Bicomplex neural networks with hypergeometric activation functions*

N. Vieira[‡]

[‡]CIDMA - Center for Research and Development in Mathematics and Applications
Department of Mathematics, University of Aveiro
Campus Universitário de Santiago, 3810-193 Aveiro, Portugal.
Email: nloureirovieira@gmail.com

March 14, 2023

Abstract

Bicomplex convolutional neural networks (BCCNN) are a natural extension of the quaternion convolutional neural networks for the bicomplex case. As it happens with the quaternionic case, BCCNN has the capability of learning and modelling external dependencies that exist between neighbour features of an input vector and internal latent dependencies within the feature. This property arises from the fact that, under certain circumstances, it is possible to deal with the bicomplex number in a component-wise way. In this paper, we present a BCCNN, and we apply it to a classification task involving the colorized version of the well-known dataset MNIST. Besides the novelty of considering bicomplex numbers, our CNN considers an activation function a Bessel-type function. As we see, our results present better results compared with the one where the classical ReLU activation function is considered.

Keywords: Artificial neural networks and deep learning; Activation functions; Bicomplex convolutional neural networks; Hypergeometric functions; Bessel functions

MSC 2010: 68T07; 68Q32; 30G35; 33C90; 33C10.

1 Introduction

Convolutional Neural Networks (CNN) are one of the most used tools in artificial intelligence. In recent years, they became a key tool in numerous fields like image classification, face recognition and machine translation (see [13] and references therein). The correct choice of the activation function can significantly affect the performance of CNN. Sometimes, a chosen activation function does not possess all the necessary properties/characteristics for a specific CNN. Usually, the process of selection of activation functions is manual and relies essentially on the architecture of the Neural Network (NN), which leads to exhaustive trial-and-error methodologies, where the NN is retrained for each activation function until the optimal configuration. Independently of the considered approach, it is clear that the task of introducing new activation functions is, for sure, not easy and its benefits are sometimes limited. This limitation comes from the fact that the several proposals presented in the literature reveal to be inconsistent and task-dependent, and therefore the classical activation functions (for example, the *ReLU*) maintain their predominance in practical applications.

The field of quaternions is the best-known extension of the field of complex numbers to the four-dimensional setting. On one hand, one of the main advantages of this extension is that quaternions form a field where we can consider all the customary operations. On the other hand, when we handle quaternions, we do not have commutativity of the product, which brings several unwanted problems in the process of extension of the theory of holomorphic functions for one complex variable. A possible way to overcome the issue of non-commutativity is to consider the bicomplex numbers, which is a four-dimensional algebra with classical operations, that contains $\mathbb{C}$ as a subalgebra

*The final version is published in *Advances in Applied Clifford Algebras*, **33**-No.22, (2023), Article No.20 (14pp.). It as available via the website https://link.springer.com/article/10.1007/s00006-023-01268-w

and preserves the commutativity. Just as in the case of complex numbers, where the individual components of the complex number can be treated independently as two real numbers, the bicomplex space can be treated as two complex or four real numbers.

The bicomplex algebra can be applied in several fields. For example, in [18], the authors point out that the so-called IHS-colour space representation (i.e., Intensity-Hue-Saturation) has broad applications, particularly in human vision, can be mathematically represented by having values in bicomplex numbers. In [9] the authors take vantage of the idempotent representation of bicomplex algebra to prove the possibility of reconstructing a bicomplex sparse signal, with high probability, from a reduced number of bicomplex random samples. Moreover, the possibility of having four separate components allows the consideration of three- and four-dimensional input feature vectors, which are associated with image processing and robot kinematics [6, 21], and also with the field of capsule networks [20]. Thereby, a bicomplex neural network-based models are able to code latent interdependencies between groups of input features during the learning process with fewer parameters than traditional NN, by taking advantage of the component-wise multiplication that can be established between bicomplex numbers. The bicomplex approach gives a better way of combining two-valued complex networks compared to the usual two-valued approach in the literature [3], as we can treat the theory as a single variable one due to its algebra structure. In our approach, the structure of the bicomplex algebra is essential in providing a convergent algorithm. For more details about the theory of bicomplex numbers and its applications we refer the interested reader to [4, 7, 10, 12].

The objective of this paper is to introduce a BCCNN where a generic activation function of Bessel-type is considered. The purpose of our idea is to generalize the results presented in the literature for the quaternionic case and to present an application of the novel activation functions of hypergeometric type introduced in [25]. The structure of the paper reads as follows: Section 2 is dedicated to recalling some basic definitions of the bicomplex numbers and special functions, which are necessary for the understanding of this work. In Section 3 based on the ideas presented in [25] we introduce a general family of hypergeometric activation functions with several trainable parameters. In the following section, we describe our BCCNN. We perform some numerical experiments in the last section using the Colored MNIST, to validate our approach.

2 Preliminaries

2.1 Bicomplex algebra

The set $\mathbb{BC}$ of bicomplex numbers is defined by

$$\mathbb{BC} := \{z_1 + \mathbf{j}z_2 : \ z_1, z_2 \in \mathbb{C}\},$$

where $\mathbb{C}$ is the set os complex numbers with the imaginary unit $\mathbf{i}$, and where $\mathbf{i}$ and $\mathbf{j}$ are commuting imaginary units, i.e., $\mathbf{ij} = \mathbf{k} = \mathbf{ji}$, $\mathbf{i}^2 = \mathbf{j}^2 = -1$, and $\mathbf{k}^2 = +1$. Bicomplex numbers can be added and multiplied, in fact, if $Z = z_1 + \mathbf{j}z_2$ and $W = w_1 + \mathbf{j}w_2$ are two bicomplex numbers, we have that

$$Z + W := (z_1 + w_1) + \mathbf{j}(z_2 + w_2), \qquad Z \cdot W := (z_1w_1 - z_2w_2) + \mathbf{j}(z_1w_2 + z_2w_1),$$

where the product is not component-wise. From the previous multiplication rules involving $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$, we decompose $\mathbb{BC}$ in two ways. Indeed, $\mathbb{BC} = \mathbb{C}(\mathbf{i}) + \mathbf{j}\mathbb{C}(\mathbf{i})$, where

$$\mathbb{C}(\mathbf{i}) = \{Z = z_1 + \mathbf{j}z_2 : \ z_2 = 0\}.$$

Likewise, we have $\mathbb{BC} = \mathbb{C}(\mathbf{j}) + \mathbf{i}\mathbb{C}(\mathbf{j})$. Moreover, if we consider $z_1 = x_1 + \mathbf{i}y_1$ and $z_2 = x_2 + \mathbf{i}x_2$, with $x_1, y_1, x_2, y_2 \in \mathbb{R}$, we have the following alternative form of presenting a bicomplex number

$$Z = z_1 + \mathbf{j}z_2 = x_1 + \mathbf{i}y_1 + \mathbf{j}x_2 + \mathbf{k}y_2.$$

The structure of $\mathbb{BC}$ suggest three possible conjugations on $\mathbb{BC}$:

- *the bar-conjugation*: $\overline{Z} = \overline{z}_1 + \mathbf{j}\overline{z}_2$;
- *the* $\dagger$*-conjugation*: $Z^\dagger = z_1 - \mathbf{j}z_2$;
- *the* $*$*-conjugation*: $Z^* = \overline{Z^\dagger} = \left(\overline{Z}\right)^\dagger = \overline{z}_1 - \mathbf{j}\overline{z}_2$,

where $\overline{z}_1$ and $\overline{z}_2$ are usual complex conjugates to $z_1, z_2 \in \mathbb{C}(i)$. The euclidian norm of a bicomplex number Z on $\mathbb{BC}$, when it is seen as

$$\mathbb{C}^2(\mathbf{i}) := \mathbb{C}(\mathbf{i}) \times \mathbb{C}(\mathbf{i}) := \{(z_1, z_2) : z_1 + \mathbf{j}z_2 \in \mathbb{BC}\}$$

or as

$$\mathbb{R}^4 = \{(x_1, y_1, x_2, y_2) : (x_1 + \mathbf{i}y_1) + \mathbf{j}(x_2 + \mathbf{i}y_2) \in \mathbb{BC}\}$$

is given by

$$\|Z\| = \sqrt{|z_1|^2 + |z_2|^2} = \sqrt{x_1^2 + y_1^2 + x_2^2 + y_2^2}. \tag{1}$$

The bicomplex space, $\mathbb{BC}$, is not a division algebra, and it has two distinguished zero divisors, namely, $\mathbf{e}_1$ and $\mathbf{e}_2$, which are idempotent, linearly independent over reals, and mutually annihilating with respect to the bicomplex multiplications (see [12, Prop. 1.6.1]):

$$\mathbf{e}_1 := \frac{1+\mathbf{k}}{2}, \qquad \mathbf{e}_2 := \frac{1-\mathbf{k}}{2}, \qquad \mathbf{e}_1 \cdot \mathbf{e}_2 = 0, \qquad \mathbf{e}_1^2 = \mathbf{e}_1, \qquad \mathbf{e}_2^2 = \mathbf{e}_2, \qquad \mathbf{e}_1 + \mathbf{e}_2 = 1, \qquad \mathbf{e}_1 - \mathbf{e}_2 = \mathbf{k}.$$

We have that $\{\mathbf{e}_1, \mathbf{e}_2\}$ form an idempotent basis of the complex algebra $\mathbb{BC}$. Considering the complex numbers $\beta_1 := z_1 - \mathbf{i}z_2$ and $\beta_2 := z_1 + \mathbf{i}z_2$ in $\mathbb{C}(\mathbf{i})$, we have that the *idempotent representation* of $Z = z_1 + \mathbf{j}z_2$ in $\mathbb{BC}(\mathbf{i})$ is given by $Z = \beta_1\mathbf{e}_1 + \beta_2\mathbf{e}_2$. This idempotent representation is the only representation for which multiplication is component-wise, as it is indicated in the next proposition

Proposition 2.1 *(cf. [12, Prop. 1.6.3]) The addition and multiplication of bicomplex numbers can be realized component-wise in the idempotent representation presented previously. Specifically, if $Z = a_1\mathbf{e}_1 + a_2\mathbf{e}_2$ and $W = b_1\mathbf{e}_1 + b_2\mathbf{e}_2$ are two bicomplex numbers, where $a_1, a_2, b_1, b_2 \in \mathbb{C}(\mathbf{i})$, then*

$$Z + W = (a_1 + b_1)\mathbf{e}_1 + (a_2 + b_2)\mathbf{e}_2, \qquad Z \cdot W = (a_1 b_1)\mathbf{e}_1 + (a_2 b_2)\mathbf{e}_2, \qquad Z^n = a_1^n\mathbf{e}_1 + a_2^2\mathbf{e}_2.$$

The multiplicative inverse of a bicomplex number $Z = a_1\mathbf{e}_1 + a_2\mathbf{e}_2$, with $a_1 \cdot a_2 \neq 0$ is given by $Z^{-1} = a_1^{-1}\mathbf{e}_1 + a_2^{-1}\mathbf{e}_2$, where a_1^{-1} and a_2^{-1} are the complex multiplicative inverses of a_1 and a_2, respectively (see [12, Thm. 1.6.5]). A bicomplex number may be identified with real 4×4 matrices (that turns out to be more suitable for computations):

$$\varphi_{\mathbb{R}} : Z = x_1 + \mathbf{i}y_1 + \mathbf{j}x_2 + \mathbf{k}y_2 \in \mathbb{BC} \mapsto \begin{pmatrix} x_1 & -y_1 & -x_2 & y_2 \\ y_1 & x_1 & -y_2 & -x_2 \\ x_2 & -y_2 & x_1 & -y_1 \\ y_2 & x_2 & y_1 & x_1 \end{pmatrix}. \tag{2}$$

We have that every 4×4 matrix determines a linear (more exactly, a real linear) transformation in $\mathbb{R}^4$, however, not all of them remain $\mathbb{BC}$-linear when $\mathbb{R}^4$ is seen as $\mathbb{BC}$. In the context of bicomplex convolutional neural networks, there are some activation functions proposed in the literature involving bicomplex numbers. For example, in [4] the authors considered the activation function $P(z) = \epsilon^l$ in $T \subset \mathbb{C}^n$, where $\epsilon = \exp\left(\frac{2\pi i}{k}\right)$ be the root of the unity of order k, and whenever $\frac{2\pi i l}{k} \leq \arg(z) < \frac{2\pi i(l+1)}{k}$.

2.2 Special functions

In this work, we make use of the generalized hypergeometric function ${}_pF_q$, which is defined by (see [19])

$${}_pF_q(a_1, \ldots, a_p; b_1, \ldots, b_q; z) = {}_pF_q\left((a_j)_{1:p}; (b_i)_{1:q}; z\right) = \sum_{l=0}^{+\infty} \frac{\prod_{j=1}^{p}(a_j)_l}{\prod_{i=1}^{q}(b_i)_l} \frac{x^l}{l!}, \tag{3}$$

where the convergence is guaranteed if one of the following conditions is satisfied:

$$p \leq q \quad \vee \quad q = p-1 \wedge |z| < 1 \quad \vee \quad q = p-1 \wedge \operatorname{Re}\left(\sum_{i=1}^{p-1} b_i - \sum_{j=1}^{p} a_j\right) > 0 \wedge |z| = 1. \tag{4}$$

Moreover, (3) is an analytical function of $a_1, \ldots, a_p$, $b_1, \ldots, b_q$ and z which is defined in $\mathbb{C}^{p+q+1}$. In the cases $p \leq q$ for fixed $a_1, \ldots, a_p$, $b_1, \ldots, b_q$, it is an entire function of z. If parameters a_k include negative integers, the function (3) degenerates to a polynomial in z.

Another special function that will play an important role in this work is the Bessel function of the first kin J_ν, which is defined by the following series (see [1])

$$J_\nu(z) = \sum_{k=0}^{+\infty} \frac{(-1)^k}{\Gamma(k+\nu+1)\, k!} \left(\frac{z}{2}\right)^{2k+\nu}. \tag{5}$$

The Bessel function is related to the hypergeometric function ${}_0F_1$ by the following relation (see [1])

$$J_\nu(z) = \frac{1}{\Gamma(1+\nu)} \left(\frac{z}{2}\right)^{\nu} {}_0F_1\left(;\, 1+\nu;\, -\frac{x^2}{4}\right), \tag{6}$$

when ν is a non-negative integer. For more details about hypergeometric functions and other special functions, we refer, for example, to [1].

3 General activation functions

The correct choice of the activation function can significantly affect the performance of CNN. Sometimes, a chosen activation function does not possess all the necessary properties/characteristics for a specific CNN. Usually, the process of selection of activation functions is manual and relies essentially on the architecture of the NN, which leads to exhaustive trial-and-error methodologies, where the NN is retrained for each activation function until the optimal configuration.

In this sense, and following the ideas presented in [25], we consider in our CNN a general multi-parametric activation function in the context of automatic activation function design. The concepts of parametric activation functions presented in [8], and the adaptative activation functions introduced in [26] inspired our work, and were adapted to deal with hypergeometric functions. More precisely, we consider the following activation function:

$$\mathcal{H}(x) = c_1 + c_2\, x^{c_3} + c_4\, x^{c_5}\, {}_pF_q\left((a_j)_{1:p}\,;\, (b_i)_{1:q}\,;\, c_6\, x^{c_7}\right), \tag{7}$$

where $c_1, c_2, c_4, c_6 \in \mathbb{R}$, $c_5 \in \mathbb{N}_0$, $c_3, c_7 \in \mathbb{N}$, and the parameters in the hypergeometric function satisfy (4). Due to a large number of parameters, it is possible to use (7) to approximate every continuous function on a compact set. Moreover, in the case where the convergence is guaranteed, it is possible to define sub-ranges of the several parameters that appear in (7) in order that the elements of the proposed class have some desirable properties that are useful for the role of the activation function.

The multi-parametric activation function (7) groups several of the standard activation functions proposed in the literature for deep NN. In Table 1, we indicate which cases are included in (7):

General Activation Functions $\mathcal{H}(\mathbf{x})$	***Activation Function Inside the Class***	Identity	Binary Step	ReLU	GELU
		ELU	SELU	PReLU	Inverse Tangent
		Softsing	Bent Identity	Sinusoid	Sinc
		Gaussian			
	Activation Functions Outside the Class	Sigmoid	Hyperbolic Tangent	SoftPlus	SNQL
		SiLu			

Table 1: General activation function $\mathcal{H}(x)$.

Moreover, in [25] is indicated in a detailed form how we derive the activation functions indicated in Table 1 from the general expression (7). For example, if we consider $c_1 = c_4 = 0$, $c_2 = 0$ (for $x < 0$) or $c_2 = 1$ (for $x \geq 0$),

$c_3, c_5, c_7 \in \mathbb{N}$, $c_6 \in \mathbb{R}$, and $p = q = 0$, we obtain

$$\mathcal{H}(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}, \tag{8}$$

which corresponds to the classical *Rectified linear unit (ReLU)*.

Let us now pay attention to a particular case of (7), that involves the Bessel function of the first kind $J_\nu(x)$, with ν a half-positive integer. In fact, if we consider

$$c_1 = c_2 = 0,\ c_4 = \sqrt{\frac{\pi}{2}}\frac{2^{-\nu}}{\Gamma(1+\nu)},\ c_5 = 2\nu,\ p = 0,\ q = 1\ (b_1 = 1+\nu),\ c_6 = -\frac{1}{4},\ c_7 = 2, \tag{9}$$

in (7), we obtain

$$\mathcal{H}(x) = \frac{2^{-\nu}}{\Gamma(1+\nu)}\, x^{2\nu}\ {}_0F_1\left(-;\, 1+\nu;\, -\frac{x^2}{4}\right) = \sqrt{\frac{\pi}{2}}\, x^\nu\, J_\nu(x), \tag{10}$$

which corresponds to a one-parameter activation function. It follows from the properties of the Bessel function of the first kind that for half-integers values ν the activation function (10) reduces to the combination of polynomials and elementary trigonometric functions such as $\sin$ and $\cos$. In fact, for the first four positive half-integers, we have that

$$\nu = \frac{1}{2} \ \Rightarrow\ \mathcal{H}(x) = \sqrt{\frac{\pi}{2}}\, x^{\frac{1}{2}} J_{\frac{1}{2}}(x) = \sin(x), \tag{11}$$

$$\nu = \frac{3}{2} \ \Rightarrow\ \mathcal{H}(x) = \sqrt{\frac{\pi}{2}}\, x^{\frac{3}{2}} J_{\frac{3}{2}}(x) = \sin(x) - x\cos(x), \tag{12}$$

$$\nu = \frac{5}{2} \ \Rightarrow\ \mathcal{H}(x) = \sqrt{\frac{\pi}{2}}\, x^{\frac{5}{2}} J_{\frac{5}{2}}(x) = -\left(x^2-3\right)\sin(x) - 3x\cos(x), \tag{13}$$

$$\nu = \frac{7}{2} \ \Rightarrow\ \mathcal{H}(x) = \sqrt{\frac{\pi}{2}}\, x^{\frac{7}{2}} J_{\frac{7}{2}}(x) = 3\left(5-2x^2\right)\sin(x) + x\left(x^2-15\right)\cos(x). \tag{14}$$

In spite of (10) (and also (11)-(14)) not being monotonic functions in all the positive real line, we can restrict our activation functions to intervals of the form $I_\nu = [0; M_\nu]$, where M_ν is the first positive zero of the Bessel function $J_{\nu-1}(x)$ and corresponds to the first maximum positive point of $J_\nu(x)$. In order to improve our results (see [25]), we consider from now on the following linear combination of (11)-(14)

$$\mathcal{B}(x) = \sqrt{\frac{\pi}{2}}\left[\beta_1\sqrt{x}J_{\frac{1}{2}}(x) + \beta_2\sqrt{x^3}\,J_{\frac{3}{2}}(x) + \beta_3\sqrt{x^5}\,J_{\frac{5}{2}}(x) + \beta_4\sqrt{x^7}\,J_{\frac{7}{2}}(x)\right], \tag{15}$$

i.e., we combine the Bessel functions (11)-(14) using trainable parameters to dynamically change how much the contribution of each Bessel function to the final activation function.

4 Bicomplex convolutional neural network

In this section, we define our BCCNN and appropriate parameter initialization. We can understand the BCCNN as a generalization of the quaternionic convolutional neural network (QCNN) (see [15–17]) and of the classical real-valued deep CNN (see [11]) to the case we deal with bicomplex numbers. Taking into account [16, 17, 23] about CNN via quaternions and the theory of bicomplex numbers [12] we have that the bicomplex convolution operation is performed via the real-number matrix representation (2). Hence, the one-dimensional convolutional layer, with a kernel that contains featured maps, is split into 4 parts: the first part equal to x_1, the second one to $\mathbf{i}y_1$, the third one to $\mathbf{j}x_2$, and the last to $\mathbf{k}y_2$ of a bicomplex number $Z = x_1 + \mathbf{i}y_1 + \mathbf{j}x_2 + \mathbf{k}y_2$.

For the activation function, we consider a combination of the so-called split activation introduced in [24] for the quaternionic case with the real-valued activation function (15) defined in terms of Bessel functions, i.e.,

$$\mathcal{F}(Z) = \mathcal{B}(x_1) + \mathbf{i}\mathcal{B}(y_1) + \mathbf{j}\mathcal{B}(x_2) + \mathbf{k}\mathcal{B}(y_2). \tag{16}$$

Taking into account the properties of the Bessel functions and the ideas presented in [4] we can introduce the concept of threshold function associated to our activation function (16).

Definition 4.1 *Let $n \geq 1$ and $T \subset \mathbb{C}^2(\mathbf{i})$. Then, a complex valued function $f : T \to \mathbb{C}$ is called a* threshold function *if there exists a weighting vector $W = (w_0, w_1, w_2)$, where $w_i \in \mathbb{C}(\mathbf{i})$ such that:*

$$f(z_1, z_2) = \mathcal{F}(w_0 + w_1 z_1 + w_2 z_2), \qquad z_1, z_2 \in T.$$

Moreover, proceeding similarly as in the proof of Theorem 3.3 of [4], we have the following result:

Theorem 4.2 *Let $T \subset \mathbb{C}^2(\mathbf{i})$ a bounded domain and $f : T \to \mathbb{C}$ a threshold function and $(w_0, 0)$ a weighting vector of $f(z_1, z_2)$. Then, there exists $w_0' \in \mathbb{C}$ and $\delta > 0$ such that (w_0', w_1, w_2) is a weighting vector of f whenever $|w_j| < \delta$ with $j = 1, 2$.*

Differentiable cost guarantees backward propagation. More precisely, the gradient with respect to a loss function J is expressed for each component of the bicomplex weights w^l that composes the matrix W^l at the layer l, being the output layer the quantification of the error with respect to the target vector for each neuron. The convolution of a bicomplex filter matrix with a bicomplex vector is performed taking into account the previous multiplications rules, in fact, let $W = X_1 + \mathbf{i}Y_1 + \mathbf{j}X_2 + \mathbf{k}Y_2$ be a bicomplex weight filter matrix, and $Z = x_1 + \mathbf{i}y_1 + \mathbf{j}x_2 + \mathbf{k}y_2$ the bicomplex input vector. The bicomplex convolution $W \otimes Z$ is defined as follows:

$$\begin{aligned} W \otimes Z = {} & (X_1x_1 - Y_1y_1 - X_2x_2 - Y_2y_2) + \mathbf{i}(X_1y_1 + Y_1x_1 + X_2y_2 - Y_2x_2) \\ & + \mathbf{j}(X_1x_2 - Y_1y_2 + X_2x_1 + Y_2y_1) + \mathbf{k}(X_1y_2 + Y_1x_2 - X_2y_1 + Y_2x_1) \end{aligned} \tag{17}$$

and can thus be expressed in a matrix form following the matrix representation (2):

$$W \otimes Z = \begin{pmatrix} x_1 & -y_1 & -x_2 & y_2 \\ y_1 & x_1 & -y_2 & -x_2 \\ x_2 & -y_2 & x_1 & -y_1 \\ y_2 & x_2 & y_1 & x_1 \end{pmatrix} * \begin{pmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \end{pmatrix} = \begin{pmatrix} x_1' \\ \mathbf{i}y_1' \\ \mathbf{j}x_2' \\ \mathbf{k}y_2' \end{pmatrix}$$

A suitable initialization scheme improves neural network convergence and reduces the risk of exploding and vanishing gradient. However, bicomplex numbers cannot be initialized component-wise as for the traditional minimization criteria. The reason for this relies on the specific bicomplex algebra and the interaction between the components. Based on the ideas presented in [16, 17], a weight component w of the weight matrix W can be sampled as follows:

$$w_0 = \lambda \cos(\theta), \qquad w_{\mathbf{i}} = \lambda \widetilde{Z}_{\mathbf{i}} \sin(\theta), \qquad w_{\mathbf{j}} = \lambda \widetilde{Z}_{\mathbf{j}} \sin(\theta), \qquad w_{\mathbf{k}} = \lambda \widetilde{Z}_{\mathbf{k}} \sin(\theta). \tag{18}$$

The angle θ is randomly generated in the interval $[-\pi, \pi]$. The bicomplex $\widetilde{Z}$ is defined as a purely normalized imaginary, and is expressed as $0 + \mathbf{i}\widetilde{Z}_{\mathbf{i}} + \mathbf{j}\widetilde{Z}_{\mathbf{j}} + \mathbf{k}\widetilde{Z}_{\mathbf{k}}$. The imaginary components $\mathbf{i}y_1, \mathbf{j}x_2$, and $\mathbf{k}y_2$ are sampled from the uniform distribution in $[0, 1]$ to obtain Z, which is then normalized via (1) to obtain $\widetilde{Z}$. The parameter λ is sampled from $[-\sigma, \sigma]$, where (see [16, 17])

$$\sigma = \frac{1}{\sqrt{2(n_{in} + n_{out})}}, \qquad \text{and} \qquad \sigma = \frac{1}{\sqrt{2n_{in}}},$$

with n_{in} and n_{out} the number of neurons on the input and the output layers.

5 Numerical examples

In this final section, we present a simple numerical implementation where we consider the Bessel-type activation function (16) and we compare its behaviour with the classical *ReLU* activation function in order to perform a comparative analysis of the results and show the effectiveness of our approach.

In our numerical simulation, we consider the *Colored MNIST* dataset and a BCCNN as a baseline model. The *MNIST* dataset consists of handwritten digits, which contains a training set of 60 000 examples, and a test set of 10 000 examples. Each sample is a 28×28 pixels image with the digits 0 to 9, the values assigned to the pixels elements range from 0 to 255. To obtain the *Colored MNIST* we display the training images using a color map and its reversed version. We emphasize that the consideration of a colorized version of the MNIST dataset will be more difficult for the network to train.

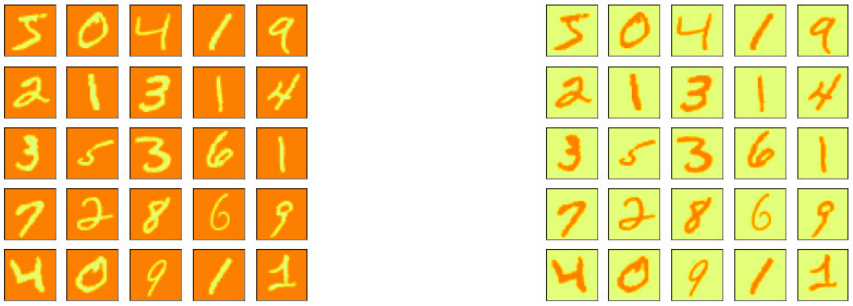
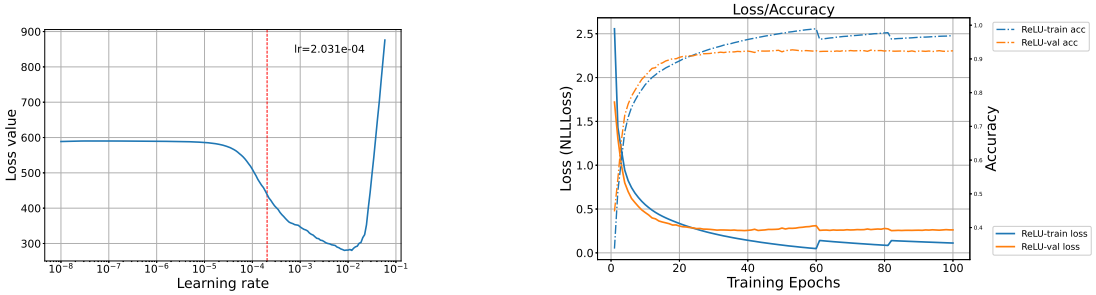


Figure 1: First 25 images of the Colorised MNIST

The BCCNN model takes into account (18) and it is built in a way that we have a convolutional group, which is composed of 2 convolutional layers, the first one has 1 convolutional filter as input and 25 convolutional filters as output, the second layer has 25 filters as input and 50 filters as outputs, each filter has a kernel size of 3×3. After the convolutional layers, we have a fully connected layer with 28800 units as input and 100 units outputs followed by the final layer with 100 units inputs and 10 units outputs which gives the final output prediction for the 10 classes expected by the Colorised MNIST dataset. We use *ReLU* and (16) as activation functions except in the last layer where we use a *LogSoftmax* activation. We employ the *negative log-likelihood loss* (NLLLoss) and *Adam* algorithm as optimisers. For the learning rate, we opted to use a dynamic value which is reduced when the loss metric has stopped improving, this is also known as *ReduceLROnPlateau*. As the initial learning rate value, we follow the guidelines from [22] and choose the value where the gradient towards the minimum loss value is steeper, which in our case, was found to be around 1.8×10^{-3}.

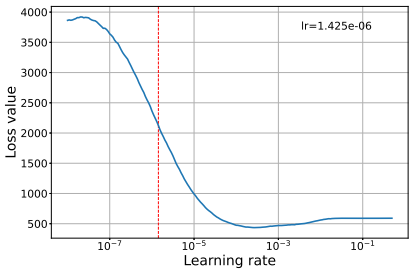


(a) Loss values vs learning rates values.

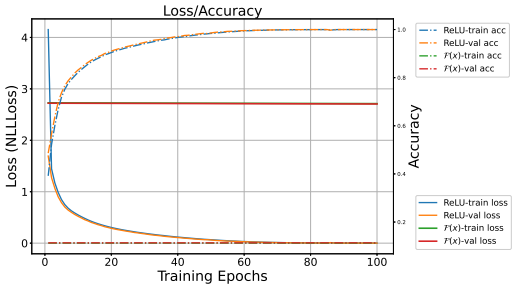
(b) Loss and accuracy history.

Figure 2: BCCNN model for the baseline models in the Colored MNIST dataset.

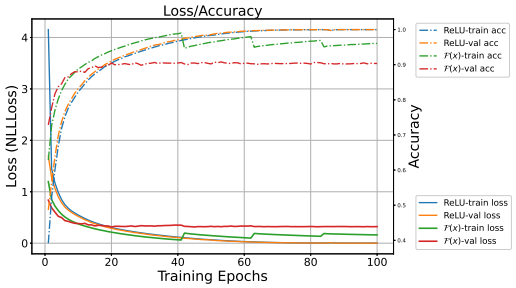
In Figure 2 we show the performance of the baseline model for the BCCNN models with *ReLU* as activation function. In Figure 2(A) the dashed red line highlight shows the layers where the gradient towards the minimum loss value is steeper, in this case, $2.031 \times$ e-04. In Figure 2(B) the continuous (resp. dot-dashed) line shows the loss (resp. accuracy) for the BCCNN model with ReLU activation function. These results will serve as a benchmark to test against the proposed new activation functions. Now we consider (16) as an activation function and we see the behaviour of the BCCNN:



(a) Loss vs learning rate values per epoch.



(b) Loss vs accuracy values per epoch.



(c) Effects of set β_i as dynamic parameter to be adjust during the training phase.

Figure 3: Performance of the baseline model for the FC models with Bessel type activation functions.

In Figure **??**, the orange (resp. blue) continuous line corresponds to the training (resp. validation) phase for the activation function $\mathcal{F}(x)$ with $\beta_i = 1$, while the dot-dashed green (resp. red) line correspond to the results for the baseline model with *ReLU* activation function training (resp. validation) phase. From the analysis of Figure **??**, we have that for the case where all β_i in (16) are equal to one (see Figure **??**(B)), the BCCNN gives poor classification accuracy and also shows a constant behaviour. If we let the values of β_i be chosen by the BCCNN as a new parameter during the training phase, we found a better result as displayed in Figure **??**(C), although the accuracy on the validation dataset stays around 90%, the maximum accuracy is reached around epoch 20, which shows the advantage of such activation against the traditional *ReLU* activation.

6 Conclusions

In this paper, we consider bicomplex neural networks with an activation function of Bessel type. The consideration of this new type of activation function leads to better results when compared with the correspondent ones obtained if we consider *ReLU*. Our numerical experiments reveal that Bessel-type functions combine, in the same activation function, the characteristics of the *ReLU* and the *sinusoid* activation functions. In fact, as we indicated in the manuscript, in the case when ν is a half-integer positive, the Bessel function reduces to a combination of trigonometric and polynomial functions. Compared with the *ReLU* activation function, Bessel-type functions reach high levels of accuracy more rapidly. Moreover, due to the influence of the *sinusoid* component, the Bessel-type activation functions have a lower saturation point when compared with the *ReLU* activation function.

In future work, it would be interesting to consider bicomplex neural networks in more challenging classification tasks, such as the classification of clinical images. Another possible direction consists in considering this new activation function in the quaternionic case, the hyperbolic case, as well in the case of higher dimension hypercomplex algebras, commutative or not. The consideration of these higher algebras simply reduces errors and their implementation. Hypercomplex valued NN allows the accumulation of several complex variables into a single variable theory that can reduce calculations and improve the accuracy of the algorithms.

Acknowledgements

The work of the author was supported by Portuguese funds through *CIDMA–Center for Research and Development in Mathematics and Applications*, and *FCT–Fundação para a Ciência e a Tecnologia*, within projects UIDB/04106/2020

and UIDP/04106/2020. He was also supported by FCT via the 2018 FCT program of Stimulus of Scientific Employment - Individual Support (Ref: CEECIND/01131/2018). N. Vieira expresses his appreciation for the support he has received from the projects *Machine Learning and Special Functions as Activation Functions in Image Processing* (Ref: CPCA/A1/421343/2021) and *Hypergeometric Functions and Machine Learning in the Diagnosis Process* (Ref: CPCA-IAC/AV/475089/2022), and from the *German Research Foundation* (Ref: SM 281/15-1).

References

[1] M. Abramowitz and I.A. Stegun, *Handbook of mathematical functions with formulas, graphs, and mathematical tables. 10th printing*, National Bureau of Standards, Wiley-Interscience Publication, John Wiley & Sons, New York etc., 1972.

[2] R. Agarwal, U.P. Sharma, and R.P. Agarwal, *Bicomplex Mittag-Leffler functions and associeted properties*, J. Nonlinear Sci. Appl. **15** (2022), 48–60.

[3] I.N. Aizenberg, N.N. Aizenberg, J. Vandewalle, *Multi-Valued and Universal Binary Neurons, Theory, Learning, and Applications*, Springer New York, NY, 2000

[4] D. Alpay, K. Dikil, and M. Vajiac, *A note on the complex and bicomplex valued neural networks*, Appl. Math. Comput. **445** (2023), Article No. 127864 (12pp.).

[5] P. Arena, L. Fortuna, I. Occhipinti, and M.G. Xibilia, *Neural networks for quaternionic-valued functions approximation*, In: 1994 IEEE Internaticnal Symposium on Circuits and Systems - ISCAS94, **6**, 307–310.

[6] N.A. Aspragathos and J.K. Dimitros, *A comparative study of three methods for robot kinematics*, Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on **28**-No.2 (1998), 135–145.

[7] H. De Bie, D. Struppa, A. Vajiac, M. Vajiac, *The Cauchy-Kowalewski product for bicomplex holomorphic functions*, Math. Nachr. **285**-No.10 (2012), 1230–1242.

[8] G. Bingham and R. Miikkulainen, *Discovering parametric activation functions*, ArXiv preprint, ArXiv:2006.03179v4, 2006.

[9] P. Cerejeiras, Y. Fu, and N. Gomes, *Bicomplex signals with sparsity constraints*, Math Meth. Appl Sci. **41**-No.13 (2018), 5140–5158

[10] F. Colombo, I. Sabadini, D.C. Struppa, A. Vajiac, and M.B. Vajiac, *Singularities of functions of one and several bicomplex variables*, Ark. Mat. **49**-No.2 (2011), 277–294.

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

[12] E. Luna-Elizarraras, M. Shapiro, and D.C. Struppa, and A. Vajiac, *Bicomplex Holomorphic Functions. The Algebra, Geometry and Analysis of Bicomplex Numbers*, Birkhüser/Springer, Cham, 2015.

[13] G. Maguolo, L. Nanni and S. Ghidoni, *Ensemble of convolutional neural networks trained with different activation functions*, Expert Syst. Appl. **166** (2021), Article No. 114048 (8pp.).

[14] J. Paneva-Konovska, *Bessel type functions as multi-index Mittag-Leffler functions: Erdélyi-Kober integral relations*, AIP Conference Proceedings 2333 (2021), Article No.060003 (8pp.).

[15] T. Parcollet, M. Ravanelli, M. Morchid, G. Linarès, C. Trabelsi, R. De Mori, and Y. Bengio, *Quaternion recurrent neural networks*, in Proceedings of the 7th International Conference on Learning Representations (ICLR 2019), 2019, Annual Conference of the International Speech Communication Association (INTERSPEECH), 2018, Article No. 149936 (19pp.).

[16] T. Parcollet, M. Morchid, G. Linarès, and R. De Mori, *Quaternion Convolutional Neural Networks for Theme Identification of Telephone Conversations*, in Proceedings of the 2018 IEEE Spoken Language Technology Workshop (SLT 2018), 2018, Article No. 145107 (7pp.).

[17] T. Parcollet, Y. Zhang, M. Morchid, C. Trabelsi, G. Linarès, R. De Mori, and Y. Bengio, *Quaternion convolutional neural networks for end-to-end automatic speech recognition*, in Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH), 2018, 22–26.

[18] S.C. Pei, J.H. Chang, and J.J. Ding, *Commutative reduced biquaternions and their fourier transform for signal and image processing applications*, IEEE Trans Signal Process. **52**-No.7 (2004), 2012–2031.

[19] A.P. Prudnikov, Yu. Brychkov, and O.I. Marichev, *Integrals and series. Volume 3: More special functions*, Transl. from the Russian by G. G. Gould, Gordon and Breach Science Publishers, New York, 1990.

[20] S. Sabour, N. Frosst, and G.E. Hinton, *Dynamic routing between capsules*, Adv Neural Inf Process Syst. **2017** (2017), 3857–3867.

[21] S.J. Sangwine, *Fourier transforms of colour images using quaternion or hypercomplex, numbers,* Electronics letters **32**No.21 (1996), 1979–1980, 1996.

[22] L.N. Smith, *A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay*, arXiv preprint arXiv:1803.09820, 2018.

[23] C. Trabelsi, O. Bilaniuk, D. Serdyuk, S. Subramanian, J.F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C.J. Pal, *Deep complex networks*, ArXiv preprint, ArXiv:1705.09792, 2017.

[24] B.C. Ujang, C.C. Took, and D.P. Mandic, *Quaternion valued nonlinear adaptative filtering*, IEEE Trans. Neural Ntew. **28**-No.8 (2011), 1193–1206.

[25] N. Vieira and F. Freitas, Hypergeometric functions as activation functions: the particular case of the Bessel type functions, submitted.

[26] J. Zamora-Esquivel, A.C. Vargas, J.R. Camacho-Perez, P.L. Meyer, H. Cordourier, and O. Tickoo, *Adaptative activation functions using fractional calculus*, 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), 2019, 2006–2013.