Set Classification with Probabilistic Learning Vector Quantization on the Grassmann Manifold

1st Mohammad Mohammadi
Dept. Intelligent Systems
Research Center for Cognitive Science & AI
Tilburg University
Tilburg, The Netherlands
mohammadimathstar@gmail.com

2nd Sreejita Ghosh
ICAI Applied AI Accelerator lab
Dept. of Orthopaedics
UMC Groningen
Groningen, The Netherlands
ghosh.sreejita@gmail.com

3nd Çiçek Güven
Dept. Intelligent Systems
Research Center for Cognitive Science & AI
Tilburg University
Tilburg, The Netherlands
C.Guven@tilburguniversity.edu

***Abstract*—Prototype-Based Networks (PBNs) provide intrinsic interpretability in image classification by learning characteristic traits of objects as evidence to support predictions. While most existing PBNs operate in Euclidean embedding spaces, recent work shows that representing images as linear subspaces provides robust and interpretable models. However, current subspace-based prototype methods are deterministic and do not explicitly account for (i) the manifold's (in this case Grassmann) geometry during optimization; (ii) the uncertainty in the model's decisions.**

We propose a probabilistic Learning Vector Quantization (LVQ) classifier, defined directly on the Grassmann manifold. Our approach models predictive uncertainty, which is important for human-centric applications such as in healthcare and justice, and employs geometry-aware optimization that respects the manifold's intrinsic structure. We also use gradient-based explanations to show which regions of the image drive the classifier's decisions. Experiments on three benchmark datasets demonstrate competitive performance compared to previous PBNs. These also show that the classifier aggregates evidence from different image regions to make its final prediction and identifies potential sources of uncertainties in the image.

***Index Terms*—Grassmann manifold, LVQ, Prototype-Based Networks, Prototypical-Part learning**

I. Introduction

Due to the rapid increase in availability of good quality digital data, the application of data-driven decision systems have become popular in many fields. Although modern machine learning (ML) models often achieve high predictive accuracy, their lack of transparency limits their applications in sensitive human-centric domains where decisions must be understandable, and trustworthy. This has led to increasing emphasis on Explainable Artificial Intelligence (XAI), both through intrinsically interpretable models [1], [2], and through model-agnostic techniques designed to explain the behavior of complex black-box systems. Together, they aim to ensure that models' decisions are not only accurate, but also understandable to human experts.

An important class of intrinsically interpretable models is Prototype-Based Networks (PBNs), where decisions are derived through a "this looks like that" reasoning process. These models embed input data into a latent space using a neural backbone, compare the resulting representations to a set of learned reference patterns, called prototypes, and produce predictions based on their presence. ProtoPNet [3] established this paradigm by associating prototypes with localized, human-interpretable patterns. Subsequent works have extended this idea in several directions: ProtoTree [4] structures decisions hierarchically using a binary tree; Deformable ProtoPNet [5] introduces spatially flexible prototypes to handle pose variations; and ProtoPShare [6] and ProtoPool [7] enable prototype sharing across classes to reduce redundancy and improve generalization. More recent approaches, such as PBN-CBC [8], connect prototype learning to radial basis function networks, guarantying robustness and principled negative reasoning. ProtoArgNet [9] on the other hand models interactions between parts using super-prototypes and argumentation theory to generate both supporting and opposing explanations.

While the aforementioned models operate in Euclidean embedding spaces, several approaches [10], [11] represent an image as a set of local CNN descriptors and model this set as a linear subspace. This representation provides a compact and noise-robust description that effectively captures variations in pose and illumination [12]. Mathematically, such subspaces reside on the Grassmann manifold, a non-Euclidean space with rich geometric structure. AChorDS-Learning Vector Quantization (LVQ) [11], operating on this manifold, has demonstrated good performance across several benchmark datasets, highlighting the effectiveness of subspace-based representations. However, existing methods with manifold-based formulations still rely on optimization schemes performed in Euclidean space, and do not explicitly account for the underlying Riemannian geometry.

Building upon these subspace-based approaches, we introduce a probabilistic LVQ classifier defined directly on the Grassmann manifold (ProGrass-LVQ). Similar to AChorDS-LVQ, our method learns prototypical parts to retain the interpretability and "this looks like that" reasoning. However, unlike the former's deterministic formulation, ProGrass LVQ uncertainties in its decision making. Simultaneously it respects the geometry of the Grassmann manifold by performing optimization directly on the manifold. In addition, we derive a gradient-based explainability mechanism that enables systematic mapping between features and predictions. Together, these contributions extend prototype-based interpretability to

probabilistic, geometry-aware subspace learning.

This paper is organized as follows: Sec. II provides background on the Grassmann manifold and formalizes the problem setting. Sec. III presents the proposed method, its probabilistic formulation, geometry-aware optimization, and its gradient-based explainability mechanism. Sec. IV reports experimental results on three benchmark datasets, comparing our approach with existing PBNs. Finally, Sec. VI concludes the paper.

II. Background

A. Grassmann Manifold

A manifold $\mathcal{M}$ is a topological space that can be locally approximated as a Euclidean space. If each tangent space is equipped with an inner product, $\mathcal{M}$ is called a Riemannian manifold, where geometric quantities such as distances, and curve lengths can be defined on it. As we use linear subspaces in this work, here we focus on the Grassmann manifold which is the mathematical framework to study them.

The Grassmann manifold $\mathcal{G}(\mathsf{d}, D)$ is the set of all d-dimensional linear subspaces of $\mathbb{R}^D$ [13]. A standard way for representing subspaces is orthonormal matrices $\mathbf{X} \in \mathbb{R}^{D\times\mathsf{d}}$, where the columns of matrices span the subspaces. Note that this representation is not unique: multiplying $\mathbf{X}$ by any orthogonal matrix $\mathbf{Q} \in \mathbb{O}(\mathsf{d})$ (i.e. $\mathbf{XQ}$) gives another basis for the same subspace. In this manifold, the tangent space of a point $\mathbf{X} \in \mathcal{G}(\mathsf{d}, D)$ is given by

$$T_{\mathbf{X}}\mathcal{G}(\mathsf{d}, D) = \{\mathbf{\Delta} \in \mathbb{R}^{D\times\mathsf{d}} : \mathbf{X}^\top\mathbf{\Delta} = \mathbf{0}\}. \tag{1}$$

A key geometric concept is geodesics, which are curves of minimal length connecting points on the manifold. These curves on the Grassmann manifold can be written in a closed form. Given a point $\mathbf{X} \in \mathcal{G}(\mathsf{d}, D)$ and a tangent vector $\mathbf{\Delta} \in T_{\mathbf{X}}\mathcal{G}(\mathsf{d}, D)$, the geodesic curve from $\mathbf{X}$ in the direction $\mathbf{\Delta}$ is obtained via the exponential map in a parametric format:

$$\gamma(t) = \mathrm{Exp}_{\mathbf{X}}(t\mathbf{\Delta}) = \mathbf{XV}\cos(\mathbf{\Sigma}t)\mathbf{V}^\top + \mathbf{U}\sin(\mathbf{\Sigma}t)\mathbf{V}^\top, \tag{2}$$

where $t \in [0, 1]$ is a scalar parameter, and $\mathbf{\Delta} = \mathbf{U\Sigma V}^\top$ is the compact signular value decomposition (SVD) of $\mathbf{\Delta}$. This expression describes the evolution of the subspace along the geodesic on the Grassmann manifold. To compute the length of geodesic curve between any two points $\mathbf{X}_1, \mathbf{X}_2 \in \mathcal{G}(\mathsf{d}, D)$, we use principal angles $\{\theta_i\}_{i=1}^{\mathsf{d}}$:

$$\delta(\mathbf{X}_1, \mathbf{X}_2) = \Big(\sum_{i=1}^{\mathsf{d}} \theta_i^2\Big)^{\frac{1}{2}} \tag{3}$$

where $\delta(.,.)$ measures the shortest path between two subspaces, and the principal angles are computed by SVD of:

$$\mathbf{X}_1^\top\mathbf{X}_2 = \mathbf{Q}_1\cos(\mathbf{\Theta})\mathbf{Q}_2^\top \tag{4}$$

where $\mathbf{Q}_1, \mathbf{Q}_2 \in \mathbb{O}(\mathsf{d})$ are rotation matrices, $\cos(\mathbf{\Theta}) = \mathrm{diag}\big(\cos(\theta_1), \ldots, \cos(\theta_{\mathsf{d}})\big)$ referred to as the *canonical correlation*, and the columns of $\mathbf{U} = \mathbf{X}_1\mathbf{Q}_1$ and $\mathbf{V} = \mathbf{X}_2\mathbf{Q}_2$ are known as *principal (or canonical) vectors*.

This foundation on the Grassmann manifold [13] provides the tools needed for developing optimization methods that respect the intrinsic structure of the manifold. Here, capital letters (e.g. X) represent matrices (containing vectors), and bold capital letters (e.g. $\mathbf{X}$) denote orthogonal matrices (representing subspaces).

B. Problem setting

Having introduced the Grassmann manifold as the space of linear subspaces, we now describe how data samples are mapped to such subspaces in our classification task. In many applications, each data point sample can be represented by a collection of feature vectors, instead of a single vector. In this paper, such sets arise from a convolutional neural network (CNN) where the final convolutional layer produces a feature map of size $W \times H \times D$. We reshape this into $n = WH$ vectors $x_i \in \mathbb{R}^D$, each describing a region of the image.

We arrange each set into a matrix $X = [x_1, \cdots, x_n] \in \mathbb{R}^{D\times n}$. Instead of working directly with this matrix, we use a compact subspace representation via the truncated SVD,

$$X = \mathbf{X}SR^T \ , \tag{5}$$

where the columns of $\mathbf{X} \in \mathbb{R}^{D\times\mathsf{d}}$ span the dominant d-dimensional subspace of the set. With this, each sample is mapped to a point on the Grassmann manifold $\mathcal{G}(\mathsf{d}, D)$, which captures its intrinsic geometric structure. This transforms our training set into $\{(\mathbf{X}_i, y_i)\}_{i=1}^N$ with all samples lying on the Grassmann manifold.

III. Method

We begin with the model formulation, including the similarity function, probabilistic interpretation, and learning objective. Next, we describe how model parameters are optimized on the Grassmann manifold. Finally, we present the explainability mechanism, which quantifies the influence of input features on the ProGrass-LVQ classifier's decisions.

A. Model Formulation

Let $\{(\mathbf{X}_i, y_i)\}_{i=1}^N$ be the training set defined on the Grassmann manifold $\mathcal{G}(\mathsf{d}, D)$. To model the distribution of classes in the data space $\mathcal{G}(\mathsf{d}, D)$, we use a set of labeled prototypes $\{(\mathbf{W}_i, c_i)\}_{i=1}^p$ defined in the data space, similar to prototypes in other LVQ classifiers. This means, each prototype $\mathbf{W}_i$ is an $(D \times \mathsf{d})$ orthonormal matrix. During training, prototypes are adapted to optimally represent the structure of their corresponding class distributions.

LVQ classifiers are (dis)similarity-based approaches, where choice of a (dis)similarity measure plays the central role. For probabilistic LVQ algorithms, this measure helps to define the class posterior distribution $\hat{p}(c|.)$. Here, as the data space is $\mathcal{G}(\mathsf{d}, D)$, we need to employ a similarity function that capture the geometry of the manifold. To do that, we define a measure using principal angles which encode valuable information about the geometrical structure of the manifold. As we aim to

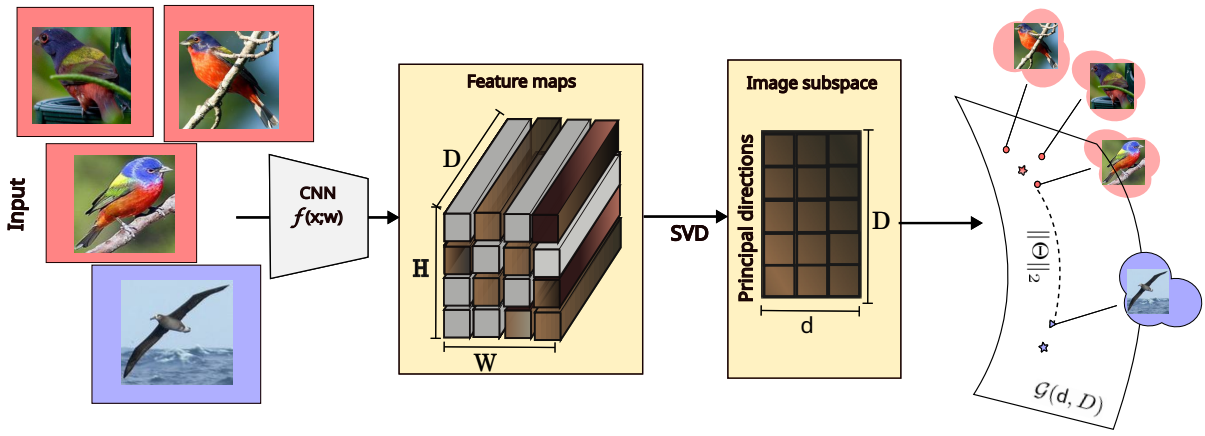

Fig. 1. A block-diagram of our model ProGrass-LVQ: a) a backbone CNN architecture takes images as inputs, b) the network generates a feature map, i.e. $W \times H$ local feature descriptor of size D, c) apply the SVD for subspace representation on the Grassmann manifold. In the right image, each circle denotes the image embedding on the Grassmann manifold, and each star represents a prototype. Note that the colors encode the labels.

define a probability function, we define similarities using the cosine of principal angles, also called *canonical correlation*:

$$s(\mathbf{X}, \mathbf{W}) = \sum_{k=1}^{\mathsf{d}} \lambda_k \cos\theta_k \quad \text{subject to} \quad \sum_{k=1}^{\mathsf{d}} \lambda_k = 1, \tag{6}$$

where $\{\lambda_k\}_{k=1}^{\mathsf{d}}$ are learnable relevance parameters capturing the importance of each principal direction in class discrimination. Thus, λ_k makes the similarity adaptive, and as shown in [12], they mitigate the impact of noisy dimensions, and as a result, reduce the sensitivity of the algorithm to changes in the hyper-parameter d.

Using this similarity measure, the class posterior probability is defined via a softmax transformation:

$$\hat{p}(c|\mathbf{X}) = \frac{\exp\left(\beta.s(\mathbf{X}, \mathbf{W}_c)\right)}{\sum_{i=1}^{C} \exp\left(\beta.s(\mathbf{X}, \mathbf{W}_i)\right)} \tag{7}$$

where β is a temperature parameter controlling the sharpness of the distribution, and $\mathbf{W}_c$ denotes the prototype associated with class c.

While classical LVQ models optimize a margin-based criterion, probabilistic LVQ models [14], [15] reformulate learning in terms of matching class posterior distributions. In this setting, classification is performed by estimating a conditional distribution over classes, and learning is achieved by minimizing a divergence between the predicted distribution ($\hat{p}(c|\mathbf{X})$) and the true class distribution ($p(c|\mathbf{X})$). We use the Kullback–Leibler (KL) divergence as the training objective where we aim to minimize the difference between $\hat{p}(c|\mathbf{X})$ and $p(c|\mathbf{X})$. However, since the KL divergence is not symmetric, this formulation gives two natural variants [14]. In the *exclusive* divergence, the predicted distribution is compared against the target distribution,

$$E_{\text{excl}}(\mathcal{W}) = \sum_{i=1}^{N} D_{\text{KL}}\big(\hat{p}(\cdot \mid \mathbf{X}_i) \,\|\, p(\cdot \mid \mathbf{X}_i)\big), \tag{8}$$

whereas the *inclusive* divergence reverses the roles of the two distributions,

$$E_{\text{incl}}(\mathcal{W}) = \sum_{i=1}^{N} D_{\text{KL}}\big(p(\cdot \mid \mathbf{X}_i) \| \hat{p}(\cdot \mid \mathbf{X}i)\big) \tag{9}$$

where $\mathcal{W} = \{\mathbf{W}_1, \ldots, \mathbf{W}_C\}$ is the set of all prototypes. In this work, we investigate both divergence variants and analyze their impact on the performance. As the label y_i of each sample $\mathbf{X}_i$ is known, the true class distribution $p(\cdot|\mathbf{X}_i)$ is represented as a one-hot encoded vector. To avoid numerical instabilities, zero entries are replaced by a small constant ϵ, followed by normalization.

B. Optimization on the Grassmann Manifold

To optimize the cost function defined in Eq. (8) and (9), we use Stochastic Gradient Descent (SGD), which jointly updates the prototype locations on the manifold and the relevance factors $\{\lambda_k\}$. For this, we need the gradient of the cost function. However, unless the geometry of manifold is accounted for, the Euclidean updates of the prototypes $\mathbf{W}_i$ might push the prototypes off of the manifold. Here, we adopt an optimization procedure ensuring updates remain valid subspaces.

For most manifolds embedded in Euclidean space, including the Grassmann manifold, the *Riemannian gradient* of a function is obtained by projecting the Euclidean gradient onto the tangent space at the current point [13], [16], [17]. Specifically, given the Euclidean gradient of the cost function with respect to a prototype $\mathbf{W}$, i.e. $\nabla_{\mathbf{W}} E$, the corresponding tangent vector in $T_{\mathbf{W}}\mathcal{G}(D, \mathsf{d})$ is computed via the projection [13]:

$$\nabla^{\mathcal{G}}_{\mathbf{W}} E = (\mathbf{I} - \mathbf{W}\mathbf{W}^\top)\, \nabla_{\mathbf{W}} E. \tag{10}$$

This projection ensures that $\nabla^{\mathcal{G}}_{\mathbf{W}} E$ lies in the tangent space, satisfying the orthonormality constraints of the Grassmann manifold (see Eq. (1)). Once the tangent vector is obtained,

the prototype can be updated along the manifold using the exponential map:

$$\mathbf{W}^{\text{new}} = \text{Exp}_{\mathbf{W}}(-\alpha \nabla^{\mathcal{G}}_{\mathbf{W}} E), \tag{11}$$

where $\alpha > 0$ is the learning rate and $\text{Exp}_{\mathbf{W}}(\cdot)$ maps a tangent vector at $\mathbf{W}$ back onto the manifold [16].

This procedure allows us to compute updates using standard Euclidean derivatives while respecting the manifold geometry. In summary, we first compute the Euclidean gradient of the objective function (see Appendix VII-A), then project it to the tangent space as in Eq. (10), and finally map the update back to the manifold using Eq. (11). By iterating this process, the prototypes converge to values such that the objective function is minimized.

C. Explainability and Feature Attribution

While several model-agnostic post-hoc interpretation techniques exist, they can be unreliable and local. For example, [18] demonstrates that such approaches can be intentionally manipulated by adversarial classifiers. In contrast, prototype-based models such as LVQ offer an *intrinsic* mechanism for interpretability, since their predictions are directly grounded in the comparison between data and prototypes. Similarly, our model has such a capability.

Given input data $X = [\mathsf{x}_1, \ldots, \mathsf{x}_n] \in \mathbb{R}^{D\times n}$, we extract its Grassmann representation via the truncated SVD, following Eq. (5), where $\mathbf{X} \in \mathbb{R}^{D\times \mathsf{d}}$ and $R \in \mathbb{R}^{n\times \mathsf{d}}$. The similarity between a data subspace $\mathbf{X}$ and prototype $\mathbf{W}$ is computed in matrix form as

$$s(\mathbf{X}, \mathbf{W}) = \text{tr}\big(\Lambda \mathbf{U}^\top \mathbf{V}\big), \tag{12}$$

where $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_{\mathsf{d}})$ contains the learned relevance factors, and $(\mathbf{U}, \mathbf{V})$ are the principal vectors associated with $(\mathbf{X}, \mathbf{W})$. The posterior probabilities $\hat{p}(c|\mathbf{X})$ are obtained by applying a soft-max function to these similarities. Because the denominator of the soft-max function is shared across classes and the exponential is strictly monotonic, the relative influence of inputs can be analyzed directly through $s(\mathbf{X}, \mathbf{W})$.

a) Relating adaptive similarity to the input: Let $\mathbf{Q}_1$ denote the orthogonal matrix from the SVD of $\mathbf{X}^\top\mathbf{W}$ that produces the principal vectors. Since $\mathbf{U} = \mathbf{X}\mathbf{Q}_1$, substituting in Eq. (5) gives $\mathbf{U} = \mathbf{X}\mathbf{Q}_1 = XRS^{-1}\mathbf{Q}_1$.

Next, from the (12) we obtain

$$s(\mathbf{X}, \mathbf{W}) = \text{tr}\big(\Lambda \mathbf{Q}_1^\top S^{-1} R^\top X^\top \mathbf{V}\big) = \text{tr}\big(X^\top \mathbf{V}\, \Lambda \mathbf{Q}_1^\top S^{-1} R^\top\big)$$

showing that $s(\mathbf{X}, \mathbf{W})$ is a *linear* function of the data X.

b) Gradient-based explanation: Gradients show how small changes in the input affect the model output, and therefore provide a sensitivity-based attribution method [19]. Differentiating the adaptive similarity with respect to X

$$\nabla_X s(\mathbf{X}, \mathbf{W}) = \mathbf{V}\, \Lambda \mathbf{Q}_1^\top S^{-1} R^\top \in \mathbb{R}^{D\times n}. \tag{13}$$

Elements of this matrix indicate how much the corresponding input influences the similarity score and, as a result, the final decision of ProGrass-LVQ.

Equivalence to contribution-based interpretation. The contribution-based explanation used in [12] decomposes the principal angles into additive terms aligned with principal vectors. Because our similarity score is *linear* in X, the contribution assigned to each input direction coincides exactly with the corresponding gradient entry. Thus, although these two viewpoints differ conceptually (the former provides a structural decomposition, while the latter measures local sensitivity), here they yield identical importance scores.

IV. Experiment

In this section, we evaluate ProGrass-LVQ in the context of image classification, although the formulation naturally extends to image-set and text classification, similar to AChorDS-LVQ. In this context, we consider each image as a set of spatial regions and use a convolutional neural network (CNN) as a feature extractor to produce a descriptor for each region (see Fig. 1). Because our ProGrass-LVQ classifier operates in a set-based manner, the spatial ordering of these regions does not influence the classification outcome, making the approach robust to local transformations and spatial permutations.

We evaluate the proposed method on three benchmark datasets: a) The CUB-200-2011: it consists of 200 fine-grained bird species, b) The Oxford-IIIT Pet: it contains 37 dog and cat breeds with large appearance variations and diverse poses. c) Brain Tumor MRI [20]: it includes MRI scans from four classes: glioma, meningioma, no tumor, pituitary. These datasets allow us to examine the behavior of the model across natural, fine-grained, and medical domains.

In all experiments, we use the ResNet-50 backbone as the feature extractor. From the last convolutional layer, we obtain local feature descriptors of dimension $D = 512$. These descriptors are treated as an unordered set and mapped to a d-dimensional subspace ($\mathsf{d} = 5$) on $\mathcal{G}(5, 512)$, which forms the input to the proposed classifier. During training, a batch size of 32 is used. The scale parameter of the softmax distribution is fixed to $\beta = 1$, and a small constant $\varepsilon = 10^{-4}$ is used for numerical stability. Separate learning rates are used for different components of the model: $lr_{\text{prototype}} = 10^{-2}, lr_{\text{relevant}} = 10^{-5}, lr_{\text{net}} = 10^{-5}$. To stabilize training and preserve meaningful representations, the CNN backbone is frozen for the first 10 epochs and then jointly fine-tuned with the remaining model parameters.

Table I reports the classification performance. Here, we use accuracy for evaluation as this enables the comparison with the other prototype-based models. Overall, ProGrass-LVQ achieves the best performance across all three datasets. In particular, on the PET dataset, it outperforms existing approaches by a clear margin. The AChorDS-LVQ method achieves competitive performance on CUB200 and Brain, highlighting the effectiveness of modeling images on the Grassmann manifold. However, while AChorDS-LVQ relies on a higher subspace dimensionality ($\mathsf{d} = 10$), ProGrass-LVQ achieves equal or better results using a more compact representation ($\mathsf{d} = 5$). A possible explanation for this improvement lies in differences in both the learning objective and the optimization strategy. In contrast to AChorDS-LVQ, which adapts only two prototypes (i.e. the closest prototypes with the same and different labels)

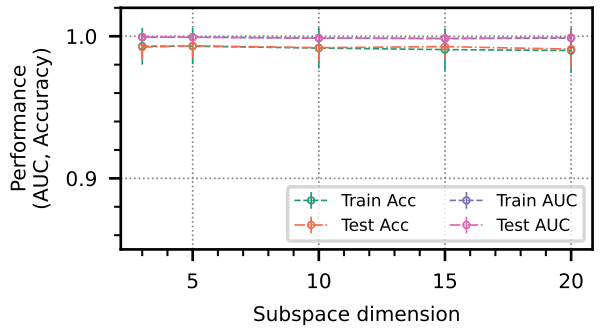

Fig. 2. Effect of subspace dimension on the performance (Accuracy and AUC) of Inclusive ProGrass-LVQ classifier on the brain tumor dataset.

Methods	CUB200	PET	Brain
ProtoPNet	79.2 (±0.1)	-	97.4 (±0.2)
D-ProtoPNet [5]	83.4 (±0.1)	-	98.4 (±0.2)
TestNet [10]	83.0 (±0.2)	-	98.2 (±0.1)
ProtoTree [4]	82.2 (±0.7)	-	98.0 (±0.3)
PIPNet [21]	82.0 (±0.3)	88.5 (±0.2)	97.5 (±0.3)
PBN-CBC [8]	83.3 (±0.3)	90.1 (±0.1)	-
ProtoArgNet [9]	85.4 (±0.3)	-	99.5 (±0.3)
AChorDS-LVQ [11]	87.4 (±0.1)	91.4 (±0.0)	**99.7** (±0.1)
Inclusive ProGrass-LVQ	**87.5** (±0.3)	93.1 (±0.0)	**99.7** (±0.1)
Exclusive ProGrass-LVQ	87.4 (±0.1)	**93.2** (±0.0)	**99.7** (±0.1)

TABLE I
THE ACCURACIES OF MODELS FOR IMAGE CLASSIFICATION.

for each training sample, ProGrass-LVQ allows all prototypes to be updated jointly, attracting prototypes of the correct class while repelling those of competing classes. Moreover, prototype updates in ProGrass-LVQ explicitly respect the geometry of the Grassmann manifold, rather than applying Euclidean updates directly. Furthermore, the performance of our newly introduced model is invariant to the subspace dimensionality, as seen in Fig. 2, in terms of both accuracy and area-under-the ROC curve (AUC). AUC is an evaluation metric that is robust against class imbalance. The figure also represents that the model does not overfit across the dimensions of the subspace, and the high correlation between our model's accuracy and AUC values, in the brain tumor dataset. Finally, the small differences between the inclusive and exclusive divergence variants suggest that the choice of divergence has a limited impact on overall classification accuracy.

Beyond good performance, our ProGrass-LVQ method provides interpretable insights by highlighting the most influential image regions for each prediction. We generate region-level importance scores by computing the gradient of the similarity measure with respect to the feature map of the last CNN layer (see Eq. (13)). The resulting maps are then upsampled using bicubic interpolation to match the original image resolution. This process produces heatmaps that indicate which parts of the image contribute most to the model's decisions. Fig. 3 illustrates this explanation for a 'painted bunting' example. As shown in the aggregated heatmap (Fig. 3, top right), the model primarily highlights pixels on the belly, wing, and head, which is well aligned with human expectations for discriminative bird characteristics. In addition to the aggregated explanation used for the final prediction, we also visualize heatmaps for

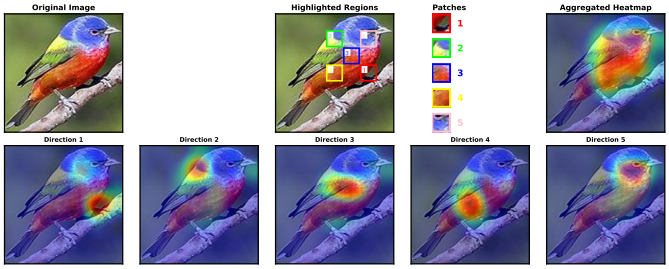

Fig. 3. An excerpt of interpretability of ProGrass-LVQ's decision when classifying an example from the CUB200 dataset. First row: Left) the example image. Middle) highlighted regions per direction. Right) the heatmap highlighting influential regions in predictions. Second row: the highlighted regions per direction.

each principal direction, as seen in Fig. 3. They reveal that different directions emphasize distinct and complementary image regions, capturing diverse aspects of similarity between the input and the prototype. Specifically, individual directions focus on characteristic patterns such as the red coloration of the belly, the green texture of the wing, and the blue coloration of the head. The final decision emerges from the combination of these directional contributions, providing a fine-grained decomposition of the model's reasoning process.

V. LIMITATIONS

We note that a limitation of this paper is that we investigated only the state-of-the-art prototype-based interpretable models. This is primarily to ensure comparable backbones of models, thereby fairness in comparison of their performances and interpretability. As previously noted, we compare the performance of ProGrass-LVQ against those of the others only in terms of accuracy. This is due to (1) inaccessibility of the exact models (i.e, their hyperparameter settings) so as to rerun them; and (2) unavailability of the other performance metrics for these models from the works we referred to. Additionally, this work showcases the interpretability of our ProGrass-LVQ only in qualitative terms. In our future works we aim to also include quantitative measures of our models' interpretability and explainability, for a more direct comparison to other relevant models. Lastly, while we mentioned that ProGrass-LVQ can be applied on image-set and text classification problems as well, in this paper we have not included any experiments related to these, due to space constraints.

VI. CONCLUSION

In this work, we presented ProGrass-LVQ, a probabilistic LVQ method for set classification, defined directly on the Grassmann manifold for interpretable image classification. By representing images as sets of regional descriptors and modeling them as low-dimensional subspaces, our approach captures the geometric structure of visual data while remaining invariant to the ordering and spatial arrangement of image regions. Unlike conventional PBNs operating in Euclidean embedding spaces, our method employs geometry-aware optimization to update prototypes along the Grassmann manifold, enabling principled learning of subspace representations.

A key contribution is the integration of KL-divergence based probabilistic LVQ with Grassmannian optimization, which allows the model to explicitly capture predictive uncertainty while preserving intrinsic interpretability. In addition, we introduced a gradient-based explanation method that quantifies the contribution of individual image regions to the model's decisions.

Experimental results demonstrate that the proposed ProGrass-LVQ approach outperforms existing prototype-based methods while relying on a more compact subspace representation than comparable approaches. Beyond improved accuracy, the resulting visual explanations show that different prototype directions focus on class-specific patterns and aggregate evidence from multiple image regions to support a prediction. These results highlight the potential of ProGrass-LVQ method for interpretable and reliable visual recognition use-cases.

ACKNOWLEDGMENT

This publication is part of the project HAICu with file number NWA.1518.22.105. of the research programme Onderzoek op Routes door Consortia 2022 which is financed by the Dutch Research Council (NWO). S.G was supported by the Next Level via Datapoort: Linking MCL & UMCG Hospital Data with LROI (LROI 2025-003).

REFERENCES

[1] M. Mohammadi, M. Biehl, A. Villmann, and T. Villmann, "Sequence learning in unsupervised and supervised vector quantization using hankel matrices," in *International Conference on Artificial Intelligence and Soft Computing*. Springer, 2017, pp. 131–142.

[2] M. Mohammadi, M. Wieling, and M. Vols, "An interpretable approach to detect case law on housing and eviction issues within the hudoc database," *Artificial Intelligence and Law*, pp. 1–22, 2025.

[3] C. Chen, O. Li, C. Tao, A. Barnett, C. Rudin, and J. Su, "This looks like that: Deep learning for interpretable image recognition," in *Advances in Neural Information Processing Systems*, 2019.

[4] M. Nauta, R. Van Bree, and C. Seifert, "Neural prototype trees for interpretable fine-grained image recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 14 933–14 943.

[5] J. Donnelly, A. J. Barnett, and C. Chen, "Deformable protopnet: An interpretable image classifier using deformable prototypes," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 265–10 275.

[6] D. Rymarczyk, Ł. Struski, J. Tabor, and B. Zieliński, "Protopshare: Prototype sharing for interpretable image classification and similarity discovery," *arXiv preprint arXiv:2011.14340*, 2020.

[7] D. Rymarczyk, Ł. Struski, M. Górszczak, K. Lewandowska, J. Tabor, and B. Zieliński, "Interpretable image classification with differentiable prototypes assignment," in *European Conference on Computer Vision*. Springer, 2022, pp. 351–368.

[8] S. Saralajew, A. Rana, T. Villmann, and A. Shaker, "A robust prototype-based network with interpretable rbf classifier foundations," *arXiv preprint arXiv:2412.15499*, 2024.

[9] H. Ayoobi, N. Potyka, and F. Toni, "Protoargnet: Interpretable image classification with super-prototypes and argumentation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, no. 2, 2025, pp. 1791–1799.

[10] J. Wang, H. Liu, X. Wang, and L. Jing, "Interpretable image recognition by constructing transparent embedding space," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 895–904.

[11] M. Mohammadi and S. Ghosh, "A prototype-based model for set classification," *arXiv preprint arXiv:2408.13720*, 2024.

[12] M. Mohammadi, M. Babai, and M. Wilkinson, "Generalized relevance learning grassmann quantization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[13] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization algorithms on matrix manifolds*. Princeton University Press, 2008.

[14] S. Ghosh, E. S. Baranowski, M. Biehl, W. Arlt, P. Tiňo, and K. Bunte, "Interpretable modelling and visualization of biomedical data," *Neurocomputing*, vol. 626, p. 129405, 2025.

[15] A. Villmann, M. Kaden, S. Saralajew, and T. Villmann, "Probabilistic learning vector quantization with cross-entropy for probabilistic class assignments in classification learning," in *International Conference on Artificial Intelligence and Soft Computing*. Springer, 2018, pp. 724–735.

[16] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM journal on Matrix Analysis and Applications*, vol. 20, no. 2, pp. 303–353, 1998.

[17] Q. Wang, J. Gao, and H. Li, "Grassmannian manifold optimization assisted sparse spectral clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5258–5266.

[18] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, "Fooling lime and shap: Adversarial attacks on post hoc explanation methods," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 2020, pp. 180–186.

[19] Y. Wang, T. Zhang, X. Guo, and Z. Shen, "Gradient based feature attribution in explainable ai: A technical review," *arXiv preprint arXiv:2403.10415*, 2024.

[20] M. Nickparvar, "Brain tumor mri dataset," 2021.

[21] M. Nauta, J. Schlötterer, M. Van Keulen, and C. Seifert, "Pip-net: Patch-based intuitive prototypes for interpretable image classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2744–2753.

[22] N. Boumal, *An introduction to optimization on smooth manifolds*. Cambridge University Press, 2023.

[23] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU press, 2013.

VII. APPENDIX

A. Gradients Computation

As discussed in Section III-B, the computation of the *Riemannian gradient* requires first obtaining the *Euclidean gradient* of the cost function with respect to the prototypes. Here, we only consider the *exclusive* variant of the loss function which is the Kullback–Leibler divergence between the target distribution $p(\cdot|\mathbf{X})$ and the model distribution $\hat{p}(\cdot|\mathbf{X})$:

$$D_{\mathrm{KL}}\big(\hat{p}(\cdot|\mathbf{X}) \,\|\, p(\cdot|\mathbf{X})\big) = \sum_{c=1}^{C} \hat{p}(c|\mathbf{X}) \, \ln\left(\frac{\hat{p}(c|\mathbf{X})}{p(c|\mathbf{X})}\right).$$

In a similar manner, one can compute the gradients for the *inclusive* variant.

1) Derivative w.r.t. prototypes: Let $\mathbf{W}_k$ denote the prototype of class k. Using the chain rule, the Euclidean gradient of the loss w.r.t. $\mathbf{W}_k$ becomes:

$$\frac{\partial D_{\mathrm{KL}}}{\partial \mathbf{W}_k} = \sum_{c=1}^{C} \frac{\partial}{\partial \hat{p}(c|\mathbf{X})} \left(\hat{p}(c|\mathbf{X}) \, \ln\left(\frac{\hat{p}(c|\mathbf{X})}{p(c|\mathbf{X})}\right)\right). \\ \frac{\partial \hat{p}(c|\mathbf{X})}{\partial s(\mathbf{X}, \mathbf{W}_k)} . \frac{\partial s(\mathbf{X}, \mathbf{W}_k)}{\partial \mathbf{W}_k}$$

a) Derivative of the first term: For each class c:

$$\frac{\partial}{\partial \hat{p}(c|\mathbf{X})} \left(\hat{p}(c|\mathbf{X}) \ln\left(\frac{\hat{p}(c|\mathbf{X})}{p(c|\mathbf{X})}\right)\right) = 1 + \ln\left(\frac{\hat{p}(c|\mathbf{X})}{p(c|\mathbf{X})}\right)$$

b) Derivative of probability: The model uses a softmax over the similarities $s(\mathbf{X}, \mathbf{W}_k)$. For clarity, we denote $\hat{p}(c) = \hat{p}(c|\mathbf{X})$. The derivative of the softmax is well known:

$$\frac{\partial \hat{p}(c)}{\partial s_k} = \begin{cases} \hat{p}(c)(1 - \hat{p}(c)) & c = k \\ -\hat{p}(c)\hat{p}(k) & c \neq k \end{cases}$$

This expression corresponds to the familiar *attraction–repulsion* mechanism in LVQ: the correct prototype is pulled closer (positive term), while prototypes from other classes are pushed away (negative term).

c) Derivative of the similarity measure:: The similarity function from Eq. (12) is:

$$s(\mathbf{X}, \mathbf{W}) = \mathrm{tr}\big(\Lambda \mathbf{U}^\top \mathbf{V}\big),$$

where $\mathbf{U} = \mathbf{X}\mathbf{Q}_1$ and $\mathbf{V} = \mathbf{W}\mathbf{Q}_2$. Replacing $\mathbf{V}$:

$$s(\mathbf{X}, \mathbf{W}) = \mathrm{tr}\big(\Lambda \mathbf{U}^\top \mathbf{W}\mathbf{Q}_2\big) = \mathrm{tr}\big(\mathbf{W}\mathbf{Q}_2 \Lambda \mathbf{U}^\top\big)$$

This gives the following result:

$$\frac{\partial}{\partial \mathbf{W}} s(\mathbf{X}, \mathbf{W}) = \mathbf{U}\Lambda\mathbf{Q}_2^\top$$

2) Derivative w.r.t. relevance factors: Let $\lambda = [\lambda_1, \cdots, \lambda_d]$ denote the vector of relevance weights. Similar to the above, we can compute the gradient with respect to the relevance factors. The only difference is in the last term:

$$\frac{\partial}{\partial \lambda} s(\mathbf{X}, \mathbf{W}) = \mathrm{tr}\big(\mathbf{U}^\top \mathbf{V}\big)$$

After any update, we also normalize the relevances.

B. Convergence Analysis on the Grassmann Manifold

In this section, we provide a convergence analysis of the proposed geometry-aware training procedure on the Grassmann manifold.

1) Objective Function: Let $\mathcal{G}(\mathsf{d}, D)$ denote the Grassmann manifold. We consider the optimization problem:

$$\min_{\mathbf{W} \in \mathcal{G}(\mathsf{d}, D)} f(\mathbf{W}),$$

Given the fixed feature subspace $\mathbf{X}$, the loss function f is based on canonical correlations, which depend smoothly on $\mathbf{W}$ (away from degenerate cases):

$$z(\mathbf{W}) = s(\mathbf{X}, \mathbf{W}),$$

The softmax cross-entropy loss (equivalently, the *inclusive* divergence) is applied to these logits. The softmax cross-entropy loss has bounded second derivatives with respect to its inputs. Therefore, f is twice continuously differentiable and has bounded Riemannian Hessian on $\mathcal{G}(\mathsf{d}, D)$. Consequently, the Riemannian gradient of f is Lipschitz continuous.

2) Convergence of the Training Procedure: We now state the main convergence result. For this, we use the following lemma ([22, Corollary 10.54]).

Lemma 7.1: Let $f : \mathcal{M} \to \mathbb{R}$ have an L-Lipschitz continuous Riemannian gradient. Then, for any $\mathbf{X} \in \mathcal{M}$ and $\mathbf{S} \in T_{\mathbf{X}}\mathcal{M}$ in the domain of the exponential map,

$$|f(\mathrm{Exp}_{\mathbf{X}}(\mathbf{S})) - f(\mathbf{X}) - \langle \mathrm{grad} f(\mathbf{X}), \mathbf{S}\rangle| \le \frac{L}{2}\|\mathbf{S}\|^2.$$

Theorem 7.2 (Convergence): Let $f : \mathcal{G}(\mathsf{d}, D) \to \mathbb{R}$ satisfy:

1) f is continuously differentiable,
2) the Riemannian gradient of f is L-Lipschitz continuous,
3) f is bounded below.

Let $\{\mathbf{W}^k\}$ be generated by:

$$\mathbf{W}^{k+1} = \mathrm{Exp}_{\mathbf{W}^k}(-\alpha\, \nabla^{\mathcal{G}} f(\mathbf{W}^k)),$$

where $\alpha \in (0, 1/L]$. Then:

1) $f(\mathbf{W}^k)$ is non-increasing,
2) $\sum_{k=0}^{\infty} \|\nabla^{\mathcal{G}} f(\mathbf{W}^k)\|^2 < \infty$,
3) $\|\nabla^{\mathcal{G}} f(\mathbf{W}^k)\| \to 0$,
4) every accumulation point is a first-order critical point.

Proof 7.3: Applying lemma 7.1 with $\mathbf{S}_k = -\alpha\, \nabla^{\mathcal{G}} f(\mathbf{W}^k)$ gives:

$$f(\mathbf{W}^{k+1}) \le f(\mathbf{W}^k) - \alpha\|\nabla^{\mathcal{G}} f(\mathbf{W}^k)\|^2 + \frac{L}{2}\alpha^2 \|\nabla^{\mathcal{G}} f(\mathbf{W}^k)\|^2.$$

For $\alpha \le 1/L$, this implies:

$$f(\mathbf{W}^{k+1}) \le f(\mathbf{W}^k) - \frac{\alpha}{2}\|\nabla^{\mathcal{G}} f(\mathbf{W}^k)\|^2.$$

Thus, $f(\mathbf{W}^k)$ is non-increasing and bounded below, hence convergent. Summing over k:

$$\sum_{k=0}^{T} \frac{\alpha}{2}\|\nabla^{\mathcal{G}} f(\mathbf{W}^k)\|^2 \le f(\mathbf{W}^0) - f(\mathbf{W}^{T+1}).$$

Taking $T \to \infty$ and using boundedness:

$$\sum_{k=0}^{\infty} \|\nabla^{\mathcal{G}} f(\mathbf{W}^k)\|^2 < \infty \Rightarrow \|\nabla^{\mathcal{G}} f(\mathbf{W}^k)\| \to 0.$$

Since $\mathcal{G}(\mathsf{d}, D)$ is compact, accumulation points exist. By continuity of the gradient, any accumulation point satisfies $\nabla^{\mathcal{G}} f(\mathbf{W}^*) = 0$..

C. Computational Complexity

Similar to other Grassmann manifold-based approaches, truncated singular value decomposition (SVD) plays a central role in our method. Specifically, it is used for: (i) obtaining subspace representations for each image (cf. (5)), (ii) computing principal vectors and canonical correlations (cf. (4)), and (iii) evaluating the exponential map on the Grassmann manifold (cf. (2)).

For a matrix of size $D \times d$ with $D \gg \mathsf{d}$, the computational cost of a truncated (thin) SVD is $\mathcal{O}(D\mathsf{d}^2)$. This is typically the dominant cost in Grassmann-based methods.

In practice, more efficient approximate techniques can further reduce the computational burden, especially for large-scale problems [23]. These methods compute low-rank approximations with reduced complexity while maintaining good accuracy. Moreover, in our setting the subspace dimension d is typically small. As demonstrated in our experiments, competitive performance can be achieved even for $\mathsf{d} = 5$, which significantly reduces the overall computational cost.