Deep Learning and Information Geometry for Drone Micro-Doppler Radar Classification

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

* Thales Land and Air Systems,
Advanced Radar Concepts
Limours, France
† LIP6, Laboratoire d'Informatique de Paris 6, F-75005,
Sorbonne Université, CNRS
Paris, France

***Abstract*—In this work, we build dedicated learning models for micro-Doppler radar time series classification. We develop both deep temporal architectures based on time-frequency representations, and also directly study the signal's underlying statistical Gaussian process using Information Geometry on Riemannian manifolds by developing and improving symmetric positive definite (SPD) neural networks. We also propose the aggregation of all proposed models in a single, highly performing classification piepline.**

I. Introduction

Machine Learning, and in particular Deep Learning, is a powerful tool to model and study the intrinsic statistical foundations of data, allowing the extraction of meaningful, human-interpretable information from otherwise unpalatable arrays of floating points. While it provides a generic solution to many problems, some particular data types exhibit strong underlying physical structure: images have spatial locality, audio has temporal sequentiality, radar has time-frequency structure. Both intuitively and formally, there can be much to gain in leveraging this structure by adapting the subsequent learning models. As convolutional architectures for images, signal properties can be encoded and harnessed within the network. Conceptually, this would allow for a more intrinsic handling of the data, potentially leading to more efficient learning models. Thus, we will aim to use known structures in the signals as model priors. Specifically, we build dedicated deep temporal architectures for time series classification, and explore the use of complex values in neural networks to further refine the analysis of structured data.

Going even further, one may wish to directly study the signal's underlying statistical process. As such, Gaussian families constitute a popular candidate. Formally, the covariance of the data fully characterizes such a distribution; developing ML algorithms on covariance matrices will thus be a central theme throughout this work. Statistical distributions inherently diverge from the Euclidean framework; as such, it is necessary to study them on the appropriate, curved Riemannian manifold, as opposed to a flat, Euclidean space. Specifically, we contribute to existing deep architectures by adding normalizations in the form of data-aware mappings, and a Riemannian batch normalization algorithm. We showcase empirical validation through a variety of different tasks, with a sharpened focus on micro-Doppler radar data for non-cooperative drone recognition. Figure 1 illustrates the diversity of possible micro-Doppler input representations.

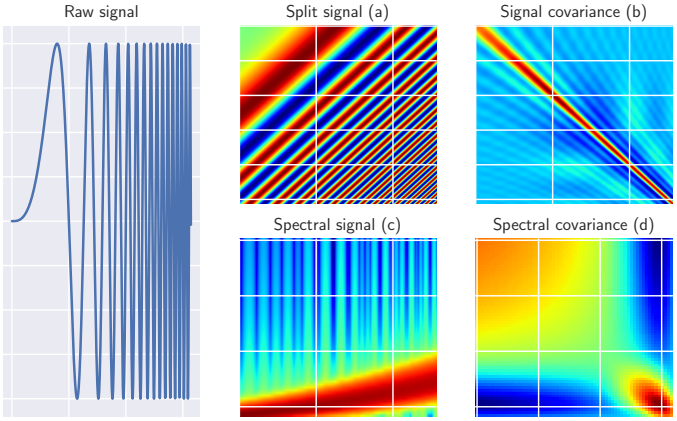

Fig. 1. Raw signal and its Fourier and covariance representations. The raw signal is split through a sliding window, from which either covariance or Fourier transform can be computed.

To summarize our proposed contributions:

- ➢ A Fully Temporal Convolutional Network (FTCN) designed to respect the temporal structure of the data;
- ➢ Two closely resembling complex declinations of the FTCN, notably:
  - → A $\mathbb{CR}$Net, with a design inspired from signal analysis principles;
  - → A Fourier convolution layer, allowing the training from raw complex temporal data, yielding a FourierNet architecture;
- ➢ A full pipeline for structured time series classification, combining multiple representations and models, called Second-Order Fully Temporal Network (SOFTNet);
- ➢ A neural architecture called DAMNet with two new layers (which are mutually exclusive):

- → A barycentric normalization layer using a Riemannian barycenter (BarNorm);
- → A parametric normalization layer using a SPD-constrained parameter (ParNorm);

➢ A neural architecture called SPDNetBN with one new layer:

- → A Riemannian batch normalization layer for SPD neural networks, respecting the manifold's geometry;

➢ A generalized gradient descent allowing to learn the DAMNet and SPDNetBN models;

➢ A convolutional layer for SPD matrices;

➢ A micro-Doppler drone radar simulator to allow for extensive experimental studies;

➢ Validation, comparison and interpretation of the various models, on both simulated and real data.

II. Second-order pipeline for temporal classification

This first section makes use of the multiplicity and connectedness of structured time series representations such as micro-Doppler radar data, to build independent learning models upon different representations, and combine these models in a single classification pipeline to harness the full underlying geometric structure of the data.

A. Learning on structured time series representations

Succintly, a temporal signal such as micro-Doppler data can take a variety of forms, each one highlighting a different set of properties within the signal (Flandrin 1998; Hlawatsch and Auger 2013). Each representation is then best fit by an individual learning model (Stoica and Moses 2005; Moruzzis and Colin 1998).

The notion of covariance, tightly linked to that of the Fourier transform, showcases a powerful, geometrically accurate yet compact micro-Doppler representation. However, its inherently curved, Riemannian nature enforces any learning algorithm to respect the underlying geometry of covariance matrices, *i.e.* Symmetric Positive Definiteness (SPD). A neural architecture, first developed in 2017 by Huang and Van Gool (2017), allows to train on SPD matrices: the SPD neural network (SPDNet). Its application to micro-Doppler classification was initiated in Yang et al. (2010), Brigant et al. (2016), Brooks et al. (2019a), and Brooks et al. (2019b).

A rather more intuitive approach for micro-Doppler classification, developed in former literature (Brooks et al. 2018), involves deep neural models on time-frequency representations, specifically a Fully-Temporal Convolutional Network (FTCN). A complex-valued version, suited to the radar application, was then proposed in Brooks et al. (2019c). An extended version, introducing a Fourier convolutional layer, which allows for the learning of a 1-D filter bank initialized with the Fourier atoms, was further introduced in Brooks et al. (2019d).

B. Full pipeline for temporal classification

We now introduce the proposed pipeline, displayed in Figure 2, and show how branching through the its blocks leads to different models on the signal representations. Four global models, noted from (1) to (4) in the figure, can be extracted from the pipeline, which we detail throughout the section.

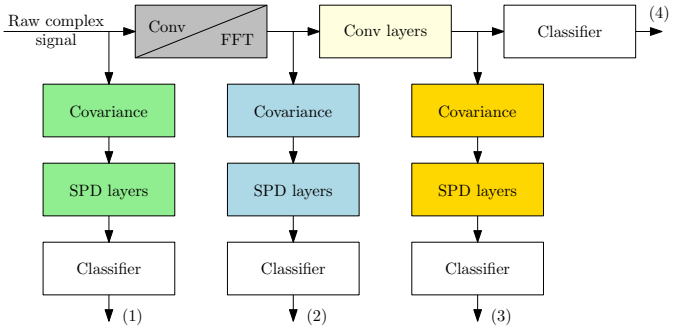

Fig. 2. **Illustration of the global pipeline proposed for the classification of time-frequency signals**.

The classification in the FTCN uses Global Average Pooling (GAP) to pool the feature maps' temporal evolution to a single dimension. However, as mentioned before it is possible to branch out in the pipeline at any stage, precisely thanks to the fully-temporal property of the network. It is precisely the property of conservation of temporal structure verified by the FTCN which allows to extract temporal feature maps, and thus, gives the possibility to study the covariance of these maps. Doing so hints towards a powerful representation learning model, making use both of a potentially complex-valued Euclidean FTCN operating on first-order moments of the data, and a Riemannian SPDNet operating on second-order moments of the data. We thus call this particular model the Second-Order Fully Temporal Network (SOFTNet), and illustrate it in Figure 3.

III. Advances in SPD neural networks

The use of covariance representation is particularly interesting in the case of structured temporal data since a global covariance matrix is a straightforward way to capture and represent the temporal fluctuations of data points of different

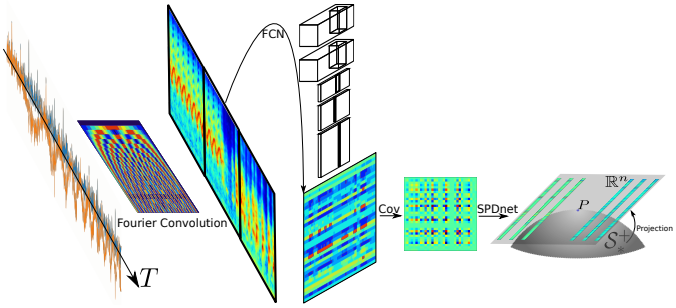

Fig. 3. **Illustration of the SOFTNet**.

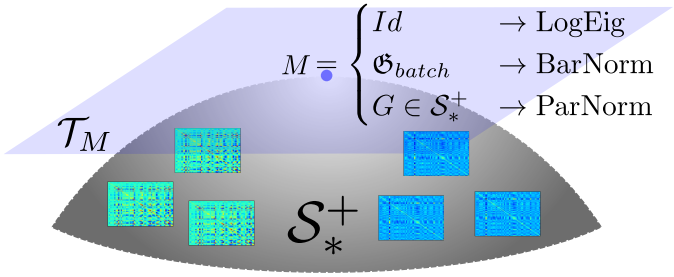

Fig. 4. **Illustration of the DAMNet architecture**. When the reference point of the Euclidean mapping is the identity matrix, it reduces to the SPDNet architecture.

lengths. Throughout this section, we propose building new blocks upon the reference architecture (Huang and Van Gool 2017), with a shared goal of normalizing its inner mechanics.

A. DAMNet

In our first proposition, we introduce a new architecture, called a Data-Aware Mapping Network (DAMNet), which focuses on the final layer of the SPDNet, the Euclidean mapping: we argue that a better projection can be done by making the layer dependent on the data. The original projection layer relies on the Log-Euclidean Metric (LEM) framework (Arsigny et al. 2006), which endows the manifold of SPD matrices with a Lie group structure. This framework is much simpler than the full Riemanian setting and allows efficient computations while keeping good theoretical properties (Pennec et al. 2006). While useful, this framework is only a particular case: we thus introduce an improved projection layer working in the broader Riemaniann setting. This new projection maps the points to the tangent space of some reference matrix and comes in two variants: a barycentric projection, called BarNorm, which uses the Riemannian barycenter as the reference matrix; and a parametric projection, called ParNorm, which uses a parameter SPD matrix, learnt during training.

In short, the difference between a SPDNet and a DAMNet is the Euclidean mapping, which in the case of the former is fixed to the matrix logarithm, contrary to the latter which depends on a data-aware reference point M, defined either as an additional parameter or a barycenter; as such; SPDNet is a special case of DAMNet, as illustrated in Equation 1 and in Figure 4:

$$X^{(k)} = log(M^{-\frac{1}{2}} P^{(k)} M^{-\frac{1}{2}}) \text{ , with}$$
$$M = \begin{cases} Id & \rightarrow \text{LogEig with no regularization} \\ \mathfrak{G}_{batch} & \rightarrow \text{BarNorm (barycenter)} \\ G \in \mathcal{S}_*^+ & \rightarrow \text{ParNorm (gradient descent)} \end{cases} \quad (1)$$

B. SPDNetBN

This section's second main contribution is inspired by the well-known and well-used BatchNorm layer, introduced in the context of (Euclidean) CNNs for Computer Vision tasks in Ioffe and Szegedy (2015). The overall architecture, which we call Batch-Normalized SPD Neural Network (SPDNetBN), is expected to perform better than either the SPDNet or DAMNet.

The Euclidean BatchNorm involves centering and biasing the batch $\mathcal{B}$, which is done via subtraction and addition. However on a curved manifold, there is no such group structure in general, so these seemingly basic operations are ill-defined. To shift SPD matrices around their mean $\mathfrak{G}$ or towards a bias parameter $G \in \mathcal{S}_*^+$, we propose to rather use parallel transport on the manifold (Amari 2016):

Centering from $\mathfrak{G} \coloneqq \text{Bary}(\mathcal{B})$:

$$\forall i \leq N,\ \bar{P}_i = \Gamma_{\mathfrak{G} \rightarrow I_d}(P_i) = \mathfrak{G}^{-\frac{1}{2}}\ P_i\ \mathfrak{G}^{-\frac{1}{2}} \quad (2a)$$

Biasing towards parameter G:

$$\forall i \leq N,\ \tilde{P}_i = \Gamma_{I_d \rightarrow G}(\bar{P}_i) = G^{\frac{1}{2}}\ \bar{P}_i\ G^{\frac{1}{2}} \quad (2b)$$

We can now write our full Riemannian BatchNorm in 1.

Algorithm 1 Riemannian batch normalization on $\mathcal{S}_*^+$, training and testing phase

TRAINING PHASE
Require: batch of N SPD matrices $\{P_i\}_{i \leq N}$, running mean $\mathfrak{G}_{\mathcal{S}}$, bias G, momentum η
$\mathfrak{G}_{\mathcal{B}} \leftarrow \text{Bary}(\{P_i\}_{i \leq N})$ // compute batch mean
$\mathfrak{G}_{\mathcal{S}} \leftarrow \text{Bary}^{\eta}(\mathfrak{G}_{\mathcal{S}}, \mathfrak{G}_{\mathcal{B}})$ // update running mean
for $i \leq N$ **do**
  $\bar{P}_i \leftarrow \Gamma_{\mathfrak{G}_{\mathcal{B}} \rightarrow I_d}(P_i)$ // center batch
  $\tilde{P}_i \leftarrow \Gamma_{I_d \rightarrow G}(\bar{P}_i)$ // bias batch
end for
return normalized batch $\{\tilde{P}_i\}_{i \leq N}$

INFERENCE PHASE
Require: batch of N SPD matrices $\{P_i\}_{i \leq N}$, final running mean $\mathfrak{G}_{\mathcal{S}}$, learnt bias G
for $i \leq N$ **do**
  $\bar{P}_i \leftarrow \Gamma_{\mathfrak{G}_{\mathcal{S}} \rightarrow I_d}(P_i)$ // center batch using set statistics
  $\tilde{P}_i \leftarrow \Gamma_{I_d \rightarrow G}(\bar{P}_i)$ // bias batch using learnt parameter
end for
return normalized batch $\{\tilde{P}_i\}_{i \leq N}$

C. Riemannian manifold-constrained optimization

The specificities of a the proposed DAMNet architecture and BatchNorm algorithm are the non-linear manipulation of manifold values in both inputs and parameters and the use of a Riemannian barycenter. We refer the reader to Edelman et al. (1998), Ionescu et al. (2015), Engin et al. (2018), and Brooks et al. (2019a) for the inner mechanics of the backpropagation of manifold-constrained structured matrix functions through a Riemannian neural network.

The bias parameter matrix G of the Riemannian BatchNorm is by construction constrained to the SPD manifold. However, the standard Stochastic Gradient Descent (SGD) has no reason to respect this constraint; instead, a Riemannian SGD is implemented, as detailed in Edelman et al. 1998 illustrated in Figure 5. In short, it consists in a two-step process of tangential projection $\Pi \mathcal{T}_G$ and exponential mapping Exp_G:

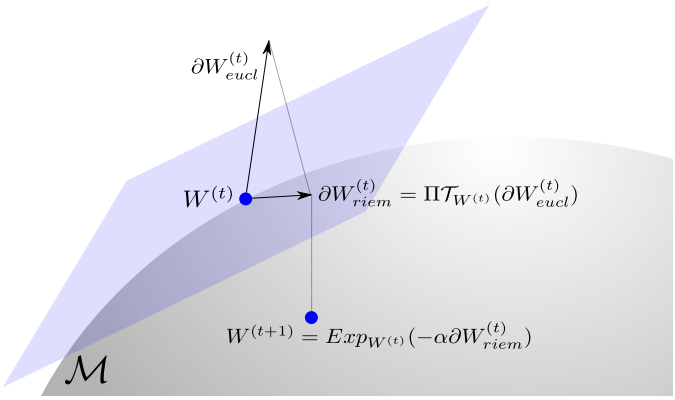

Fig. 5. **Illustration of manifold-constrained gradient update**. The Euclidean gradient is projected to the tangent space, then mapped to the manifold through retraction, which can be the exponential mapping.

$$\forall P, \qquad \Pi\mathcal{T}_G(P) = G\frac{P+P^T}{2}G \in \mathcal{T}_G \subset \mathcal{S}^+ \tag{3}$$

$$\forall P \in \mathcal{T}_G,\ \mathrm{Exp}_G(P) = G^{\frac{1}{2}} log(G^{-\frac{1}{2}}PG^{-\frac{1}{2}})G^{\frac{1}{2}} \in \mathcal{S}^+_* \tag{4}$$

However, this is still not enough for the optimization of the layer, as the BatchNorm involves not simply G and $\mathfrak{G}$, but $G^{\frac{1}{2}}$ and $\mathfrak{G}^{-\frac{1}{2}}$, which are structured matrix functions of G, *i.e.* which act non-linearly on the matrices' eigenvalues without affecting its associated eigenspace. The next paragraph deals with the backpropagation through such functions.

We summarize the backpropagation formulas as such: given the function $P \longmapsto X := g(P)$ and the succeeding gradient $\frac{\partial L^{(k+1)}}{\partial X}$, the output gradient $\frac{\partial L^{(k)}}{\partial P}$ is:

$$\frac{\partial L^{(k)}}{\partial P} = U\left(L \odot (U^T(\frac{\partial L^{(k+1)}}{\partial X})U)\right)U^T \tag{5}$$

$$L_{ij} = \begin{cases} \frac{g(\sigma_i)-g(\sigma_j)}{\sigma_i-\sigma_j} & \text{if } \sigma_i \neq \sigma_j \\ g'(\sigma_i) & \text{otherwise} \end{cases} \tag{6}$$

D. Convolution for covariance time series

Here we introduce a novel temporal convolution layer for time series of SPD matrices, inspired from the usual Euclidean 1D convolution layer, and using the BiMap layer defined above. This layer produced no satisfactory results, so does not appear in the experimental validations.

The convolution as defined above takes the form of an arithmetic mean; one could instead imagine using a Riemannian mean, better-suited to the SPD data:

$$\begin{aligned} &X := P *^w K \\ &\forall t \leq T_o,\ X_t = \mathrm{Bary}^w_{l\leq k}\ (K_l^T\ P_{t+k-1-l}\ K_l) \end{aligned} \tag{7}$$

In the equation above, one must take care not to confuse the weighted Fréchet mean of the operands with the unweighted Fréchet mean of the weighted operands.

Figure 6 illustrates the somewhat convoluted process of performing the SPD BiMap convolution.

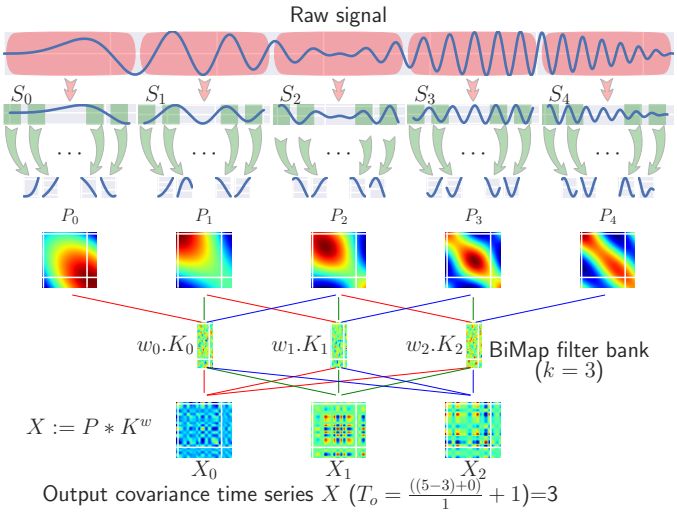

Fig. 6. **The convolutional BiMap**.

More realistically, we wish to instigate the network's potential expressiveness by further building multi-channeled SPD representations. We do so in the usual fashion; input P is now of shape (T_i, C_i, n_i, n_i), kernel K of shape (C_o, C_i, k, n_i, n_o), weights w are still of shape $(k,)$, and output X of shape (T_o, C_o, n_o, n_o), and is computed as follows:

$$\forall c_o \leq C_o,\ X_{c_o} = \sum_{c_i \leq C_i} P_{c_i} *^w K_{c_o,c_i} \tag{8}$$

In all definitions above, the convolution uses the BiMap layer (and therefore induces dimension reduction), which is novel.

IV. Experimental results

Here we evaluate the performances of the different proposed architectures on micro-Doppler radar data classification, along with baseline reference algorithms.

A. Drones recognition task

Experiments are conducted on a confidential dataset of real recordings issued from a NATO working group [1]. To spur reproducibility, we also experiment on synthetic, publicly available data.

The raw micro-Doppler signal is split in windows of length $n = 20$, the series of which a single covariance matrix of size 20×20 is sampled from, which represents one radar data point. We refer to Figure 7 for a visual clarification of the splitting operation. The NATO data features 10 classes of drones, whereas the synthetic data is generated by a realistic simulator of 3 different classes of drones following the protocol previously described.

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

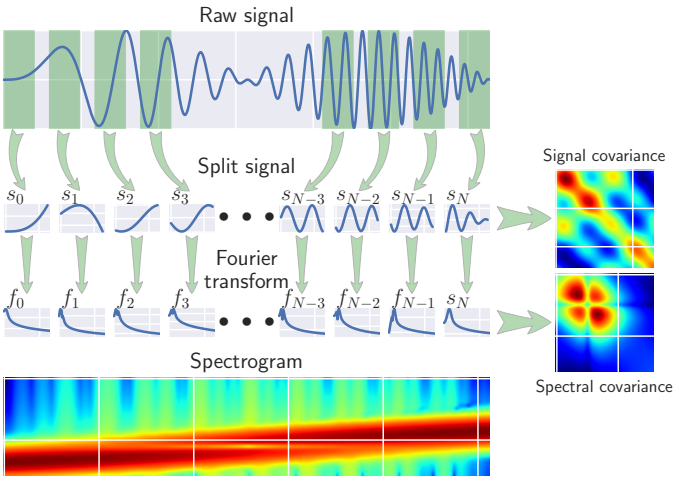

Fig. 7. **Structured time series representations**.

TABLE I
PERFORMANCE COMPARISON OF FIRST- AND SECOND-ORDER MODELS ON RADAR DATA

Train size	100%	20%	5%
SPDNet	92.6 ± 0.54	91.5 ± 0.74	88.4 ± 3.06
HPDNet	94.4 ± 0.76	91.8 ± 0.80	87.1 ± 1.11
FTCN	98.9 ± 0.44	93.4 ± 1.21	84.3 ± 2.51
FourierNet	99.4 ± 0.17	96.2 ± 1.12	87.4 ± 1.94
SpectroSPD	95.1 ± 0.49	91.9 ± 0.82	84.6 ± 3.49
SOFTNet	99.5 ± 0.16	97.2 ± 0.90	93.9 ± 0.74

B. Comparison with second-order models and validation of the full pipeline

The same windowing is used for covariance and spectral representations to keep comparisons fair, as per illustrated in Figure 7. We bifurcate the introduced pipeline at various stages in various configurations, which amounts to different learning models which we relate to in Figure 2. Furthermore, we repeat the experiments with decreasing amount of training data in the hope that injecting geometric information in the learning through second-order modelling would compensate for lack in data volume. Results are displayed in Table I.

The key takeaway from these results follows: Riemannian SPD-based models don't compete against Euclidean deep models when data is plentiful. However, the exhibit strong robustness to the lack of data, outperforming the latter in scarcity scenarios. The full SOFTNet pipeline exhibits complimentary behaviours, yielding competitive performance throughout all situations.

Table II reports the average accuracies and variances of the proposed DAMNet and SPDNetBN architectures, compared with the original SPDNet. We observe from these results a strong gain in performance on the SPDNetBN and DAMNet over the SPDNet and over the small FTCN, which validates the usage of the Riemannian BatchNorm along with the exploitation of the geometric structure underlying the data. All in all, we reach better performance with much fewer parameters, which again is a key feature for radar classification, and much others.

TABLE II
PERFORMANCE OF SPDNET, DAMNET AND SPDNETBN ON THE NATO DATASET

Model	SPDNet	DAMNet		SPDNetBN
Normalization	-	BarNorm	ParNorm	BatchNorm
Accuracy	$72.6\% \pm 0.61$	$79.9\% \pm 1.19$	$80.3\% \pm 0.55$	$\mathbf{82.3}\% \pm 0.80$
Acc. (10% data)	$69.1\% \pm 0.97$	$73.8\% \pm 0.25$	$70.2\% \pm 1.74$	$\mathbf{77.7}\% \pm 0.95$

The raw SPDNet, though not competing with the FTCN when all data is available, remains more robust to the lack data. The SPDNetBN on the other hand, exhibits a strong gain in performance to the point of competing with the deep model.

For the sake of expression, we also show in Figure 8 a confusion matrix of the SPDNetBN model for the NATO dataset.

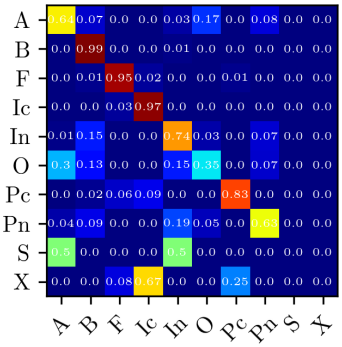

Fig. 8. **Confusion matrix on the NATO dataset**.

As stated previously, it is of great interest to consider the robustness of learning algorithms when faced with a critically low amount of data. The previous results show that when given only 10% of available training data, the SPD-based models remain highly robust to the lack of data while the FTCN plummet. Further, we study robustness on synthetic data, artificially varying the amount of training data while comparing performance over the same test set. As the simulator is unbounded on potential training data, we also increase the initial training set up to double its original size. Results are reported in Figure 9.

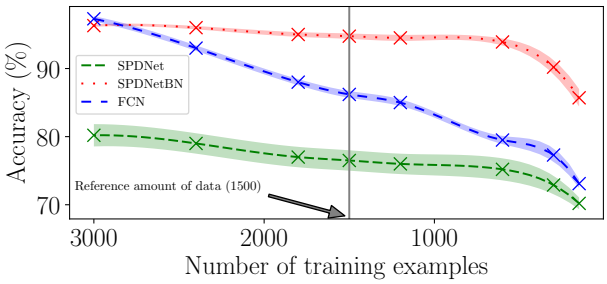

Fig. 9. **Performance of all models in function of the amount of synthetic radar data**.

V. Conclusion

The primary goal of this work was finding learning models, with the internal structure naturally suited to the input data. Two main classes of representations were considered in addition to the raw time series: a time-frequency spectrogram, *i.e.* some form of Fourier transform, and a covariance matrix, which respectively yield Euclidean, convolution-based, and Riemannian, geometry-based, neural models. We also provided a congregating solution to the signal's representation multiplicity, by making use of all representations and models in a single pipeline, called SOFTNet.

We can conclude from the experiments that the SPDNetBN both exhibits higher robustness to lack of data and performs much better than the Euclidean deep methods with much fewer parameters. When the available training data allowed skyrockets, we do observe that the FTCN comes back to par with the SPDNetBN to the point of outperforming it by a small margin in the extremal scenario; in the meantime, the SPDNet lags behind by a large margin to the SPDNetBN, which thus seems to benefit strongly from the normalization.

References

[1] P. Flandrin. *Time-Frequency/Time-Scale Analysis, Volume 10*. 1st. Orlando, FL, USA: Academic Press, Inc., 1998 (cit. on p. 2).

[2] F. Hlawatsch and F. Auger. *Time-Frequency Analysis*. en. Google-Books-ID: tOeeJyP95IQC. John Wiley & Sons, Mar. 2013 (cit. on p. 2).

[3] P. Stoica and R. L. Moses. *Spectral analysis of signals*. Upper Saddle River, N.J: Pearson/Prentice Hall, 2005 (cit. on p. 2).

[4] M. Moruzzis and N. Colin. "Automatic recognition of air targets for future shorad radars". In: *RTO SCI Symposium on" Non cooperative Air Target Identification Using Radar", Mannheim, Germany*. 1998 (cit. on p. 2).

[5] Z. Huang and L. Van Gool. "A Riemannian Network for SPD Matrix Learning". In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. AAAI'17. event-place: San Francisco, California, USA. AAAI Press, 2017, pp. 2036–2042 (cit. on pp. 2, 3).

[6] L. Yang, M. Arnaudon, and F. Barbaresco. "Riemannian median, geometry of covariance matrices and radar target detection". In: *The 7th European Radar Conference*. Sept. 2010, pp. 415–418 (cit. on p. 2).

[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)*. May 2016, pp. 1–6 (cit. on p. 2).

[8] D. Brooks, O. Schwander, F. Barbaresco, J.-Y. Schneider, and M. Cord. "Riemannian batch normalization for SPD neural networks". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Curran Associates, Inc., 2019, pp. 15463–15474 (cit. on pp. 2, 3).

[9] D. Brooks, O. Schwander, F. Barbaresco, J.-Y. Schneider, and M. Cord. "Second-Order Networks in PyTorch". en. In: *Geometric Science of Information*. Ed. by F. Nielsen and F. Barbaresco. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 751–758 (cit. on p. 2).

[10] D. 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)*. June 2018, pp. 1–10 (cit. on p. 2).

[11] D. Brooks, O. Schwander, F. Barbaresco, J.-Y. Schneider, and M. Cord. "Complex-valued neural networks for fully-temporal micro-Doppler classification". In: *2019 20th International Radar Symposium (IRS)*. ISSN: 2155-5753, 2155-5745. June 2019, pp. 1–10 (cit. on p. 2).

[12] D. Brooks, O. Schwander, F. Barbaresco, J. Schneider, and M. Cord. "Exploring Complex Time-series Representations for Riemannian Machine Learning of Radar Data". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. May 2019, pp. 3672–3676 (cit. on p. 2).

[13] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. "Log-Euclidean metrics for fast and simple calculus on diffusion tensors". en. In: *Magnetic Resonance in Medicine* 56.2 (Aug. 2006), pp. 411–421 (cit. on p. 3).

[14] X. Pennec, P. Fillard, and N. Ayache. "A Riemannian Framework for Tensor Computing". en. In: *International Journal of Computer Vision* 66.1 (Jan. 2006), pp. 41–66 (cit. on p. 3).

[15] S. Ioffe and C. Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". en. In: *International Conference on Machine Learning*. June 2015, pp. 448–456 (cit. on p. 3).

[16] S.-i. Amari. *Information Geometry and Its Applications*. en. Applied Mathematical Sciences. Springer Japan, 2016 (cit. on p. 3).

[17] A. Edelman, T. Arias, and S. Smith. "The Geometry of Algorithms with Orthogonality Constraints". In: *SIAM Journal on Matrix Analysis and Applications* 20.2 (Jan. 1998), pp. 303–353 (cit. on p. 3).

[18] C. Ionescu, O. Vantzos, and C. Sminchisescu. "Matrix Backpropagation for Deep Networks with Structured Layers". en. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. Santiago, Chile: IEEE, Dec. 2015, pp. 2965–2973 (cit. on p. 3).

[19] M. Engin, L. Wang, L. Zhou, and X. Liu. "DeepKSPD: Learning Kernel-Matrix-Based SPD Representation For Fine-Grained Image Recognition". en. In: *Computer Vision – ECCV 2018*. Ed. by V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss. Vol. 11206. Cham: Springer International Publishing, 2018, pp. 629–645 (cit. on p. 3).