2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) | 978-1-6654-6382-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/AVSS56176.2022.9959227

Dual-valued Neural Networks

Dmitry Kozlov[†] Stanislav Pavlov[†‡*] Alexander Zuev[†] Mikhail Bakulin[†]

Mariya Krylova[†‡] Igor Kharchikov[†‡]

[†]NN AI Team, Huawei Russian Research Institute
[‡]National Research University Higher School of Economics
[*]Volga State University of Water Transport
Nizhny Novgorod, Russia
{kozlov.dmitry, pavlov.stas, zuev.alexander1, bakulin.mikhail, krylova.mariya, kharchikov.igor}@huawei.com

Abstract

The majority of existing neural networks operate with real-valued representation of data. However, there are multiple tasks in which the input is complex-valued. The complex-valued data is considered to be more informative in terms of larger representational capacity. These reasons motivate researchers to develop neural networks using complex numbers instead of real-valued ones. In this paper, we take a step forward in the generalization of neural networks. We develop the basic building blocks for dual-valued neural networks based on dual numbers. We adjust basic layers such as Linear, Convolution, Average Pooling, ReLU to the dual domain and present an algorithm for Dual Batch Normalization. We construct several dual-valued neural networks for classification tasks basing on classical CV problems and the MusicNet and G2Net datasets. We show that dual-valued models outperform analogous complex-valued neural networks in execution time and have higher or at least the same accuracy.

1. Introduction

Generalization of neural networks to the complex domain is a popular subject of recent studies. The reason for such heightened interest is expected since the original data is often presented in a complex form. Recent studies show multiple advantages of using complex-valued neural networks, as opposed to the real-valued ones, including faster learning [3] and larger representational capacity [19]. Furthermore, the state-of-the-art works introduce equivariant and invariant complex-valued layers and activation functions for neural networks, which allow researchers to construct robust models with better convergence and accuracy [7, 22]. These results are based on the fact that complex numbers can be represented in the polar form in terms of their amplitude and phase. This representation also has a significant drawback in particular, it is difficult to find a sum of complex numbers in this form. That is why in this work we will focus on algebraic representation of complex numbers.

A survey [4] of existing complex-valued networks and developed approaches also shows growing interest in this field of artificial intelligence.

These outstanding results inspire us to perform further generalization of neural networks to the dual domain $\mathbb{D}$. The dual numbers are a special class of numbers, whose elements can be written as $x+\varepsilon y$, where x, y are real numbers, and ε is a nilpotent element, which satisfies the relations: $\varepsilon^2 = 0, \varepsilon \neq 0$. The basic mathematical operations for dual numbers are

$$(x_1 + \varepsilon y_1) \pm (x_2 + \varepsilon y_2) = (x_1 \pm x_2) + \varepsilon(y_1 \pm y_2), \quad (1)$$

$$(x_1 + \varepsilon y_1)(x_2 + \varepsilon y_2) = x_1 x_2 + \varepsilon(x_1 y_2 + y_1 x_2), \quad (2)$$

$$\frac{(x_1 + \varepsilon y_1)}{(x_2 + \varepsilon y_2)} = \frac{x_1}{x_2} + \varepsilon \frac{(y_1 x_2 - x_1 y_2)}{x_2^2}, \quad (3)$$

$$\overline{(x + \varepsilon y)} = (x - \varepsilon y). \quad (4)$$

As well as complex numbers, dual numbers have found applications in physics, specifically in the Screw theory [9]. Dual numbers also make it possible to automatically compute derivatives of functions [5, 14]. There are no obstacles

978-1-6654-6382-9/22/$31.00 ©2022 IEEE

to use dual numbers for Deep Learning. Presumably, the first and only attempt to investigate dual-valued neural network is [20], but this work does not exploit any properties of dual numbers, except for $\varepsilon^2 = 0$.

Our increased interest in such algebra is driven by the following problem: when real-valued layers are transferred to a complex domain, the number of operations is significantly increased. For example, convolution of complex-valued input $z = x + iy$ with complex-valued filter matrix $W = A + iB$ can be rewritten as $Wz = Ax - By + i(Ay + Bx)$, which has four real-valued matrix multiplications (see Fig. 1). At the same time, we can consider x, y as separate groups and apply real-valued convolutions to each channels independently. In this case, x and y are not be linked with each other. To overcome this obstacle, we suggest using dual numbers. Dual-valued convolution of dual filter $W = A + \varepsilon B$ and dual input $z = x + \varepsilon y$, due to distributive property and $\varepsilon^2 = 0$, is $Wz = Ax + \varepsilon(Ay + Bx)$. This convolution has only three real-valued matrix multiplications (see Fig. 1), which results in 25% performance speed-up compared to complex-valued convolution, and links x and y channels as well. This result has persuaded us into further research.

In this paper, we present a dual-valued neural network. We provide definitions of the basic components of dual-valued neural network, such as layers and activation functions. In addition, we develop special normalization techniques that utilize matrix representation of dual numbers. As it has been shown for the real-valued networks [12], batch normalization helps us accelerate the training process and reach better accuracy.

2. Methodology

This section is dedicated to definitions of key elements of a dual-valued neural network.

2.1. Generation of Dual-valued Data

Dual numbers are not natively implemented in any framework. Therefore, to emulate dual-valued operations, we use two real-valued channels, considering the first channel to be a real part of a dual-valued entity and the second channel to be its dual component. The peculiarities of dual algebra are considered internally.

The first thing to be clarified is how we get dual-valued input. To generate dual-valued input for complex-valued problems (MusicNet, G2Net), we associate the imaginary part of the original input with a dual component, namely

$$x + iy \implies x + \varepsilon y. \tag{5}$$

It is not a theoretically based approach, it is only our assumption. Here we also apply dual-valued neural networks for classical computer vision classification tasks (CIFAR-10, CIFAR-100, SVHN). To make dual-valued input, we use a more general method, comparing to the one proposed in [22]. In [22], the authors convert $[R, G, B]$ real-valued image to a complex form by

$$[R, G, B] \implies [R + iG, G + iB], \tag{6}$$

claiming that this type of encoding captures channel correlations and hue shift. In this paper, we generalize the idea from [22], namely, we use a linear transformation with trainable parameters to convert $[R, G, B]$ into six real-valued channels, which are later reshaped into three complex/dual-valued channels.

2.2. Dual-valued Operations

To be able to construct a dual-valued neural network of any kind, we need to define its building blocks. Here we define the key operations needed for developing dual-valued networks.

2.2.1 Dual-valued Convolution

One of the most important operations in neural networks is convolution. In order to illustrate convolution in the dual domain, we exploit the matrix representation of dual numbers that uses only real-valued entities. Algebra of dual numbers $z = x + \varepsilon y$ is known to be isomorphic to the algebra of second-order real-valued matrices of the form $A = \begin{pmatrix} x & y \\ 0 & x \end{pmatrix}$. Therefore, convolution of a dual-valued filter $W = W_r + \varepsilon W_d$ and a dual-valued input matrix $z = x + \varepsilon y$ is identical to the following matrix manipulation

$$W * z = \begin{pmatrix} W_r & W_d \\ 0 & W_r \end{pmatrix} * \begin{pmatrix} x & y \\ 0 & x \end{pmatrix} = \\ = \begin{pmatrix} W_r * x & W_r * y + W_d * x \\ 0 & W_r * x \end{pmatrix}, \tag{7}$$

where * denotes convolution .

Complex numbers are also isomorphic to the algebra of second-order real matrices of the form $A = \begin{pmatrix} x & -y \\ y & x \end{pmatrix}$. So, complex-valued convolution is expressed in the matrix notation as

$$W * z = \begin{pmatrix} W_r & -W_i \\ W_i & W_r \end{pmatrix} * \begin{pmatrix} x & -y \\ y & x \end{pmatrix} = \\ = \begin{pmatrix} W_r * x - W_i * y & -(W_r * y + W_i * x) \\ W_r * y + W_i * x & W_r * x - W_i * y \end{pmatrix}. \tag{8}$$

Both of these convolutions have only two independent terms. As it was mentioned earlier, we should make four real-valued multiplications in the complex domain and only three in the dual case to calculate them.

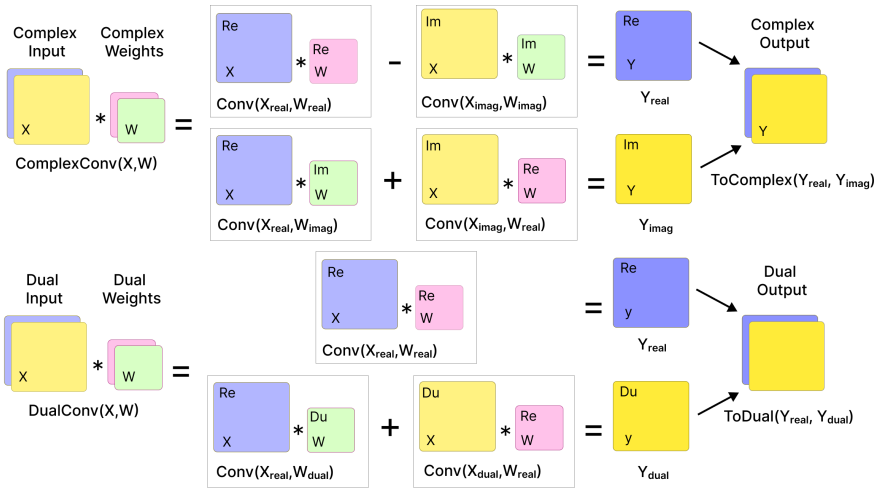

Figure 1. Comparison of complex-valued and dual-valued convolutions. Complex-valued convolution is equivalent to four real-valued matrix multiplications, while the dual-valued one consists of only three real-valued matrix multiplications.

2.2.2 Dual-valued Linear Layer

Linear or fully-connected layers are commonly used in neural networks. This kind of layers connects every input channel to every output channel. To generalize linear layer for the dual domain, we again use the matrix representation of dual numbers as well as the distributive and associative properties of dual numbers. This leads to the similar result as the one we get in dual-valued convolution described above. Namely, a dual-valued linear layer is equal to superposition of three real-valued linear layers:

$$\begin{aligned}\mathbb{D}Linear(W,b,z) = Linear(W_r,b_r,z_r)+\\ +\varepsilon(Linear(W_r,0,z_d)+Linear(W_d,b_d,z_r)),\end{aligned} \quad (9)$$

where $Linear(w,b,x)$ stands for a real-valued linear layer, in which w represents the weights, b is for the bias, and x is for the input.

2.2.3 Dual-valued Average Pooling

Average pooling operation means calculating the average for each patch of the feature map. This implies that each nxn square of the feature map, where n is the kernel size, is downsampled to the average value of the square. An average value of a set of dual numbers is average values of their real and purely dual parts, so we define average pooling in the dual domain as a real-valued average pool that is applied independently to real and purely dual parts of the dual-valued input:

$$\begin{aligned}\mathbb{D}AvgPool(z) =\\ = AvgPool(Re(z))+\varepsilon AvgPool(Du(z)).\end{aligned} \quad (10)$$

2.2.4 Dual-valued ReLU

Activation functions are used to introduce non-linearity into a system. Among the variety of real-valued activation functions, $ReLU$-type ones do not suffer from the vanishing gradient, so they are more stable. Functions of this kind can be extended to the complex domain in multiple ways. For example, a complex cardioid was applied in [27] for magnitude resonance imaging fingerprinting:

$$f(z) = \frac{1}{2}(1+\cos\angle z)z. \quad (11)$$

It is also possible to apply $ReLU$ to the real and imaginary parts independently, as it is done in [26]:

$$\mathbb{C}ReLU(z) = Relu(Re(z)) + iRelu(Im(z)). \quad (12)$$

Here, we apply ReLU activation function in the same way:

$$\mathbb{D}ReLU(z) = Relu(Re(z)) + \varepsilon Relu(Du(z)), \quad (13)$$

where $Du(z)$ is a dual part of z. We also experimented with others activation functions, but this one shows the best results.

2.3. Norm of Dual Numbers

The key feature of this work is formulation of dual batch normalization. It is based on the norm (or modulus) of a dual number. In mathematical or physical literature, the modulus of a dual number is usually taken as its real part, since

$$|z| = \sqrt{z\bar{z}} = \sqrt{(x+\varepsilon y)(x-\varepsilon y)} = |x|. \quad (14)$$

But several works like [15, 16] propose more sophisticated expression for the modulus

$$|z| = |x+\varepsilon y| = x^2 - y, \quad (15)$$

which describes the contour line of orbits of some Möbius map. Unfortunately, the dual modulus defined by (2) can be negative and leads to division by zero in normalization. That is why we have to develop a new expression for the dual norm.

To do this, we again exploit the representation of a dual number $z = x + \varepsilon y$ as a matrix $A = \begin{pmatrix} x & y \\ 0 & x \end{pmatrix}$. Then we associate the norm of this matrix with the norm of the original dual number. There are multiple ways to determine the matrix norm. First, let us define $\mathbb{R}^{m\times n}$ as a vector space of matrices with m rows and n columns entries in the field of real numbers $\mathbb{R}$. We use the matrix norm induced by a vector norm $\|\cdot\|_2$ on $\mathbb{R}^{2\times 2}$ and a vector norm $\|\cdot\|_2$ on $\mathbb{R}^{2\times 1}$ and define the dual norm as

$$\|z\|^2 = sup\left\{\|At\|_2^2 : t \in \mathbb{R}^{2\times 1}, \|t\|_2^2 = 1\right\} \tag{16}$$

It is also called the operator norm. For the matrix that corresponds to the dual number $z = x + \varepsilon y$, the formula looks as follows

$$\|At\|_2^2 = \left(\begin{pmatrix} x & y \\ 0 & x \end{pmatrix}\begin{pmatrix} t_1 \\ t_2 \end{pmatrix}\right)^T \left(\begin{pmatrix} x & y \\ 0 & x \end{pmatrix}\begin{pmatrix} t_1 \\ t_2 \end{pmatrix}\right) =$$
$$= (xt_1 + yt_2)^2 + x^2t_2^2 = \begin{bmatrix} t_1^2 + t_2^2 = 1 \\ t_1 = \cos\varphi \\ t_2 = \sin\varphi \end{bmatrix} =$$
$$= x^2 + \frac{y^2}{2} + y\left(-\frac{1}{2}y\cos 2\varphi + x\sin 2\varphi\right)$$

Then we find the extremum of this function:

$$f'(\varphi) = y\left(y\sin 2\varphi + 2x\cos 2\varphi\right) = 0$$
$$\tan 2\varphi = \frac{-2x}{y} \implies \begin{matrix} \cos 2\varphi = \dfrac{\mp y}{\sqrt{4x^2+y^2}} \\ \sin 2\varphi = \dfrac{\pm 2x}{\sqrt{4x^2+y^2}} \end{matrix}.$$

And we eventually get the maximum of the function

$$\|z\|^2 = x^2 + \frac{y^2}{2} + \left|\frac{y}{2}\right|\sqrt{4x^2+y^2}. \tag{17}$$

It is easy to show that

$$\|z\| = \left|\frac{y}{2}\right| + \sqrt{x^2 + \left(\frac{y}{2}\right)^2}. \tag{18}$$

Now we should consider the limiting cases. For a real number, $\|z\| = |x|$, and for a purely dual number, $\|z\| = |y|$. It is also interesting to find a contour line corresponding to the constant value of the dual norm:

$$|y| = \frac{\|z\|^2 - x^2}{\|z\|}. \tag{19}$$

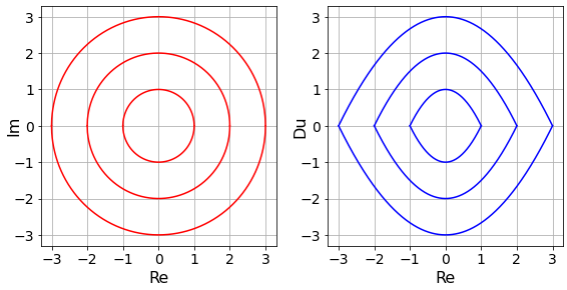

Figure 2. Contour lines correspond to the constant value of the complex norm (left red circles) and dual norm (right blue parabolas).

To clarify the difference between the proposed dual norm and the modulus of a complex number, we depicted both contour lines at Fig. 2. One can see from Fig. 2 that the contour line of constant dual norm is closer to the center, so it influences the distance between two dual numbers, definition of which is essential to data normalization. In addition, as with modulus defined by (15), our contour lines of the constant dual norm (19) are parabolas, but they have different slopes of their branches due to $\|z\|$ in the denominator.

2.4. Dual Batch Normalization

One of the ways to accelerate the learning process of deep networks and reach better data generalization is Batch Normalization. Batch Normalization in its original form [12] can only be applied to real-valued models. This approach includes normalization of each dimension of the k-dimensional input

$$\check{x}^{(k)} = \frac{x^{(k)} - E[x^{(k)}]}{\sqrt{Var[x^{(k)}]}} \tag{20}$$

and additional scaling with shift

$$RBN[\check{x}^{(k)}] = \check{\gamma}^{(k)}\check{x}^{(k)} + \check{\beta}^{(k)}. \tag{21}$$

The linear transformation is needed to ensure that the total function can be the identity operator. There is also an extension of it to the complex values [26]. The main idea of such generalization is to treat complex values as $2D$ vectors, which allows us to decorrelate real and imaginary parts of data. Otherwise, if we apply the original Batch Normalization to real and imaginary parts separately, it can end up with a high eccentricity elliptical distribution. Therefore, the authors of [26] define Complex Batch Normalization as

$$\tilde{x}^{(k)} = \frac{x^{(k)} - E[x^{(k)}]}{\sqrt{V[x^{(k)}]}}, \tag{22}$$

where the covariance matrix $V[x^{(k)}]$ is defined as follows:

$$V[x^{(k)}] = \begin{pmatrix} Cov(x_r^{(k)}, x_r^{(k)}) & Cov(x_r^{(k)}, x_i^{(k)}) \\ Cov(x_i^{(k)}, x_r^{(k)}) & Cov(x_i^{(k)}, x_i^{(k)}) \end{pmatrix} \tag{23}$$

Similar to the original real-valued approach, Complex Batch Normalization also has an additional linear transformation with two parameters $\check{\gamma}^{(k)}$, $\check{\beta}^{(k)}$ with real-valued learning components

$$\tilde{\gamma}^{(k)} = \begin{pmatrix} \tilde{\gamma}_{rr}^{(k)} & \tilde{\gamma}_{ri}^{(k)} \\ \tilde{\gamma}_{ir}^{(k)} & \tilde{\gamma}_{ii}^{(k)} \end{pmatrix}, \quad \tilde{\beta}^{(k)} = \begin{pmatrix} \tilde{\beta}_{r}^{(k)} \\ \tilde{\beta}_{i}^{(k)} \end{pmatrix}. \tag{24}$$

Finally, Complex Batch Normalization is

$$CBN[\tilde{x}^{(k)}] = \tilde{\gamma}^{(k)}\tilde{x}^{(k)} + \tilde{\beta}^{(k)}. \tag{25}$$

Here we generalize the batch normalization process for dual values. To achieve this goal, we use the dual norm proposed in the previous section. We should emphasize that we cannot use the exact same procedure as for Complex Batch Normalization [26] due to the following reason:

$$\mu^{(k)} = E[x^{(k)}] = \mu_r^{(k)} + \varepsilon\mu_d^{(k)} \tag{26}$$

$$\begin{aligned}\Gamma^{(k)} = E[(x^{(k)} - \mu^{(k)})^2] = E[(x_r^{(k)} - \mu_r^{(k)})^2] + \\ + 2\varepsilon E[(x_r^{(k)} - \mu_r^{(k)})(x_d^{(k)} - \mu_d^{(k)})]\end{aligned} \tag{27}$$

$$C^{(k)} = E[(x^{(k)} - \mu^{(k)})\overline{(x^{(k)} - \mu^{(k)})}] = E[(x_r^{(k)} - \mu_r^{(k)})^2] \tag{28}$$

Here we can see that neither covariance Γ nor pseudo-covariance C depend on $E[(x_d^{(k)} - \mu_d^{(k)})^2]$. This problem is similar to the independence of dual number norm (14) on its dual part. To overcome this obstacle, we exploit the norm of dual numbers proposed in the previous section. To make Dual Batch Normalization, we need to define the mean value and the variance first:

$$\mu^{(k)} = E[x^{(k)}] = \mu_r^{(k)} + \varepsilon\mu_d^{(k)} \tag{29}$$

$$(\sigma^{(k)})^2 = \frac{1}{m}\sum_{i=1}^{m} \|x_i^{(k)} - \mu^{(k)}\|^2, \tag{30}$$

where $\|\cdot\|$ is the norm determined by (18) for dual numbers. Then we transform the input in the following way

$$\hat{x}^{(k)} = \frac{x^{(k)} - \mu^{(k)}}{\sqrt{(\sigma^2)^{(k)} + \delta}}, \tag{31}$$

where $\delta = 10^{-7}$ is needed to avoid division by zero. The last step is a linear transformation with dual-valued scaling and shift:

$$DBN[\hat{x}^{(k)}] = \hat{\gamma}^{(k)}\hat{x}^{(k)} + \hat{\beta}^{(k)}, \tag{32}$$

where $\begin{matrix}\hat{\gamma}^{(k)} = \hat{\gamma}_r^{(k)} + \varepsilon\hat{\gamma}_d^{(k)} \\ \hat{\beta}^{(k)} = \hat{\beta}^{(k)} + \varepsilon\hat{\beta}^{(k)}\end{matrix}$. We should emphasize that Dual Batch Normalization is closer to $RBN[\check{x}^{(k)}]$ than to $CBN[\tilde{x}^{(k)}]$, but instead of L_2 norm, it uses the dual norm defined by our formula (18).

Algorithm 1 Dual Batch Normalization

Input: Dual-valued $x^{(k)}$ over a batch:
$\mathfrak{B} = \{x_1^{(k)}, \dots, x_m^{(k)}\}$;
Trainable parameters: $\hat{\gamma_r}^{(k)}, \hat{\gamma}_d^{(k)}, \hat{\beta}_r^{(k)}, \hat{\beta}_d^{(k)}$
Output: $y_i^{(k)} = DBN[\hat{x}_i^{(k)}]$
$\mu^{(k)} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i^{(k)}$ // batch mean
$(\sigma^{(k)})^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} \|x_i^{(k)} - \mu^{(k)}\|^2$ // batch variance with dual norm
$\hat{x}^{(k)} \leftarrow \frac{x^{(k)} - \mu^{(k)}}{\sqrt{(\sigma^2)^{(k)} + \delta}}$ // normalization
$y_i^{(k)} \leftarrow \hat{\gamma}^{(k)}\hat{x}_i^{(k)} + \hat{\beta}^{(k)} \equiv DBN[\hat{x}_i^{(k)}]$ // dual scale and shift

3. Implementation of Dual-valued Networks

To explore the capabilities of dual-valued neural networks, we have conducted a series of experiments with our models for signal and image classification tasks and compared the results of real and complex-valued neural networks with the same architecture. We also use a fixed training strategy to focus on influence of the new algebras rather than hyperparameters.

3.1. Music Transcription Task

In this part we show the results for automatic music transcription. The experiments are performed on the MusicNet dataset [24]. To gain the computational efficiency, we resample the original signal from 44.1kHz to 11kHz basing on the algorithm described in [23]. This downsampling allows us to decrease computational cost without any significant information loss. Following [24], '2303', '2382', '1819' records are used as a test subset and other 327 files are used as a training set. We conduct all the experiments with complex representation of the frequency spectrum. For the real model, we consider the real and imaginary components of the spectrum as separate channels.

We also exploit the DeepConvNet architecture developed in [26]. Our network consists of six $1D$-convolutional layers. The first one has the filter of size six and other layers have kernel sizes equal three. Convolution blocks are followed by a real-valued linear layer with 2048 connections for a real model or a complex/dual-valued linear layer with 1024 connections for a complex/dual model and a $ReLU$ activation function. Before passing through the last layer, the data representation needs to be changed from the dual/complex form to the real one. To preserve all information, we concatenate real and imaginary/dual components to one joint channel. Finally, we apply real-valued linear layer with 84 connections and a sigmoid activation function. The number of units in the last operator corresponds to the number of notes present in the dataset. For the real, dual, and complex models, we use $ReLU$ and its counterparts $\mathbb{D}ReLU$ and

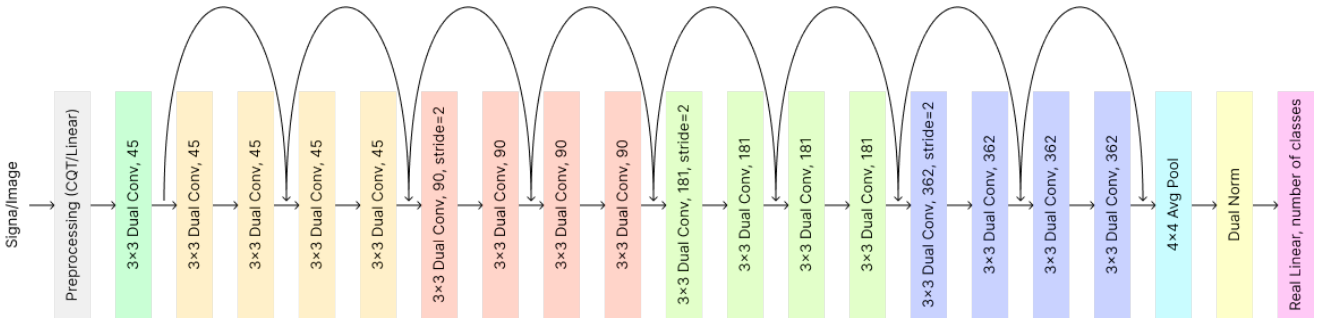

Figure 3. A dual ResNet18 architecture for gravitational wave detection (G2Net) and computer vision problems. The same architecture was also used in complex-valued neural networks up to replacement of dual operations with the complex ones.

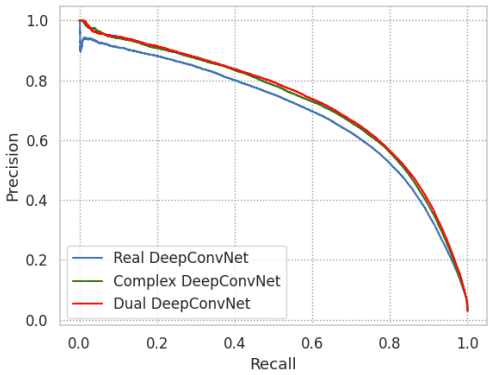

Figure 4. Precision-Recall curve for the MusicNet dataset.

$\mathbb{C}ReLU$ respectively, which were defined in Section 2.2. In all of our experiments we used an input window of 4096 samples or its FFT (which corresponds to the 8192-window used in the baseline) and predicted notes in the center of the window. All networks were optimized with Adam [13]. We started with the learning rate at 10^{-3} for the first 10 epochs and then decreased it by a factor of 10 at each of the epochs 10, 100, 120, and 150. The complex network was initialized using the unitary initialization scheme respecting the He criterion as it was described in [26]. The dual-valued and real-valued models were initialized by the He initialization according to the method proposed in [10], basing on a uniform distribution. The results are summarized in Table 1. Precision-Recall dependency is depicted in Fig.4.

Table 1. Results of experiments on the MusicNet dataset.

Model	Average Precision, %	Parameters, MB
Real	68.9	43.90
Dual	**73.0**	43.83
Complex	72.5	43.86

From Table 1, we can observe that the best average precision is achieved by the dual-valued neural network. Moreover, as it will be shown in Section 3.4, the dual-valued model has x0.8 inference time of the complex one. Meanwhile, the precision of the complex-valued model is close to the value reported in [26]. These results show that a dual-valued neural network has the best accuracy-performance trade-off, because it is faster than a complex-valued model and more accurate than other models. We consider dual-valued neural networks to be promising solutions for the tasks with the complex representation of the input data.

3.2. Gravitational Wave Detection

This part is dedicated to the implementation of dual-valued networks with respect to signal processing, in particular, gravitational wave detection based on the G2NET dataset. This training set consists of time series data, which essentially is a number of noised synthetic gravitational wave instances. It mimics measurements of a system of three ground-based laser interferometers (LIGO Hanford, LIGO Livingston, and Virgo) [1, 21, 25]. The gravitational waves are radiated by accelerated huge masses. For example, this phenomenon can be created during neutron star merging or black holes collision.

Each sample of G2Net is represented as 3-channel (one for each detector) time series spanning 2 sec with sampling rate of 2,048 Hz [25]. The solution of this task consists of two stages. The first one is the construction of the representative frequency spectrum of signals based on transform algorithms [2], or also known as a spectral portrait. The second part is the detection of waves' presence or absence with the help of neural network [2, 28].

In this paper we concatenate records to one signal track. Since the signals have different relative amplitudes due to sensitivity and geographical location of sensors, data was normalized before concatenation. Afterwards, the CQT algorithm (Constant-Q Transform [6]), which is considered to be one of the preferred algorithms for gravitational waves signal processing [2], is applied as a main preprocessing algorithm to generate complex-valued scalogram of the track. Complex-valued CQT output is used as input of complex-valued and dual-valued networks without any changes (see Section 2.1). In a real-valued case, modulus of CQT output is used as input, but we should notice that it leads to loss of

Table 2. Results of gravitational wave detection.

Model	Average Precision, %	Parameters, MB
Real	77.02	34.2
Dual	**78.19**	34.2
Complex	77.85	34.2

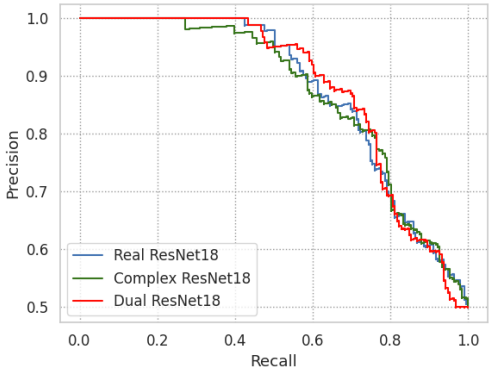

Figure 5. Precision-Recall curve for the G2Net dataset.

important information. Subsequently, the resulting scalogram is processed as an image in the classification problem, which is solved by a convolutional neural network.

To detect gravitational waves on the scalogram, we adapt an architecture developed in [11]. Namely, we use ResNet18 with the same original architecture [11] as in the real-valued case but with $\sqrt{2}$ times fewer features in case of dual or complex networks (see Fig. 3). In order to emulate the dual and complex numbers, we have added an extra dimension of length two, where the first element corresponds to the real part and the second one to the dual (or complex) part of a number. All operations of real-valued neural networks are changed to analogous dual- or complex-valued ones. Before the last layer of the complex-valued and dual-valued networks outputs are transferred to real domain via applying ad-hoc norm. Models are optimized with RAdam [17] algorithm with starting learning rate set to 0.05 changing by StepLR scheduler with $\gamma = 0.1$ and $step = 10$. Networks are trained for 30 epochs. Dual-valued and complex-valued ResNet18 have average precision that is higher then real-valued one (Table 2 and Fig. 5).

3.3. Computer Vision Problems

For CV tasks, we reuse ResNet18 (Fig. 3) developed for the G2Net dataset. The main difference is caused by the preprocessing (Section 2.1), which leads to three dual-valued or complex-valued channels instead of three real-valued ones. There is a study of the effect of color representation on the accuracy of a neural network [8]. It shows that the RGB representation is not optimal. Thus, a trainable linear layer, which we use to convert colors into the dual/complex domain, allows our neural networks to determine the best color space for the task.

To train dual and complex models, we use stochastic gradient descent [18] with 0.9 momentum to optimize real-valued loss functions by treating the real and dual or imaginary values as separate real-valued channels. So, we use a standard approach for calculating dual and complex derivatives. The cross entropy loss between the input and the target is used as a criterion for these problems. We also apply cosine annealing schedule, proposed in [10], with $T_{max} = 300$ and $\eta_{min} = 0$. The training lasts 300 epochs. Our results of image classification on CIFAR-10, CIFAR-100, and SVHN (Street View House Numbers) are presented in Table 3.

Table 3. Accuracy (%) of our models for classical CV problems.

Model	CIFAR-10	CIFAR-100	SVHN
Real	94.5	75.9	96.3
Dual	**95.7**	**78.3**	**96.8**
Complex	**95.7**	78.2	96.7

One can observe in Table 3 that the dual-valued neural network achieves higher or at least the same accuracy compared to the real/complex-valued ones in all cases. In addition, a dual-valued model shows better inference time than a complex-valued one, but a real-valued neural network has the best performance time (Section 3.4). Despite the fact that these results are based on the simplest preprocessing (Section 2.1), a dual-valued neural network shows good results. Our main idea is that dual-valued neural networks, as well as complex-valued ones, allow different channels to interact with each other due to the internal connection between real and dual parts. This inner complexity should be exploited; possibly, more sophisticated input preprocessing can lead to even better accuracy for these more complicated networks.

3.4. Inference and Computational Complexity

As it was presented in Introduction, the theoretical computational complexity of a dual-valued convolution is 25% less of the complex-valued one. However, they are 3 and 4 times, correspondingly, slower than a real-valued convolution in case of the same amount of channels.

We modify the number of channels and input size so that our models have the same number of weights in any algebra (see Tables 1, 2). This implies that the theoretical inference complexity of the DeepConvNet and ResNet18 models is accordingly $\times 1.5$ and $\times 2$ more expensive for the dual and complex domains than for the real numbers. The difference between experimental and theoretical figures is explained by the cost of high-level implementation of Dual and Complex Batch Normalization, which are not natively integrated in cuDNN. Table 4 presents our measurements of inference time for our models. The technical details are

SW: PyTorch 1.11.0, CUDA 11.3; HW: CPU: 11th Gen Intel(R) Core(TM) i9-11900KF 3.50GHz, GPU: NVIDIA GeForce RTX 3080Ti.

Table 4. Inference time (μs) of considered models.

Model	DeepConvNet	ResNet18
Real	37	11
Dual	51	20
Complex	64	34

Table 4 indicates that the usage of dual or complex numbers in neural networks leads to a significant increase in the number of operations. Nevertheless, dual-valued and complex-valued networks reach higher metrics than the real one.

4. Conclusion

The key feature of this work is a proposition of dual-valued models with dual number based layers: Convolution, Batch Normalization, Linear, Average Pooling, ReLU. We have shown that using dual numbers instead of complex ones with proposed normalization allows us to improve the average precision score (or reach the same) and significantly speed up the execution time of neural networks, comparing with the complex-valued models. Further modification of data preprocessing can improve the generalization characteristics of a dual neural network. We hope that our work will inspire other researchers to use dual-valued neural networks for more difficult problems such as image segmentation and NLP.
There are still several unsolved problems such as transformation of input to the dual numbers and backward error propagation, which will be subjects of future research.

References

[1] B. P. Abbott et al. Observation of gravitational waves from a binary black hole merger. *Phys. Rev. Lett.*, 116:061102, Feb 2016. 6

[2] P. Addesso et al. Localization of gravitational sources from time-frequency maps. *5th IEEE International Workshop on Metrology for AeroSpace, MetroAeroSpace 2018 - Proceedings*, pages 538–543, 2018. 6

[3] M. Arjovsky, A. Shah, and Y. Bengio. Unitary evolution recurrent neural networks, 2015. 1

[4] J. Bassey, L. Qian, and X. Li. A survey of complex-valued neural networks. *ArXiv*, abs/2101.12249, 2021. 1

[5] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind. Automatic differentiation in machine learning: a survey. 2015. 1

[6] J. Brown. Calculation of a constant Q spectral transform. *JASA*, 89:425, 01 1991. 6

[7] R. Chakraborty, Y. Xing, and S. X. Yu. Surreal: Complex-valued learning as principled transformations on a scaling and rotation manifold. *IEEE Trans. Neural Networks Learn. Syst.*, 33(3):940—-951, 2020. 1

[8] J. Diaz et al. The effect of color channel representations on the transferability of convolutional neural networks. pages 27–38, 04 2019. 7

[9] F. M. Dimentberg. The screw calculus and its applications in mechanics, 1968. WP-AFB, Ohio. 1

[10] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proc. IEEE Int. Conf. Comput. Vis.*, pages 1026—-1034, 2015. 6, 7

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE CVPR*, pages 770–778, 2016. 7

[12] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, page 448–456, 2015. 2, 4

[13] D. P. Kingma and J. L. Ba. Adam: A method for stochastic optimization. *3rd Int. Conf. Learn. Represent. ICLR 2015*, pages 1—15, 2015. 6

[14] R. Kiran and K. Khandelwal. Automatic implementation of finite strain anisotropic hyperelastic models using hyper-dual numbers. *Computational Mechanics*, 55, 12 2014. 1

[15] V. V. Kisil. Erlangen program at large–2: Inventing a wheel. the parabolic one. *arXiv: General Mathematics*, 2007. 3

[16] V. V. Kisil. Erlangen Program at Large-1: Geometry of Invariants. *SIGMA*, 6:076, Sept. 2010. 3

[17] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han. On the variance of the adaptive learning rate and beyond, 2019. 7

[18] I. Loshchilov and F. Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. 7

[19] T. Nitta. The computational power of complex-valued neuron. In *Artificial Neural Networks and Neural Information Processing, ICANN/ICONIP2003*, pages 993–1000, 2003. 1

[20] Y. Okawa and T. Nitta. Learning properties of feedforward neural networks using dual numbers. In *2021 APSIPA ASC*, pages 187–192, 2021. 2

[21] R. M. Shannon et al. Gravitational waves from binary supermassive black holes missing in pulsar observations. *Science*, 349(6255):1522–1525, 2015. 6

[22] U. Singhal, Y. Xing, and S. X. Yu. Co-domain symmetry for complex-valued deep learning, 2021. 1, 2

[23] J. O. Smith. Digital audio resampling. 5

[24] J. Thickstun, Z. Harchaoui, and S. M. Kakade. Learning features of music from scratch. *5th Int. Conf. Learn. Represent. ICLR 2017*, pages 1—14, 2017. 5

[25] K. S. Thorne. Gravitational waves, 1995. 6

[26] C. Trabelsi et al. Deep complex networks, 2018. 3, 4, 5, 6

[27] P. Virtue, S. X. Yu, and M. Lustig. Better than real: Complex-valued neural nets for mri fingerprinting, 2017. 3

[28] D. Zhang. Detecting gravitational waves using constant-Q transform and convolutional neural networks. In *2021 The 4th International Conference on Computational Intelligence and Intelligent Systems*, page 37–43, 2021. 6