Journal of the Brazilian Computer Society, 2026, 32:1, doi: 10.5753/jbcs.2026.5482

Subspace representations in deep neural networks: A survey

Stéfane Rêgo Gandra [**Federal University of Amazonas** | ***stefane.rego@icomp.ufam.edu.br***]
Bernardo Bentes Gatto [**Frontier Technology Division, MTI** | ***gatto_b@mti.co.jp***]
Eulanda Miranda dos Santos [**Federal University of Amazonas** | ***emsantos@icomp.ufam.edu.br***]

Institute of Computing, Federal University of Amazonas, Av. Gen. Rodrigo Octávio, 6200, Coroado, Manaus, AM, 69080-900, Brazil.

Received: 28 January 2025 • **Accepted:** 16 November 2025 • **Published:** 27 March 2026

Abstract. Computer vision applications often involve processing large-scale multidimensional data, requiring methods that are both efficient and accurate. Traditional pattern recognition methods based on subspace representations offer low computational complexity but typically underperform compared to deep learning models in terms of recognition accuracy. This study aims to explore and analyze the integration of subspace representations within deep learning frameworks to leverage the advantages of both approaches. We conducted a comprehensive survey of existing methods that combine subspace representation techniques with deep neural networks. We propose a taxonomy to categorize these methods into three distinct groups based on their integration strategies. The reviewed methods demonstrate that incorporating subspace representations can enhance the performance and efficiency of deep learning models. The taxonomy helps to clarify the landscape of these hybrid approaches and identifies trends in methodological development. The surveyed approaches demonstrate a clear methodological evolution, contributing to enhanced outcomes in various real-world applications.

Keywords: Deep Learning, Subspace Methods, Manifold Learning

1 Introduction

Multidimensional data require advanced techniques for efficient analysis, representation, and interpretation due to the exponential growth of data in various domains. Consequently, developing methods to extract meaningful insights from such data has become crucial in applications across various fields, especially in computer vision.

An example of a task involving multidimensional data is image set classification, which focuses on recognizing patterns from sets of images rather than individual images. It has been thoroughly explored and applied in several different applications, such as image diagnosis [Jiang *et al*., 2023], autonomous vehicles [Kuutti *et al*., 2020], robotics [Pierson and Gashler, 2017], among others. An effective pattern set model requires robustness to appropriately handle corrupted data and sets of varying sizes without increasing computational complexity [Gatto *et al*., 2021]. In this context, subspace representations address these challenges, enabling efficient processing of large and noisy datasets.

Subspace representations are pattern recognition methods where each class is represented in terms of a linear subspace of the Euclidean pattern or feature space [Oja and Kut, 1983]. In image set classification, a set of images can be represented by a fixed-dimensional subspace. This type of representation has mainly two advantages [Gatto *et al*., 2021]: the first relates to robustness to noise in the input data. Even when the collected images are not highly representative of the object, the inherent noise allows the learning process to adjust and create robust pattern representations. The second advantage is related to compression. Subspaces are low dimensional, which ensures fixed complexity when processing a set as a subspace, even if it contains a large number of input images. An example of how compression is a decisive advantage of using subspace representation is the fact that a subspace can represent a set with only 20% of its original memory space [Gatto *et al*., 2021].

Among the applications using linear subspaces in computer vision, we can mention: face verification [Sun *et al*., 2014], emotion recognition [Sun *et al*., 2018] [Zhiwu and Van Gool, 2017], and activity recognition [Lu *et al*., 2018]. These methods trace their origins back to multivariate statistical analysis, particularly Principal Component Analysis (PCA) and Canonical Correlation Analysis (CCA), developed by Hotelling [Katayama, 2005]. Such methods have attracted much attention in recent years due to their ability to provide accurate state-space models for multivariate linear systems directly from input-output data [Katayama, 2005].

In their turn, deep learning methods, particularly Convolutional Neural Networks (CNNs), have revolutionized computer vision by achieving state-of-the-art results in numerous tasks. However, they often require large amount of training data and may not inherently exploit the geometric structures present in data. Combining deep learning with subspace representations offers a promising avenue to take advantage of the strengths of both approaches, enabling efficient learning from limited data and preserving important structural information.

More precisely, methods based on subspace representations may offer a complementary approach capable of mitigating the limitations of deep learning by projecting data into lower-dimensional latent spaces, preserving relevant information and discarding noise or redundancies. Among the several benefits, subspace representations may help by reducing the complexity of the input space, which makes the learning process more efficient and less prone to overfitting; improving

the robustness of the extracted representations, thereby enhancing generalization to new data, especially in contexts with limited data or computational resources; and potentially increasing interpretability by inducing more organized and semantically coherent latent structures.

Therefore, incorporating subspace methods into the deep learning framework not only enhances the technical performance of models but also helps address fundamental limitations of traditional deep learning. This approach is particularly relevant in critical applications that demand not only accuracy and reliability but also transparency in decision-making processes.

To conduct this research, a systematic literature review was carried out with the aim of identifying studies that address the integration between subspace methods and deep neural networks. The search was performed in scientific databases using the following specific keyword combinations: "DEEP LEARNING" AND "SUBSPACE METHODS" NOT "CLUSTERING", "REVIEW" AND "SUBSPACE METHODS", and "REVIEW" AND "DEEP SUBSPACE LEARNING". From the results obtained, the abstracts were analyzed to select those most relevant to the scope of this research, with an emphasis on supervised learning approaches that employ subspace representations throughout the entire architecture of deep networks, rather than only in intermediate stages or preprocessing steps. During this process, it was observed that the first studies effectively exploring deep networks based on subspaces—integrated and end-to-end—began to emerge around 2020. This temporal milestone highlights the novelty of the topic and supports the need for a critical analysis discussing recent advances, remaining gaps, and future perspectives in this emerging research area.

1.1 The need for this survey

Figure 3 shows the amount of papers dealing with this research domain which were found in our search in the Elsevier, IEEE Xplore and Springer Link databases since 2020. Despite this interest, there is a lack of comprehensive surveys that systematically review and categorize these methods. The nonexistence of a clear taxonomy for these methods makes it difficult to compare existing approaches and identify gaps in the literature, thereby hindering the systematic advancement of the field.

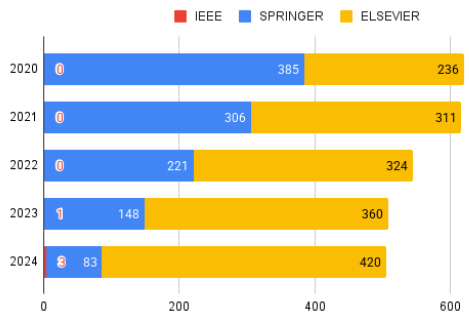

Figure 1. Timeline of research publications on the use of subspace methods combined with deep learning.

Therefore, this paper aims to fill the above-mentioned gaps by providing a structured and comprehensive review of the methods that integrate subspace representations within deep neural networks. We categorize existing approaches based on their underlying mathematical structures, specifically focusing on Riemannian manifolds, Grassmann manifolds, and Lie groups. We highlight practical implementations and comparative analyses of them. It is important to mention that there are previous surveys that discuss subspace learning methods and their theoretical foundations, such as [Turan *et al*., 2021] and [Fei *et al*., 2023]. However, unlike this work, they do not focus on the integration of subspace learning and deep neural network architectures.

The main contributions of this work are as follows:

- We introduce a novel taxonomy to organize existing methods into manifolds and groups, providing clarity on their relationships and differences;
- We analyze and compare these methods, discussing their advantages, limitations, and applications;
- We identify current challenges and suggest directions for future research in this emerging field.

This review provides a clear mapping of the strategies developed to date through critical analysis and categorization of methods. The proposed taxonomy helps to identify the strengths and weaknesses of each method, enabling future research to select, adapt, or combine methodologies according to specific needs. Finally, this work serves as a solid foundation for researchers to develop more effective solutions, overcoming the individual limitations of each isolated approach in tasks involving multidimensional data.

The rest of this paper is organized as follows. Section 2 describes the taxonomy proposed. Section 3 summarizes theoretical concepts of subspaces structures to allow a better understanding of the works discussed. Then, sections 4, 5 and 6 review methods combining subspace representation and deep neural networks. In sequence, Section 7 provides a detailed analysis considering computational performance. Finally, Section 8 presents conclusions and future work.

2 Taxonomy

Our taxonomy organizes the reviewed methods into a hierarchical structure that reflects the mathematical foundations underlying subspace representations in deep learning, as depicted in Figure 2. At the highest level, we divide the methods into two main categories based on the mathematical structures they employ: manifold-based methods and group-based methods. Manifold-based methods are further subdivided into Riemannian and Grassmann manifolds, while group-based methods focus on Lie groups. This categorization not only highlights the intrinsic relationships between the methods but also provides a clear framework for analyzing their similarities and differences. Below, we provide a brief description of each category and its significance in the context of deep learning.

Grassmann manifolds, Riemannian manifolds, and Lie groups are mathematical structures that serve as the foundation for the methods reviewed in this survey. Understanding their characteristics is essential to realize how they contribute to subspace representations in deep networks. While each

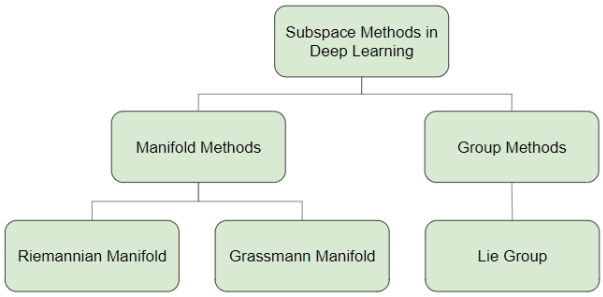

Figure 2. A taxonomy of subspace methods in deep learning, illustrating the hierarchical categorization based on mathematical structures.

structure has distinct properties, they also share common features that facilitate their integration with deep learning algorithms. We outline both the similarities and differences below to provide a comprehensive understanding of their roles. We also provide examples of the applications of these structures in computer vision tasks.

- **Geometric Structures:** All three structures have a strong geometric foundation. The Grassmann manifold describes projective subspaces in vector spaces. In its turn, the Riemannian manifold encompasses a broad class of smooth manifolds with a Riemannian metric, including subsets of positive definite symmetric matrices. Lie groups, on the other hand, are differentiable manifolds that also have a group structure, exhibiting important geometric properties. These characteristics can contribute to a better understanding and interpretation of results.
- **Differentiable Manifolds:** Grassmann and Riemannian manifold, as well as Lie groups are all differentiable structures, which means that they support smooth operations and the application of differential calculus. However, Lie groups uniquely combine manifold and group properties, where the group operations (multiplication and inversion) are smooth, providing additional algebraic structure useful in modeling transformations.
- **Applications in Computer Vision:** There are significant applications of these three structures in the field of computer vision, contributing to the analysis and processing of images in different ways.

Regarding the differences, we can highlight the following:

- **Dimensions and Characteristics:** Grassmann manifolds have dimensions that depend on the dimensions of the subspaces they represent. Riemannian manifolds are a broad class of manifolds equipped with a Riemannian metric, and their dimensions depend on the specific manifold under consideration. The space of positive definite symmetric matrices, for example, is a specific type of Riemannian manifold. Finally, Lie groups have dimensions that correspond to the dimensions of their underlying Lie algebras.
- **Group Structure:** Grassmann and Riemannian are manifolds and do not inherently have a group structure, although specific examples may have additional structures. Lie groups, on the other hand, are manifolds that are also groups, with group operations that are smooth.

These structures have been extensively employed in computer vision applications:

- **Grassmann Manifolds:** They are widely used for dimensionality reduction and data representation because they enable efficient compression of image data by projecting high-dimensional data onto lower-dimensional subspaces [Haitman *et al*., 2021a; Wang *et al*., 2017]. This is particularly beneficial in tasks like face recognition [Johnson and Savakis, 2015; Huang *et al*., 2015], where images are represented as points on the manifold to facilitate comparison and classification.
- **Riemannian Manifolds:** They are employed in medical imaging to represent complex anatomical structures. For example, symmetric positive definite (SPD) matrices capture the shape and connectivity in magnetic resonance imaging (MRI) [Ke *et al*., 2021] and diffusion tensor imaging (DTI) [Liu *et al*., 2019]. In image classification, networks such as SPDNet [Zhiwu and Van Gool, 2017] use Riemannian geometry to improve accuracy by effectively modeling spatial correlations of images.
- **Lie Groups:** They are applied in modeling geometric transformations, such as rotations and translations, which are essential in robotics and augmented reality [Rhif *et al*., 2018]. They also facilitate the representation and manipulation of object poses in 3D space. In 3D vision and virtual reality, Lie groups help to accurately render scenes from multiple points of view by modeling camera transformations [Xu *et al*., 2017].

In summary, Grassmann manifolds, Riemannian manifolds, and Lie groups are powerful mathematical tools applied across various fields of computer vision, from representing and analyzing image data to describing geometric transformations in three-dimensional scenarios. These structures significantly enhance the capability of processing and interpreting visual information in advanced computer vision systems. It is worth noting that there are several types of manifolds with diverse applications that have not yet been fully explored or have been underexplored in the literature with deep learning, such as Lorentzian Manifolds, Complex Projective Space, Symmetric Spaces, Flag Manifolds, Spheres, and Hyperspheres, among others. However, in this paper we describe only works that use Grassmann manifolds, Riemannian manifolds and/or Lie Groups.

3 Theoretical Background

The works described in this review are divided into three categories: Riemannian and Grassmann manifolds, as well as Lie group. As previously mentioned, the three categories represent mathematical tools which present distinct structures. In this section we summarize important theoretical concepts of these structures in order to allow a better understanding of the works discussed in the next sections.

3.1 Mathematical Notations

To facilitate understanding the mathematical concepts presented in this paper, we provide in Table 1 a summary of the main notations used throughout the text.

3.2 Riemannian Manifold

It is a differentiable manifold, i.e. a generalization of the concept of a curve or surface, equipped with a Riemannian metric. This metric is a way of measuring distance and angle at each point of the manifold. Consequently, it provides a notion of intrinsic geometry on the manifold, allowing the definition of curve lengths, angles between tangents, curvature, and so on. It is commonly represented by an SPD matrix at each point of the manifold, used to define the inner product between vectors in the tangent space. In general, the matrix representation of the Riemannian metric depends on the choice of coordinates and on specific properties of the manifold. Hence, it is an SPD matrix whose exact form varies depending on the context of the Riemannian manifold.

Mathematically, an SPD $\mathbf{M}$ of order n is a real square symmetric matrix ($\mathbf{M} = \mathbf{M}^T$) that satisfies the following condition for every non-zero column vector $\mathbf{x}$ of size n:

$$\mathbf{x}^T\mathbf{M}\mathbf{x} > 0 \tag{1}$$

This inequality means that the inner product of the vector $\mathbf{x}$ with the matrix $\mathbf{M}$ results in a strictly positive value. In other words, for any non-zero vector $\mathbf{x}$, the matrix $\mathbf{M}$ ensures that the inner product of $\mathbf{x}$ with $\mathbf{M}\mathbf{x}$ is always greater than zero.

In Section 4, we describe the most recent works that employ Riemannian manifolds in the context of deep learning. All reviewed methods were developed based on the research presented in [Zhiwu and Van Gool, 2017], which uses SPDs matrices defined in deep learning, and investigate tasks involving video data. For this reason, we also present here some mathematical definitions in the context of video data.

Consider $S_k = [s_1, s_2, ..., s_{n_k}]$ a set of frames where S_k represents a sequence with n_k frames and $s_j \in \mathbf{R}^{d\times 1}$ denotes the j^{th} vectorized frame [Wang *et al*., 2023a]. The corresponding covariance matrix for S_k is computed by:

$$\mathbf{X_k} = \frac{1}{n_k - 1}\sum_{j=1}^{n_k}(s_j - u_k)(s_j - u_k)^T. \tag{2}$$

Here, $u_k = \frac{1}{n_k}\sum_{j=1}^{n_k} s_j$ is the mean vector of the frames in S_k. Given that $\mathbf{X_k}$ does not satisfy the positive definite condition (there may exist a non-zero vector y such that the quadratic form $y^T\mathbf{X_k}y$ is non-positive), it is necessary to regularize the matrix $\mathbf{X_k}$, which will be defined as:

$$\mathbf{X_k} + \lambda I_d \tag{3}$$

where I_d is the identity matrix of order $d \times d$, and λ is defined as the trace of $\mathbf{X_k}$—the sum of the elements on the main diagonal of $\mathbf{X_k}$. Therefore, after this operation, the regularized matrix $\mathbf{X_k}$ becomes an SPD matrix.

3.3 Grassmann Manifold

A Grassmann manifold $Gr(q, D)$ is a dimensional compact Riemannian manifold $q(D - q)$, which is the set of q-dimensional linear subspaces of $\mathbf{R}^D$. Each point in $Gr(q, D)$ is a linear subspace that is generated by the related orthonormal basis matrix $\mathbf{M}x$ of dimension *D* x *q*, such that $\mathbf{M}^T\mathbf{M}$ = $\mathbf{I}_q$, where $\mathbf{I}_q$ is the identity matrix of order *q* x *q*. One of the most popular approaches to representing linear subspaces is to approximate the true Grassmannian geodesic distance from the projection mapping structure $\phi(\mathbf{M}) = \mathbf{M}\mathbf{M}^\mathbf{T}$, proposed in Edelman *et al*. [1998]. Since the $\phi(\mathbf{M})$ is a symmetric matrix *D* x *D*, a natural choice of inner product is $\langle \mathbf{M_1}, \mathbf{M_2} \rangle_\phi = tr(\phi(\mathbf{M_1})^T\phi(\mathbf{M_2}))$, which induces a distance metric called the projection metric:

$$d_p(\mathbf{M_1}, \mathbf{M_2}) = 2^{-\frac{1}{2}} \left\| \mathbf{M_1}\mathbf{M_1}^T - \mathbf{M_2}\mathbf{M_2}^T \right\|_F \tag{4}$$

where $\|\|_F$ indicates the Frobenius norm of the matrix. As proven in [Harandi *et al*., 2011], the projection metric can approximate the true geodesic distance up to a scale of $\sqrt{2}$.

To perform discriminative learning on Grassmann manifolds, some studies [Hamm and Lee, 2008b,a; Cetingul and Vidal, 2009; Harandi *et al*., 2011] use tangent space approximation of the underlying manifolds or exploit kernel functions to incorporate the manifolds in Hilbert spaces. In both cases, any existing Euclidean techniques can be applied to the data, as Hilbert spaces respect Euclidean geometry.

3.4 Lie group

A Lie group is a set of elements whose group operation satisfies the properties of associativity, identity, and inversion. Additionally, Lie groups have a special characteristic: they are "smooth" or "differentiable", meaning they can be studied in terms of smooth functions, especially differentiable functions. More precisely, these groups possess a continuous structure that is compatible with operations such as multiplication and inversion. They also exhibit a structure that allows for differential calculus and the application of concepts from differential geometry to group elements. These characteristics enable Lie groups to be studied using tools from differential analysis and differential geometry. Consequently, they can be equipped with structures that are compatible with a Riemannian metric.

This structure can be employed in computer vision problems as a tool for modeling transformations and symmetries in video data. For instance, in action recognition tasks in videos, the Lie group theory is used to describe the spatiotemporal transformations that occur in videos, helping to capture important information such as motion dynamics, pose variations, and action representation from different viewpoints. This provides a way to learn robust and compact representations of human actions. This approach typically involves applying convolutional layers with Lie group-constrained parameters in deep neural network architectures [Vemulapalli *et al*., 2014], allowing the network to learn features that are invariant to certain geometric and temporal transformations.

Considering this context, all works discussed in Section 6 employ Lie group representations in action recognition

Table 1. Summary of Mathematical Notations

Notation	Description
$\mathbf{M}$	Symmetric Positive Definite (SPD) matrix
$\mathbf{x}$	Non-zero column vector of size n
$\mathbf{X}_k$	Covariance matrix of the k-th sequence
S_k	Set of frames in the k-th sequence
s_j	j-th vectorized frame
d	Dimension of the feature vector
u_k	Mean vector of frames in S_k
λ	Regularization parameter (trace of $\mathbf{X}_k$)
I_d	Identity matrix of size $d \times d$
$Gr(q, D)$	Grassmann manifold of q-dimensional subspaces in $\mathbb{R}^D$
$\mathbf{M}_x$	Orthonormal basis matrix generating a point on $Gr(q, D)$
$\phi(\mathbf{M})$	Projection mapping $\phi(\mathbf{M}) = \mathbf{M}\mathbf{M}^T$
$\langle \mathbf{M}_1, \mathbf{M}_2 \rangle_\phi$	Inner product defined as $\mathrm{tr}(\phi(\mathbf{M}_1)^T \phi(\mathbf{M}_2))$
$d_p(\mathbf{M}_1, \mathbf{M}_2)$	Projection metric distance between $\mathbf{M}_1$ and $\mathbf{M}_2$
$\|\cdot\|_F$	Frobenius norm of a matrix
$SE(3)$	Special Euclidean group representing rigid transformations in 3D
R	Rotation matrix in $SE(3)$
t	Translation vector in $SE(3)$
$\times$	Direct product between Lie groups

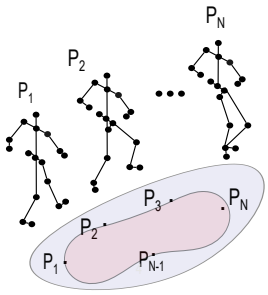

Figure 3. Representation of an action from a sequence of skeletons as a curve in the Lie group $SE(3) \times \ldots \times SE(3)$. Source: [Vemulapalli *et al.*, 2014].

problems, specifically using human skeleton keypoint data as input. The authors based their work on the approach presented in [Vemulapalli *et al.*, 2014], which represents the relative geometry between pairs of body parts as a point in the special Euclidean group $SE(3)$, and the entire human skeleton as a point in the Lie group $SE(3) \times \ldots \times SE(3)$, where $\times$ denotes the direct product between Lie groups.

A special Euclidean group is a Euclidean terminology in a 2D or 3D space. Special Euclidean transformations include rotations, translations, and reflections that preserve distances and angles. Thus, the Lie group $SE(3)$ represents rigid three-dimensional transformations, i.e., transformations that preserve distances and angles in 3D spaces. This group consists of ordered pairs (R, t), where R is a 3x3 rotation matrix describing a rotation in 3D space, and t is a 3D translation vector. Therefore, human actions can be modeled as curves in the Lie group, and action recognition can be performed by classifying these curves. Figure 3 illustrates such representation. The Lie group $SE(3)$ is commonly used in geometry, robotics, and computer vision to describe the three-dimensional motion of objects or systems.

Once these theoretical definitions have been presented, in the next sections each of the nodes of the proposed taxonomy is explored and the reviewed works are detailed.

4 Methods Based on Riemannian Manifolds

The first network based on the Riemannian manifold proposed is called the *Symmetric Positive Definite Matrix Network* (SPDNet) [Zhiwu and Van Gool, 2017]. The input layer of SPDNet receives the SPD matrix $\mathbf{X}_0$, i.e., a covariance matrix derived from the input data, defined according to equations 2 and 3. Subsequently, two types of Riemannian operations layers are employed: 1) bilinear mapping (BiMap); and 2) eigenvalue rectification (ReEig). The BiMap layers are designed to transform the input SPD matrices into new SPD matrices with lower dimensionality and more discriminative features. This is done through a bilinear mapping f_b, expressed as $\mathbf{X_k} = f_b^{(k)}(\mathbf{X_{k-1}}, W_k) = W_k \mathbf{X_{k-1}} W_k^T$, where W_k is a column transformation matrix with semi-orthogonality to be learned during training and $\mathbf{X_{k-1}}$ is the input SPD matrix at the k-th layer. This type of layer is analogous to the dense layer in traditional artificial neural networks, but it is defined for SPD data.

The ReEig layers, similar to the classical ReLU layers in dense networks, introduce non-linearity into the SPDNet. This is done by rectifying the resulting SPD matrices through a non-linear rectification function f_r, formulated as $\mathbf{X_k} = f_r^{(k)}(\mathbf{X_{k-1}}) = U_{k-1} \max(\epsilon I, \Sigma_{k-1}) U_{k-1}^T$, where $\mathbf{U_{k-1}}$ and Σ_{k-1} are found through singular value decomposition, ϵ is a small activation threshold, I is the identity matrix, and $\max(\epsilon I, \Sigma_{k-1})$ is the diagonal matrix of Σ_{k-1} whose elements are defined as: A(i,i) = $\sum_{k-1}(i,i)$ if $\sum_{k-1}(i,i) > \epsilon$ and ϵ if $\sum_{k-1}(i,i) \leq \epsilon$.

Additionally, due to the fact that SPD matrices reside on the Riemannian manifold, which is non-Euclidean, it is necessary to include a third type of layer, called the eigenvalue logarithm layer (LogEig). This layer performs Riemannian computation on the SPD matrices and aims to produce the Euclidean forms of the SPD matrices for

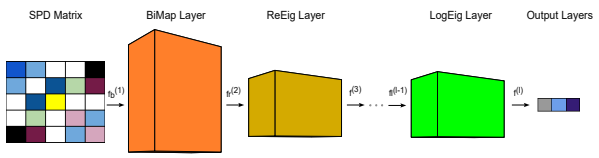

Figure 4. Diagram of the SPDNet Structure. The network is composed of BiMap, ReEig, and LogEig layers. Source: [Zhiwu and Van Gool, 2017].

any regular output layer of the network. The LogEig layer is designed to perform the following mathematical operation: $\mathbf{X_k} = f_r^{(k)}(\mathbf{X_{k-1}}) = U_{k-1}\log(\Sigma_{k-1})U_{k-1}^T$, where $\mathbf{X_{k-1}} = U_{k-1}\Sigma_{k-1}U_{k-1}^T$ represents the singular value decomposition operation and $\log(\Sigma_{k-1})$ is the element-wise logarithm of the diagonal elements of Σ_{k-1}. With this operation, conventional fully connected (FC) layers can be used on the resulting Euclidean space for classification tasks. Figure 4 shows a representation of the SPDNet architecture, highlighting the three types of layers that make up the network.

The authors used backpropagation of the Riemannian matrix, so that, for each layer k, the gradients of the weights W_k are computed. The weights in the BiMap layers are updated using a configuration of the stochastic gradient descent (SGD) algorithm [Bonnabel, 2013a] on Stiefel manifolds. In their approach, the authors expanded the backpropagation matrix framework, initially studied in [Ionescu *et al*., 2015], to compute the gradients of SPD matrices within the ReEig and LogEig layers, ensuring more accurate and efficient gradient computation in these layers. This generalization not only extends the applicability of backpropagation but also enhances the learning capabilities of SPDNet. Overall, SPDNet provides the possibility of nonlinear learning of the SPD matrix, as well as a new backpropagation with SGD configuration on Stiefel manifolds.

However, in [Chen *et al*., 2022], the authors argue that very few solutions explicitly explore the local geometric information in deep representations of SPD features in the Riemannian manifold of SPD matrices. According to them, considering the success of local mechanisms in Euclidean methods, it is very important to ensure the preservation of geometric information in SPD networks. Taking into account this context, they proposed the MSNet architecture, whose structure can be observed in Figure 5. Firstly, the authors employed the SPDNet [Zhiwu and Van Gool, 2017] as a backbone to extract lower-dimensional, yet more discriminative SPD features representations. Next, branches are created, with each branch exploring a BiMap-ReEig block to obtain SPD representations. Finally, the LogEig layer is applied to map each submatrix feature into a Euclidean space, as done in [Zhiwu and Van Gool, 2017]. After that they apply a process called TrilCon, which involves the following three steps: extraction of a lower triangular matrix, vectorization, and data concatenation. In the end, all vectors from the different branches are concatenated through the Concat layer. This, in turn, is followed by any regular output layers, such as the softmax layer.

In [Wang *et al*., 2022], the SPDNet architecture is modified. The authors propose two new layers for the analysis and processing of data points on Riemannian manifolds with the aim of improving the capability of nonlinear representation learning of the original SPDNet, as shown in Figure 6. It is worth noting that the BiMap, ReEig, LogMap, and FC layers are the same as those defined in [Zhiwu and Van Gool, 2017]. According to the authors, in the context of neural networks with multiple SPDs, the variability information conveyed by the inputs is not explicitly encoded during training, causing the lower-dimensional geometric representations resulting from each transformation layer to potentially be insignificant for producing a powerful network. Thus, the first module of the network (SPDNetRP) is the Riemannian Batch Regularization (RBR) module and is located before the BiMap layer. RBR is primarily used to provide more suitable initial filters for the BiMap layers in each forward pass, encoding and learning the variations in the representations. This task is accomplished by encoding intra-class variability information and inter-class similarity information present in a data batch.

The transformation matrices to be learned in this process are guided by supervision based on labels and distribution. Consequently, the features resulting from these transformations will exhibit high intra-class compactness and inter-class separability. The proposed RBR module is trained together with SPDNet through Riemannian matrix backpropagation. Thus, the initial weights of the RBR layers are set to be equal to those of the corresponding BiMap layers. It is important to note that the RBR module is flexible in its connection with other components, allowing arbitrary adjustments in its configuration.

The second proposed module is a pooling layer that precedes the ReEig layer. Due to the non-Euclidean geometry of the representations, the authors proposed the Riemannian pooling operation implemented in three steps. The first step involves the use of the theoretical and practical framework of the Riemannian barycenter to calculate a batch of input SPD matrices. Then, in the second step, the pooling operation can be expressed as an unsupervised metric learning problem that aims to find an orthogonal mapping to generate a lower-dimensional manifold with maximum data variance. Finally, the task of learning the embedding mapping is performed, which is an optimization problem on the Grassmannian manifold solved using the Grassmannian Conjugate Gradient (GCG) method.

In [Wang *et al*., 2023a], a new architecture for processing and classifying SPD matrices is proposed, aiming to avoid the problems of stacking many layers in Riemannian networks. The authors argue that simply increasing the network depth does not guarantee accuracy gains, as this solution's main obstacle is the degradation of statistical information as the network becomes deeper. As the network deepens, the learned representations lose their ability to effectively capture the structural information of the original data, leading to impaired accuracy. Considering this challenge, they proposed a deep discriminative Riemannian network called DreamNet. This network consists of a Stacked Riemannian Autoencoder (SRAE) added at the end of the *backbone* (SPDNet) to reconstruct the remaining structural details of the input block by block. The advantage of this approach lies in the fact that it not only performs discriminative feature selection but also

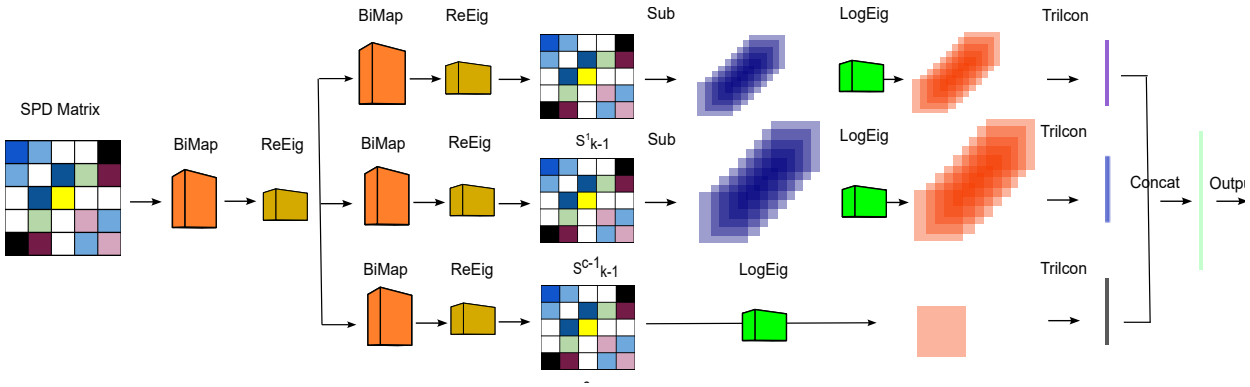

Figure 5. Diagram of the MSNet structure. Source: [Chen *et al*., 2022].

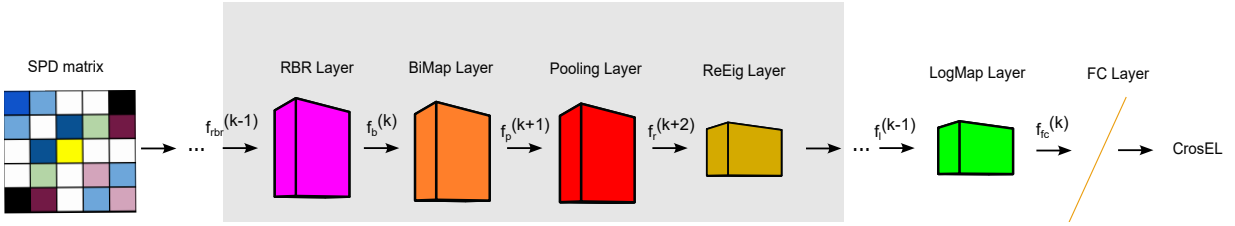

Figure 6. Diagram of the SPDNetRP structure. Source: [Wang *et al*., 2022].

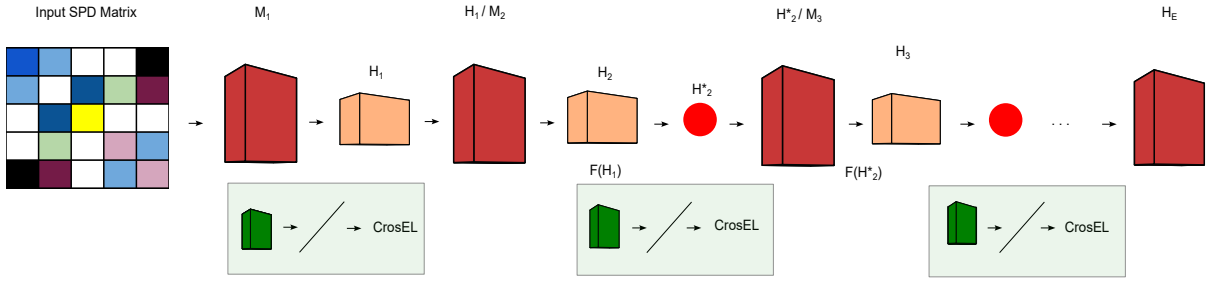

Figure 7. Architecture of the DreamNet network. Source: [Wang *et al*., 2023a].

preserves the Riemannian geometric structure of the original data manifold.

The architecture diagram of DreamNet can be seen in Figure 7. The encoder and decoder components of the SRAE are formed by layers of the original SPDNet. It is a cascade of Riemannian Autoencoders (RAEs), where the output feature maps of each RAE are used as input for the adjacent RAE. Each RAE is strictly defined in the context of SPD matrices, as is the SRAE and the entire network. Residual blocks are added to facilitate the reconstruction of the remaining residue by the SRAE. Additionally, the minimization of reconstruction error allows SRAE to remain highly sensitive to variability in representations in the new generated subspaces. Furthermore, each RAE also connects to a classification network through the LogEig and FC layers, using the cross-entropy loss function. To better explain how SRAE works, it is necessary to understand the following steps:

1) Considering the definitions given in equations 2 and 3, X_i represents the SPD matrices of the *backbone* network. The authors employed the following 5 layers in this network: $X_i \rightarrow$ BiMap $\rightarrow$ ReEig $\rightarrow$ BiMap $\rightarrow$ ReEig $\rightarrow$ BiMap;

2) In the SRAE module, there are M_e, H_e, and H_e^*, which respectively represent the input, the output of the hidden layer, and the reconstruction of the e-th input of each RAE. These layers perform the same operations as described in [Wang *et al*., 2023a].

3) The classifier module consists of 3 layers: $H_e \rightarrow$ LogEig $\rightarrow$ FC $\rightarrow$ Cross-Entropy. Cross-entropy was used to minimize the classification error of input-target pairs, and the reconstruction error measures the discrepancy between the input sample and its corresponding reconstruction.

The last work described in this section is [Wang *et al*., 2023b], where another model based on autoencoders is proposed to also address the need to handle the problem of degradation of structural information that may occur during feature transformation across multiple layer stacking stages. The proposed network is called U-SPDNet, which is a deep neural network for visual classification. This network consists

of two subsystems. The first is an encoder in the form of SPDNet [Zhiwu and Van Gool, 2017] to capture compact representations of the input data. The second is the decoder, designed to increase the dimension of the SPD matrix. As is typical in autoencoder models, the decoder has a structure symmetric to that of the encoder, and reconstruction error is utilized. The authors advocate that the model gradually alleviates the problem of data degradation during training.

Figure 8 presents a conceptual illustration of the U-SPDNet architecture. The structure of the encoder network (symmetric decoder) consists of five layers: $M_0 \rightarrow$ BiMap $\rightarrow$ ReEig $\rightarrow$ BiMap $\rightarrow$ ReEig $\rightarrow$ BiMap. Additionally, residual connections from the encoder to the decoder are added through the LFE (Log-Fusion-Exp) module to enhance the model's representation capacity. The LFE operates as follows: the input SPD matrices are first mapped to the logarithmic domain using the LogEig operation. Next, since the mapped space is compatible with Euclidean geometry, the Euclidean barycenter $\tilde{A}$ is computed. Finally, $\tilde{A}$ is incorporated back into the SPD manifold through the exponential mapping of the matrix, where $\tilde{A} = U\Sigma U^T$ denotes the eigenvalue decomposition and the diagonal matrix composed of exponential eigenvalues. Furthermore, the outputs of the encoder are also fed into a classification network, consisting of a LogEig layer and an FC layer, guided by cross-entropy.

It is important to note that both the inputs and outputs of the network are structured SPD matrices, in other words, the network is strictly defined on the Riemannian manifold. Furthermore, optimizing the network parameters involves exploring SGD on the Stiefel manifold while employing Riemannian matrix backpropagation, accurately characterizing and preserving the intrinsic Riemannian geometry of the SPD data. This approach not only maintains the mathematical integrity of the data but also enhances the network's ability to learn and generalize from the complex, high-dimensional structures within the SPD matrices.

4.1 Comparison between the works

The works described in this section are the most recent researches regarding the use of SPD matrices applied in deep networks that were found in our literature review. Table 2 provides a brief summary of the main highlights of the architectures described here. Based on this table, some observations can be made.

Among the described works, two of them use SPDNet as a backbone: MSNet and DreamNet. In the case of MSNet, the authors aim to explore the geometric information of SPD matrices, as these matrices have geometric interpretations related to convexity preservation, ellipse modeling, distance measurement, PCA, optimization, and various other applications in mathematics and science. They play a fundamental role in many fields and have practical applications in a wide range of problems. On the other hand, in DreamNet, the goal is to prevent the degradation of information in SPD matrices by using a network architecture in which SPDNet serves as the backbone of a stack of autoencoders.

SPDnetRP and U-SPDNet do not use SPDNet as a backbone. In the case of SPDnetRP, the authors modified the original SPDNet architecture to enhance its learning capability, while U-SPDNet employs an encoder-decoder to address the issue of information degradation. It should be mentioned that, despite focusing on reducing information degradation using autoencoders, U-SPDNet differs in structure and methodology from DreamNet, mainly in their input handling: DreamNet utilizes SPDNet as a backbone, whereas U-SPDNet uses a single SPD matrix. Additionally, their architectures differ significantly: DreamNet consists of a cascade of Riemannian autoencoders, whereas U-SPDNet employs a single encoder-decoder pair.

In addition, except for SPDNetRP, all works use SGD in the learning process, guided by the cross-entropy loss function as used in [Zhiwu and Van Gool, 2017]. The only work that took a different approach is the one developed in [Wang *et al*., 2022], which uses SGD in the SPDNet layers (BiMap and ReEig) and GCG in the pooling layer. This choice is due to the task of transforming the learning mapping into an optimization problem on the Grassmannian manifold.

Finally, the main difference between the works lies in the changes made to the SPDNet architecture. In [Chen *et al*., 2022], the authors proposed a multiscale submanifold block that utilizes SPDNet throughout the architecture. In their turn, [Wang *et al*., 2023a] proposed a stacked Riemannian autoencoder network that uses layers from SPDNet. The model more similar to the original SPDNet was developed in [Wang *et al*., 2022]. These authors applied Riemannian batch regularization (RBR) before the BiMap layer and a pooling layer before the ReEig layer. Conversely, the network with the most modifications was presented in [Wang *et al*., 2023b]. The architecture is U-shaped and consists of an encoder-decoder setup to enhance the sampling of encoded features. In terms of applications, the networks based on the Riemannian manifold presented here were employed to emotion recognition, action recognition, facial verification, dynamic scene classification, among others.

5 Methods Based on Grassmann Manifold

The Grassmannian manifold has been incorporated into deep learning to speed up its operation or enhance performance in specific applications such as data augmentation [Hong *et al*., 2019], dimensionality reduction [Haitman *et al*., 2021b; Liu *et al*., 2018], and in the design of deep networks, which is the focus of this review.

The research developed by Zhiwu *et al*. [2018] was the first to propose a deep network architecture on the Grassmann manifold, called GrNet. Figure 9 illustrates the architecture of GrNet. Its input is an orthonormal matrix. Similarly to CNNs, GrNet also constructs a block-wise projection network containing convolutional and FC layers. The first block, named the Projection Block, consists of the FRMap and ReOrth layers. The first transforms the orthonormal matrices of input subspaces into new matrices through a linear mapping function. Since the result of this transformation is generally not an orthonormal basis matrix, the ReOrth layer uses a QR decomposition normalization strategy to address this issue. It is worth noting that a QR decomposition (also known as QR factorization) of a matrix $\mathbf{M}$ is a decomposition of $\mathbf{M}$ into a

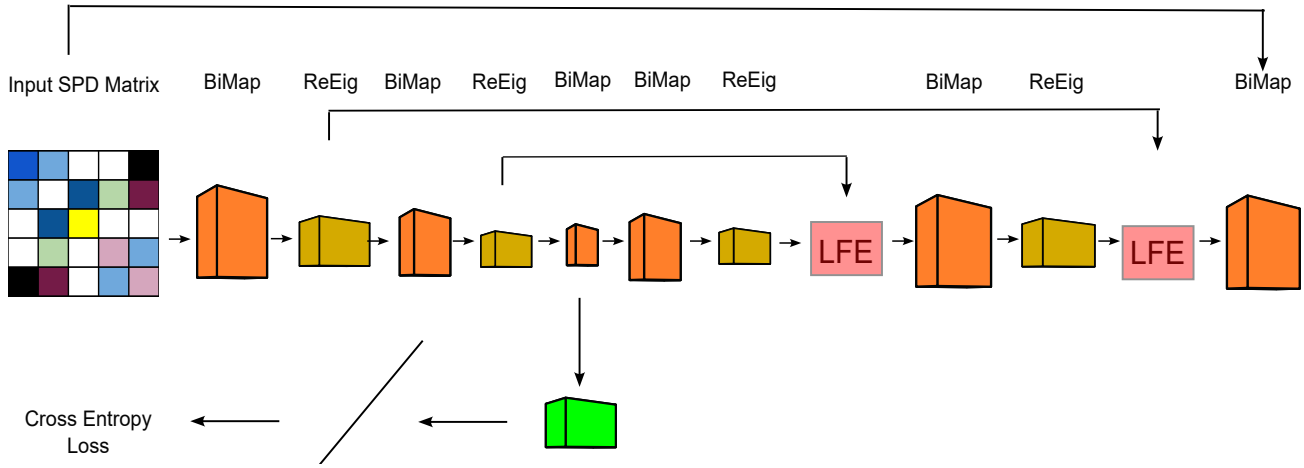

Figure 8. Schematic diagram of U-SPDNet. Source: [Wang *et al*., 2023b].

Table 2. Summary of the architectures using SPDNet.

Proposed Method	SPDNet as backbone	Number of network layers	Optimization Algorithm	Architecture Modifications	Applications
SPDNet	-	8	Riemannian SGD	-	Emotion recognition, action recognition and facial verification
MSNet	Yes	7	SGD	Yes	Action recognition
SPDNetRP	-	12	GCG	Yes	Video-based emotion recognition, dynamic scene classification and manual action recognition
DreamNet	Yes, with an SRAE built in tail	13	SGD	Yes	Video-based facial emotion recognition, skeleton-based hand action recognition and skeleton-based human action recognition
U-SPDNet	-	13	SGD	Yes	Dynamic scene classification, cell identification and skeleton-based hand action recognition

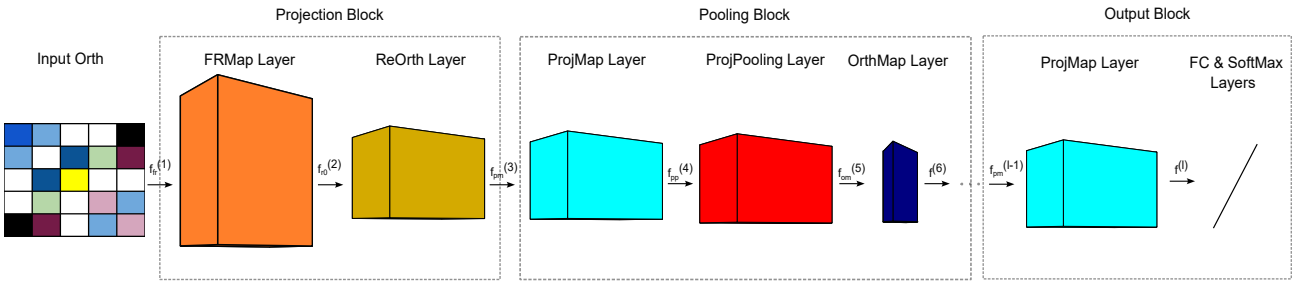

Figure 9. GrNet architecture. Source: [Zhiwu *et al*., 2018].

product $\mathbf{M} = QR$ of an orthogonal matrix Q and an upper triangular matrix R. QR decomposition is commonly used to solve the linear least squares problem and forms the basis for the QR eigenvalue algorithm.

The second block, named the Pooling Block, consists of the layers ProjMap, ProjPooling, and OrthMap, respectively. Grassmannian data is first mapped to the space of projection matrices through the ProjMap layer. Following this, with the resulting m projection matrices, the ProjPooling layer performs mean pooling over Grassmannian points. The OrthMap layer then transforms the resulting average projection matrix back into orthonormal data. This block is designed similarly to classical pooling layers, with pooling functions that reduce the size of representations to alleviate computational effort and prevent overfitting.

Finally, in the Output Block, the ProjMap layer is applied

again, resulting in outputs that represent projection matrices in Euclidean space. These outputs can then be converted into vector forms. Consequently, it becomes possible to use existing classic layers for Euclidean space data on top of the ProjMap layer. For instance, FC layers can be employed with Softmax in classification problems.

Several layers are expressed through complex matrix factorization functions throughout GrNet. Due to this complexity, backpropagation of matrices cannot be derived using the traditional matrix handling elementary operations format. Consequently, using traditional backpropagation would result in configuration failures. To address this issue, the authors used the SGD algorithm proposed in [Bottou, 2010] and [Bonnabel, 2013b]. The GrNet performance was compared to four groups of existing methods, including SPDNet, across three visual classification tasks: emotion recognition, action recognition, and facial verification. Among the results obtained, GrNet only outperformed the VGGDeepFace and DeepO2P methods.

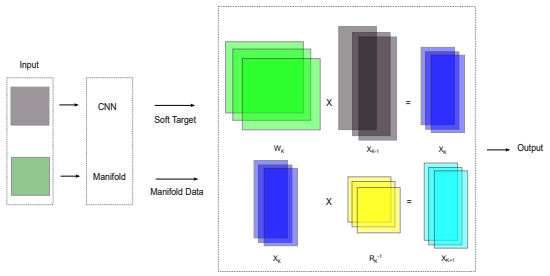

Figure 10. Knowledge Distillation from a Grassmannian Variety Network for Remote Sensing Scene Classification. Source: [Tian *et al*., 2021].

Another work using Grassmannian manifold in deep networks is presented in [Tian *et al*., 2021]. The authors employ GrNet as part of the proposed method, which is based on the knowledge distillation approach. This model is designed to handle remote sensing data. An overview of the method is shown in Figure 10. In the first stage, remote sensing images are processed by a CNN, which acts as the teacher network (TNet). This network is trained to produce soft targets from the input images. These soft targets provide additional supervision information to aid in training the second network of the model, the manifold network. GrNet forms the Manifold Network block, where the original remote sensing images are transformed into manifold data. This process involves converting the images into points on the Grassmannian manifold, which is the appropriate input space for GrNet. This transformation is crucial for representing data in a suitable manner for training the manifold network effectively.

Finally, GrNet is trained using the manifold data obtained in the previous stage as input. The soft targets and the real targets are used as supervision information. The soft targets generated by the TNet contain knowledge distilled from a large dataset. On the other hand, the real targets are the predefined labels that serve as reference information during training. According to the authors, with these combined steps, the resulting model benefits from the accumulated knowledge in TNet and the more appropriate data representation in the manifold space (GrNet). This training approach with soft and real targets allows the manifold network to converge more rapidly and make more accurate predictions compared to training solely on raw data without knowledge distillation.

The last work described in this section is presented by Souza *et al*. [2023], where the authors first introduce a structure for the Learning Mutual Subspace Method (LMSM) and subsequently propose a Grassmannian version of this method: Grassmannian-LMSM (G-LMSM). The authors employed a CNN as a feature extraction backbone and the proposed methods as classifiers, creating an end-to-end framework applied to image set recognition. A conceptual representation of the key idea behind LMSM and G-LMSM methods can be seen in Figure 11.

From this figure, it can be observed that LMSM learns and optimizes subspaces in Euclidean space. This process is expected to result in losing the structure of the subspaces. G-LMSM was proposed to overcome this limitation of LMSM. To address this issue, G-LMSM employs the structure-preserving properties of the Grassmannian manifold by performing learning via stochastic Riemannian gradient descent (RSGD), maintaining the geometric integrity of the subspaces. The following variations for G-LMSM are proposed: 1) a repulsion loss to maximize the representativeness of G-LMSM; 2) a square root activation function to mitigate overconfidence; 3) an FC layer after a G-LMSM layer to represent more abstract objects; and 4) the use of softmax to control model confidence and scale similarity values.

Among the variations proposed, G-LMSM + softmax, whose architecture can be seen in Figure 12, stood out by surpassing baselines such as Grassmann Discriminant Analysis (GDA) [Hamm and Lee, 2008b] and Graph embedding discriminant analysis on Grassmannian manifolds (GGDA) [Harandi *et al*., 2011] for improved image set matching, which have a similar purpose of mapping multiple Grassmann data into a vector representation. This variant also outperformed GRNet and SPDNet because, as stated by the authors, G-LMSM + softmax achieved competitive results with fewer layers and without decompositions, making it naturally parallelizable and scalable.

5.1 Comparison between the works

Table 3 provides a brief summary of Grassmann manifold-based architectures to establish a comparison between these architectures. This table shows that all three studies differ in their focus and approaches. In summary, the first explores building deep networks directly on the Grassmann manifold rather than conventional networks in vector spaces. It is proposed a deep neural network architecture to exploit the geometric characteristics of this manifold to enhance the performance of machine learning tasks, as classification and pattern recognition, leveraging the intrinsic subspaces structure.

The second study focuses on the specific task of remote sensing classification. Knowledge distillation is employed to

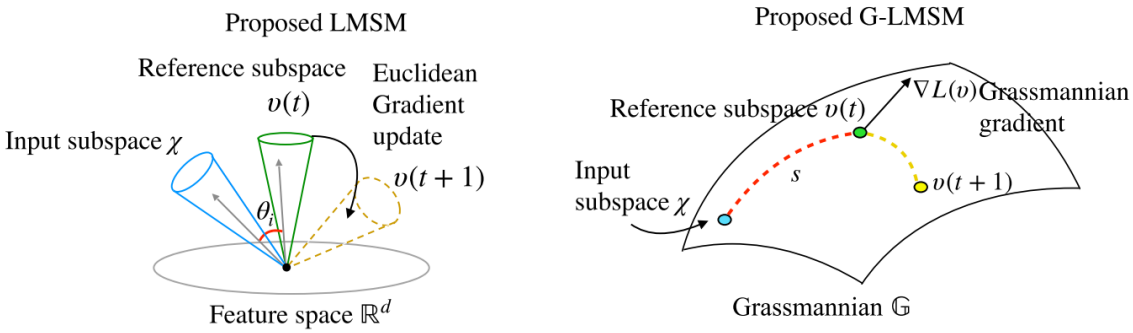

Figure 11. Representation of the Learning Mutual Subspace Method (LMSM) and Grassmannian-LMSM (G-LMSM). Source: [Souza *et al*., 2023].

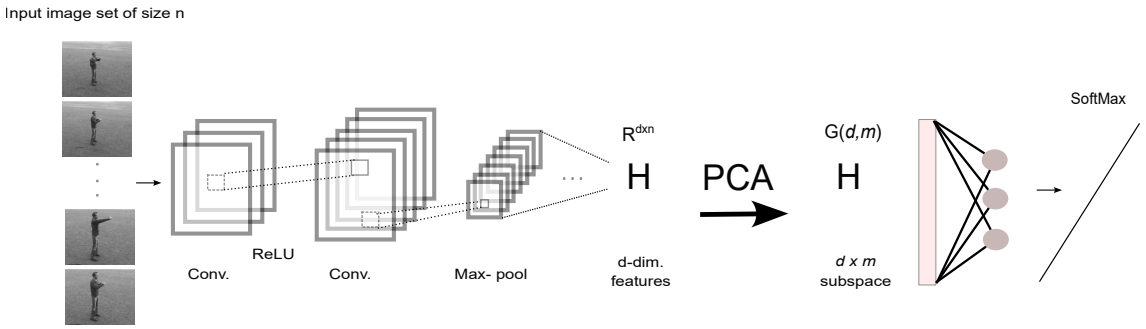

Figure 12. G-LMSM + SOFTMAX. Source: [Souza *et al*., 2023].

transfer information from a CNN to GRNet. The emphasis is on compressing information from a complex model to a simpler model while maintaining performance. Finally, the third study addresses the problem of image set classification. The authors use a mutual subspace learning method on a Grassmann manifold to capture shared information among images in each set. The focus is on leveraging subspace relationships between images to improve recognition performance.

In addition to these differences, Table 3 indicates variations in the use of loss functions and different optimizers across the studies. Therefore, it can be concluded that using the Grassmann manifold in deep networks is beneficial for representing and manipulating high-dimensional data, a common characteristic in many practical problems. These methods rely on projecting data into a lower-dimensional space while preserving relevant information. However, to date, there have been relatively few studies exploring the use of Grassmann manifold in deep networks.

6 Methods Based on Lie group

All works described in this section use Lie group representations in action recognition problems, specifically with human skeleton keypoint as input data. However, before describing these works, it is important to mention that one of the early research efforts using Lie groups in deep networks is the work developed in [Yang and Li, 2017]. In this study, the authors introduce two algorithms. The first is a single-layer Lie group model, demonstrating how the representation learning algorithm can be stacked to produce a deep architecture. The second is a Lie group-based gradient descent algorithm designed to solve the network weight learning problem. The algorithms are evaluated using the MNIST dataset. The results showed that the proposed techniques yield significantly more suitable representations for training deep networks and are also computationally efficient.

Zhiwu *et al*. [2017] proposed a deep network architecture to learn a Lie group and generate skeleton data representations, called LieNet. Similar to conventional CNNs, LieNet also has convolutional layers and pooling layers, referred to as rotation mapping layers (RotMap) and rotation pooling layers (RotPooling), respectively. The architecture of LieNet is shown in Figure 13. The proposed RotMap layers perform transformations on the input rotation matrices to generate new rotation matrices that possess the same multiple property [Tenenbaum *et al*., 2000; Roweis and Saul, 2000; Belkin and Niyogi, 2003]. To reduce the dimensionality of the Lie group, the RotPooling layers aim to pool the rotation matrix results at both the spatial and temporal levels. Moreover, due to the fact that rotation matrices are in non-Euclidean manifolds, the logarithm mapping layer (LogMap) is designed to perform Riemannian calculations. Finally, SGD was used to train LieNet.

The authors evaluated LieNet on three 3D human action datasets to verify the effectiveness of the proposed method. The datasets were G3D-Gaming [Bloom *et al*., 2012], HDM05 [Müller *et al*., 2007], and NTU RGB+D [Shahroudy *et al*., 2016]. LieNet with 3 blocks achieved better results compared to its versions with 1 and 2 blocks. In addition, it outperformed the following baseline methods: RBM+HMM Nie

Table 3. Summary of architectures using Grassmann manifold.

Proposed Method	Network Type	Type of Optimizer	Loss Function	Advantages	Disadvantages
GRNet	DNN(Deep Neural Network) composed of projection blocks containing convolution and FC layers	SGD	Euclidean	It is more robust to data variations because the Grassmann space geometry provides some invariance to geometric transformations, e.g. rotations and scale change	High computational cost
[Tian *et al*., 2021]	Grassmannian Network in Knowledge Distillation	SGD	Cross Entropy	It uses information learned by CNNs to guide the training of GRNet	Problem-dependent design: remote sensing
G-LMSM	DNN with FC layers and CNNs as feature extractors. It uses LMSM of Grassmann in an end-to-end model	RSGD	Cross Entropy	It integrates subspace-based methods into modern DNNs, maintaining end-to-end training capability	Fewer libraries, and documentation available compared to other architectures

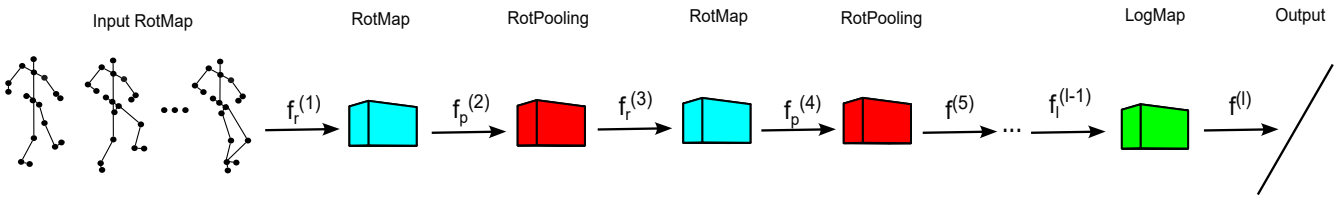

Figure 13. Architecture of the LieNet network. Source: [Zhiwu *et al*., 2017].

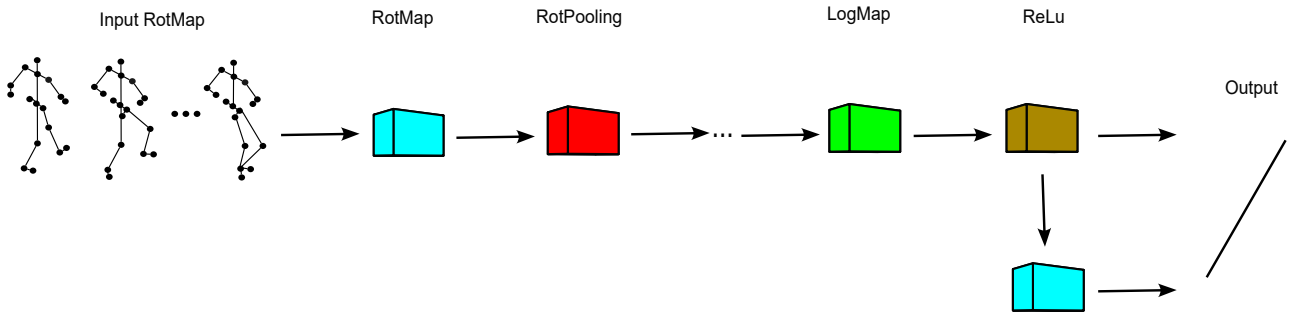

Figure 14. Architecture of the LS-LieNet network. Source: [Li *et al*., 2019].

and Ji [2014], SE [Vemulapalli *et al*., 2014], SO [Vemulapalli and Chellapa, 2016], SPDNet [Zhiwu and Van Gool, 2017], HBRNN [Du *et al*., 2015], and Deep RNN [Shahroudy *et al*., 2016].

In [Li *et al*., 2019], a modification of LieNet is proposed, called LS-LieNet. It combines the convolutional layers of LieNet with Bi-LSTM (bidirectional long short-term memory) recurrent layers. The network was also employed in the task of skeleton-based action recognition to learn the Lie group representation of the skeleton data. LS-LieNet consists of: 1. a convolutional layer called RotMap; 2. a pooling layer called RotPooling; 3. a logarithm mapping layer called LogMap, which maps the Lie group to the Lie algebra; 4. a Bi-LSTM layer; 5. an FC layer; and 6. a softmax layer. The architecture is summarized in Figure 14.

As proposed in LieNet, the RotMap and RotPooling layers are specially designed to Lie group features. LogMap transforms the Lie group features into Lie algebra features. In the sequence, a conventional CNN classification process is performed, with dense layers and softmax. In parallel, the Lie algebra attributes are cascaded into the Bi-LSTM layer to learn more detailed temporal information, which is followed by dense layers and softmax. Finally, the final classification result is achieved by merging the two softmax outputs,

taking into account both the spatial information from the conventional CNN and the temporal dynamics captured by the Bi-LSTM. This dual-pathway approach ensures that the model benefits from the strengths of both convolutional and recurrent layers, providing a comprehensive understanding of the data. The experiments indicated that the proposed LS-LieNet reached superior performance compared to LieNet [Zhiwu *et al*., 2017] in terms of action recognition accuracy.

The approach presented in [Rhif *et al*., 2018] also uses LSTM layers. The proposed deep network, named Long-term Recurrent Convolutional Network on Lie groups (LRCNLG), whose architecture is shown in Figure 15, was applied to the problem of human action recognition from sequences of 3D skeleton data. Initially, the intrinsic nature of the data characterized by Lie group geometry is incorporated into the network to learn more suitable geometric features for 3D action recognition. Subsequently, the model properties follow those of classical CNNs and LSTMs, trained with SGD.

As shown in the figure, the architecture of LRCNLG consists of stacked layers of one-dimensional convolution across the entire temporal domain. Subsequently, the feature maps generated by the 1D CNN are used to train an LSTM model to recognize action categories. After this, the outputs from the LSTM layers are concatenated, and a dropout layer is introduced to mitigate overfitting issues. Finally, an FC layer with a softmax function is used for class label prediction. The results obtained from experiments conducted on three human action datasets consistently demonstrated the effectiveness of the proposed approach.

6.1 Comparison between the works

Table 4 presents a summary of the works described in this section. These methods share the following similarities. All of them apply deep learning techniques to the task of action recognition from 3D skeletons and use the SGD optimizer. Additionally, the Lie group plays a crucial role in representing the input skeleton data. In the case of LieNet and LS-LieNet, data is processed in the form of Lie group representations throughout the deep network, while in LRCNLG, the Lie group is used to represent the input data.

On other hand, they differ in how they approach skeleton data representation and handle temporal information loss, with LieNet, LS-LieNet, and LRCNLG providing unique perspectives on the problem. In LieNet, the authors integrated the advantages of convolutional layers to represent the data. However, there was still a loss of temporal information related to skeleton data. In response to this issue, the authors of LS-LieNet enhanced LieNet by adding LSTM recurrent layers to mitigate these temporal errors. Finally, within the context of LRCNLG, the authors developed a neural network model that goes further into learning geometric characteristics captured by Lie group representations of skeleton data using 1D convolutional layers and LSTM layers.

7 A Brief Computational Performance Analysis

The computational performance of the methods varies significantly due to differences in the mathematical operations required by each manifold or group. Below, we provide a detailed analysis of why certain methods are faster than others, considering computational complexity and the nature of the data processing involved.

Fastest Methods: Grassmann Manifold-Based Methods
These methods are generally the fastest among the manifold methods discussed in this paper. This is primarily due to the efficient computational properties of operations on Grassmann manifolds. These methods involve computations on subspaces that can be represented by orthonormal matrices of relatively low dimensions. The key operations include matrix multiplications and QR decompositions, which are computationally less intensive compared to operations like eigenvalue decompositions required in other manifolds.

Moreover, the use of subspaces allows for dimensionality reduction, leading to a significant decrease in the size of data being processed. This reduction not only minimizes memory usage but also reduces the computational cost of subsequent operations. Additionally, Grassmann manifold methods inherently capture geometric invariances, such as rotations and scaling, which means they can process data without the need for additional computational resources to handle these transformations. By exploiting these geometric properties, Grassmann methods achieve efficient computations that are often faster than traditional Euclidean methods, which may require complex preprocessing or data augmentation to handle invariances.

Intermediate Speed: Lie Group-Based Methods
These methods exhibit moderate computational costs. They work with data represented on Lie groups, which often involve rotations and translations in three-dimensional space. The operations required include matrix exponential and logarithms, which are more computationally demanding than basic linear algebra operations but are still manageable, especially when the matrices involved are small (e.g., 3×3 rotation matrices).

The data processed by Lie group methods, such as human skeleton joints, are typically of lower dimensionality, which helps to mitigate computational demands. Moreover, Lie groups naturally model continuous transformations and symmetries, allowing these methods to inherently handle geometric invariances without additional computational overhead. This reduces the need for complex networks or extensive data preprocessing, leading to computational efficiency that is superior to some Euclidean methods.

Slowest Methods: Riemannian Manifold-Based Methods
These methods, particularly those utilizing SPD matrices, tend to be the most computationally intensive. This is due to the complex mathematical operations required, such as eigenvalue decompositions, matrix logarithms, and exponential, which have high computational complexity (for instance,

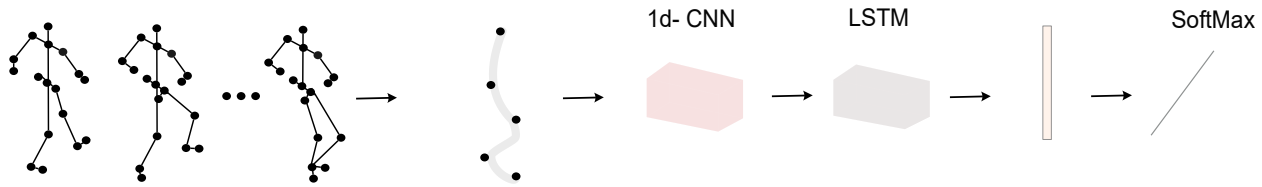

Figure 15. Architecture of the LRCNLG network. Source: [Rhif *et al*., 2018].

Table 4. Summary of architectures using Lie group.

Proposed Method	Network Type	Advantages	Disadvantages
LieNet	Deep network with convolutional layers and pooling layers (RotMap and RotPooling)	Handles key issues related to speed variations and high dimensionality of Lie group features	Loss of some temporal information from skeleton sequences
LS-LieNet	Extension of LieNet with Bi-LSTM layers	LSTM layers handle temporal domain information, while convolutional layers handle spatial information	Implementing Lie groups in the network can be complex and requires advanced knowledge in mathematics and linear algebra
LRCNLG	Lie group representing skeleton keypoints as input to network with 1D convolutional layers and LSTM layers	Effective modeling of human action sequences. Robust to variations in action duration, speed, and order	Training models using Lie group representations may be more complex than training on other representations

$O(n^3)$ for eigenvalue decomposition of an $n \times n$ matrix). Additionally, SPD matrices are often high-dimensional, which further increases computational demands.

These methods need to maintain the geometric structure of the manifold through the network layers, requiring specialized operations and careful numerical implementations. While they capture the intrinsic geometry of the data effectively, the computational overhead associated with these operations makes Riemannian-based methods slower compared to Grassmann and Lie group methods.

Comparison with Euclidean Methods

Manifold-based methods, despite operating on complex mathematical structures, can be faster than traditional Euclidean ones. Euclidean methods often require deeper network architectures, extensive data augmentation, or complex preprocessing steps to handle geometric invariances such as rotations, scaling, and translations. These additional steps increase computational complexity and processing time.

In contrast, manifold methods inherently encode these invariances within their mathematical framework, allowing them to achieve similar or better performance with fewer parameters and shallower networks. By leveraging the geometric properties and subspace representations, manifold methods process data more efficiently, reducing computational costs. Therefore, manifold-based methods not only offer better representation of the data but also improve computational efficiency over Euclidean methods.

Practical Considerations

In practice, the computational performance of Euclidean methods depends on factors such as data dimensionality, network architecture, optimization algorithms, and implementation efficiency. High-dimensional data or complex network structures can increase computational demands. However, efficient numerical libraries, parallel computing and approximation algorithms, etc can mitigate these costs.

Methods that minimize the need for computationally intensive operations, such as the Grassmann manifold-based, are generally faster and more scalable. By exploiting geometric invariances and working with lower-dimensional representations, these methods reduce the overall computational complexity, making them suitable for real-time applications and large-scale data processing. A comparison of the accuracy attained by deep networks in five benchmark datasets is shown in Table 5. These datasets are widely used in tasks involving human action, emotion, and gesture recognition in videos, each with specific characteristics that make them relevant to different methodological approaches.

Table 5. Comparison of deep network accuracy.

Method	AFEW	HDM05	CG	UCF-sub	UAV-Human
SPDNet	34.23%	61.45%	89.03%	59.93%	42.31%
MSNet	-	-	**91.25%**	60.87%	-
DreamNet	37.47%	-	-	-	**46.28%**
SPDNetRP	35.85%	-	39.49%	**89.17%**	-
U-SPDNet	21,41%	-	-	-	43.39%
GRNet	**59.23%**	**80.52%**	85.69%	35.80%	35.23%
G-LMSM+FC	38.27%	-	-	-	-
LieNet	-	75.78%	-	-	-

- *AFEW (Acted Facial Expressions in the Wild): is a dataset designed for emotion recognition. It contains*

1,345 video clips of facial expressions extracted from films, categorized into seven classes of emotions. Its collection in real-world-like environments facilitates the evaluation of models under challenging scenarios.

- *HDM05 (Human Motion Database 2005): it provides high-quality skeletal data for modeling 3D human motion, encompassing a wide range of actions and actors, making it particularly relevant for applications in computer vision and deep learning.*
- *CG (Cambridge-Gesture): is a dataset composed of 900 videos featuring five types of gestures performed by different individuals under varying lighting conditions, poses, and camera movements. Due to its controlled yet challenging structure, the CG dataset is ideal for studies involving model generalization and robustness.*
- *UCF-101: is one of the most widely used benchmarks for action recognition in uncontrolled videos, containing 13,320 video clips across 101 categories collected from YouTube, which introduces high variability and complexity to the task.*
- *UAV-Human: a dataset designed for human action recognition and person tracking in videos captured by drones. It provides a relevant scenario for applications in surveillance, security, and aerial monitoring.*

It is important to mention that Table 5 presents the accuracy values for the networks discussed in previous sections, as originally reported in the works of their respective authors. Considering these results, we present below a brief performance analysis of the networks on the AFEW dataset, which is the dataset used in almost all the previously described works, with the exception of the Lienet and MSNet networks. This analysis is based on the original works and experiments performed with U-SPDNet in this work.

The authors in Zhiwu and Van Gool [2017] explored SPDNet's performance with various configurations, providing several key observations. First, they observed a drop in accuracy without the LogEig layer, highlighting the importance of Riemannian computation. They also found that the original SPDNet outperformed a model that learned directly over log-Euclidean forms, demonstrating the relevance of SPD subspace-based layers. Finally, their results showed that improvements are not solely due to network depth but also due to the specific contributions of BiMap and ReEig layers.

In contrast, DreamNet [Wang *et al*., 2023a] exhibited improved performance with increased depth. The 47-layer and 92-layer models both surpassed the 27-layer version, with the 92-layer version achieving the lowest test errors, reinforcing the effectiveness of deeper networks. The authors also noted that these models are easier to train and converge faster than SPDNet, a benefit they attributed to the reconstruction error term (RT).

For the SPDNetRP [Wang *et al*., 2022], experiments demonstrated that adding RBR layers affects performance but not in a direct proportion to the number of modules. This is mainly due to the complexity of the AFEW dataset, which requires high-level semantic features for better discrimination. In terms of U-SPDNet [Wang *et al*., 2023b], we implemented and observed that learning rate strongly influences stability and convergence. Lower learning rates values allowed more gradual parameter updates, better adhering to the manifold geometry and improving performance. These findings highlight the importance of careful hyperparameter tuning for neural networks that operate in non-Euclidean spaces.

Finally, GrNet [Zhiwu *et al*., 2018] reached superior performance compared to traditional Grassmann learning methods and standard ConvNets. It also demonstrated advantages in speed and in requiring fewer training epochs. However, recent studies highlighted that both SPDNet and GrNet face scalability limitations due to complex operations, whereas lighter approaches such as GLSM+FC can achieve competitive results with lower computational cost.

Limitations and Challenges of the Study

This study acknowledges important limitations, such as the restricted scope of comparisons due to the unavailability of certain architectures (e.g., MSNet). We also point out limitations of the methods that combine subspace representation techniques with deep neural network. For instance, the sensitivity of models like SPDNet to hyperparameter selection in non-Euclidean spaces, and the increased computational cost and complexity associated with deeper architectures like SPDNet and GrNet. Future directions should include developing lightweight and efficient architectures that preserve effectiveness while reducing computational demands; refining optimization methods for non-Euclidean spaces to improve stability; expanding open-source availability for fairer comparisons; and incorporating self-supervised or transfer learning to address small or imbalanced datasets. Additionally, the creation of diverse and realistic datasets is essential for training more robust and generalizable models, particularly in complex scenarios such as aerial action recognition and spontaneous facial expression analysis.

8 Conclusion

This study provides a structured and comprehensive review of recent advances in the integration of subspace representations within deep neural networks. By systematically categorizing existing approaches based on their underlying mathematical structures—such as Riemannian manifolds, Grassmann manifolds, and Lie groups—this work provides a novel taxonomy that clarifies the relationships among diverse methodologies. This classification facilitates the identification of categories of methods in the literature and serves as a foundation for the systematic development of new models. Through critical comparison and analysis, this survey highlights the performance, strengths, and limitations of subspace-based architectures, offering insights into their design, effectiveness, and potential applications.

From a theoretical standpoint, the proposed taxonomy and analysis contribute to a deeper understanding of how subspace learning principles can be embedded into deep learning frameworks, especially in non-Euclidean spaces. This encourages the development of more interpretable and mathematically grounded models. From a practical standpoint, subspace-based approaches exhibit significant advantages, particularly in terms of robustness to noise and data redundancy, as well

as the ability to perform effective dimensionality reduction. These characteristics make them particularly suitable for real-world applications involving high-dimensional data, including recognition, classification, and segmentation tasks in complex environments.

The use of subspace representations enables neural network models to compress large volumes of information into lower-dimensional forms without compromising performance. This results in increased computational efficiency, which is crucial for deployment on resource-constrained devices. Furthermore, the inherent robustness of these models to noise makes them highly suitable for applications in domains with imprecise, corrupted, or incomplete data, such as aerial surveillance or facial expression recognition.

However, this study also presents some limitations. The most important is the absence of direct comparisons, reducing the comprehensiveness of the benchmarking analysis. This is due to the unavailability of certain architectures, such as MSNet. In terms of the investigated architectures, the sensitivity of models like SPDNet to hyperparameter selection, especially the learning rate, poses a challenge in non-Euclidean spaces, where inappropriate configurations can compromise convergence and training stability. Moreover, although increasing network depth tends to improve performance, it also leads to higher computational costs and algorithmic complexity, limiting scalability and posing challenges for efficient GPU implementation.

In this work we could identify multiple possibilities for future research. One promising direction involves the development of lightweight and computationally efficient architectures that retain the expressive power of models such as SPDNet and GrNet, possibly through approximations that reduce reliance on expensive matrix decompositions in Riemannian manifolds. Another important challenge is the refinement of optimization techniques specifically designed for non-Euclidean spaces, which would allow for stable training even with higher learning rates and reduce dependence on extensive hyperparameter tuning. Enhancing reproducibility and fairness in comparative studies also requires the release of public implementations for all relevant architectures. Additionally, integrating self-supervised and transfer learning techniques could help mitigate issues arising from limited or imbalanced datasets. Lastly, the design of new benchmark datasets that combine realism, diversity, and high-quality annotations would be fundamental in evaluating and advancing the generalization capacity of subspace-based deep models, particularly in underexplored areas such as UAV-based action recognition and spontaneous facial expression analysis.

An especially promising avenue is the exploration of dynamic subspace selection techniques, where the subspace basis adapts during training to better capture evolving patterns in the data. Additionally, deeper integration of manifold learning principles within neural network architectures could enhance geometric consistency and model interpretability. Finally, the combination of subspace methods with modern attention mechanisms and transformer-based architectures has the potential to significantly boost performance in sequence modeling and cross-modal tasks.

Authors' Contributions

SRG contributed to the conception of the study, including investigation, data analysis, and discussion of the results, and is the lead author of this research. BBG and EMS supervised the planning and execution of the study and contributed with relevant revisions and discussions. All authors read and approved the final version of the manuscript.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

It does not apply to the content of this paper.

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES-PROEX) - Finance Code 001. This work was partially supported by Amazonas State Research Support Foundation - FAPEAM - through the POSGRAD project 2024/2025.

References

Belkin, M. and Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396. DOI: 10.1162/089976603321780317.

Bloom, V., Makris, D., and Argyriou, V. (2012). G3d: A gaming action dataset and real time action recognition evaluation framework. In *2012 IEEE Computer society conference on computer vision and pattern recognition workshops*, pages 7–12. IEEE. DOI: 10.1109/cvprw.2012.6239175.

Bonnabel, S. (2013a). Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229. DOI: 10.1109/tac.2013.2254619.

Bonnabel, S. (2013b). Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229. DOI: 10.1109/tac.2013.2254619.

Bottou, L. (2010). Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, pages 177–186. Springer. DOI: 10.1201/b11429-4.

Cetingul, H. E. and Vidal, R. (2009). Intrinsic mean shift for clustering on stiefel and grassmann manifolds. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1896–1902. IEEE. DOI: 10.1109/cvpr.2009.5206806.

Chen, Z., Wu, X., Xu, T., Wang, R., huang2017deep2017, Z., and Kittler, J. (2022). Msnet: A deep multi-scale submanifold network for visual classification. *CoRR*,

abs/2201.10145. Available at:https://arxiv.org/abs/2201.10145.

Du, Y., Wang, W., and Wang, L. (2015). Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1110–1118. DOI: 10.1109/cvpr.2015.7298714.

Edelman, A., Arias, T. A., and Smith, S. T. (1998). The geometry of algorithms with orthogonality constraints. *SIAM journal on Matrix Analysis and Applications*, 20(2):303–353. DOI: 10.1137/s0895479895290954.

Fei, Y., Wei, X., Liu, Y., Li, Z., and Chen, M. (2023). A survey of geometric optimization for deep learning: From euclidean space to riemannian manifold. *arXiv preprint arXiv:2302.08210*. DOI: 10.1145/3708498.

Gatto, B. B., dos Santos, E. M., Koerich, A. L., Fukui, K., and Junior, W. S. (2021). Tensor analysis with n-mode generalized difference subspace. *Expert Systems with Applications*, 171:114559. DOI: 10.1016/j.eswa.2020.114559.

Haitman, Y., Francos, J. M., and Scharf, L. L. (2021a). Grassmannian dimensionality reduction for optimized universal manifold embedding representation of 3d point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4213–4221. DOI: 10.1109/iccvw54120.2021.00468.

Haitman, Y., Francos, J. M., and Scharf, L. L. (2021b). Grassmannian dimensionality reduction for optimized universal manifold embedding representation of 3d point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4213–4221. DOI: 10.1109/iccvw54120.2021.00468.

Hamm, J. and Lee, D. (2008a). Extended grassmann kernels for subspace-based learning. *Advances in neural information processing systems*, 21. Available at:https://proceedings.neurips.cc/paper_files/paper/2008/file/e7f8a7fb0b77bcb3b283af5be021448f-Paper.pdf.

Hamm, J. and Lee, D. D. (2008b). Grassmann discriminant analysis: A unifying view on subspace-based learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 376–383, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/1390156.1390204.

Harandi, M. T., Sanderson, C., Shirazi, S., and Lovell, B. C. (2011). Graph embedding discriminant analysis on grassmannian manifolds for improved image set matching. In *CVPR 2011*, pages 2705–2712. IEEE. DOI: 10.1109/cvpr.2011.5995564.

Hong, J., Li, Y., and Chen, H. (2019). Variant grassmann manifolds: A representation augmentation method for action recognition. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(2):1–23. DOI: 10.1145/3314203.

Huang, Z., Wang, R., Shan, S., and Chen, X. (2015). Projection metric learning on grassmann manifold with application to video based face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 140–149. DOI: 10.1109/cvpr.2015.7298609.

Ionescu, C., Vantzos, O., and Sminchisescu, C. (2015). Matrix backpropagation for deep networks with structured layers. In *Proceedings of the IEEE international conference on computer vision*, pages 2965–2973. DOI: 10.1109/iccv.2015.339.

Jiang, X., Hu, Z., Wang, S., and Zhang, Y. (2023). Deep learning for medical image-based cancer diagnosis. *Cancers*, 15(14):3608. DOI: 10.3390/cancers15143608.

Johnson, M. and Savakis, A. (2015). L1-grassmann manifolds for robust face recognition. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 482–486. IEEE. DOI: 10.1109/icip.2015.7350845.

Katayama, T. (2005). *Subspace methods for system identification*. Springer, 1 edition. DOI: 10.1007/1-84628-158-x.

Ke, Z., Cui, Z.-X., Huang, W., Cheng, J., Jia, S., Ying, L., Zhu, Y., and Liang, D. (2021). Deep manifold learning for dynamic mr imaging. *IEEE Transactions on Computational Imaging*, 7:1314–1327. DOI: 10.1109/tci.2021.3131564.

Kuutti, S., Bowden, R., Jin, Y., Barber, P., and Fallah, S. (2020). A survey of deep learning applications to autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 22(2):712–733. DOI: 10.1109/tits.2019.2962338.

Li, Y., Guo, T., Liu, X., and Xia, R. (2019). Skeleton-based action recognition with lie group and deep neural networks. In *2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP)*, pages 26–30. IEEE. DOI: 10.1109/siprocess.2019.8868548.

Liu, S., Zhao, C., An, Y., Li, P., Zhao, J., and Zhang, Y. (2019). Diffusion tensor imaging denoising based on riemannian geometric framework and sparse bayesian learning. *Journal of Medical Imaging and Health Informatics*, 9(9):1993–2003. DOI: 10.1166/jmihi.2019.2832.

Liu, T., Shi, Z., Liu, Y., and Li, C. (2018). Dimensionality reduction on grassmannian: A good practice. In *2018 Eighth International Conference on Instrumentation & Measurement, Computer, Communication and Control (IMCCC)*, pages 943–948. IEEE. DOI: 10.1109/imccc.2018.00199.

Lu, W., Fan, F., Chu, J., Jing, P., and Yuting, S. (2018). Wearable computing for internet of things: A discriminant approach for human activity recognition. *IEEE Internet of Things Journal*, 6(2):2749–2759. DOI: 10.1109/jiot.2018.2873594.

Müller, M., Röder, T., Clausen, M., Eberhardt, B., Krüger, B., and Weber, A. (2007). Mocap database hdm05. *Institut für Informatik II, Universität Bonn*, 2(7). Available at:https://resources.mpi-inf.mpg.de/HDM05/.

Nie, S. and Ji, Q. (2014). Capturing global and local dynamics for human action recognition. In *2014 22nd International Conference on Pattern Recognition*, pages 1946–1951. IEEE. DOI: 10.1109/icpr.2014.340.

Oja, E. and Kut, M. (1983). The alsm algorithm - an improved subspace method of classification. Pattern Recoflnition, 16 edition. DOI: 10.1016/0031-3203(83)90064-x.

Pierson, H. A. and Gashler, M. S. (2017). Deep learning in robotics: a review of recent research. *Advanced Robotics*, 31(16):821–835. DOI: 10.1080/01691864.2017.1365009.

Rhif, M., Wannous, H., and Farah, I. R. (2018). Action recognition from 3d skeleton sequences using deep networks on lie group features. In *2018 24th International Conference*

on Pattern Recognition (ICPR), pages 3427–3432. IEEE. DOI: 10.1109/icpr.2018.8546027.

Roweis, S. T. and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326. DOI: 10.1126/science.290.5500.2323.

Shahroudy, A., Liu, J., Ng, T.-T., and Wang, G. (2016). Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1010–1019. DOI: 10.1109/cvpr.2016.115.

Souza, L. S., Sogi, N., Gatto, B. B., Kobayashi, T., and Fukui, K. (2023). Grassmannian learning mutual subspace method for image set recognition. *Neurocomputing*, 517:20–33. DOI: 10.1016/j.neucom.2022.10.040.

Sun, Y., Chen, Y., Wang, X., and Tang, X. (2014). Deep learning face representation by joint identification-verification. *Advances in neural information processing systems*, 27. DOI: 10.48550/arxiv.1406.4773.

Sun, Z., Hu, Z.-P., Chiong, R., Wang, M., and He, W. (2018). Combining the kernel collaboration representation and deep subspace learning for facial expression recognition. *Journal of circuits, systems and computers*, 27(08):1850121. DOI: 10.1142/s0218126618501219.

Tenenbaum, J. B., Silva, V. d., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323. DOI: 10.1126/science.290.5500.2319.

Tian, L., Wang, Z., He, B., He, C., Wang, D., and Li, D. (2021). Knowledge distillation of grassmann manifold network for remote sensing scene classification. *Remote Sensing*, 13(22):4537. DOI: 10.3390/rs13224537.

Turan, C., Zhao, R., Lam, K.-M., and He, X. (2021). Subspace learning for facial expression recognition: an overview and a new perspective. *APSIPA Transactions on Signal and Information Processing*, 10:e1. DOI: 10.1017/atsip.2020.27.

Vemulapalli, R., Arrate, F., and Chellappa, R. (2014). Human action recognition by representing 3d skeletons as points in a lie group. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 588–595. DOI: 10.1109/cvpr.2014.82.

Vemulapalli, R. and Chellapa, R. (2016). Rolling rotations for recognizing human actions from 3d skeletal data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4471–4479. DOI: 10.1109/cvpr.2016.484.

Wang, Q., Gao, J., and Li, H. (2017). Grassmannian manifold optimization assisted sparse spectral clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5258–5266. DOI: 10.1109/cvpr.2017.335.

Wang, R., Wu, X.-J., Chen, Z., Xu, T., and Kittler, J. (2022). Learning a discriminative spd manifold neural network for image set classification. *Neural Networks*, 151:94–110. DOI: 10.1016/j.neunet.2022.03.012.

Wang, R., Wu, X.-J., Chen, Z., Xu, T., and Kittler, J. (2023a). Dreamnet: A deep riemannian manifold network for spd matrix learning. In *Computer Vision – ACCV 2022: 16th Asian Conference on Computer Vision, Macao, China, December 4–8, 2022, Proceedings, Part VI*, page 646–663, Berlin, Heidelberg. Springer-Verlag. DOI: 10.1007/978-3-031-26351-4$_3$9.

Wang, R., Wu, X.-J., Xu, T., Hu, C., and Kittler, J. (2023b). U-spdnet: An spd manifold learning-based neural network for visual classification. *Neural networks*, 161:382–396. DOI: 10.1016/j.neunet.2022.11.030.

Xu, C., Govindarajan, L. N., Zhang, Y., and Cheng, L. (2017). Lie-x: Depth image based articulated object pose estimation, tracking, and action recognition on lie groups. *International Journal of Computer Vision*, 123:454–478. DOI: 10.1007/s11263-017-0998-6.

Yang, M. and Li, F. (2017). Lie group impression for deep learning. In *2017 International Smart Cities Conference (ISC2)*, pages 1–5. DOI: 10.1109/ISC2.2017.8090853.

Zhiwu and Van Gool, L. (2017). A riemannian network for spd matrix learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31. DOI: 10.1609/aaai.v31i1.10866.

Zhiwu, Wan, C., Probst, T., and Van Gool, L. (2017). Deep learning on lie groups for skeleton-based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6099–6108. DOI: 10.1109/cvpr.2017.137.

Zhiwu, Wu, J., and Van Gool, L. (2018). Building deep networks on grassmann manifolds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32. DOI: 10.1609/aaai.v32i1.11725.