**ABSTRACT**

Complex-valued deep learning has attracted increasing attention in recent years, due to its versatility and ability to capture more information. However, the lack of well-defined complex-valued operations remains a bottleneck for further advancement. In this work, we propose a geometric way to define deep neural networks on the space of complex numbers by utilizing weighted Fréchet mean. We mathematically prove the viability of our algorithm. We also define basic building blocks such as convolution, non-linearity, and residual connections tailored for the space of complex numbers. To demonstrate the effectiveness of our proposed model, we compare our complex-valued network comprehensively with its real state-of-the-art counterpart on the MSTAR classification task and achieve better performance, while utilizing less than 1\% of the parameters.

**Keywords** Fréchet Mean · Equivariance · Invariance

1 Introduction

In recent years, deep learning based learning models become powerful for various computer vision/machine learning tasks [LeCun et al., 1998, Bengio et al., 2009, Krizhevsky et al., 2012, He et al., 2016, LeCun et al., 2015]. Most of the deep learning based techniques are applicable to data that lie on vector spaces, but there is a dire necessity for incorporating geometry information into the network to develop techniques for “geometric deep learning”. Recently several researchers [Cohen and Welling, 2016, Chakraborty et al., 2018a,b, Esteves et al., 2018, Bronstein et al., 2017, Chakraborty et al., 2018c] have proposed deep learning based techniques for manifold-valued data. Here, we study the task of extending deep learning to naturally complex-valued data, where useful information is intertwined in both magnitudes and phases. For example, synthetic aperture radar (SAR) images, magnetic resonance (MR) images, and radio frequency (RF) signals are acquired in complex numbers, with the magnitude often encoding the amount of energy and the phase indicating the size of contrast or geometrical shapes. Even for real-valued images, their complex-valued representations could be more successful for many pattern recognition tasks; the most notable examples are the Fourier spectrum and spectrum-based computer vision techniques ranging from steerable filters [Freeman and Adelson, 1991] to spectral graph embedding [Maire et al., 2016, Yu, 2011].

A trivial solution is to extend the real-valued deep learning to the complex-valued data by treating the complex values as two-channel reals. But, such an Euclidean space embedding would not respect the intrinsic geometry of the space of complex numbers. For example, in MR and SAR images, the pixel intensity value could be subject to complex-valued scaling. One can train the model with explicit data augmentation [Krizhevsky et al., 2012, Dieleman et al., 2015, Wang et al., 2017] to overcome such ambiguity. But such extrinsic data manipulation requires an additional training time and is rather ineffective. Ideally, deep learning on complex valued data should be invariant to the group of non-zero scaling and planar rotation in the complex plane.
We treat each complex-valued data sample as a field in the space of complex numbers –a non-Euclidean space. This allows us to develop novel definitions of convolution and fully connected layers that can achieve respective equivariance and invariance to complex scaling.

A bottleneck to extend the definition of convolution from Euclidean spaces to non-Euclidean spaces is the lack of a proper vector space structure. As in the Euclidean space, we can move from one point to another using an element from the group of translations, the standard convolution operator is equivariant to the action of the group of translations. In the non-Euclidean space, for instance a hypersphere, however, translation equivariance is not meaningful, as it is not the group to move from one point to another. Instead, there always exists a rotation to go from one point to another on a hypersphere. On a space, we say a group $G$ acts transitively if there exists a $g \in G$ in order to go from one point to another on the space. Thus, for the Euclidean space, the group of translations act transitively; while on a hypersphere, it is instead the group of rotations. Thus, in order to extend the definition of convolution operator on a non-Euclidean space, we want the definition of convolution operator to be equivariant to the action of the group that transitively acts on the space.

This manifold view applies to both the domain and range of the data. In this work, we focus on the manifold perspective towards the range space of the data in order to extend deep learning to complex-valued images or signals. There is a long line of work that define convolution in a non-Euclidean space by treating each data sample as a function in that space [Worrall et al., 2017, Cohen and Welling, 2016, Cohen et al., 2017, Esteves et al., 2018, Chakraborty et al., 2018a, Kondor and Trivedi, 2018].

Our key insight is to represent a complex number by its polar form, such that the general group that acts on this space is the product group of planar rotations in the complex plane and non-zero scaling. This polar representation of the complex plane makes it into a Riemannian manifold. And in order to define deep convolutional neural networks, we define a convolution operator which is equivariant to the action of the aforementioned product group.

When a data sample is a field on a Riemannian manifold,

- Convolution defined by weighted Fréchet mean (wFM) [Fréchet, 1948] is equivariant to the group that naturally acts on that manifold [Chakraborty et al., 2018b].
- Since wFM is non-linear and acts like a contraction mapping [Mallat, 2016] analogous to ReLU or sigmoid, non-linear activation functions such as ReLU may not be needed.
- By taking the Riemannian geometric point of view, we could also use tangent ReLU/transport operator for better accuracy.
- We further propose a distance transform as a fully-connected layer operator that is invariant to complex scaling. It takes complex-valued responses at a previous layer to the real domain, where all kinds of standard CNN functions can be subsequently used.

Figure 1: An overview of our model: given a complex valued image we use multiple complex convolutions followed by real valued standard convolution. The complex valued response is color-coded in HSV with hue and value represent phase and magnitude respectively, we use saturation to be 1.
A neural network equipped with our wFM filtering and distance transform on complex-valued data has a group invariant property similar to the standard CNN on real-valued data. Existing complex-valued CNNs tend to extend the real-valued counterpart to the complex domain based on the form of functions [Bunte et al., 2012; Trabelsi et al., 2017], e.g., convolution or batch normalization. None of the complex-valued CNNs are derived by studying the desired property of functions, such as equivariance or linearity. Our complex-valued CNN is composed of layer functions with all the desired properties and is a theoretically justified analog of the real-valued CNN.

On evaluation of the MSTAR classification task, for example, our model consistently achieves higher accuracy compared to the baseline real-valued CNN. We achieved a highest of 99.25 percent testing accuracy compared to 99 percent of the ResNet50 model, while using less than 1 percent of the baseline model parameters.

To summarize, we make three major contributions.

1. We propose novel complex-valued CNNs with theoretically proven equivariance and invariance properties.
2. We provide experimental validation of our method on complex-valued data classification tasks, demonstrating significant performance gain at a fraction of the baseline model size.
3. Furthermore, we extend our framework and propose a complex valued residual network.

These results demonstrate significant benefits of proposing CNN layer functions in terms of desirable intrinsic properties on the complex plane as opposed to applying the standard CNN to the 2D Euclidean embedding of complex numbers.

2 Related Works and Motivation

Complex signal analysis has yielded major interest along recent years. The demand of visual and audio signal classification tasks in modern communication systems has resulted in a variety of new methods being developed. While traditional methods utilize higher order statistics such as variance fractal dimension trajectory [Kinsner and Grieder, 2010] and spectral analysis [Reichert, 1992] present adequate predictions, neural approaches have attracted considerable attention in recent years.

Complex representation of data has increasingly became a research of interest. Complex valued data poses a variety of advantages over real-valued data, including easier optimization, better generalization characteristics, faster learning, and to allow for noise-robust memory mechanisms [Nitta, 2002; Hirose and Yoshida, 2012; Arjovsky et al., 2016]. Furthermore, complex-valued representations have been used to construct more biologically plausible models [Reichert and Serre, 2013].

The advantages of complex-valued data has inspired development in complex-valued deep learning. [Maire et al., 2016], utilizes a complex representation to simultaneously embed both the prediction confidence and the relative depth between neighboring pixels. Similarly, [Amin and Murase, 2009] attempts to learn a mapping from finite range of real values to the complex unit circle. [Cadieu and Olshausen, 2012] trains a complex-valued sparse coding model to capture edge structure and motion structure. [Arjovsky et al., 2016] learns complex embeddings in neural networks by using complex-valued weight matrices at each hidden layer.

3 A convolutional neural network on complex valued data

In this section, we present a convolutional neural network on the complex valued field, denoted by \( C \). We identify the space of complex numbers in its polar form and identify the space of complex numbers with a non-Euclidean Riemannian manifold. With this identification, we naturally need to generalize the definition of standard convolution operator on \( C \). In order to generalize, the first natural thing to ask is what property to retain in this generalized definition of convolution? We define a new convolution operator on \( C \) and have shown that it retains the equivariance property. We have also introduced non-linearities and invariant layer to make the entire network invariant under isometric transformations. In order to develop the deep network on \( C \) valued field, we first present the geometry of the manifold of complex numbers, i.e., \( C \). Then, we develop a convolutional neural network (CNN) framework for complex-valued data. Before developing this framework, we will talk about key properties of a CNN architecture, specifically, (a) the equivariance property of the convolution operator (b) the invariance property of a CNN. We will also point out the implications of these key properties after presenting the necessary background related to the space of complex numbers.

**Space of complex numbers:** A (smooth) manifold of complex numbers, \( C \), consists of elements of the form \( a + ib \), where \( a, b \in \mathbb{R} \). It is a Riemannian manifold [Boothby, 1986] and the distance induced by the canonical Riemannian
Another way to represent a complex number is by its polar form, i.e., represent each complex number by its phase and magnitude. As expected, we can show that this identification is a bijective function. In the rest of the paper, we will use the polar form to represent a complex number. We formally define the polar representation as:

**Definition 1.** We identify each complex number, \( a + ib \), with its polar form, i.e., \( r \exp i\theta \), where \( r \) and \( \theta \) are the absolute value/magnitude (\( \text{abs} \)) and argument (\( \text{arg} \)) of \( a + ib \). Hence, we can identify \( \mathbb{C} := \mathbb{C} \setminus \{0 + i0\} \) as \( \mathbb{R}^+ \times \text{SO}(2) \), where \( \mathbb{R}^+ \) is the manifold of positive reals and \( \text{SO}(2) \) is the manifold of planar rotations. Let \( F : \mathbb{C} \to \mathbb{R}^+ \times \text{SO}(2) \) be the mapping which is given by

\[
a + ib \mapsto \left( r, \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \right),
\]

where, \( r = \text{abs}(a + ib) = \sqrt{a^2 + b^2} \) and \( \theta = \text{arg}(a + ib) = \arctan(y/x) \).

Observe that \( F \) is a bijective mapping. Hence, we can go back and forth between \( \tilde{\mathbb{C}} \) and \( \mathbb{R}^+ \times \text{SO}(2) \) using \( F \) and its inverse. In this representation, the induced distance from Eq. (1) is defined as follows. Using the identification, we can define a metric on \( \tilde{\mathbb{C}} \) as follows. Given, \( z_1, z_2 \in \tilde{\mathbb{C}} \), let, \( (r_1, R_1) = F(z_1) \) and \( (r_2, R_2) = F(z_2) \). Then,

\[
d(z_1, z_2) = \sqrt{\log \left( \frac{r_1}{r_2} \right)^2 + \| \log m \left( R_1^{-1}R_2 \right) \|^2},
\]

where, \( \log m \) is the matrix logarithm. Furthermore with the above identification, \( \tilde{\mathbb{C}} \) is a Riemannian homogeneous space [Helgason, 1962]. The next two propositions is crucial for the construction of the convolution operator. Before proving the next proposition, we restate some definition borrowed from group theory literature [Dummit and Foote, 2004].

**Definition 2.** [Dummit and Foote, 2004]

1. Given a (Riemannian) manifold \( \mathcal{M} \) and a group \( G \), we say that \( G \) acts on \( \mathcal{M} \) (from left) if there exists a mapping \( L : \mathcal{M} \times G \rightarrow \mathcal{M} \) given by \( (X, g) \mapsto g.X \) satisfies (a) \( L(X,e) = e.X = X \) (b) \( g.(h.X) = (gh).X = g.(h.X) \).

2. An action is called a transitive action iff given \( X, Y \in \mathcal{M} \), \( \exists \ g \in G \), such that \( Y = g.X \).

**Proposition 1.** Using the identification in Def. (1), the group, \( G := \{ R \setminus \{0\} \} \times \text{SO}(2) \) transitively acts on \( \tilde{\mathbb{C}} \) and the action is given by \(( (r, R), (r_g, R_g) )) \mapsto (r_g r r_g, R_g R) \).

**Proof.** Using Definition (2), we can easily show that \(( (r, R), (r_g, R_g) )) \mapsto (r_g r r_g, R_g R) \) is a valid group action. Observe that given \( (r_1, R_1) \) and \( (r_2, R_2) \), we can choose \( r_g = \sqrt{r_2/r_1} \) and \( R_g = R_2 R_1^{-1} \) such that \( Y = g.X \), where, \( g = (r_g, R_g) \), \( X = (r_1, R_1) \) and \( Y = (r_2, R_2) \).

Now that we know the group \( G := \{ R \setminus \{0\} \} \times \text{SO}(2) \) acts on \( \tilde{\mathbb{C}} \) transitively, we show that elements of the group \( G \) preserves the distance as defined in Eq. (2), i.e., \( G \) is a set of isometries on \( \tilde{\mathbb{C}} \). This is formally stated in the following proposition.

**Proposition 2.** Given \( z_1 = (r_1, R_1), z_2 = (r_2, R_2) \in \mathbb{C} \) and \( g = (r_g, R_g) \in G \), \( d(g.z_1, g.z_2) = d(z_1, z_2) \).

**Proof.** The proof follows from Eq. (2) as

\[
d(g.z_1, g.z_2) = \left( \log \left( \frac{r_1}{r_2} \right)^2 + \| \log m \left( R_1^{-1}R_2 \right) \|^2 \right)^{0.5}
\]

\[
= \left( \log \left( \frac{r_1}{r_2} \right)^2 + \| \log m \left( R_1^{-1}R_2 \right) \|^2 \right)^{0.5}
\]

\[
= d(z_1, z_2)
\]
Equipped with the basic properties of complex numbers, we first formally define the convolution operator and then in-turn define a CNN. Afterwards, we show that our definition of convolution and CNN satisfy the aforementioned two properties, *equivariance of convolution operator* and *invariance of CNN*. Moreover, we will show the similarities of our convolution operator with the standard Euclidean convolution operator.

**Convolution operator:** We will use *weighted Fréchet mean (FM)* ([Fréchet, 1948](#)) to define convolution layer (the operator is denoted by *). Given \{z_i\}_{i=1}^K \subset \mathbb{C} and \{w_i\}_{i=1}^K \subset (0,1] with \sum_i w_i = 1, (the) *weighted Fréchet mean (FM)* (wFM) is defined as:

\[
wFM(\{z_i\}, \{w_i\}) = \arg \min_{m \in \mathbb{C}} \sum_{i=1}^K w_i d^2(z_i, m),
\]

(3)

Here, \(d\) is the distance defined in Eq. (2). And we define the convolution as

\[
\{z_i\} * \{w_i\} = \text{wFM}(\{z_i\}, \{w_i\})
\]

In the above definition wFM can be regarded as the minimizer of the weighted variance. Notice that in the above defined convolution operator, \{w_i\} is the learnable filter and wFM (\{z_i\}, \{w_i\}) \in \bar{\mathbb{C}} is the convolution output. The filter \{w_i\} will be learned through stochastic gradient descent. As \(w_i\) is real valued, one can use the standard SGD, but additionally we will ensure the convexity constraint on \{w\}.

We like to point out that, analogous to the standard convolution network, here for each complex convolution layer we have multiple channels. So for each convolution layer with input \{f_{i,j}\}_{i=1,j=1}^{N_1,c_{in}} \subset \mathbb{C} with \(c_{in}\) number of input channels, we write the convolution with \(c_{out}\) output channels as

\[
\hat{*}(\{f_{i,j}\}, \{w_{k,j}\}, c_{in}, c_{out}) = \text{wFM}(\{f_{i,j}\}, \{w_{k,j}\}, c_{in}, c_{out})
\]

(4)

with \(l \in \{1, \cdots, c_{out}\}\), \(k \in \{1, \cdots, K\}\), and \(w_{k,j} > 0\) and \(\sum_k w_{k,j} = 1\). Notice that in the above notation of convolution using wFM, we implicitly share the kernel \(\{w_{k,j}\}\) across the input dimension \(N_1\).

Thus, in order to apply a kernel of size \(K\) on a complex valued signal with number of input channels \(c_{in}\) and to generate an output with \(c_{out}\) channels, we need to learn filters, \(w\) of size \(c_{in} \times K \times c_{out}\).

Now, that we have a valid definition of a convolution, we show what properties this new definition of convolution preserves.

**Equivariance property of convolution:** In the Euclidean convolution definition, i.e., defined on \(\mathbb{R}^n\), the convolution operator is equivariant to translation, i.e., fixing the kernel of convolution if the input signal is translated by \(t\), the convolution output is going to be translated by \(t\) as well. This is a desirable property for standard convolution as it enables weight sharing across the entire domain (image for most computer vision applications). But a natural question to ask is *What is so special about translation?* Observe that the group of translations is the group of isometries for \(\mathbb{R}^n\) and moreover transitively acts on \(\mathbb{R}^n\) (as defined in Def. (2)).

From Props. (1) and (3), we know that on \(\bar{\mathbb{C}}\), \(G = \{\mathbb{R} \setminus \{0\}\} \times \text{SO}(2)\) transitively acts and is a group of isometries, hence in order to generalize the Euclidean convolution operator on \(\mathbb{C}\), we need to define an operator which is equivariant to the action of \(G\). This motivates us to explore a definition of convolution operator suitable for \(\bar{\mathbb{C}}\). Recently in [Chakraborty et al., 2018b](#), the authors proposed a CNN framework on manifold valued data. They defined a convolution operator on a manifold \(\mathcal{M}\) which is equivariant to the group, \(G\), that acts on \(\mathcal{M}\). In this setting, \(\mathcal{M} = \mathbb{C}\) and \(G = \text{SO}(2) \times \mathbb{R} \setminus \{0\}\) and hence we defined the convolution operator on \(\mathbb{C}\) as in Eq. (5).

Now, we will formally state the equivariance property of a convolution operator and then justify the choice of wFM by drawing some analogy from standard convolution operator.

**Proposition 3.** The convolution definition as given in Eq. (5) is equivariant to the action of \(G = \{\mathbb{R} \setminus \{0\}\} \times \text{SO}(2)\).

Proof. The statement of the proposition can be written analytically as

\[
\hat{*}(\{g(z_i)\}, \{w_i\}) = g.\hat{*}(\{z_i\}, \{w_i\})
\]

(5)

Observe that, using Prop. (2), one can easily show the above identity. \(\square\)

In Fig (4), we have shown the equivariance property with respect to rotation and scaling. While the colored dots are the sample points on the complex plane, the white dots are the wFM. Observe from the figure that, if all the colored dots are transported using an element of \(G\), then the wFM also going to be transported by the same element.
Below, we justify the usage of wFM as our choice of convolution operator.

Why wFM?: In order to justify the choice of wFM as convolution operator in Eq. (3), we first remind the readers of the definition of standard Euclidean convolution operator. Notice that the standard convolution operator can be written as \( \sum_i w_i x_i \), where \( \{w_i\} \) is the filter and \( \{x_i\} \) is the signal. Now notice that with the convexity constraint on \( \{w_i\} \), \( \sum_i w_i x_i \) is the wFM on the Euclidean space as it is the minimizer of the weighted variance as defined in Eq. (3). Now, observe that the convexity constraint is to ensure that the result stays on the space \( \mathcal{M} \). Thus the choice of utilizing wFM as the convolution operator may seem arbitrary at first, is an obvious choice if we look at the Euclidean convolution operator as the minimizer of weighted variance. Now that we have a definition of convolution on our hand, we will define a non-linear activation function. Though in [Chakraborty et al., 2018b], the authors argued that as the convolution operator defined in Eq. (3) is non-linear and is a contraction mapping [Chakraborty et al., 2018b], we found out that using a ReLU like activation accelerates the optimization step. Hence, we define two possible non-linear operators, namely \text{tangent ReLU} and \text{G-transport}.

\text{tangent ReLU (tReLU)}: Similar to the ReLU operator on \( \mathbb{R}^n \), tReLU is a function from \( \mathcal{M} \) to \( \mathcal{M} \) as is defined as: \((r, R) \mapsto \text{Exp} \left( \iota^{-1} (\text{ReLU} (\log (r), \iota (\logm (R)))) \right)\), where, \( \text{Exp} (a, B) = (\exp (a), \expm (B)) \). \( \expm \) is the matrix exponential operator. Note that, here \( \iota \) is the mapping from the space of \( 2 \times 2 \) skew-symmetric matrices (output of \( \logm \)) to \( \mathbb{R} \), i.e., the corresponding vector space identification. A pictorial description of tReLU is given in Fig. (3).

\text{G-transport}: Given \((r, R) \in \hat{\mathcal{C}}\), we transport it to \((r g^2, R g R)\), where we learn \((r g, R g) \in G\). Given a complex-valued field with values \( \{(r_i, R_i)\}_{i \in I}\), we learn \((r g, R g) \in G\) and transport the values to \( \{(r_i g^2, R_i g R_i)\}_{i \in I} \).

With the convolution operator and a non-linear operator in our hand, we are now ready to define a deep convolutional network. Notice that analogous to standard CNN, we want our complex valued CNN to be invariant (the remaining desired property of a CNN) which is our next topic of discussion.

Invariance property of CNN: In standard Euclidean CNN, the entire network is invariant to the action of the group of translations. This ensures that the output does not change if the input is translated. This invariance is achieved by the
last softmax fully connected (FC) layer. It is easy to show that the standard softmax FC layer is invariant to translation. Similar to our earlier discussion, on $\tilde{C}$ we want the CNN to be invariant to the action of $G$. We will now define the invariant layer and will show that this layer is invariant to the action of $G$.

**Invariant layer:** Let $\{t_{i,j}\}_{i=1,j=1}^{d,c} \subset \tilde{C}$ be the output of the last complex convolution layer, where $d$ is the spatial resolution and $c$ is the number of channels of the last complex convolution layer. Then, we define the invariant layer with inputs $\{t_{i,j}\}_{i=1,j=1}^{d,c}$ and outputs $\{u_{i,j}\}_{i=1,j=1}^{d,c} \subset \mathbb{R}$ as

$$u_{i,j} = d(t_{i,j}, t^u),$$

(6)

where, $t^u = \ast \left(\{t_i\}, \left\{v^j_i\right\}\right)$. We will learn the filter $\left\{v^j_i\right\}$ as before. As obvious from the construction itself, we can show that this layer is invariant to the action of $G = \{\mathbb{R} \setminus \{0\}\} \times SO(2)$ as formally stated next.

**Proposition 4.** The above defined layer is invariant to the action of $G$.

**Proof.** If all $\{t_{i,j}\}_{i=1,j=1}^{d,c}$ are transported using an element $g \in G$, then so is $t^u$ using proposition (3). Hence, after the action using the element $g \in G$, $t_{i,j} \mapsto g \cdot t_{i,j}$ and $t^u \mapsto g \cdot t^u$. Now, as $g \in G$,

$$\forall i, j, d(t_{i,j}, t^u) = d(g \cdot t_{i,j}, g \cdot t^u)$$

(7)

This proves the invariance under the action of $G$.

The aforementioned layer essentially quotient out the action of $G$. This is an informal explanation that the output of this layer is $\mathbb{R}$ valued. In Fig. (2), we have also shown the invariance property with respect to rotation and scaling. This can be observed by noticing the invariance of the distances between $wFM$ and the sample points.

**Standard Convolution and Fully connected layer:** The output of the invariant layer is $\{u_{i,j}\}_{i=1,j=1}^{d,c}$, where $d$ is the spatial dimension and $c$ is the channel dimension. Then, one can use standard convolution layer with input a 1-D field of dimension $d$ and $c$ the number of channels (note that, for a 2-D complex valued input field, the output of the invariant layer is a 2-D real valued field). As we quotient out $G$ in the invariant layer, using standard Euclidean convolution afterwards is meaningful. After standard convolution layer(s), we will use fully connected softmax layer for classification.

**RotLieNet:** With the above building blocks, we have a complex valued CNN framework (dubbed as RotLieNet) which is invariant to the action of $G$. A schematic description of this network is given in Fig. (4).

![Figure 4: Schematic diagram on RotLieNet](image-url)

With the inception of a residual network, it is a well-known fact that a residual network outperforms a standard CNN for a deep network. A natural question to ask is that **how we can define a complex residual network?** which we explain next.

7
**Complex Residual Network:** Residual structures are known for better performance in deeper networks. It addresses and prevents both the explosive/vanishing gradient problem and the accuracy degradation dilemma. Inspired by ResNet [He et al., 2016], we propose a residual structure for our complex-valued convolution. While addition is natural for residual structures in the real number field, it does not make sense to utilize the same method in the complex field, where distance metrics are inherently different. We first formulate our residual block as follows. Given \( \{f_{i,j}\}_{i=1,j=1}^{N_2,c_2} \), \( \{\tilde{f}_{i,j}\}_{i=1,j=1}^{N_1,c_2} \subset \mathbb{C} \) as the two complex valued fields with \( N_1 < N_2 \), we first use wFM to reduce the spatial dimension of \( \tilde{f} \) as

\[
\tilde{\ast}_1 \left( \{\tilde{f}_{i,j}\}, \{\tilde{w}_{k,j}\}, c_2, c_2 \right).
\]

Let \( \{\tilde{f}_{i,j}\}_{i=1,j=1}^{N_1,c_2} \subset \mathbb{C} \) be the output of wFM with spatial dimension \( N_1 \) and number of output channels \( c_2 \).

Then, we concatenate two fields namely \( \{\tilde{f}_{i,j}\} \) and \( \{f_{i,l}\} \) along the channel dimension to get \( \{g_{i,m}\} \) as

\[
\{g_{i,m}\} = \left[ \{\tilde{f}_{i,j}\} | \{f_{i,l}\} \right]
\]

Note that, \( g \) has spatial dimension \( N_1 \) and number of channels \( c_1 + c_2 \). Then, we define the convolution on \( g \) as

\[
\tilde{\ast}_2 \left( \{g_{i,j}\}, \{w_j^l\}, c_1 + c_2, c_3 \right)
\]

with output channels \( c_3 \) and the kernel \( w_j^l > 0 \) with \( \sum_{j=1}^{c_1+c_2} w_j^l = 1 \) for all \( l \in \{1, \cdots, c_3\} \).

We can replace the complex convolution layer with this residual layer and can build a residual complex convolution network. Furthermore, we introduce another essential block in our framework, namely tensor ring, to reduce the number of parameter is our model.

**Tensor Ring:** By decomposing the convolution filter \( W \in \mathbb{R}^{l_1 \times l_2 \cdots \times l_d} \) of \( \prod_{i=1}^d I_i \) degrees of freedom into \( d \) independent rank \( r \) tensors, \( \{T_i \in \mathbb{R}^{r \times I_i \times r}\}_{i=1}^d \) such that, \( \forall i_1, \cdots, i_d, \)

\[
W(i_1, \cdots, i_d) = \text{trace} \left( T_1(:, i_1, :) \times \cdots \times T_d(:, i_d, :) \right),
\]

where, \( \times \) is matrix multiplication. This decomposition needs \( r^2 \sum_i I_i \) parameters instead of \( \prod_{i=1}^d I_i \). In [Zhao et al., 2016; Oseledets, 2011], the authors show that the tensor ring decomposition can achieve arbitrary approximation error.

With these essential building blocks for a complex valued deep neural network, we will now present some results for experimental validations.

**4 Experimentation**

We comprehensively evaluate the effectiveness of our method on the MSTAR dataset [Keydel et al., 1996]. The MSTAR dataset is a collection of complex-valued X-band SAR images of 11 distinct target types. The dataset consists of roughly 15,000 samples, of which six thousand are images taken at 17-degree depression and another six thousand are taken at 15-degree depression. All images are of the size \( 128 \times 128 \). We preprocess the images by center cropping the images into size of \( 100 \times 100 \), and use polar representation, i.e., phase and magnitude to represent each pixel. We compare our results on the MSTAR classification problem with a ResNet50 [He et al., 2016], which is also used as baseline from [Shao et al., 2018]. The success of ResNet50 on various image classification tasks makes it an excellent baseline for comparing our model. While ResNet contains a total of 23 million parameters, our model is extremely light-weight, and yields between 40,000 to 60,000 parameters. We have reduced the parameters further by replacing the standard convolution layers with tensor-ring architecture [Oseledets, 2011; Zhao et al., 2016] as described in section (3). The class distribution for the MSTAR dataset is shown in Fig. (8). Now, we give the specification of the network architecture.

**4.1 Model Architecture and Implementation Details**

We propose RotLieNet with and without the residual connection. In Fig. 7, we have given a schematic description of our proposed RotLieNet. Both these models are trained using the GeForce RTX 2080 GPU for a total of 120 epochs, using Adam optimizer and cross-entropy loss. As mentioned before, our model consists of a combination of both complex and real convolutional layers. While regular convolutional layers are known to process real-valued data efficiently, our complex convolutional layers can learn both phase and magnitude information with greater ease. After
Figure 5: Using minimal amount of training data, our model can achieve relatively higher classification accuracy compared to the baseline ResNet50 model.

Table 1: Our RotLieNet Architecture

| Layer Type                  | Input Shape  | Kernel Size | Stride | Output Shape |
|-----------------------------|--------------|-------------|--------|--------------|
| Complex Convolution         | [2, 1, 100, 100] | (5, 5)      | (2, 2) | [2, 20, 48, 48] |
| G-transport                 | [2, 20, 48, 48] | -           | -      | [2, 20, 48, 48] |
| Complex Convolution         | [2, 20, 48, 48] | (5, 5)      | (2, 2) | [2, 20, 48, 48] |
| Invariant layer             | [2, 20, 22, 22] | -           | -      | [2, 20, 22, 22] |
| Batch Norm and ReLU         | [30, 18, 18]  | (5, 5)      | (1, 1) | [30, 18, 18]  |
| Conv2d                      | [20, 18, 18]  | (2, 2)      | -      | [30, 9, 9]     |
| Batch Norm and ReLU         | [30, 2, 2]    | (5, 5)      | (3, 3) | [30, 2, 2]     |
| Conv2d                      | [30, 2, 2]    | -           | -      | [30, 1, 1]     |
| Batch Norm and ReLU         | [30, 1, 1]    | (2, 2)      | (1, 1) | [30, 1, 1]     |
| FC Layer                    | [30]          | -           | -      | [50, 10, 10]   |
| FC Layer                    | [50]          | -           | -      | [10, 1, 1]     |

Table 2: Our Complex Residual Network Architecture

| Layer Type                  | Input Shape  | Input Shape 2 | Kernel Size | Stride | Output Shape |
|-----------------------------|--------------|---------------|-------------|--------|--------------|
| Complex Convolution         | [2, 1, 100, 100] | -             | (5, 5)      | (2, 2) | [2, 20, 48, 48] |
| G-transport                 | [2, 20, 48, 48] | -             | -           | -      | [2, 20, 48, 48] |
| Complex Convolution         | [2, 20, 48, 48] | -             | (5, 5)      | (2, 2) | [2, 20, 48, 48] |
| Invariant layer             | [2, 20, 22, 22] | -             | -           | -      | [2, 20, 22, 22] |
| Batch Norm and ReLU         | [30, 18, 18]  | (5, 5)       | (1, 1)     | -      | [30, 18, 18]  |
| Real Residual Block         | [30, 18, 18]  | -             | -           | -      | [30, 18, 18]  |
| Maxpool2d                   | [40, 18, 18]  | (2, 2)       | (2, 2)     | -      | [40, 9, 9]     |
| Conv2d                      | [40, 18, 18]  | (5, 5)       | (3, 3)     | -      | [30, 2, 2]     |
| Batch Norm and ReLU         | [50, 2, 2]    | -             | -           | -      | [30, 2, 2]     |
| Real Residual Block         | [50, 2, 2]    | (2, 2)       | (1, 1)     | -      | [50, 1, 1]     |
| Batch Norm and ReLU         | [50, 1, 1]    | -             | -           | -      | [30, 1, 1]     |
| FC Layer                    | [70]          | -             | -           | -      | [30, 1, 1]     |
| FC Layer                    | [30]          | -             | -           | -      | [10, 1, 1]     |

using the alternative cascaded complex convolution layers with tReLU/G-transport layers, we use the invariant layer to map the complex valued outputs to real valued field (as mentioned in section (3)). Afterwards, we can use cascaded 2D standard convolution layers to extract features by looking at the spatial interaction in real valued field. Finally, we use fully connected layer(s) to output our final prediction. We also like to point out that a choice of tReLU or G-transport did not affect the experimental results.

**Residual Architecture:** In both $15^\circ/17^\circ$ and seen/unseen data split, we use the residual RotLieNet to boost the performance. On $15^\circ/17^\circ$ data partition, we achieve 0.3% performance gain, while on seen/unseen data split, we get 1% classification accuracy gain. This though may not look like a significant performance gain at first, but a careful observation reveals that in both these cases baseline models reach 100% training accuracy while achieving above 97% testing accuracy. Hence, performance boost even after the apparent saturation of the training accuracy proves the
effectiveness of our proposed residual structure. Furthermore, for seen/unseen data partition, we report the confusion matrix using our proposed residual RotLieNet in Fig. 12. We can see that compared with the confusion matrix using RotLieNet in Fig. 12, the residual counterpart significantly better in some classes which justifies the overall performance boost.

Figure 6: A visualization for the convolution pipeline for all 11 classes of the MSTAR dataset. For conciseness, we only include 10 channels for every convolution filter output we visualize. We encode the complex-valued filter outputs using the hsv colormap, and encode the real-valued filter outputs using gray-scale colormap. The feature column represents the final input to the fully connected layer.

For our residual RotLieNet model, we utilize both complex and real residual blocks. We effectuate our complex residual blocks by concatenating outputs from the first and second complex convolution layers and learn a weighted Fréchet mean transform as the output. We implement our real residual blocks by stacking three 2d convolutions with stride (1, 1) and kernel size (1, 1), (3, 3), (1, 1) respectively. In order to preserve spatial dimensions, we zero pad the inputs. We use batch norm and ReLU followed by maxpool. For residual network, we position all residual blocks after batch-norm and ReLU layers.

We have presented the model architecture for our RotLieNet and residual RotLieNet in Tables 1 and 2 respectively. We compare model performances with a variety of splitting metrics in order to substantiate the superiority of our model.

Figure 7: Sample architecture of a complex-valued CNN (Left: complex convolutional network, Right: complex residual network) used for MSTAR classification. It consists of complex-valued convolution layers, manifold ReLU nonlinear activation layers, a wFM invariant layer, regular 2d-convolution layers and a fully connected layer at the end, together achieving invariance to complex scaling in the range domain.

4.2 Results

In this section, we present the experimental results in terms of classification accuracy and number of parameters used. We use different partition of the MSTAR data to evaluate the effectiveness of our proposed methods. We have shown...
Figure 8: Class distribution for the original MSTAR dataset (in log scale). It consists of 10 distinct target classes, as well as an additional clutter class.

The number of parameters used in our models compared to baseline in Fig. (5). We can see that we are just using < 1% of the baseline model. Below, we will see that using a much leaner model we can achieve better performance than the baseline model.

10-class Random Split: We randomly shuffle the MSTAR data using a fixed seed. We then train both our model and the baseline ResNet50 using different ratios of the original data, namely 1%, 5%, 10%, 20%, 30% of the original data is used for training and the rest for testing. In this split of the dataset, we evaluate the performance of both of our methods on the 10-class MSTAR classification task, ignoring the 11th clutter class. We show the results of both the baseline and our model with different training splits in figure (5). For smaller training percentage, we can see significant boost in the performance of our proposed method. We can achieve 3% performance gain while using 10% of the data during training. For larger fraction of training data, the performance of both the baseline and our model are comparable and one possible reason can be the convergence of the training which can be verified by observing that using larger training partition, training accuracy reaches the maximum as evident from Fig. (9). Because proposals of our model are competitive with the baseline on the 10-class classification task, we may expect our model to achieve better accuracy on the 11-class classification task due to our models ability to extract extra information from the polar representation of the complex valued data. The following experiments justify this hypothesis.

15°/17° split: We include all 11 classes for experimentation in the proceeding sections. To demonstrate the resilience of our model, we partition the MSTAR dataset into groups of images taken at 15 degree depression and at 17 degree depression as in [Ding et al., 2016]. We utilize the images taken at 17 degree depression for training, and images taken at 15 degree depression for testing. The resulting model performances are shown on Fig. (11). We can see that, our model can achieve 99.145% testing accuracy compare to 99.054% using the baseline model. The class distribution for the 15°/17° split dataset is shown in Fig. (10). Due to the heterogeneous nature of the MSTAR dataset, further elaborate partition of data is possible for experimentation.

Table 3: Seen / Unseen Dataset Split

| Dataset | BMP2 | BTR70 | T72 | BTR60 | 2S1 | BRDM2 | D7 | T62 | ZIL131 | ZSU23/4 | Clutter |
|---------|------|-------|-----|-------|-----|-------|----|-----|-------|--------|--------|
| Training| sn21 | sn9563 | sn9563 | c71 | sn132 | sn812 | sn87 | c256 | 299 | 298 | 299 | 299 | 299 | 299 |
| Testing | 196 | 195 | 196 | 196 | 196 | 195 | 191 | 195 | 274 | 274 | 274 | 273 | 274 | 274 | 1159 |

Seen/Unseen Splitting: Following the work of [Wang et al., 2017], we partition the test data to two separate sets to evaluate the performance of the proposed method on never-seen model sub-classes. Due to the in-class variance of the MSTAR dataset, certain subclass models of target classes are only present in the testing dataset. We sort the testing set separately on seen and unseen target serial numbers. The unseen test set consists of targets with serial numbers never shown to the network in the training phase, specifically BMP2 serial numbers sn9563 and sn9566 and T72 serial...
numbers sn812 and sns7. The seen test set contains the rest of the test samples. The split is given in Table (3). We present the performance of our proposed model in compared with the baseline model in Fig. (11). As a baseline we have used the model proposed by [Wang et al., 2017]. We can see that using our proposed RotLieNet, we can achieve 97.688% compared to 97.407% classification accuracy using the model proposed in [Wang et al., 2017]. Furthermore, we have shown in Fig. (12), the confusion matrix using our proposed RotLieNet. We can see that on the largest three classes (in terms of number of samples), we achieved perfect classification accuracy.

5 Conclusions and future directions

Deep learning becomes ubiquitous in most of the machine learning/computer vision tasks. Though most of the deep learning based techniques assume that the data lies on a vector spaces, this assumption is not actually practical. Using celebrated Nash embedding theorem [Boothby, 1986], though one can embed the data lying on a manifold in a higher dimensional vector space, this results in an increase in complexity of the trainable model. Several researchers in the recent past have developed techniques for “geometric deep learning” by defining deep learning tools for spaces with some geometry constraint including graphs, surfaces etc.. One of the popular way to deal with complex numbers is to
Figure 11: MSTAR classification accuracy plot for the 15/17 angle depression split and the unseen testing set split. Our model demonstrates better performance in both scenarios, justifying the capability of our model to learn both phase and magnitude information, in contrast with real-valued convolutional neural networks.

Figure 12: Confusion matrices (Left: RotLieNet, Right: RotLieNet with Residual Structure) for MSTAR classification. Both models are trained on the seen/unseen split. Our residual-augmented model demonstrates further increased accuracy in the majority of classes.

treat them as two-channel reals, though this can alleviate the bottleneck of using deep network for the space of complex numbers, this does not respect the underlying geometry. In this paper, we proposed a geometric way to define deep network on the space of complex numbers. We have defined convolution, non-linearity, residual connections tailored for the space of complex numbers. Several experimental results demonstrate the superior performance of our proposed complexnet over the baseline model. This clearly suggests the usefulness our proposed complex valued deep learning. Furthermore, because our proposed model utilizes the underlying geometry, we can utilize a much leaner model to achieve the superior performance. As a possible future direction, we will generalize other neural network operations for complex valued data and see the effectiveness of our proposed complexnet on other datasets including radio-frequency and MR imaging datasets.

Acknowledgements

The authors thank Liu Yang for the generous help with creating figures used in this manuscript.
References

Md Faijul Amin and Kazuyuki Murase. Single-layered complex-valued neural network for real-valued classification problems. *Neurocomputing*, 72(4-6):945–955, 2009.

Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary evolution recurrent neural networks. In *International Conference on Machine Learning*, pages 1120–1128, 2016.

Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

William M Boothby. *An introduction to differentiable manifolds and Riemannian geometry*, volume 120. Academic press, 1986.

Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

Kerstin Bunte, Frank-Michael Schleif, and Michael Biehl. Adaptive learning for complex-valued data. In *ESANN*. Citeseer, 2012.

Charles F Cadieu and Bruno A Olshausen. Learning intermediate-level representations of form and motion from natural movies. *Neural computation*, 24(4):827–866, 2012.

Rudrasis Chakraborty, Monami Banerjee, and Baba C Vemuri. H-cnns: Convolutional neural networks for riemannian homogeneous spaces. *arXiv preprint arXiv:1805.05487*, 2, 2018a.

Rudrasis Chakraborty, Jose Bouza, Jonathan Manton, and Baba C Vemuri. Manifoldnet: A deep network framework for manifold-valued data. *arXiv preprint arXiv:1809.06211*, 2018b.

Rudrasis Chakraborty, Chun-Hao Yang, Xingjian Zhen, Monami Banerjee, Derek Archer, David Vaillancourt, Vikas Singh, and Baba Vemuri. A statistical recurrent model on the manifold of symmetric positive definite matrices. In *Advances in Neural Information Processing Systems*, pages 8883–8894, 2018c.

Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999, 2016.

Taco Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Convolutional networks for spherical signals. *arXiv preprint arXiv:1709.04893*, 2017.

Sander Dieleman, Kyle W Willett, and Joni Dambre. Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly notices of the royal astronomical society*, 450(2):1441–1459, 2015.

Jun Ding, Bo Chen, Hongwei Liu, and Mengyuan Huang. Convolutional neural network with data augmentation for sar target recognition. *IEEE Geoscience and remote sensing letters*, 13(3):364–368, 2016.

David Steven Dummit and Richard M Foote. *Abstract algebra*, volume 3. Wiley Hoboken, 2004.

Carlos Esteves, Christine Allen-Blanchette, Xiaowei Zhou, and Kostas Daniilidis. Polar transformer networks. In *International Conference on Learning Representations*, 2018.

Maurice Fréchet. Les éléments aléatoires de nature quelconque dans un espace distancié. In *Annales de l’institut Henri Poincaré*, volume 10, pages 215–310, 1948.

William T. Freeman and Edward H Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (9):891–906, 1991.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

S Helgason. Differential geometry and symmetric spaces, acad. *Press, New York*, 21, 1962.

Akira Hirose and Shotaro Yoshida. Generalization characteristics of complex-valued feedforward neural networks in relation to signal coherence. *IEEE Transactions on Neural Networks and learning systems*, 23(4):541–551, 2012.

Eric R Keydel, Shung Wu Lee, and John T Moore. Mstar extended operating conditions: A tutorial. In *Algorithms for Synthetic Aperture Radar Imagery III*, volume 2757, pages 228–242. International Society for Optics and Photonics, 1996.

Witold Kinsner and Warren Grieder. Amplification of signal features using variance fractal dimension trajectory. *Int. J. Cogn. Inform. Nat. Intell.*, pages 1–17, 2010.

Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. *arXiv preprint arXiv:1802.03690*, 2018.
Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012.

Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. nature, 521(7553):436, 2015.

Michael Maire, Takuya Narihira, and Stella X Yu. Affinity cnn: Learning pixel-centric pairwise relations for figure/ground embedding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 174–182, 2016.

Stéphane Mallat. Understanding deep convolutional networks. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 374(2065):20150203, 2016.

T Nitta. On the critical points of the complex-valued neural network. In Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP’02., volume 3, pages 1099–1103. IEEE, 2002.

Ivan V Oseledets. Tensor-train decomposition. SIAM Journal on Scientific Computing, 33(5):2295–2317, 2011.

David P Reichert and Thomas Serre. Neuronal synchrony in complex-valued deep networks. arXiv preprint arXiv:1312.6115, 2013.

Juergen Reichert. Automatic classification of communication signals using higher order statistics. In [Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing, volume 5, pages 221–224. IEEE, 1992.

Jiaqi Shao, Changwen Qu, Jianwei Li, and Shujuan Peng. A lightweight convolutional neural network based on visual attention for sar image target classification. Sensors, 18(9):3039, 2018.

Chiheb Trabelsi, Oleksa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep complex networks. arXiv preprint arXiv:1705.09792, 2017.

Jiayun Wang, Patrick Virtue, and Stella X Yu. Joint embedding and classification for sar target recognition. arXiv preprint arXiv:1712.01511, 2017.

Daniel E Worrall, Stephan J Garbin, Daniyar Turmukhambetov, and Gabriel J Brostow. Harmonic networks: Deep translation and rotation equivariance. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5028–5037, 2017.

Stella Yu. Angular embedding: A robust quadratic criterion. IEEE transactions on pattern analysis and machine intelligence, 34(1):158–173, 2011.

Qibin Zhao, Guoxu Zhou, Shengli Xie, Liqing Zhang, and Andrzej Cichocki. Tensor ring decomposition. arXiv preprint arXiv:1606.05535, 2016.