A Hermitian Positive Definite neural network for micro-Doppler complex covariance processing

Daniel A Brooks, Olivier Schwander, Frédéric Barbaresco, Jean-Yves Schneider, Matthieu Cord

A Hermitian Positive Definite neural network for micro-Doppler complex covariance processing

Daniel Brooks
Advanced Radar Concepts
Thales Land and Air Systems
Limours, FRANCE
LIP6, Laboratoire d'Informatique de Paris 6, F-75005
Sorbonne Université, CNRS
Paris, FRANCE
daniel.brooks@lip6.fr

Olivier Schwander
LIP6, Laboratoire d'Informatique de Paris 6, F-75005
Sorbonne Université, CNRS
Paris, FRANCE

Frédéric Barbaresco
Advanced Radar Concepts
Thales Land and Air Systems
Limours, FRANCE

Jean-Yves Schneider
Advanced Radar Concepts
Thales Land and Air Systems
Limours, FRANCE

Matthieu Cord
LIP6, Laboratoire d'Informatique de Paris 6, F-75005
Sorbonne Université, CNRS
Paris, FRANCE

***Abstract*—In its raw form, micro-Doppler radar data takes the form of a complex time-series, which can be seen as multiple realizations of a Gaussian process. As such, a complex covariance matrix constitutes a viable and synthetic representation of such data. In this paper, we introduce a neural network on Hermitian Positive Definite (HPD) matrices, that is complex-valued Symmetric Positive Definite (SPD) matrices, or complex covariance matrices. We validate this new architecture on synthetic data, comparing against previous similar methods.**

***Index Terms*—covariance matrices, complex numbers, micro-Doppler, drone classification, neural networks**

I. Introduction

The usage of deep learning methods has steadily been emerging in the radar community, specifically for micro-Doppler classification [9]; furthermore, the diversity of possible representations of micro-Doppler signals induce a variety of methods, such as recurrent neural networks (RNNs) in [14], or convolutional neural networks (CNNs) in [17] and [8]. In parallel, exploiting second-order feature moments, or covariance, is gaining momentum in certain subfields of machine learning, such as in EEG/ECG [4], facial recognition [1] or texture classification [11]. The natural representation of micro-Doppler signals as covariance (or self-correlation) led to several classification methods based on real-valued covariances matrices: in [7], authors use Riemannian barycenters on covariance reflexion coefficients; in [3], a minimum-distance-to-median scheme on SPD matrices was developed. Of all previously cited methods operating on covariance, the most promising in terms of potential development may be the ones based on neural-like processing: specifically, a first version of networks on SPD matrices, SPDNet, was first introduced in 2017 in [12], upon which further works expanded on. We also follow this blooming trend, by extending the theoretical framework of the SPDNet to complex values, thus introducing the HPDNet, or neural network operating on HPD matrices.

In the following section, we review the two theoretical learning frameworks we wish to fuse together. Then we describe our proposed HPDNet architecture and detail its specific mechanics. Finally, we validate the usage of a HPDNet over a SPDNet by studying micro-Doppler drone classification on synthetic radar data.

II. Manifold values in neural networks

Both machine learning frameworks we wish to fuse involve manifold values; the first type involves SPD matrices, the second complex values.

A. SPD matrices in neural networks

The fundamental interest of the SPD neural network is to take into account the information geometric structure of the curved Riemannian manifold of SPD matrices, noted $\mathcal{S}_*^+$. Statistical learning on curved manifolds is part of the field of information geometry [2]. Here, the learning model involved is a neural network, the layers of which we describe here. As in a standard Euclidean network, the SPDNet builds a hierarchy of activations on linear transformations, ended by a loss function to minimize by gradient descent [12]. This linear transformation, a fully-connected layer in perceptrons, a convolution in convolutional networks, becomes a bilinear mappping in the SPDNet, referred to as the BiMap layer, which we illustrate in Fig. 1. The activation function, usually set to the rectified linear unit (ReLU) in Euclidean networks, becomes the ReEig (rectified eigenvalues) in the SPDNet:

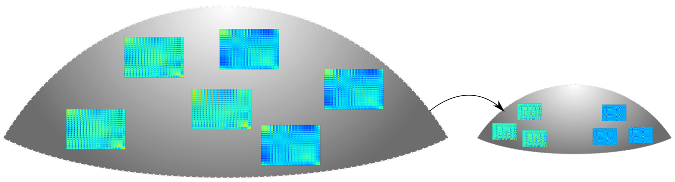

Fig. 1. Illustration of the BiMap layer. These are successively set in order to reduce the feature dimension, and more importantly separate the data. Each BiMap is followed by the ReEig activation.

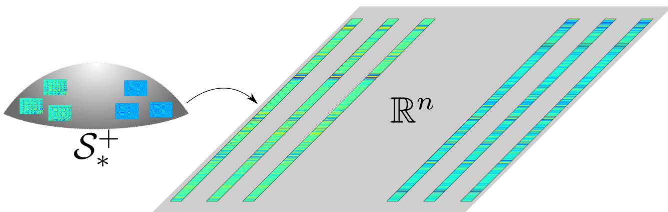

Fig. 2. Illustration of the LogEig. The final feature SPD manifold is logarithmically mapped to a Euclidean space to allow for classification.

$$\begin{aligned} P^{(l)} &= {W^{(l)}}^T X^{(l-1)} W^{(l)} \text{ with } W^{(l)} \in \mathcal{O}(n_{l-1}, n_l) \\ X^{(l)} &= U^{(l)} \max(\Sigma^{(l)}, \epsilon I_n) {U^{(l)}}^T \text{ with } P^{(l)} = U^{(l)} \Sigma^{(l)} {U^{(l)}}^T \end{aligned} \tag{1}$$

Thus, the building block of the SPDNet is $X^{(l-1)} \mapsto P^{(l)} = BiMap^{(l)}(X^{(l-1)}) \mapsto X^{(l)} = ReEig(P^{(l)})$. In the BiMap layer, $\mathcal{O}(n_{l-1}, n_l)$ is the space of semi-orthogonal rectangular matrices, also called Stiefel manifold, meaning the bilinear mapping is none other than an orthogonal basis change. In order for the parameter W to be non-singular, we must have $n_{l-1} \leq n_l$. In the ReEig equation, ϵ is a fixed parameter, which thresholds close to zero eigenvlues. Note the activation acts directly on the eigenvalues, thus necessitating a prior eigen-decomposition; such a function is sometimes referred to as a structured matrix function [13]. A final layer, of central importance to the SPDNet, is the logarithmic mapping, or LogEig, which maps the final SPD representation to a Euclidean space to perform the actual classification, which we illustrate in Fig. 2; the LogEig is simply the matrix logarithm. The training of an SPDNet entails two main technical difficulties: the optimization constrained to the Stiefel manifold for the update of the BiMap orthogonal parameters on the one hand, and on the other hand the propagation of gradients through the non-linear structured matrix functions found in the ReEig and LogEig layers. Interested readers may refer to [13] and [10] for theoretical background on these matters.

B. Complex numbers in neural networks

Here we explain how to integrate complex values in standard learning frameworks. Real-valued neural networks handle Euclidean data, which can be represented as vectors. A practical approach to handling Hermitian values, i.e. complex values, would simply to concatenate the real and imaginary parts in two independent channels. However, this method fails to take into account the structure of complex numbers. However, given some constraints, it is possible to formally state the equivalence between $\mathbb{C}$ and $\mathbb{R}^2$, which is directly translatable to $\mathbb{C}^n$ and $\mathbb{R}^{2n}$. This equivalence was first discovered by Wirtinger in 1927 [18], and adapted in [6] and [5] for electrical engineering purposes; it is sometimes referred to as Wirtinger calculus, or $\mathbb{CR}$ calculus, due to the intimacy shared between real and 2D real vectors exposed below. Traditional complex calculus is presented in the context of holomorphic functions; the aforementioned developments aimed to broaden this limited set of functions, nowadays allowing for instance differentiation of complex neural blocks, which are for the most part non-holomorphic, an operation fundamental to statistical learning. The key idea of these developments is to equate the Taylor expansion of a function $f : z \in \mathbb{C} \mapsto y \in \mathbb{R}$ with a two-dimensional, real-valued counterpart, which by abuse of notation is also noted $f : (u,v)^T \in \mathbb{R}^2 \mapsto y \in \mathbb{R}$:

$$f(z) \approx f(z_0) + \nabla_z f_{z_0}(z - z0) \tag{2}$$

$$f(u,v) \approx f(u_0,v_0) + \begin{bmatrix} \nabla_u f & \nabla_v f \end{bmatrix}_{(u_0,v_0)} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \tag{3}$$

We now introduce the 2×2 real-to-complex matrix T:

$$T := \begin{bmatrix} 1 & j \\ 1 & -j \end{bmatrix} \text{ such that: } \begin{cases} T^H T = 2I = T T^H \\ T \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} u + jv \\ u - jv \end{bmatrix} = \underbrace{\begin{bmatrix} x \\ x^* \end{bmatrix}}_{\underline{\mathrm{x}}} \end{cases} \tag{4}$$

As such, we can now write:

$$\begin{aligned} \begin{bmatrix} \nabla_u f \\ \nabla_v f \end{bmatrix}^T_{(u_0,v_0)} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} &= \left(\frac{1}{2} \begin{bmatrix} \nabla_u f \\ \nabla_v f \end{bmatrix}^T_{(u_0,v_0)} T^H \right) \left(T \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \right) \\ &:= \left(\nabla_{\underline{\mathrm{x}}} f_{\underline{\mathrm{x}}_0} \right) \left(\underline{\mathrm{x}} - \underline{\mathrm{x}}_0 \right) \end{aligned} \tag{5}$$

We have isolated a term, $\nabla_{\underline{\mathrm{x}}} f$, behaving like a gradient, artificially introduced in the $\mathbb{CR}$ calculations:

$$\nabla_{\underline{\mathrm{x}}} f_{\underline{\mathrm{x}}_0} = \begin{bmatrix} \nabla_x f_{x_0} \\ \nabla_{x^*} f_{x_0} \end{bmatrix}^T = \begin{bmatrix} \frac{1}{2}(\nabla_u f - j\nabla_v f)_{(u_0,v_0)} \\ \frac{1}{2}(\nabla_u f + j\nabla_v f)_{(u_0,v_0)} \end{bmatrix}^T \tag{6}$$

It is now possible to perform optimization on a non-holomorphic complex-valued function f by zeroing $\nabla_{\underline{\mathrm{x}}} f$. It is now crucial to note that doing so is equivalent to zeroing either $\nabla_x f$ or $\nabla_{x^*} f$, which is exactly how [18] showed that such a function f can be seen as operating on two independent variables x and x^*. We see below how the choice of $\nabla_x f$ over $\nabla_{x^*} f$ in a neural architecture will lead to a very fluid computational scheme.

III. HPDNet: a HPD-valued neural network

In this section, we present how we adapt the SPDNet layers to complex values, making use of the formal two-channel representation previously exposed. We note $\mathcal{H}_*^+$ the manifold of HPD matrices.

A. Formal HPD representation in a real-valued computational framework

Because calculus of non-holomorphic functions is to this day not widespread in computational frameworks (for instance as of end 2018, complex tensors are not integrated in the deep learning framework PyTorch [15]), it remains interesting to provide a custom integration, especially given the simplicity of the resulting implementation. From the results derived above, and taking inspiration from a seminal paper on deep complex networks [16], we can thus represent an HPD matrix $H = H_\Re + jH_\Im$ as a two-channel SPD matrix $(H_\Re, H_\Im)$. In any neural architecture, the gradient of the loss function l involving $(H_\Re, H_\Im)$ is computed as $\nabla_{H_\Re} l + j\nabla_{H_\Im} l$, which corresponds exactly to $\nabla_H l$. For this reason, it is natural to choose $\nabla_H l$ over $\nabla_{H^*} l$, as foreseen above. The backpropagation through the network can therefore be done with no additional pain, using any out-of-the-box backpropagation algorithm in a real-valued computational framework. However, the inference still needs to respect the internal structure of complex numbers. Below we show how to adapt the BiMap layer to adopt complex numbers.

B. Complex bilinear mapping

Here we show how to generalise the BiMap layer described previously to complex values, using the formalism exposed above. First we write the complex expresison of the bilinear mapping, then structure it to fit the $\mathbb{CR}$ framework:

$$
\begin{aligned}
P &= WXW^H \\
&= (W_\Re + jW_\Im)(X_\Re + jX_\Im)(W_\Re^H + jW_\Im^H) \\
&= \Bigg(W_\Re X_\Re W_\Re^H - W_\Im X_\Im W_\Re^H \\
&\qquad - W_\Re X_\Im W_\Im^H - W_\Im X_\Re W_\Im^H \Bigg) \\
&\quad + j\Bigg(W_\Re X_\Re W_\Im^H - W_\Im X_\Im W_\Im^H \\
&\qquad + W_\Re X_\Im W_\Re^H + W_\Im X_\Re W_\Re^H \Bigg) \\
&:= \begin{bmatrix} W_\Re X_\Re W_\Re^H - W_\Im X_\Im W_\Re^H - W_\Re X_\Im W_\Im^H - W_\Im X_\Re W_\Im^H \\ W_\Re X_\Re W_\Im^H - W_\Im X_\Im W_\Im^H + W_\Re X_\Im W_\Re^H + W_\Im X_\Re W_\Re^H \end{bmatrix}
\end{aligned} \tag{7}
$$

Note that the parameter matrix is now unitary (that is, complex orthogonal), and the transpose becomes a transconjugate.

C. Complex structured non-linearities

We study here the case of the non-linear structured complex functions involved in the HPDNet, the complex ReEig and complex LogEig. As stated previously, both take the form a non-linear function f acting on the an HPD matrix P's eigenvalues. We assume an eigen-decomposition of $P = U\Sigma U^H$. Note that although U is unitary, Σ is real because P is Hermitian. This is a very strong result, which greatly simplifies the generalisation of aforementioned functions to the complex setting: in fact, there is nothing in f to actually generalise, since the input eigenvalues are already real. However, one must take care to correctly handle the eigen-decomposition itself, separating the real and imaginary parts in order to respect the $\mathbb{CR}$ formalism:

$$
\begin{aligned}
X &= f(P) \\
&= Uf(\Sigma)U^H \\
&= (U_\Re + jU_\Im)f(\Sigma)(U_\Re - jU_\Im)^T \\
&= \Bigg(U_\Re f(\Sigma)U_\Re^T - U_\Im f(\Sigma)U_\Im^T \Bigg) \\
&\quad + j\Bigg(U_\Re f(\Sigma)U_\Im^T + U_\Im f(\Sigma)U_\Re^T \Bigg) \\
&:= \left[U_\Re f(\Sigma)U_\Re^T - U_\Im f(\Sigma)U_\Im^T U_\Re f(\Sigma)U_\Im^T + U_\Im f(\Sigma)U_\Re^T \right]
\end{aligned} \tag{8}
$$

Using the equations above, we are now able to perform inference through the complex BiMap, ReEig and LogEig layers. Posterior to the LogEig, a fully-connected layer (or several) handles the hyperplanar separation for classification. At any point in the network, it is possible to go back to $\mathbb{R}$ using the $\mathbb{C}2\mathbb{R}$ transfer function below:

$$
\forall H \in \mathcal{H}_*^+, \ \frac{1}{2}(H_\Re + H_\Im) = S \in \mathcal{S}_*^+ \tag{9}
$$

This is necessary because, again, we cannot yet perform the classification in the complex manifold. Furthermore, it is not obvious that the best performing model would maximize the use of complex numbers; it remains of interest to study the influence of the $\mathbb{C}2\mathbb{R}$'s positioning in the networks hierarchy to optimize its performance andd robustness.

IV. Experiments

In this section we test the HPDNet architecture against an equivalent SPDNet counterpart on both synthetic and two real micro-Doppler datasets. We also evaluate the robustness to lack of available training data.

A. Description of the synthetic data

The synthetic micro-Doppler data generated from a simulator introduced in [8]. The simulator models three drone classes: a helicopter, a quadrocopter and an octocopter as shown in Fig. 3.

Approximately 4 minutes of signal are synthetisized per class. We set the observation conditions such that the signal-to-noise ration (SNR) varies around $5dB$ before coherent integration. We consider an elementary timeframe to correspond roughly to one or two blade rotations; the fastest propeller having an RPM of 4800 rounds per minute, we set the elementary period of observation to $10ms$. We also set a

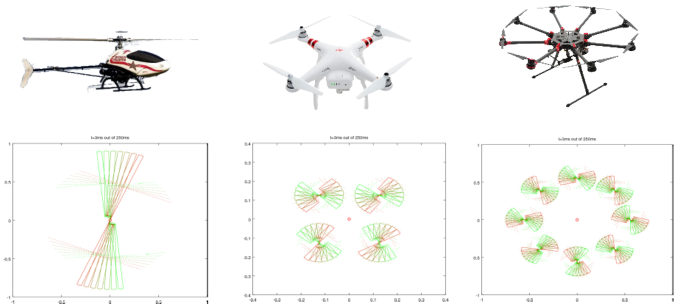

Fig. 3. Photographs and corresponding models of the three drones to classify. Scatter points along their surface define the reflected radar wave which in turn constitutes the received micro-Doppler signal.

reasonably low period of repetition frequency (PRF) of $2kHz$, which results in an elementary signal of 20 discrete timesteps, which translates to setting the input feature dimension to 20. In other words, one radar signal is modeled as a point z in $\mathbb{C}^n$, with $n = 20$. However, the final objective here is the classification on the underlying Gaussian process; in order to estimate its corresponding covariance, we theoretically need at least 20 independent samples in order for the resulting matrix to be non-singular. In practice, we sample $N = 20$ successive samples; that is, the strict minimum. We use the unbiased covariance estimator:

$$X = \frac{1}{N-1} \sum_{i \leq N} \bar{z}_i \bar{z}_i^H \in \mathcal{H}_*^+(n) \tag{10}$$

In the equation above, $\bar{z}_i$ is the centered version of z_i. We implement a 2-layer network reducing the matrix dimension from 20 to 16 and 8.

B. Description of the NATO database

To challenge our model to a real-world setting, we make use of a micro-Doppler drone database provided by the NATO organization.

The NATO database consists of recordings of 8 different aiborne subjects, including one class encompassing birds and 7 different drones:

- DJI Phantom 3 (carbon fiber & nylon blades) [quadcopter];
- 3DR X8 (carbon fiber blades) [quadcopter];
- 3DR Iris (carbon fiber & nylon blades) [quadcopter];
- Firefly [hybrid]
- Anaconda [fixed wing]
- Opterra [fixed wing]
- Skywalker [fixed wing]

The Phantom and the Iris come in two versions: carbon fiber or nylon blades, which can consitute child classes. The drones are categorized in three types: quadcopter, fixed wing and hybrid, which can consitute parent classes. The radar signals are accessible in their raw form of time series of complex points of amplitude and phase. The subjects were furthermore recorded in 6 frequency bands (L,S,C,X,Ku and Ka) and with both vertical and horizontal polarization (except for the L band). In our classification setup, we choose to consider all class variation as independent objects; for instance, the Phantom drones with carbon or nylon blades are set as different classes altogether, which we hope to descriminate. All in all, this leads to 10 separate classes.

Each datafile is the continuous recording of one of the above targets; those recordings vary in lenth from $7s$ to $672s$. The PRF is set to $25kHz$, meaning a signal of $1min$ totals $1.5e6$ complex numbers. All recordings are split in elementary segments of $M = 1024$ points (i.e. $40ms$) destined for further Fourier processing. Thus, a $1min$ long signal would yield a file of $1024 * 1465$ complex numbers organized in a complex matrix of shape $(1024, 1465)$.

Having a large amount of data sampled at a high PRF allows for customization to worstly-conditioned scenarios, by using only part of the data and downsampling the signals. We set ourselves in a particular scenario, making use of only 3% of the available data, downsampled 3 times (for a PRF of about $8kHz$).

We transform the inputs, as we do the synthetic data, to 20×20 HPD matrices, and also use the same architecture and learning scheme.

C. Description of the Aveillant database

To further asses the algorithm's sturdiness, we repeat experiments on a dataset provided by the Aveillant company.

The data's main focus is the discrimination of the I1-D drone from other airborne targets. Although three classes are proposed ("bird", "car", "drone"), for which there are respectively 10, 2 and 12 files, we chose to collapse the task to a 2-class problem ("other", "drone"). Each file in the dataset is a series of nearly continuous frames of length $279ms$ sampled to $M = 2048$ points, i.e. at a PRF of about $8kHz$. The number of frames in the files varies from 72 to 823 with a median value of 207, i.e. the durations of the recordings vary from $20s$ to $4min$ with a median duration of $1min$. As for the NATO data, classification is performed frame by frame, i.e. one data point consists of 2048 consecutive complex points. A data point is split through a sliding window of length $n = 128$ with a hop length of 32 (so a 75% overlap); a single covariance matrix of size 128×128 is thus sampled from all the resulting sub-frames and passed on to an SPDNet or HPDNet. We implement a 2-layer network reducing the matrix dimension from 20 to 16 and 8. The following paragraph describes the results on both datasets.

D. Comparison of SPDNet and HPDNet

Upon these 20×20 Hermitian matrices, we build a 2-layer HPDNet with hidden dimensions 16 and 8. After the LogEig Euclidean mapping, the resulting matrix is passed to the real domain via the $\mathbb{C}2\mathbb{R}$ layer and vectorized to a $8 * 8 = 64$-dimensional vector, which is finally mapped to the 3-dimensional space of class distribution. All experiments are run on a 5-fold cross-validation using 25% of all data for validation, and tested on another fixed 25% held-out set.

The remaining 50% is used for training. We compare the HPDNet described above with the equivalent SPDNet. In this experimental configuration, the performance of both models, measured as overall accuracy over the synthetic test set, are as follows:

- SPDNet: $90.2\% \pm 3.28$
- HPDNet: $93.5\% \pm 1.13$

As for the NATO dataset, we get:

- SPDNet: $82.8\% \pm 0.72$
- HPDNet: $86.5\% \pm 0.53$

As for the Aveillant dataset, we get:

- SPDNet: $86.7\% \pm 1.61$
- HPDNet: $89.2\% \pm 1.42$

We see that for all datasets the HPDNet performs slightly better, while also presenting a smaller standard deviation, which indicates higher stability from using complex values. For visual purpose, we plot the training and validation accuracy curves along with the confusion matrix upon convergence for one cross-validation instance of the training on both NATO and Aveillant data using the HPDNet in figures 4 and 5.

E. Model robustness to lack of data

In the context of machine learning on radar data, one is often faced with an unevitable lack of data, due to the underlying cost and sensitivity. To simulate such a scenario, we repeat the experiment using only 10% of the available training data. The performance obtained on the synthetic data are then:

- SPDNet: $83.2\% \pm 4.18$
- HPDNet: $85.1\% \pm 1.49$

As for the NATO data:

- SPDNet: $79.9\% \pm 3.18$
- HPDNet: $81.4\% \pm 1.95$

As for the Aveillant dataset, we get:

- SPDNet: $83.2\% \pm 1.97$
- HPDNet: $85.1\% \pm 1.56$

Although the gap is reduced, the HPDNet still outperforms the SPDNet in both cases; this results was not obvious, as, because it involves complex numbers, the HPDNet intrinsically exhibits twice as many parameters than its real counterpart. Allowing too many paramaters to a neural network can lead to overfitting on the training set, resulting in worsened performance at test time; however, we do not observe this phenomenon, justifying the interest of correctly handling the geometric structure of complex numbers. Again, the HPDNet exhibits a smaller standard deviation, which makes it a suitable generalisation of the SPDNet.

CONCLUSION

Radar signals exhibit a strong structure, which makes way for a variety of meaningful representations; the covariance of the underlying Gaussian process, modeled as a Symmetric Positive Definite (SPD) matrix, is one of them. However, this representation's major drawback is the discarding of phase information; in this work, we introduce a neural network on Hermitian Positive Definite (HPD) matrices, which model the complex covariance of the signal, thus integrating any phase information in the raw signal. The theoretical background of this model is grounded in both complex-valued, and SPD-valued neural networks. We experimentally show, on synthetic micro-Doppler data and larger-scale real-world NATO and Aveillant datasets, that the HPDNet not only increases the performance compared to a real-valued SPDNet counterpart, but also exhibits higher stability and robustness to lack of data, all the while possessing a small amount of parameters compared to standard neural networks. For all these reasons, we believe it to be a suitable and scalable model for radar micro-Doppler classification.

ACKNOWLEDGMENT

We would like to thank the NATO working group SET245 for providing the drone micro-Doppler database and allowing for publication of classification results.

REFERENCES

[1] D. Acharya, Z. Huang, D. Paudel, and L. Van Gool. Covariance Pooling For Facial Expression Recognition. *arXiv:1805.04855 [cs]*, May 2018. arXiv: 1805.04855.

[2] S.-i. Amari. *Information Geometry and Its Applications*. Applied Mathematical Sciences. Springer Japan, 2016.

[3] M. Arnaudon, F. Barbaresco, and L. Yang. Medians and Means in Riemannian Geometry: Existence, Uniqueness and Computation. Nov. 2011.

[4] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten. Multiclass BrainComputer Interface Classification by Riemannian Geometry. *IEEE Transactions on Biomedical Engineering*, 59(4):920–928, Apr. 2012.

[5] A. v. d. Bos. Complex gradient and Hessian. *IEE Proceedings - Vision, Image and Signal Processing*, 141(6):380–383, Dec. 1994.

[6] D. H. Brandwood. A complex gradient operator and its application in adaptive array theory. *IEE Proceedings H - Microwaves, Optics and Antennas*, 130(1):11–16, Feb. 1983.

[7] A. L. Brigant, F. Barbaresco, and M. Arnaudon. Geometric barycenters of time/Doppler spectra for the recognition of non-stationary targets. In *2016 17th International Radar Symposium (IRS)*, pages 1–6, May 2016.

[8] D. A. Brooks, O. Schwander, F. Barbaresco, J. Schneider, and M. Cord. Temporal Deep Learning for Drone Micro-Doppler Classification. In *2018 19th International Radar Symposium (IRS)*, pages 1–10, June 2018.

[9] V. C. Chen, F. Li, S.-S. Ho, and H. Wechsler. Micro-Doppler effect in radar: phenomenon, model, and simulation study. *IEEE Transactions on Aerospace and electronic systems*, 42(1):2–21, 2006.

[10] A. Edelman, T. Arias, and S. Smith. The Geometry of Algorithms with Orthogonality Constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, Jan. 1998.

[11] M. T. Harandi, M. Salzmann, and R. Hartley. From Manifold to Manifold: Geometry-Aware Dimensionality Reduction for SPD Matrices. *arXiv:1407.1120 [cs]*, July 2014. arXiv: 1407.1120.

[12] Z. Huang and L. J. Van Gool. A Riemannian Network for SPD Matrix Learning. In *AAAI*, volume 1, page 3, 2017.

[13] C. Ionescu, O. Vantzos, and C. Sminchisescu. Matrix Backpropagation for Deep Networks with Structured Layers. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2965–2973, Santiago, Chile, Dec. 2015. IEEE.

[14] P. Molchanov, R. I. Harmanny, J. J. de Wit, K. Egiazarian, and J. Astola. Classification of small UAVs and birds by micro-Doppler signatures. *International Journal of Microwave and Wireless Technologies*, 6(3-4):435–444, June 2014.

[15] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. Oct. 2017.

[16] C. Trabelsi, O. Bilaniuk, Y. Zhang, D. Serdyuk, S. Subramanian, J. F. Santos, S. Mehri, N. Rostamzadeh, Y. Bengio, and C. J. Pal. Deep Complex Networks. *arXiv:1705.09792 [cs]*, May 2017. arXiv: 1705.09792.

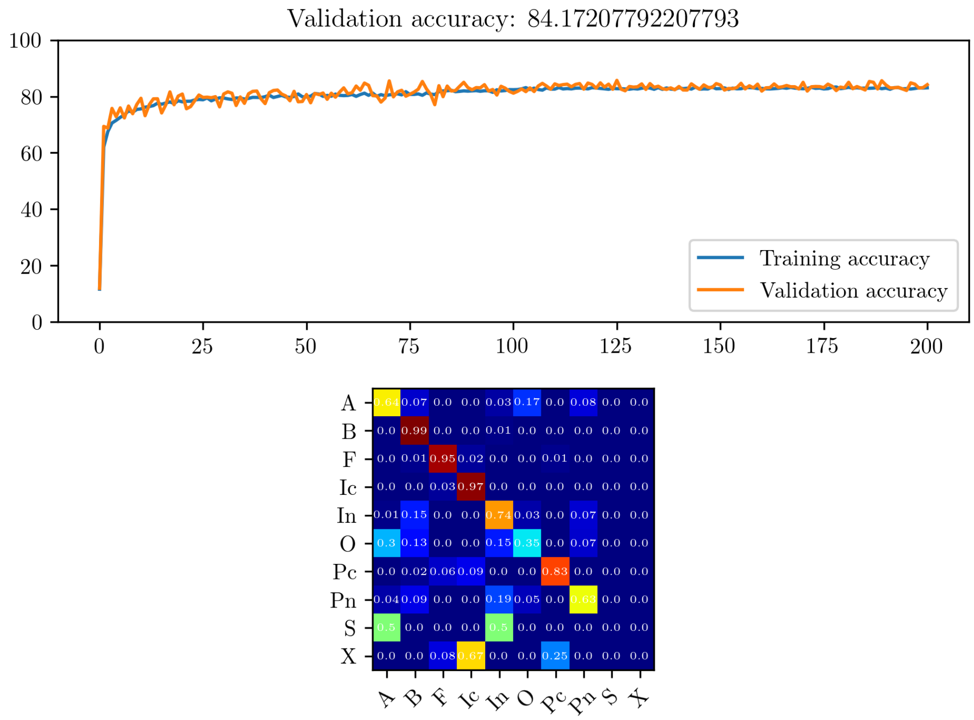

Fig. 4. Training and validation accuracy during the learning of the HPDNet on NATO data, along with the final normalized confusion matrix; true class is written on the left, predicted class underneath. Full class names can be deduced from the letters displayed on the matrix; for instance, class 'B' designates the birds, 'Pc' the Phantom drone with carbon blades. We notice that the Skywalker and X8 drones are never predicted, perhaps due to their challengingly evasive presence within the database.

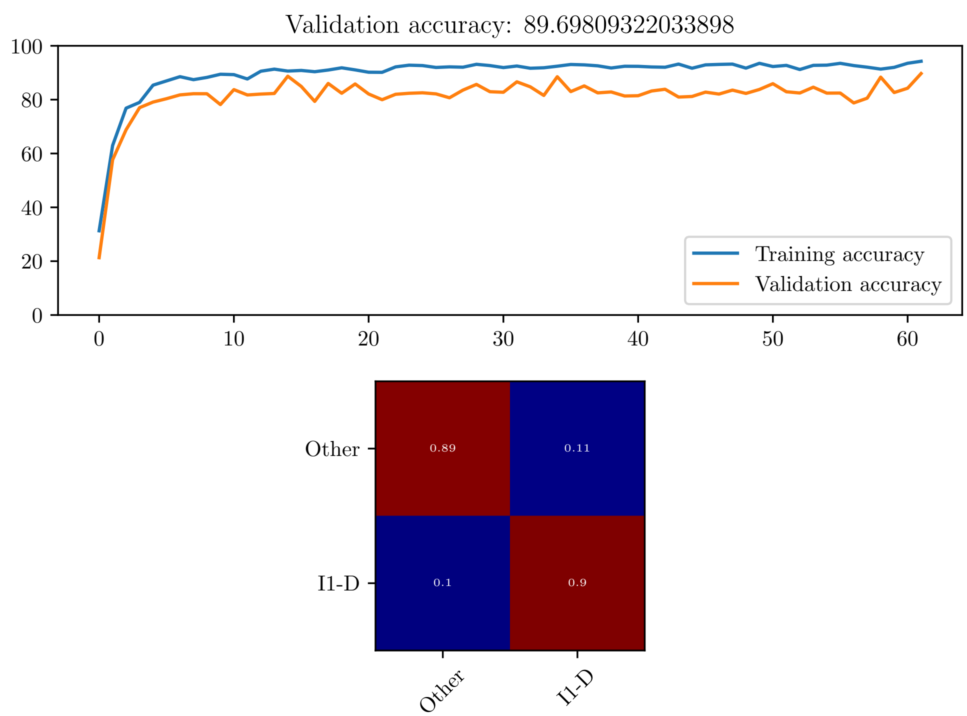

Fig. 5. Training and validation accuracy during the learning of the HPDNet on NATO data, along with the final normalized confusion matrix; true class is written on the left, predicted class underneath. Full class names can be deduced from the letters displayed on the matrix; for instance, class 'B' designates the birds, 'Pc' the Phantom drone with carbon blades. We notice that the Skywalker and X8 drones are never predicted, perhaps due to their challengingly evasive presence within the database.

[17] R. P. Trommel, R. I. A. Harmanny, L. Cifola, and J. N. Driessen. Multi-target human gait classification using deep convolutional neural networks on micro-doppler spectrograms. In *2016 European Radar Conference (EuRAD)*, pages 81–84, Oct. 2016.

[18] W. Wirtinger. Zur formalen Theorie der Funktionen von mehr komplexen Vernderlichen. *Mathematische Annalen*, 97(1):357–375, Dec. 1927.