1024$ the error leveled off, for larger values the error is expected to be above the indicated error bound. Although the theoretical results provide an upper bound for the error, there has to be also a lower bound due to computational reasons. According to graphs (see Figure 2), the error decay may be described as linear for $n \leq 16$, that is, order of $O(1/n)$, whereas on the right tail of MSEs, it is best described as $O(1)$. However, this possibly can be explained by a large number of neurons, and consequently, large number of parameters. For Silvescu NN we have $(2d + 1)n = 7n$ parameters. This potentially can cause over-parametrization during training without the regularization. Another reason could be existence of a lower bound for an approximation by Sivlsecu FNN. This is also probable, because the error stops decaying around 10^{-3}, which is even smaller than the error for our generated functions ($\varepsilon = 0.03$). Then, small fluctuations in the right tail can be explained by Adam Optimizer finding different local minima during training stages. Because we redefine the network each time we change the hidden layer size n.

6 | CONCLUSION

Theoretically, we have proven that there exist such parameters of FNN, for which the network's error is bounded by a constant divided by the number of hidden neurons. However, the existence of such parameters does not imply that the bound is the same for all trained models. This is the best-case scenario, and obtaining such solution is not guaranteed in practice. Main experimental concern would be an algorithm for solving the minimization problem. Additional regularization techniques and constraining network weights as in Section 3 could help to mitigate these challenges. Nevertheless, the importance of this paper is in obtaining error bounds for networks, which fall out of the paradigm of "weighted sum of activation of a linearly transformed input". All standard models were described by Hornik [10] and Cybenko [3], and Barron [2] provided lower (Kolmogorov n-width) and upper bound for approximation by sigmoidal neural networks. Silvescu FNN, on the other hand, has a non-trivial activation, which makes it harder to analyze. The constant term 4^{d-1} in the error bound appears also because of the way the activation was constructed. To counter-balance this term, the number of neurons n has to be exponential in the dimensionality d.

First possible improvement of the work could be extending the results for linear and polynomial functions. For these two functions, functions defined on integers and function on other bounded domains, Barron used suitable spline extrapolations to satisfy his assumption $\int |w||\widetilde{f}(w)|dw < C$ [2]. This could help us considering more real-life data with such nature. The lower bound for an approximation can be derived. Barron obtains it by considering Kolmogorov's n-width, the closest distance between the function and a collection of basis functions. More research in this direction could shed a light on the analysis of the approximation by linear spans of Silvescu functions.

ACKNOWLEDGMENTS

This work was supported by the Nazarbayev University faculty-development competitive research grants program, Grant Number 240919FD3921, and by the Committee of Science of the Ministry of Education and Science of the Republic of Kazakhstan, IRN AP05133700.

DATA AVAILABILITY

The data that support the findings of this study are openly available at https://github.com/zh3nis.

ORCID

Zhenisbek Assylbekov https://orcid.org/0000-0003-0095-9409

REFERENCES

1. A. Silvescu, Fourier neural networks. IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339), 1999, IEEE, vol. 1, pp. 488–491.
2. A. R. Barron, *Universal approximation bounds for superpositions of a sigmoidal function*, IEEE Trans. Inf. Theory 39(3) (1993), 930–945.

3. G. Cybenko, *Approximation by superpositions of a sigmoidal function*, Math. Control Signals Syst. 2(4) (1989), 303–314.
4. K. Hornik, *Approximation capabilities of multilayer feedforward networks*, Neural Netw. 4(2) (1991), 251–257.
5. R. Eldan and O. Shamir, *The power of depth for feedforward neural networks*, Conf. Learn. Theory. 49 (2016), 907–940.
6. A. R. Gallant and H. White, There exists a neural network that does not make avoidable mistakes. *Proceedings of the Second Annual IEEE Conference on Neural Networks, San Diego, CA, I*, 1988.
7. H. Tan, *Fourier neural networks and generalized single hidden layer networks in aircraft engine fault diagnostics*, J. Eng. Gas Turbines Power 128(4) (2006), 773–782.
8. W. Zuo and L. Cai, *Adaptive-fourier-neural-network-based control for a class of uncertain nonlinear systems*, IEEE Trans. Neural Netw. 19(10) (2008), 1689–1701.
9. S. Liu, Fourier neural network for machine learning. *2013 International Conference on Machine Learning and Cybernetics*, IEEE, vol. 1, 2013, pp. 285–290.
10. K. Hornik, M. Stinchcombe, and H. White, *Multilayer feedforward networks are universal approximators*, Neural Netw. 2(5) (1989), 359–366.
11. L. K. Jones et al., *A simple lemma on greedy approximation in hilbert space and convergence rates for projection pursuit regression and neural network training*, Ann. Stat. 20(1) (1992), 608–613.
12. J. Buescu and A. Paixao, *Real and complex variable positive definite functions*, São Paulo J. Math. Sci. 6(2) (2012), 155–169.
13. P. D. Lax, *Functional analysis*, John Wiley & Sons, New York, NY, 2002.
14. M. Elias and R. S. Stein, *Fourier analysis: An introduction*, Princeton University Press, Princeton, NJ, 2003.
15. G. B. Folland, *Fourier analysis and its applications*, Vol 4, American Mathematical Society, Providence, RI, 2009.
16. J.-P. Montillet, *Sobolev spaces, schwartz spaces, and a definition of the electromagnetic and gravitational coupling*, J. Mod. Phys. 8 (2017), 1700–1722.
17. M. A. Pinsky, *Introduction to Fourier analysis and wavelets*, Vol 102, American Mathematical Soc, Providence, RI, 2008.
18. M. Abramowitz and I. A. Stegun, *Handbook of mathematical functions with formulas, graphs, and mathematical tables*, Vol 55, US Government Printing Office, Washington, D.C., 1948.
19. M. H. Kutner, *Applied linear statistical models*, 5nd ed., McGraw-Hill Education, New York, NY, 2005.
20. J. Benesty, M. M. Sondhi, and Y. Huang, *Springer handbook of speech processing*, Springer, Berlin, Germany, 2007.
21. T. Oliphant, *NumPy: A guide to NumPy*. Trelgol Publishing, Austin, TX, 2006.
22. M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. *12th {USENIX} Symposium on Operating Systems Design and Implementation {OSDI}*, vol. 16, 2016, pp. 265–283.
23. D. P. Kingma and J. Ba, Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

How to cite this article: Zhumekenov A, Takhanov R, Castro AJ, Assylbekov Z. Approximation error of Fourier neural networks. *Stat Anal Data Min: The ASA Data Sci Journal*. 2021;14:258–270. https://doi.org/10.1002/sam.11506