Investigating Desirable Properties of Inverse Projections and Decision Maps

Yu Wang$^{1[0000-0001-6066-0279]}$ and Alexandru Telea$^{1[0000-0002-1129-4628]}$

Department of Information and Computing Science, Utrecht University, 3584 CS Utrecht, The Netherlands
{y.wang6, a.c.telea}@uu.nl

Abstract. Inverse projection techniques enable the creation of decision maps which help the visual exploration of trained classification models. However, different inverse projections lead to significantly different decision maps for the same model, leading to uncertainty in their interpretation. Recent work compared three inverse projection techniques from the perspective of their intrinsic dimensionality and showed that all three techniques visualize only two-dimensional substructures in the data space. We extend this evaluation in several directions. First, we consider three additional inverse projections thereby covering, to our knowledge, all such techniques in existence. Secondly, we correlate the quality of the inverse projections with their ability to depict certain types of data structures. Finally, we study the smoothness of the structures created by inverse projections. Our results show that all inverse projection techniques essentially cover only two-dimensional structures in the data space and that the smoothness of such structures is inversely correlated with their ability to approximate data points. Based on our findings, we also propose ways to select inverse projections which lead to interpretable decision maps.

Keywords: Dimensionality reduction · Inverse projections · Decision maps · Intrinsic dimensionality.

1 INTRODUCTION

Dimensionality reduction (DR) methods, also called projections, map high-dimensional data samples to a low-dimensional space (typically 2D or 3D for visualization purposes) while aiming to keep neighborhood and/or distance relations between the data samples. DR methods scale very well both in the number of samples and number of dimensions, which has made them widespread candidates for building visualization applications for high-dimensional data.

Inverse projections, also sometimes called backprojections, aim to reverse – in a broad sense – the mapping produced by a DR method. Inverse projections have enabled multiple applications such as shape or image morphing [1], data imputation [18], and constructing so-called classifier *decision maps* [39, 43, 34] that depict the behavior of a trained machine learning (ML) model. Key

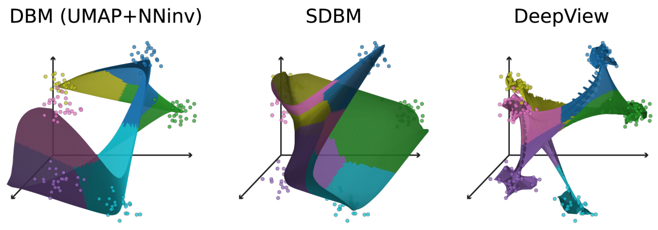


Fig. 1. A first indication of the limitations of three inverse projection techniques (NNInv [39], SDBM [34], DeepView [43]) outlined by Wang *et al.* [49]. For a Logistic regression classifier trained on a simple 3D synthetic 6-blobs dataset, all three inverse projections generate relatively smooth *surfaces* (embedded in the considered 3D data space) which interpolate between the training-set samples. Both surfaces and samples are colored to indicate the class labels at the respective locations in 3D space.

to all these applications of inverse projections is their ability to *extrapolate* the low-to-high dimensional mapping away from the data points which are projected from the high to the low dimensions by a DR technique. This ability is crucial for all aforementioned applications – inferring data values in the 'empty space' between the points created by a projection, generating shapes or images that 'morph' (that is, interpolate) between a selected set of examples, and inferring how a ML model behaves on data values outside a given training or test set.

Wang *et al.* [49] first explored the behavior of inverse projections by a simple experiment which depicted the decision maps of a linear regressor model trained on a synthetic 3D dataset. All three inverse projections they examined (NNInv [39], SDBM [34], DeepView [43]) created essentially smooth *surfaces* that interpolate between the 3D samples (see Fig. 1). Practically, this means that decision maps created by these methods will only show the model's behavior on a *small* 2D subset of the entire 3D data space the model works on, namely the aforementioned surfaces. How the model behaves on samples away from these surfaces is not shown.

In our recent work [50], we aimed to answer several questions to further explore the surface-like behavior of inverse projections observed by Wang *et al.*:

Q1 How do decision maps look like for different ML models than the one used by Wang *et al.*?
Q2 How do the boundaries we see in Figure 1 relate to the actual decision boundaries of a classifier?
Q3 Which parts of the data space do decision maps cover for data spaces having many more than three dimensions?
Q4 How do different inverse projection techniques influence the answers obtained for Q1-Q3?

To answer Q1-Q4, we studied the behavior of NNInv, SDBM, and DeepView on a combination of several datasets (of varying dimensionality) and classification models. Next, we proposed a way to measure how far an inverse projection can produce structures away from a single surface using intrinsic dimension estimation [6, 10, 11, 4]. Jointly put, our findings showed that, for all datasets, all three inverse projections essentially create surface like structures – with varying local smoothness – when mapping all points of the 2D space, except for points very close to the ones created by the direct projection.

However, the experiments in [50] leave several questions unanswered:

Q5 How are Q1-Q3 answered for additional direct and inverse projection techniques?
Q6 How is the smoothness of the backprojection influenced by the direct-and-inverse projection technique choice?

Answering Q5 is important for practitioners aiming to choose which (inverse) projections to use for any of the aforementioned applications (imputation, morphing, decision maps). In our work, we address this by studying several additional combinations of direct projections (UMAP [31], MDS [8]) and inverse projections (iLAMP [41], RBF [1], and iMDS [7]). Answering Q6 is important as the earlier experiments in [50] showed, but did not explain, variations of the backprojection smoothness. A smooth backprojection is essential for interactive applications such as morphing where users want that small changes of a selected 2D point yield only small changes of the inferred data sample [41]. Conversely, a backprojection that aims to cover as much as possible from the data space will be more effective in *e.g.* creating decision maps that capture more of a ML model's behavior, but will be likely less smooth. Understanding the inverse projection's smoothness is thus important for users to make informed choices for such applications.

The structure of this paper is as follows. Section 2 introduces related work. Section 3 presents our results for 3D datasets, for which direct visual evaluation can be used to answer our questions. Section 4 extends our evaluation with new methods that address high-dimensional data. Section 5 discusses our findings. Finally, Section 6 concludes our paper.

2 BACKGROUND AND RELATED WORK

Definitions: We first introduce the notations and concepts further used in this paper. Let $\mathbf{x} \in \mathbb{R}^n$ be an n-dimensional sample or data point and $D = \{\mathbf{x}_j\}$, $j = 1, 2, \dots, N$, a dataset of N such samples.

A *classification model* (or classifier) is a function

$$f : \mathbb{R}^n \to C \tag{1}$$

that maps a sample $\mathbf{x}$ to a label $f(\mathbf{x})$ in a given label-set C. A *decision zone* of f, for class $y \in C$, is the point set $\{\mathbf{x} \in \mathbb{R}^n | f(\mathbf{x}) = y\}$; the boundaries separating decision zones are called the *decision (hyper)surfaces* of f.

A *projection*, also called dimensionality reduction (DR), is a function

$$P : \mathcal{P}(\mathbb{R}^n) \to \mathcal{P}(\mathbb{R}^q), \tag{2}$$

where $\mathcal{P}$ denotes the power set. That is, P maps datasets $D \subset \mathbb{R}^n$ to datasets $P(D) \in \mathbb{R}^q$, where $q \ll n$. We further use the term projection to denote both the operation P and also its output $P(D)$ for a given input D, depending on the context. For visualization purposes, one typically uses $q = 2$. By abusing notation, we denote $P(\mathbf{x})$ to be the point in $P(D)$ that corresponds to the sample $\mathbf{x} \in D$. Tens of different projection techniques exist with different abilities to capture data-space similarities $\mathbb{R}^n$ by corresponding similarities in the projection space $\mathbb{R}^q$, computational speed, robustness to noise, out-of-sample ability, and ease of use and implementation. Extensive surveys cover all these aspects [16, 32, 25, 45, 15].

An *inverse projection*, or unprojection, is a function

$$P^{-1} : \mathbb{R}^q \rightarrow \mathbb{R}^n, \tag{3}$$

which aims to reverse the mapping of a given projection P for a given dataset D. Typically P^{-1} is constructed by minimizing errors of the form $\sum_{\mathbf{x} \in D} \|P^{-1}(P(\mathbf{x})) - \mathbf{x}\|$. In most cases, P^{-1} is not the mathematical inverse of P since P may not be injective *i.e.* it can map different samples in $\mathbb{R}^n$ to the same point in $\mathbb{R}^q$. Moreover, P^{-1} is definitely not surjective since P only maps the points in the finite set D to $\mathbb{R}^q$. Hence, there is an infinity of points in $\mathbb{R}^q$ that are not covered by P but need to be handled by P^{-1} to enable applications such as shape or image morphing [41], data imputation [18], and constructing classifier decision maps (detailed further below).

An important difference between direct and inverse projections follows from the above. A direct projection aims to map a given, finite, *set* of samples from high to low dimensions. Formally speaking, direct projections do not need to be *functions* – they only need to produce the q-dimensional counterparts of a n-dimensional sample set D while obeying certain quality criteria [16, 32]. For example, DR techniques like t-SNE [28] are non-parametric (thus are not functions) since, for the same input sample set D, they can generate different outputs $P(D)$. In contrast, inverse projections aim to produce a *function* that not only reverses the effect of a direct projection for the aforementioned D, but also smoothly extrapolates this effect to any point in the q-dimensional space in a deterministic way.

While many direct projection algorithms exist [16], only a few inverse projection techniques have been proposed. This can be explained by the fact that computing inverse projections is significantly harder than computing direct projections. The key problems are that (a) inverse projections need to create high-dimensional data from a (very) low-dimensional input in a meaningful way; and, even more difficult, (b) are used specifically to infer data samples for q-dimensional points for which no ground-truth projection information is available. Concerning existing inverse projection methods, iLAMP [41] uses local information in $P(D)$ to build affine transformations that map $\mathbb{R}^2$ to $\mathbb{R}^n$. Although iLAMP was proposed to reverse the LAMP projection technique [26], it can be used to reverse other projections P with reasonable results [40, 20]. The same authors next proposed an inverse projection method using Radial Basis Functions

(RBFs) to gain continuity and global behavior. NNinv [20] constructs P^{-1} by deep learning to map the points of any given 2D scatterplot $P(D)$, constructed by any projection technique P, to corresponding samples in D. SSNP [19] uses semi-supervised deep learning to construct both a P and its inverse P^{-1}, following an autoencoder principle [24]. DeepView [43] customizes the UMAP projection [31] with a classifier to produce a discriminative projection while a modified UMAP was used for inverse projection. More recently, Blumberg *et al.* [7] proposed iMDS to invert MDS projections using multilateration with randomly selected samples to estimate the inverse projections.

Decision maps aim to construct dense visualizations of a trained ML model f as follows. Let $I = \{\mathbf{p} \in \mathbb{R}^2\}$ be an image that samples the 2D zone containing the projection $P(D)$. A decision map is simply the image I whose pixels $\mathbf{p}$ are colored to depict the labels $f(P^{-1}(\mathbf{p}))$ inferred by the model f. Same-color areas in I thus show the decision zones of f; neighbor pixels of different colors indicate the decision boundaries of f. The set $I^{-1} = \{P^{-1}(\mathbf{p}) | \mathbf{p} \in I\}$, optionally colored as mentioned above, is called the *backprojection* of the decision map; the surfaces in Fig. 1 are examples hereof. As such, Q3 and Q6 (see Sec. 1) relate to how much of the data space does I^{-1} cover, respectively how smooth is I^{-1}. Decision maps assist many tasks in ML engineering such as understanding how a model generalizes from a training set [20, 44, 43], studying how different models agree [18], dynamic data imputation [18], and studying a model's brittleness [29].

In principle, decision maps can be constructed using any inverse projection P^{-1}, suitably constructed from any direct projection P. However, only a limited number of (P, P^{-1}) combinations have been used to this end, as follows. Espadoto et al. [17] tested 28 projection techniques P with iLAMP as the inverse projection to construct decision maps and concluded that t-SNE and UMAP were the best choices for P. Following this, DBM [40] used UMAP [31] or t-SNE [28] for P and NNinv [20] for P^{-1}. Recently, DBM was accelerated by using bisection techniques to reduce the number of times that a given P^{-1} needs to be invoked [23]. Espadoto *et al.* [20] used UMAP and t-SNE for direct projection while using iLAMP or RBF for the inverse one. DBM was improved to produce less noisy images by filtering out poorly projected samples [39]. SDBM [34] uses SSNP which, as already mentioned, provides both P and P^{-1}. DeepView [43] leverages discriminative dimensionality reduction [42] to enhance the direct projection UMAP [31], which also provides an inverse projection. Finally, the recent inverse projection iMDS [7] can also potentially be used to construct decision maps if P is set to MDS – though this was not tested in practice.

Evaluations of decision maps involve the following aspects:

– *visual quality:* Decision maps created by different methods are compared against each other with the best one chosen based on agreement with ground-truth information on the visualized model [40, 39, 34, 17];
– *stability:* Oliveira *et al.* [33] studied how much DBM and SDBM maps would change in presence of small training set perturbations and concluded that these methods are quite robust to such changes;

– *accuracy and speed:* Wang *et al.* [49] provided a detailed quantitative evaluation of the quality and speed of decision map methods by extending classical ML performance metrics [43] with several visualizations;
– *coverage:* Wang *et al.* [49] also showed that, for the simple example of a linear regressor trained on a 3D dataset, DBM, SDBM, and DeepView only depict a *surface* (see again Fig. 1). We recently extended this evaluation to consider datasets of varying dimensionality and several more complex classifiers [50].

3 VISUAL EVALUATION ON 3D DATA

To answer question Q5 (Sec. 1), we first extend the visual evaluation for 3D datasets in [50] to use more inverse projection techniques. We next evaluate these techniques on high-dimensional data (Sec. 4).

3.1 Method

Dataset: We conduct this evaluation using the well-known three-class Iris flower dataset [22]. As explained in [50], the key idea of visual evaluation is to directly *draw* the backprojected decision maps I^{-1} and see how these actually cover the data space of the trained ML model they are supposed to depict. To be able to create such visualizations, we need our dataset to have maximally $n = 3$ dimensions. We achieve this by restricting the Iris dataset to its last three features.

Decision map methods: Besides the three decision map methods used in [50], *i.e.*, DBM [40], SDBM [34], and DeepView [43], we also consider three additional inverse projection techniques: iLAMP [41], RBF [20], and iMDS [7]. For all methods except SDBM and DeepView (which use their own direct projection techniques, see Sec. 2), we now use MDS as direct projection. This is because (a) one of the considered inverse projections, iMDS, only works with MDS as direct projection; and (b) this setting allows us to minimize the number of direct projection techniques we use and thus provide a more intuitive comparison.

Classifiers: We study the behavior of decision maps using six classifiers: Logistic Regression [13], Support Vector Machines [12, SVM], Random Forests [9], Neural Networks, Decision Trees, and K-Nearest Neighbors (KNN). All are implemented using Scikit-Learn [38] with default parameters, except Neural Networks, which uses three hidden layers each with 256 units. For each classifier, we not only construct the backprojected decision maps I^{-1} (see Sec. 2) for the six studied decision map techniques, but also visualize the actual decision boundaries in the 3D data space.

3.2 Results

Preliminary comparison: Figure 2 (top two rows) shows the backprojected decision maps, each from two different viewpoints (for better interpretation), for the six studied direct-and-inverse projection combinations (columns). For ease of interpretation of the results, we use here only the simple Logistic Regression classifier. The corresponding 2D decision maps are shown in Fig. 2 (bottom row). This preliminary investigation already reveals several interesting facts.

Firstly, we see that the backprojected decision maps for the first three methods (DBM, SDBM, DeepView) have very similar smooth-surface-like shapes as the ones shown in Fig. 1 for the synthetic blobs dataset. The backprojected surfaces of DBM and SDBM are quite smooth and, as such, cannot get very close to (all) the actual data points; In contrast, DeepView creates a much more noisy surface which 'connects' the data points better. This is also observed in the actual 2D decision maps (bottom row in Fig. 2): The maps for DBM and SDBM have far smoother decision boundaries than the DeepView map. This tells us that DBM and SDBM can depict the classifier's behavior *further* from the training set (extrapolation), whereas DeepView shows this behavior *close to and inside* this set (interpolation). Further on, we see that the backprojected decision maps for MDS+iLAMP, MDS+RBF, and MDS+iMDS behave very differently from the first three techniques. The latter two generate decision maps and backprojections which are very similar to each other and also quite close to a planar surface. Slight differences exist though: MDS+RBF creates a quite smooth backprojection that strictly passes through every sample $\mathbf{x}$, *i.e.*, $P^{-1}(\mathbf{x}) = \mathbf{x}$ for all $\mathbf{x} \in D$. In contrast, MDS+iMDS creates a noisier backprojection that does not strictly pass through the data samples. The most noticeable outlier is the result of MDS+iLAMP. It shows the appearance of a 'triangle soup' that exhibits practically no smoothness. In contrast, its backprojection covers far more of the 3D data space than all other compared methods. It is worth mentioning here that such discontinuities, originating from the iLAMP inverse projection, is precisely why the iLAMP authors next proposed the RBF inverse projection which is continuous and smooth [1].

Detailed comparison: We now extend the findings obtained so far using the Linear Regressor classifier to all six classifiers mentioned in Sec. 3.1. At the same time, we extend the visual exploration used in Fig. 2 to show not only the backprojections I^{-1} but also the actual decision zones and decision surfaces. As this creates quite complex imagery, we now restrict the Iris dataset to its last two classes. This will decrease the amount of colors we need to use in our visualizations to two. Note that these two classes are not fully linearly separable, which makes our classification task more challenging than the synthetic blob dataset used in Fig. 1.

Figure 3a shows the actual decision zones and decision boundaries and the backprojected decision maps for the six classifiers mentioned in Sec. 3.1 and the same six decision map methods already explored in Fig. 2. The actual 2D decision maps are shown in Fig. 3b. We further study the differences between the backprojected decision map and the *actual* decision zones and surfaces as follows. We sample the 3D data space on a voxel grid of size 100^3 (to limit computational effort); compute, for each voxel $\mathbf{v}$, the predicted class $f(\mathbf{v})$, and color code it;

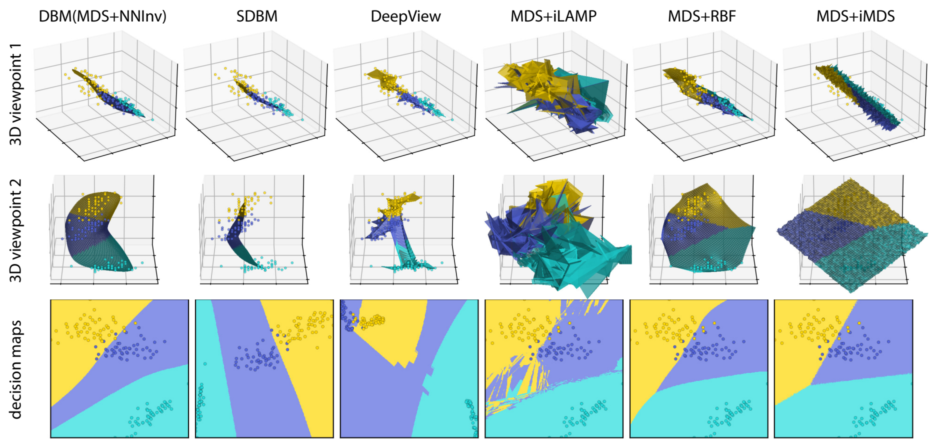
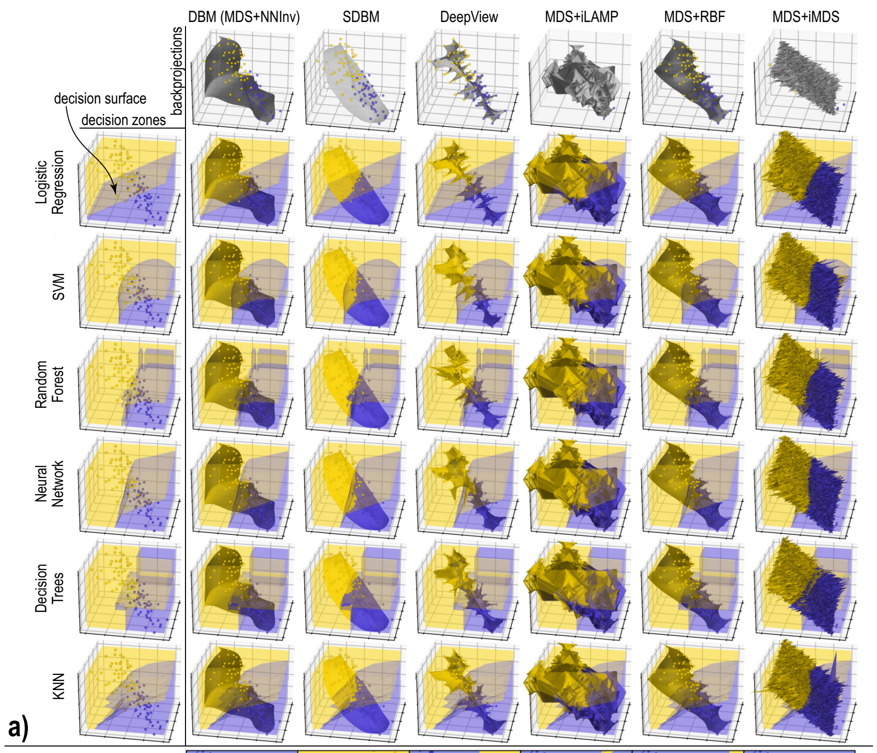


Fig. 2. Top two rows: Backprojections of the decision maps constructed by 6 inverse projection techniques (columns) with Logistic Regression on IRIS dataset, viewed from two viewpoints. Bottom row: Corresponding (2D) decision maps.

and draw this color-coded volume half-transparently (Fig. 3a, bottom 6 rows). The yellow, respectively purple, volumes are thus the *actual* decision zones of f. For clarity, we show these volumes, without the backprojection I^{-1}, in the leftmost column in Fig. 3a. Also, we draw the actual decision boundary $\mathcal{S}$ that separates the two decision zones in beige – see Fig. 3a, leftmost column, top cell for an example.

Figure 3 leads us to several insights. First, we see that the backprojected decision maps I^{-1} (shaded surfaces in Fig. 3a, top row), *i.e.*, the part of the data space that a decision map visualizes, are roughly orthogonal to, and intersecting, the actual decision surfaces (pale brown in Fig. 3a). That is, the boundaries which we *see* in a decision map (curves where yellow meets purple in Fig. 3b) are the intersection $\mathcal{S} \cap I^{-1}$. Separately, if we scan a column in Fig. 3a, we see that the backprojections I^{-1} are the same – or almost the same in the case of MDS+iMDS and DeepView (the reason for this is discussed separately below). However, as the classifiers change (rows), the decision boundaries change – see Fig. 3a, leftmost column. Hence, the *intersection* of I^{-1} with the actual decision boundary will change. As mentioned, this intersection is precisely what we see as color boundaries in a 2D decision map. So, if the backprojection I^{-1} is not smooth, this intersection can change *significantly*, even when the visualized classifier model changes only slightly. Simply put, this means that decision maps whose backprojections look ‘crumpled’ (non-smooth) can be very unstable and thus unsuited for practical use. Even stronger: the non-smoothness of the backprojection can create the false impression that a classifier has complex decision maps. Take for instance Logistic Regression, SVM, or Neural Networks; these classifiers show very smooth decision boundaries (Fig. 3b, leftmost column). However, their 2D decision maps created with DeepView or MDS+iLAMP show



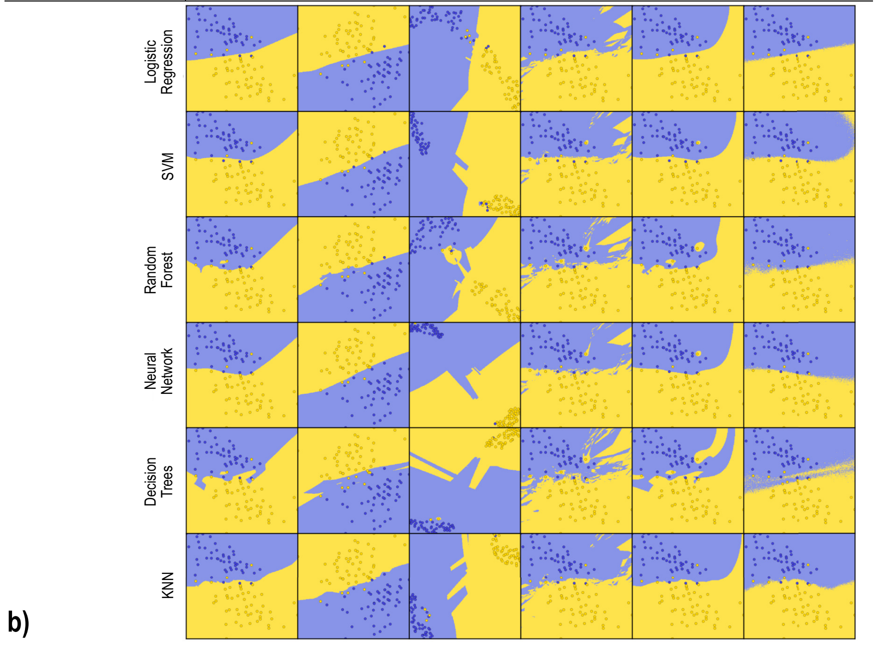


Fig. 3. Decision maps (a) backprojected in 3D; (b) original in 2D of six classifiers, modified Iris dataset, computed by six techniques. The decision zones are yellow, respectively purple; the decision surface separating them is beige. The shaded surfaces are the backprojected decision maps. Figure extends Fig. 2 in [50].

complex, non-smooth boundaries, which is clearly misleading. All in all, the above insights argue in favor of *e.g.* (S)DBM and MDS+RBF as techniques for creating decision maps and definitely against DeepView and MDS+iLAMP.

Secondly, we see that no decision map technique can actually depict the *full* decision boundaries of any classifier. For example, the linear decision boundary of Logistic Regression is not well captured, except by MDS+iMDS. The other decision maps show non-linear boundaries or even disconnected decision zones, see *e.g.* DeepView and MDS+iLAMP. Another example is for Decision Trees. We see that the actual decision zone (purple) is split into two disconnected components (top and bottom purple cubes (Fig. 3a, leftmost column)). However, none of the tested decision map techniques shows two such separated purple decision zones (Fig. 3b).

Finally, let us revisit the issue of the backprojection shapes generated by a given technique. DBM, SDBM, iLAMP, and RBF produce exactly the same shapes regardless of the classifier they depict – indeed, their P and P^{-1} do not depend on the classifier. In contrast, MDS+iMDS and DeepView can generate (slightly) different shapes for different classifiers, for different reasons, as follows. By design, DeepView uses discriminative dimensionality reduction [42], so its P depends on f. As for the reason why MDS+iMDS has different shapes for rows, this is because iMDS uses *random* selections of samples to compute its P^{-1}. While one can argue that DeepView's design shows more information on f, *controlling* how DeepView's decision maps actually sample the data space as a function of f is unclear. As such, we believe that the approaches of (S)DBM, iLAMP and RBF where this sampling only depends on the training set, are more intuitive and stable.

4 EVALUATION ON HIGH DIMENSIONAL DATA

We next extend the quantitative evaluation of inverse projections and decision maps for high-dimensional datasets in [50] to use all six inverse projection techniques listed in Sec. 3.1. Additionally, we substitute UMAP for MDS when using it in combination with the inverse projection techniques NNInv, RBF, and iLAMP, as UMAP is far better suited to handle high-dimensional data than MDS. We also present additional quantitative measurements that gauge the quality of the studied inverse projections and corresponding decision maps, as well as a visual exploration of the smoothness of the studied inverse projection techniques.

4.1 Method

Since decision maps fundamentally depend on inverse projections, it makes sense to first and foremost quantify the quality of P^{-1}. Further on, for $n > 3$ dimensional data, we cannot directly *draw* the backprojected images I^{-1}, as already mentioned in Sec. 3.1. Recall now our question Q3 (Sec. 1). To answer it, we measure how far I^{-1} is, locally, from a two-dimensional manifold embedded in $\mathbb{R}^n$. For this, we use intrinsic dimensionality (ID) estimation [4] with a linear method, *i.e.*, Principal Component Analysis (PCA), due to its intuitiveness, computational efficiency, ease of use, and popularity [16, 4, 46]. Finally, we use the gradient map technique [18] to get insights into the decision maps' smoothness. All these steps are detailed further below.

Datasets. We use five synthetic and real-world datasets, all having $N = 5000$ samples (Tab. 1). The synthetic datasets, with dimensionality n of 10, 30, and 100, consist of each of $C = 10$ isotropic Gaussian blobs. Using isotropic blobs ensures that the ID is the same as the dimension count n for these datasets. As real-world datasets, we use HAR [2] and MNIST [27]. The intrinsic dimensionality of these datasets has been estimated by prior work [14, 21, 3, 5]. We use Logistic Regression as an example classifier f. Note that this does not affect the ID estimation, as f is not involved in P^{-1}'s construction.

Dataset	n	N	$\lvert C\rvert$
Blobs 10D (synthetic)	10	5000	10
Blobs 30D (synthetic)	30	5000	10
Blobs 100D (synthetic)	100	5000	10
HAR [2]	561	5000	6
MNIST [27]	784	5000	10

Table 1. Datasets used for ID estimation. For each dataset, we list the provenance, dimensionality n, sample count N, and class count $|C|$. Table taken from [50].

Error of the Inverse Projection. We measure the quality of an inverse projection P^{-1} for a given dataset D and its projection $P(D)$ by the mean squared error (MSE) of the backprojection $D' = P^{-1}(P(D))$ which is defined as

$$MSE = \frac{1}{|D|} \sum_{\mathbf{x} \in D} \|\mathbf{x} - P^{-1}(P(\mathbf{x}))\|^2. \quad (4)$$

As explained in Sec. 2, an ideal inverse projection P^{-1} should yield $P^{-1}(P(\mathbf{x})) = \mathbf{x}$ for all $\mathbf{x} \in D$, *i.e.*, have zero MSE. Conversely, if this error is large, then the inverse projection is likely poor and will lead to meaningless decision maps.

Intrinsic Dimensionality Estimation. Let X be a dataset in $\mathbb{R}^n$ with $S(\mathbf{x})$ being its k nearest neighbors in X. Let $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \dots, \lambda_n)$ be the n eigenvalues of $S(\mathbf{x})$'s covariance matrix, sorted decreasingly. Wang *et al.* [50] proposed to define the ID of $S(\mathbf{x})$ as the smallest d value so that the sum of the first d eigenvalues is larger than a given threshold θ, where θ was set to a value close to 1, specifically 0.95 in their experiments. This method is also known under the name *total variance* [46]. When using the total variance method for computing d_i, we found that, in the case of iLAMP (an inverse projection method they didn't study but we do), sometimes the first two eigenvalues capture a significant portion of the variance (*e.g.,* 85%); however, to arrive at 95%, one would need a large number of additional eigenvalues (*e.g.*, over 500 on MNIST dataset), each contributing less than 1% to the total variance. Obviously, this is not desirable, as it would highly overestimate the intrinsic dimensionality. To cope with this, we adopted the alternative definition of intrinsic dimensionality known as *minimal variance* which solves precisely this problem [46] – that is, we define ID as the number of eigenvalues each accounting for at least θ percent of the data variance, where θ is set typically to a small value.

Algorithm 1 shows our computation of ID values. We set $\theta = 0.01$, thereby identifying the principal components that capture more than 1% of the total variance as significant for the intrinsic dimensionality. The size k of the local neighborhood $S(\mathbf{x})$ needs careful setting. A too large k leads to overestimating the local ID. Conversely, too small k values lead to noisy estimations. Note that $d+1$ independent vectors are required to span d dimensions, so k should be at least equal to the actual ID of $S(\mathbf{x})$ [47]. We have ID estimations ranging from 13 to 33 for MNIST [21, 3, 5]; and from 15 to 61 for HAR, depending on the estimation method [14]; our synthetic datasets have known ID values of 10, 30, and 100 (see Tab. 1). To cover all the above cases, we globally set $k = 120$.

Algorithm 1: Intrinsic Dimensionality Estimation

```
Data: $X$, set of data points in $\mathbb{R}^n$ (can be $D$, $D'$, or $I^{-1}$); neighborhood size
      $k = 120$; threshold $\theta = 0.01$
Result: $\bar{d}$, the estimated ID of $X$ (average among all local neighborhoods)
1 begin
2     for $\mathbf{x} \in X$ do
3         Find the $k$ nearest neighbors $S(\mathbf{x})$ of $\mathbf{x}_i$ in $X$;
4         Compute the covariance matrix $\mathbf{Cov}$ of $S(\mathbf{x})$;
5         Compute the eigenvalues $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_n)$ of $\mathbf{Cov}$;
6         Sort $\boldsymbol{\lambda}$ in descending order;
7         Calculate ID $d(\mathbf{x})$ of $S(\mathbf{x})$ as
              $d(\mathbf{x}) = \left| \left\{ \frac{\lambda_i}{\sum_{i=1}^{n} \lambda_i} \geq \theta, 1 \leq i \leq n \right\} \right|$;
8     Calculate average ID $\bar{d} = \sum_{\mathbf{x} \in X} d(\mathbf{x}) / |X|$;
```

We perform two different ID estimations, as follows. First, for a given dataset D and its 2D projection $P(D)$, we compute the average ID of the backprojection $D' = P^{-1}(P(D))$ over all neighborhoods $S(\mathbf{x})$, denoted $ID_{D'}$. We then compare $ID_{D'}$ with the ground-truth average ID of D, denoted ID_D. Both ID_D and $ID_{D'}$ are computed using Alg. 1 with D and D' as inputs, respectively. For an ideal inverse projection P^{-1} that perfectly reverses the effects of a direct projection P on D, we would obtain $ID_{D'} = ID_D$. Secondly, to study how well a decision map covers the data space it aims to depict (see Q3, Sec. 1), we create a pixel grid I of size 500^2 and backproject it by P^{-1} to obtain a sample set I^{-1}. We next measure the ID at each sample $\mathbf{p} \in I^{-1}$ using Alg. 1 with I^{-1} as input. Let the resulting value at $\mathbf{p}$ be called ID_p. Finally, we color the image I by the values ID_p and also compute the average $\overline{ID_p}$ over all pixels in I.

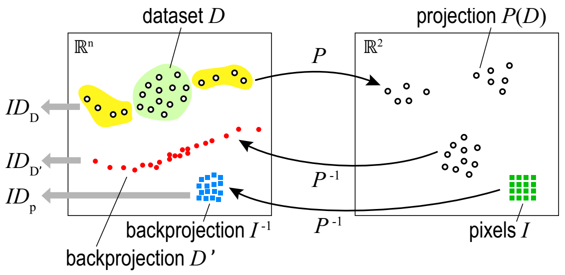


Fig. 4. Computing the intrinsic dimensionality of data, backprojection of data, and backprojection of pixels. Image taken from [50].

Figure 4 depicts all the above processes: Given a dataset D, we compute its 2D projection $P(D)$. We inversely project these points via P^{-1} to get the backprojection D'. Separately, we inversely project all pixels in the image I to get the sample set I^{-1}. In this example, the intrinsic dimensionality ID_D is the same to $ID_{D'}$ for the yellow areas in D; and higher than $ID_{D'}$ for the green areas in D, respectively.

Gradient Maps. To study the smoothness of the computed decision maps, we use the gradient map technique [18], which works as follows. Since I^{-1} is a function of two variables (the x and y coordinates of $\mathbb{R}^2$), we can estimate its gradient magnitude $\|\nabla I^{-1}\| = \sqrt{(\partial I^{-1}/\partial x)^2 + (\partial I^{-1}/\partial y)^2}$ using central differences as

$$\frac{\partial I^{-1}}{\partial x}(\mathbf{p}) \simeq \frac{P^{-1}(\mathbf{p}+(w,0)) - P^{-1}(\mathbf{p}-(w,0))}{2},$$

$$\frac{\partial I^{-1}}{\partial y}(\mathbf{p}) \simeq \frac{P^{-1}(\mathbf{p}+(0,w)) - P^{-1}(\mathbf{p}-(0,w))}{2},$$

$$G(\mathbf{p}) = \|\nabla I^{-1}(\mathbf{p})\| = \sqrt{\left(\frac{\partial I^{-1}}{\partial x}(\mathbf{p})\right)^2 + \left(\frac{\partial I^{-1}}{\partial y}(\mathbf{p})\right)^2}, \tag{5}$$

where w is a small step size, set to the size of one pixel for all our experiments. Areas in a decision map where G is large mean that neighboring pixels are backprojected by I^{-1} far away from each other in the data space, hence the map is unreliable at those locations. Conversely, areas in a decision map with low G mean that neighboring pixels sample the $\mathbb{R}^n$ space at close locations. Assuming a (relatively) smoothly evolving classifier f over $\mathbb{R}^n$, such areas will accurately capture the local behavior of f.

4.2 Results

Error Assessment. Table 2 shows the MSE results for all our datasets and decision map computation methods. Values for the iLAMP and RBF inverse projections are exactly zero since these methods enforce that $P^{-1}(P(\mathbf{x}_i)) = x_i$ for all $\mathbf{x}_i \in D$ by construction. We see that the MSEs of DBM, SDBM, and DeepView are quite low and comparable across all datasets, indicating that these inverse projections are similar and reliable. In contrast, the MSE of MDS+iMDS is significantly higher. On the synthetic datasets (Blobs), the errors of MDS+iMDS are 2 to 3 orders of magnitude higher than for the other tested methods, which may be acceptable in the limit. However, on the real-world datasets (HAR and MNIST), the errors of MDS+iMDS become much higher. This indicates that the backprojections of MDS+iMDS, and thus the corresponding decision maps, are likely meaningless. Figure 5 confirms this by running a simple test on the MNIST dataset. For 14 images $\mathbf{x}_i$ in this dataset (top row), we show the corresponding inverse projections $P^{-1}(\mathbf{x}_i)$ computed by DBM, SDBM, DeepView, and MDS+iMDS. The first three methods yield very similar images to the original ones, as expected due to the low MDS thereof. In contrast, MDS+iMDS yields basically noise.

	Blobs 10D	**Blobs 30D**	**Blobs 100D**	**HAR**	**MNIST**
DBM	1.83×10^{-3}	2.13×10^{-3}	2.16×10^{-3}	5.30×10^{-3}	3.67×10^{-2}
SDBM	2.22×10^{-3}	2.06×10^{-3}	2.16×10^{-3}	8.69×10^{-3}	5.28×10^{-2}
DeepView	1.42×10^{-3}	1.67×10^{-3}	1.90×10^{-3}	4.40×10^{-3}	2.92×10^{-2}
UMAP+iLAMP	0	0	0	0	0
UMAP+RBF	0	0	0	0	0
MDS+iMDS	1.07×10^{-1}	6.11×10^{-1}	5.71×10^{0}	5.15×10^{33}	6.19×10^{5}

Table 2. MSE of the backprojection for the studied datasets.

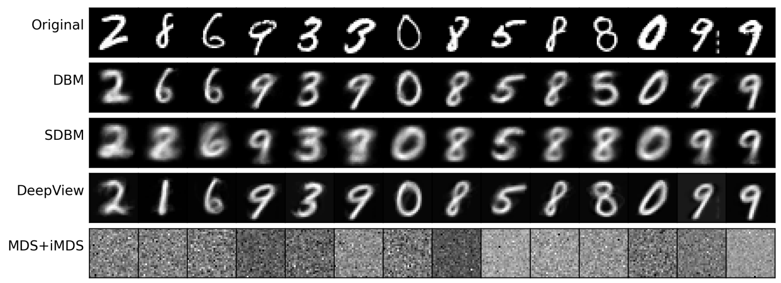


Fig. 5. Selected samples from MNIST and their corresponding backprojections using different P^{-1} methods. Compare with the MSE values in Tab. 2.

Intrinsic Dimensionality Estimation. To answer Q3, we first test how well an inverse projection P^{-1} covers the *data space* D it aims to depict. For this, we compare the estimated ID of the actual data (ID_D) with that of the round-trip consisting of the direct and inverse projections ($ID_{D'}$). As explained in Sec. 4.1, an ideal inverse projection would yield $ID_{D'} = ID_D$. Table 3 shows the results for our five studied datasets. A first consistency check is to see how good the estimated ID_D is. We see that these values align well with the expected (ground-truth) ID values for most datasets. The largest difference occurs for Blobs 100D, which is due to the fact that this dataset isotropically spreads in 100 dimensions at every blob by construction (see Sec. 4.1); the ID estimation by PCA (Alg. 1) is heavily affected by the well-known curse of dimensionality.

Given the above, we can next compare the estimated ID_D with the round-trip estimation $ID_{D'}$ to gauge the inverse projection quality (see Tab. 3). Just as for the 3D data discussed in Sec. 3.2, we see that (S)DBM creates basically a *two-dimensional*, surface-like, structure in the data space. DeepView is slightly better in capturing the ID_D of the data – which matches the fact observed for 3D datasets that its backprojected surfaces have more complex shapes that aim to connect the data samples (Fig. 3). Still, DeepView's $ID_{D'}$ values are much lower than the estimated ID_D. Note that iLAMP and RBF are not included in the comparison as they have $P^{-1}(P(\mathbf{x}_i)) = \mathbf{x}_i$ for all $\mathbf{x}_i \in D$ by construction, which means $ID_{D'} = ID_D$. Finally, for MDS+iMDS, we see that $ID_{D'}$ is much closer to the estimated ID_D than for all other methods. This may suggest that MDS+iMDS is better at capturing the data space. Yet, as observed earlier, this method has a very high MSE (Tab. 2) and also generates meaningless inverse projections (Fig. 5). As such, the high $ID_{D'}$ for this method is rather an indication of its random sampling pertaining to its implementation (see [7] for details) than its intrinsic higher quality. The authors of iMDS also noted that this inverse projection may not be effective for datasets of high intrinsic dimensionality. Our experiment here confirmed this observation.

	Blobs 10D	**Blobs 30D**	**Blobs 100D**	**HAR**	**MNIST**
Expected ID	10	30	100	15-61	13-33
ID_D	10.00	29.03	39.63	24.62	20.04
$ID_{D'}$ DBM	2.04	2.10	2.04	3.56	4.71
$ID_{D'}$ SDBM	2.23	2.14	2.11	2.09	2.47
$ID_{D'}$ DeepView	4.98	4.71	4.63	8.25	7.60
$ID_{D'}$ UMAP+iLAMP	-	-	-	-	-
$ID_{D'}$ UMAP+RBF	-	-	-	-	-
$ID_{D'}$ MDS+iMDS	10.00	22.95	37.69	11.77	28.68

Table 3. Estimated intrinsic dimensionalities ID_D and $ID_{D'}$ for our studied datasets. The expected ID values for HAR and MNIST are taken from prior studies [14, 21, 3, 5].

To further answer Q3, we want to know how well a decision map image covers the entire data space of the classifier it aims to visualize. We measure this by comparing the ID of the backprojected decision map image ID_p at each pixel (see Sec. 4.1) with the ID_D of the dataset D the classifier is trained (or tested) on. Areas where ID_p is close to ID_D indicate that the decision map covers well the local distribution of D; areas where $ID_p \ll ID_D$ indicate that the decision map can only capture a part of this local distribution.

Figures 6−8 show this comparison. In each figure, the top row shows the actual decision maps computed by our six decision map techniques for the studied Logistic Regression classifier. Colors in these images indicate the inferred class by the trained model f at each pixel; brightness encodes the confidence of f at those locations (dark values indicate low confidence); for details of this computation, see [40, 39]. These decision map images are only provided for illustration purposes *e.g.,* showing the location of decision boundaries and the data clusters; the ID analysis presented next does not depend on the classifier choice.

The second rows in Figs. 6-8 show the estimated ID_p at each decision map pixel, with the average value $\overline{ID_p}$ over the entire map shown bottom-right in the images. The results are very interesting to examine.

For DBM and SDBM, the estimated ID_p are *exactly 2* almost everywhere, which means that these decision maps precisely correspond to *surfaces* in the data space. This extends our earlier findings (Sec. 3.2) to $n > 3$ dimensions. DeepView yields higher ID_p values (but still much lower than ID_D, peaking at 10 for the HAR dataset) close to the actual data points; and values roughly equal to 2 further away from these points, with an $\overline{ID_p}$ over all datasets of 2.49 ± 0.03.

For UMAP+RBF, in areas close to the data points, the estimated ID_p is high (about the same as ID_D), see the red-colored spots in Figs. 6-8 (second rows). This is expected by the design of the RBF method, as mentioned earlier. Between the data points, the estimated ID_p for this method is exactly 2. UMAP+iLAMP shows more complicated patterns: This method also yields high ID_p values (basically equal to ID_D) close to the data points areas, as expected by construction for this method, as explained earlier. Between the data points, this method yields an estimated ID_p roughly equal to 2. However, in contrast to all other methods, UMAP+iLAMP shows strong radial-like patterns of high estimated ID_p that 'fan out' from the data points. These radial patterns in the ID_p images match similar ones in the decision maps (Figs. 6-8, first rows). We see here again an instance of iLAMP's behavior discussed in Fig. 3 for the 3-dimensional dataset case: iLAMP covers the data space better than other methods (thus answers Q3 better) but does this at the expense of continuity – that is, it produces decision maps which can be hard to interpret.

To further understand the high ID_p values for iLAMP in image areas far away from data samples, we select an area having such high values for the MNIST dataset (Fig. 8, second row, white square). We next oversample this area at a resolution of 500^2 pixels to compute ID_p. The result (Fig. 8 bottom row) shows that ID_p is actually almost 2 in such areas as well, apart from very close to the data points. Hence, the observed higher intrinsic dimensionality ID_p for iLAMP (and, actually, all other tested methods) is only an effect of the image resolution; all methods have the low ID_p values they exhibit virtually

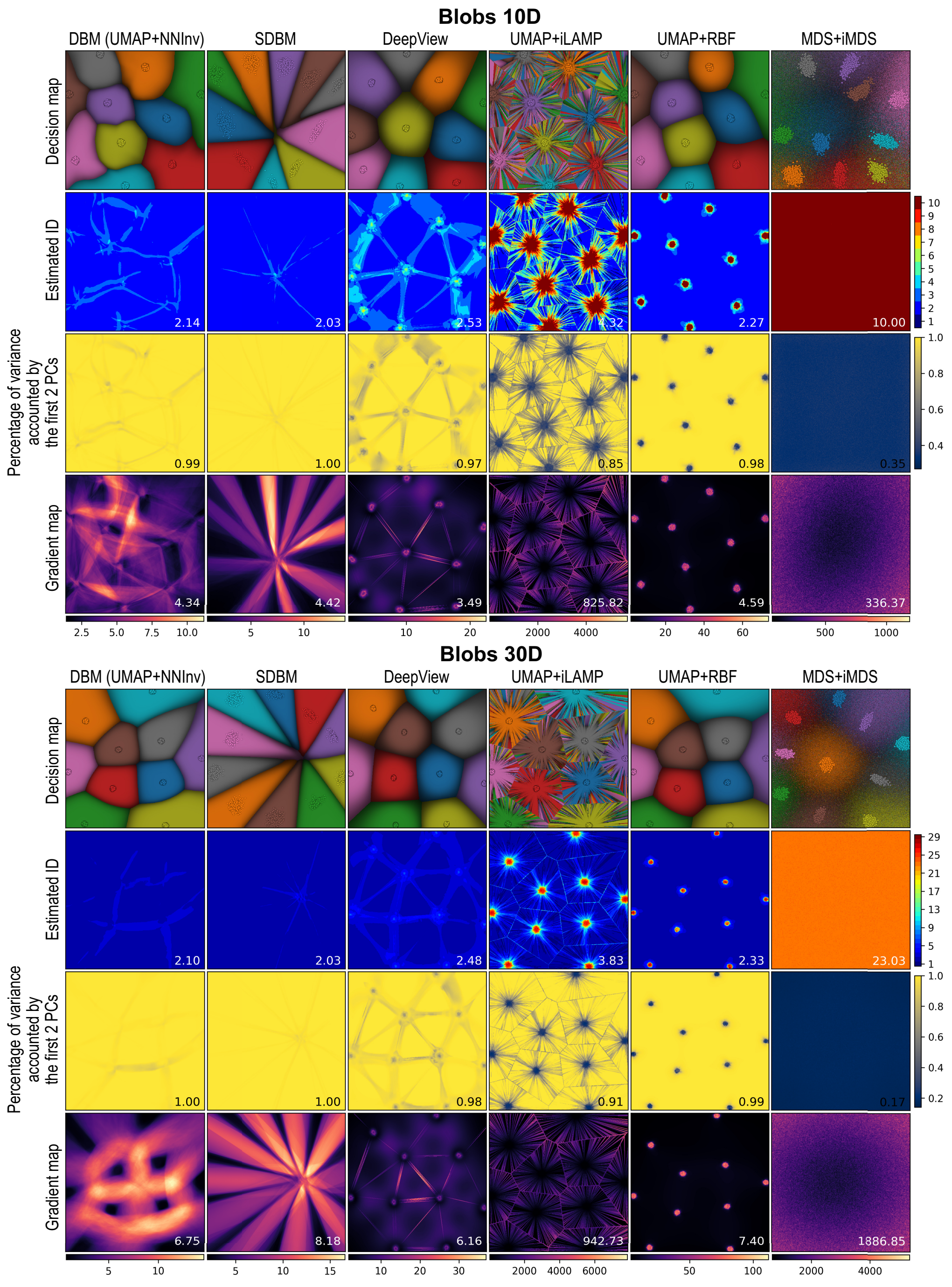


Fig. 6. Decision maps and ID estimation, Blobs 10D and 30D.

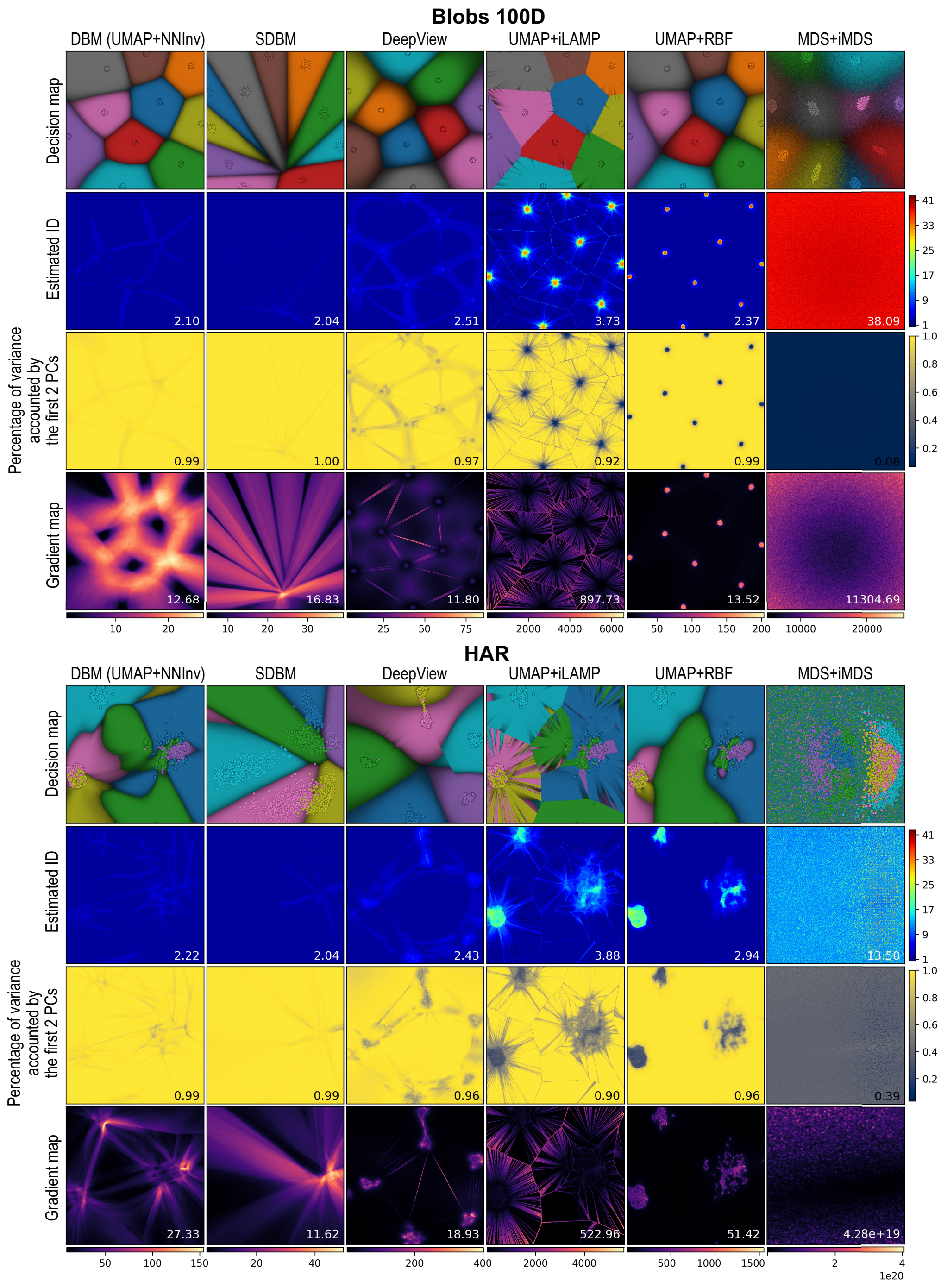


Fig. 7. Decision maps and ID estimation, Blob 100D and HAR.

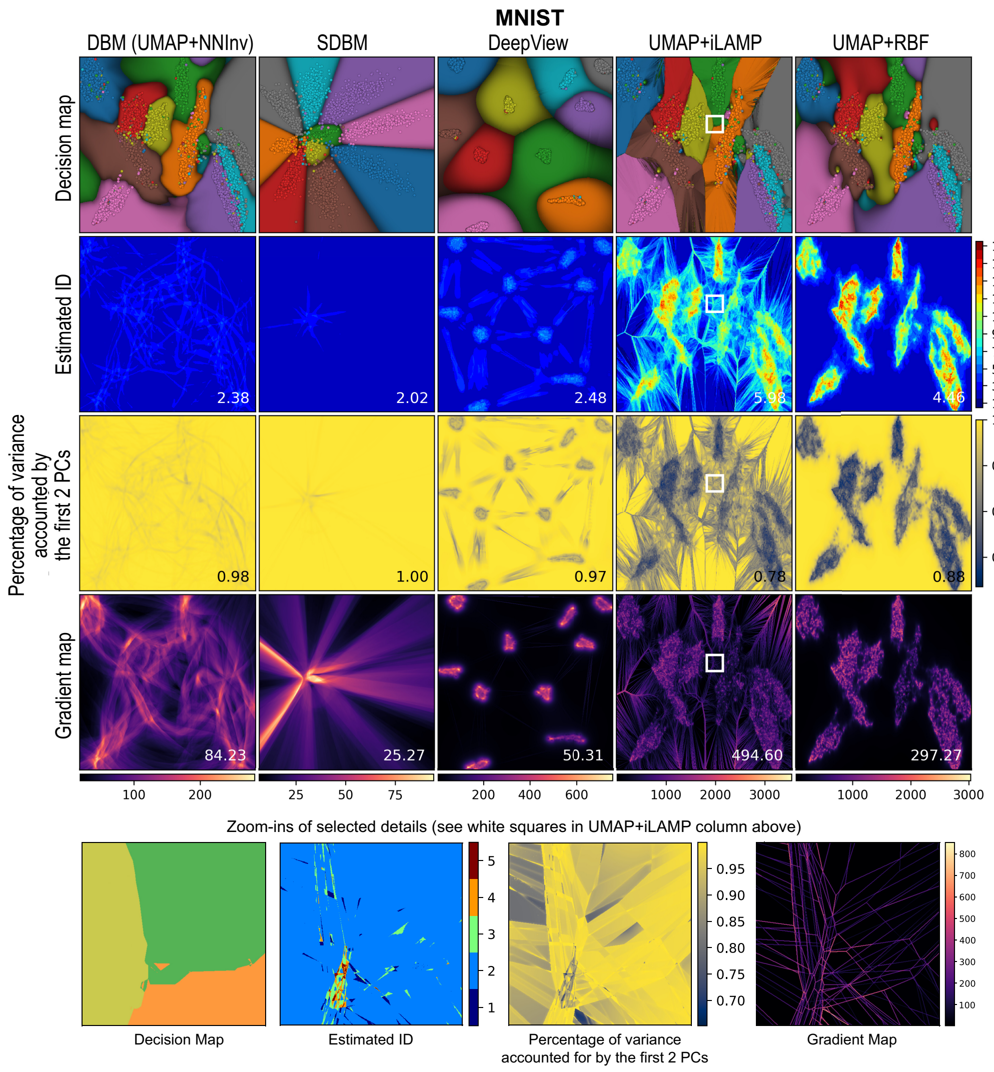


Fig. 8. Decision maps and ID estimation, MNIST Bottom images show selected detail zones sampled at a high resolution of 500^2 pixels.

everywhere except infinitesimal neighborhoods around the data points. Further, an interesting observation is that the aforementioned radial patterns seem to be less noisy as the data dimensionality increases – compare the Blobs 10D, 30D, and 100D images in Figs. 6-7. Indeed, as the data is increasingly higher dimensional, iLAMP has more difficulties to 'cover' the entire data space with a two-dimensional map, even close to the data points.

Finally, MDS+iMDS shows a quite different result: The ID_p values it produces are nearly identical over the entire image and also roughly equal to ID_D. The fact that ID_p is nearly constant matches the linear behavior of this inverse projection method that we discussed for the 3D dataset case (Sec. 3). Separately, the fact that $ID_p \simeq ID_D$ is due to the random sampling process used by iMDS, see Sec. 2. We also see that the decision maps for this method are quite dark in all areas, even in those near the actual samples. This correlates with the relatively high MSE of MDS+iMDS (Tab. 2). Intuitively put, these findings indicate that this inverse projection quickly 'goes away' from the data samples $\mathbf{x}_i$ for pixels which are not very close to the locations $P(\mathbf{x}_i)$. Again, this is due to the linear nature of iMDS – the backprojected surface I^{-1} cannot, by construction, follow the likely curved manifolds on which the samples $\mathbf{x}_i$ are spread. Separately, we see strong noise in the decision maps for this method, which is due to the aforementioned random sampling process. Again, we see here the earlier-mentioned trade-off between coverage and continuity.

As iMDS has a global linear behavior, we can study it in further detail as follows. We compute the covariance matrix for the whole set of backprojected points I^{-1} and then analyze its eigenvalues $\lambda_1, \ldots, \lambda_{100}$ (Fig.9). We observe that there is always a clear drop from the second to the third eigenvalue, indicating that the backprojected points are also dominated by a 2D planar-like structure. This matches the visual observation for the 3D dataset shown in Fig. 3. Hence, the earlier discussed fact that ID_p is overall high (Figs. 6-7, second rows) is purely due to the random sampling of iMDS. Separately, we see that as the dimensionality of the data increases, this drop becomes less significant. This suggests that the structure becomes more dominated by noise as the dimensionality increases, which correlates with the fact that the MSE of MDS+iMDS is significantly higher for higher-dimensional data (Tab.2).

The third and fourth rows in Figs. 6−8 refine the above insights. The third rows show the percentage of data variance in a neighborhood S captured by the eigenvectors corresponding to the two largest eigenvalues λ_1 and λ_2. Yellow values indicate that almost all the data variance is captured by these two eigenvalues, so the inverse projection creates locally planar structures there. Dark blue values indicate that the opposite, *i.e.*, the inverse projection creates high-dimensional structures in the respective areas. The fourth rows in the figures show the gradient maps of the inverse projections. Dark values in these maps indicate low values of G (Eqn. 5), *i.e.*, areas where the backprojection changes slowly and smoothly. Bright areas indicate the converse phenomenon – rapid and potentially non-smooth changes in the backprojection. The computation details are described in Sec. 4.1.

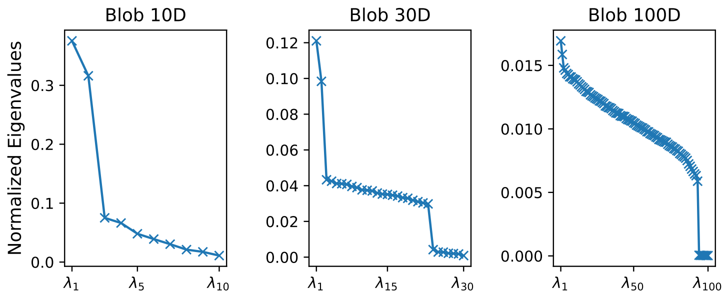


Fig. 9. Eigenvalues of the covariance matrix of the whole set of backprojected points I^{-1} for MDS+iMDS.

These visualizations lead us to additional interesting observations. First, we see that, in nearly all cases, all inverse projection methods except MDS+iMDS create large yellow areas far away from the data points (third rows in Fig. 6−8) – that is, they essentially create two-dimensional surface-like backprojections I^{-1}. In contrast, MDS+iMDS shows dark blue values nearly everywhere in these images, *i.e.*, it creates nearly everywhere a high-dimensional sampling of the data space. As explained earlier, this is due to the random sampling process inherent to this method.

A second observation pertains to the presence of 1D dark filament-like linear structures that connect the projected data points which we notice for DeepView and UMAP+iLAMP. These filaments seem to connect the projected points much like a Delaunay triangulation. These structures match quite well high values in the corresponding gradient maps. Taken together, these findings indicate that the backprojections of DeepView and UMAP+iLAMP consist of a set of planar-like facets, separated by sharp creases or gaps. This generalizes our earlier findings on the 'crumpled' aspect of these backprojections, observed for 3D datasets (Fig. 3a) to higher dimensions.

The gradient maps allow us to draw some other insights on the behavior of the inverse projections. For (S)DBM, these maps have high values that align quite well with the corresponding decision boundaries shown in Figs. 6−8, first rows. In contrast, UMAP+RBF has high gradients systematically close to the projected data samples only. UMAP+iLAMP shows an almost complementary behavior to UMAP+RBF, that is, high gradient values on the aforementioned filaments connecting the projected data points and relatively low gradient values close to the data points. Overall, these insights tell that the studied inverse projection methods have very different smoothness behaviors: (S)DBM is relatively smooth overall except close to the decision boundaries; UMAP+RBF is also quite smooth except close to the data samples; and UMAP+iLAMP is overall smooth except close to lines that connect neighboring data samples. All these findings match our earlier observations in the visual study of these backprojections for the 3D dataset case (Fig. 3a).

5 DISCUSSION

We next discuss our findings on the interpretation, added value, and found limitations of decision maps and their accompanying inverse projections, and summarize our answers to the questions Q1-Q6 listed in Sec. 1.

5.1 Surface behavior of inverse projections and decision maps

All six inverse projection pipelines we studied essentially generate surface-like structures embedded in the high-dimensional data space, with some local differences. (S)DBM tends to create relatively smooth and compact surfaces that closely interpolate the data samples. UMAP+RBF does the same but passes exactly through the data samples while being slightly less smooth. DeepView and UMAP+iLAMP create highly twisted surfaces with a similar type of trade-off, *i.e.*, DeepView interpolates the data points less accurately but yields smoother surfaces, while UMAP+iLAMP interpolates the data points exactly but yields very non-smooth results. Finally, MDS+iMDS yields a structure which formally speaking has higher intrinsic dimensionality than a surface, upon closer examination, we see that this structure is essentially a plane jittered by high amounts of noise (Q1, Q4, Q5). We also saw that this surface-like property does not depend on the intrinsic or total dimensionality of the studied datasets (Q1), the studied classifiers (Q1), or resolutions of the decision map images (see Figs. 5).

5.2 Coverage of decision maps

Given the aforementioned surface property, we conclude that current decision maps only depict a *small* part of the behavior of a given classifier (Q3). The only (relative) exception here is MDS+iMDS which succeeds in covering a higher proportion of the data space. However, this is done by using a random sampling mechanism which leads to high inverse projection errors (Tab. 2) and noisy results in both the inverse projections (Fig. 5) and decision maps (Figs. 6, 7). We conclude that this method is not suitable for creating general-purpose inverse projections and decision maps for high-dimensional data.

The boundaries shown by the studied decision maps (1D curves separating same-color regions in *e.g.* Fig. 3) are actually the intersections of the aforementioned surfaces with the actual decision boundaries in high-dimensional space (Q2). Intuitively put, a decision map thus shows a 'slice' through the high dimensional data space – its pixels are located on the aforementioned 2D surface; and its decision boundaries are 1D curve subsets of the actual decision surfaces. It is tempting to argue that, since inverse projections take a 2D space as input, they will always produce also a 2D surface as output and not a higher-dimensional object. Yet, this does not need to be so. Space-filling curves [37] and space-filling surfaces [36] can map low-dimensional sets to higher-dimensional ones in a continuous fashion. By combining such primitives, we could in principle create continuous mappings of intervals between any two dimensions q and n, $q < n$. Our study – in particular, the ID and gradient map estimations – showed that all evaluated inverse projections (DBM, SDBM, DeepView, iLAMP, RBF, iMDS) do not even get close to such behavior – which can be explained by the fact that they are constructed by differentiable mappings which cannot in principle exhibit fractal behavior. iMDS has the highest coverage but, as we saw, this is achieved by random sampling, which completely loses continuity.

5.3 Comparing decision map methods

Different decision map techniques sample the high-dimensional space quite differently (Q4). As such, they produce different maps for the *same* classifier (which, obviously, has a unique set of actual decision surfaces). Each such map provides its own insights for the same classifier (see *e.g.* Figs. 3, 6, 7, 8), each with its own advantages and limitations. At a global level, we see a clear trade-off between *smoothness* and *precision* (Q6). Methods that generate the smoothest surfaces (DBM, SDBM) cannot approximate very well the data samples. Conversely, methods that pass very close or exactly through the data samples (DeepView, UMAP+iLAMP) generate non-smooth surfaces. UMAP+RBF falls somewhere in the middle of these two types. These aspects affect in turn the *interpretability* and ultimately *usability* of the corresponding decision maps. Smooth-surface methods yield maps which are easier to interpret and show better how a classifier *extrapolates* from its training set but are harder to control in terms of *where* they are actually constructed; tighter-surface methods approximate data samples better and, for the case of DeepView, are also easier to control in terms of where they sample the data space. However, they only *interpolate* the classifier behavior close to and between the training points, and can create decision maps which are hard to interpret (UMAP+iLAMP). Summarizing the above, we believe that smooth-surface methods are overall preferred to tight-surface ones – they ultimately yield decision maps which are easier to interpret at the small cost of not perfectly approximating the data samples.

As a separate point, we note that none of the studied techniques aims to explicitly sample a classifier close to its *actual* decision boundaries – which, arguably, are the most interesting areas to understand (Q2). For this task, new inverse projections and/or decision map methods need to be devised.

5.4 Limitations caused by the low dimensionality of decision maps

Inverse projection tasks are structurally similar to data reconstruction or data generation tasks – all of these aim to output high-dimensional data from low-dimensional representations. From the perspective of data reconstruction, the projection and inverse projection pipeline (P, P^{-1}) can be seen as a special case of an encoder-decoder structure, where the bottleneck, or latent space, is two-dimensional. Existing works show that the dimensionality of the latent space, which is analogous to the input of P^{-1}, is a critical factor for the reconstruction or generation quality [48, 30, 35].

Inspired by these findings, we wonder how the dimensionality q of the latent representation affects the quality (MSE and ID) of an inverse projection. To explore this, we ran DBM (UMAP+NNInv) and SDBM (SSNP) with q values in the range 2 to 25. We chose these methods since they are easily modifiable to use a different latent dimensionality than 2 and also since, following our earlier results, they seem to offer a good balance in terms of desirable properties of inverse projections. For each q value, we recorded the $ID_{D'}$ and MSE of the inverse projection again on D. The results, shown in Fig. 10, reveal that, although DBM and SDBM exhibit similar surface behavior, their outcomes differ. The MSE of both methods decreases with q increasing, which is expected – a higher q is closer to the data dimensionality n, so both P and P^{-1} have an easier task. This drop in MSE is however more pronounced for DBM. The $ID_{D'}$ of both methods increases until reaching a plateau around 10-20 (for DBM) and 5 (for SSNP). This tells that the inverse projection task is *fundamentally* harder than the direct projection – indeed, even when having a much higher number of dimensions q than two as input for P^{-1}, it is not always possible to fully recover the full dimensionality n of the data.

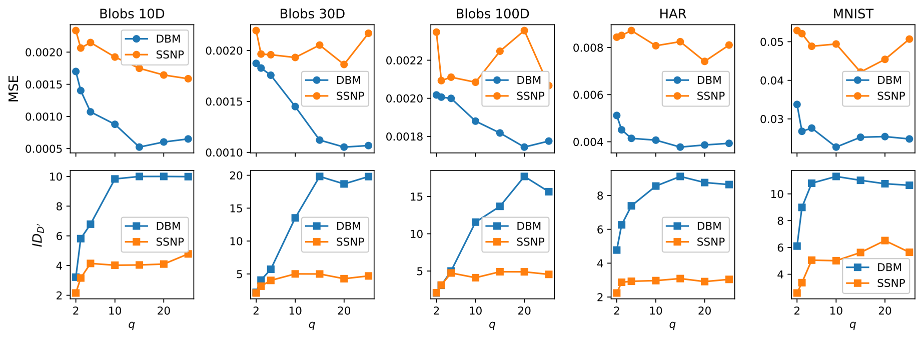


Fig. 10. How MSE and $ID_{D'}$ change with p changes for DBM (UMAP+NNinv) and SDBM (SSNP).

From this experiment, we infer that the 2D bottleneck of a (P, P^{-1}) transformation strongly affects the quality (error and ID) of the inverse projections. Increasing the number of dimensions q available as input for inverse projections can increase the quality of their output, but only up to a given limit. Moreover, from a practical viewpoint, increasing q is not evident – after all, we need to have $q \in \{2, 3\}$ if we want to use direct visualization of the decision maps. Exploring how this barrier can be overcome, *e.g.*, by more sophisticated inverse projection methods, are an important direction for future work.

5.5 Limitations

Our results are limited in their power by several factors. We used only two real-world datasets. Datasets having different characteristics, *e.g.*, local intrinsic dimensionality, data distribution, sparsity, or dimensionality, could potentially lead to new insights on how the tested decision maps and inverse projections work. A challenge here is to find datasets having ground-truth estimations of their intrinsic dimensionality. Separately, apart from the technical estimation of MSE, ID estimation, and gradient maps, we gauged the *suitability* of decision maps to practical applications only by qualitatively interpreting the visual smoothness of the resulting decision zones and boundaries. It is expected that, for the tested datasets and classifiers, such zones and boundaries should be smooth [33]. A more powerful ranking of decision maps would need to consider their actual use in ML engineering scenarios such as data augmentation or adversarial attacks, see *e.g.* [29].

6 CONCLUSIONS

We have analyzed the limitations of current inverse projection and decision map techniques used to visualize the behavior of machine learning classification models. Specifically, we compared the decision zones and boundaries depicted by six inverse projection techniques (and corresponding decision maps), with the actual zones and boundaries created by six classifiers on a three-dimensional real-world dataset. We found out that, in all cases, all the studied maps essentially capture a 2D structure embedded in the data space. We further extended our analysis to high-dimensional data by comparing the intrinsic dimensionality of the data with that of the inverse projection and backprojection of the map to the data space. We found that, as for the 3D data case, all studied map techniques still only cover essentially two-dimensional structures in the data space (modulo a certain amount of noise). We found that this surface-like limitation is particularly visible in areas located between the projected data points. Apart from this common aspect, we found several differences between the studied methods in terms of smoothness of the generated surface and accuracy by which it approximates the data points. Our work extends the earlier study on the same topic [50] by studying three additional techniques (iLAMP, RBF, and iMDS); correlating the MSE of the inverse projection with its exhibited behavior; and analyzing the inverse projection smoothness using gradient maps.

Our conclusion is that, when selecting inverse projection methods for constructing decision maps, methods which create smoother surfaces, *i.e.*, DBM, SDBM, and UMAP+RBF, are preferred in terms of predictability and ease of interpretation of the resulting maps, to methods that create tighter-fitting, less smooth, surfaces, and thus harder to interpret decision maps, *i.e.*, UMAP+iLAMP and DeepView. Finally, we showed that the recent MDS+iMDS inverse projection method is not suitable for constructing meaningful decision maps. Our work highlights fundamental limitations of all studied decision map techniques in terms of how much of a classifier's behavior they capture, but also where and how they choose to capture this behavior. These limitations are essential to understand when choosing which such technique to use in practice to construct decision maps but also when actually interpreting the resulting maps.

Future work can advance in a number of directions. The key one, we believe, is overcoming the 'surface limitation' of current decision map techniques. Likely, capturing a full high-dimensional space in a 2D map is not possible in general. Rather, one can focus on capturing specific areas in this space which are important to ML engineering, such as low-dimensional (curved) subspaces which contain most of a given dataset; areas close to the actual decision zones, where a classifier is most interesting to study; or areas where a classifier exhibits poor testing performance. An alternative way is to involve interacting allowing the user to move the current backprojected surfaces so as to sweep interesting zones of the data space, by *e.g.* generalizing the approach of Sohns *et al.* [44], which interactively explores decision boundaries by the simple but limited PCA projection. Last but not least, acceleration techniques, in the spirit of [23], can be further designed to compute decision maps at interactive rates, a prerequisite to the interactive exploration mentioned above.

References

1. Amorim, E., Vital Brazil, E., Mena-Chalco, J., Velho, L., Nonato, L.G., Samavati, F., Costa Sousa, M.: Facing the high-dimensions: Inverse projection with radial basis functions. Computers & Graphics **48**, 35–47 (2015)
2. Anguita, D., Ghio, A., Oneto, L., Parra, X., Reyes-Ortiz, J.L.: Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In: Proc. Intl. Workshop on ambient assisted living. pp. 216–223. Springer (2012)
3. Aumüller, M., Ceccarello, M.: The Role of Local Intrinsic Dimensionality in Benchmarking Nearest Neighbor Search (2019), arXiv:1907.07387 [cs]
4. Bac, J., Mirkes, E.M., Gorban, A.N., Tyukin, I., Zinovyev, A.: Scikit-Dimension: A Python Package for Intrinsic Dimension Estimation. Entropy **23**(10), 1368 (2021)
5. Bahadur, N., Paffenroth, R.: Dimension Estimation Using Autoencoders (2019), arXiv:1909.10702 [cs, stat]
6. Bennett, R.: The intrinsic dimensionality of signal collections. IEEE Trans. Inform. Theory **15**(5), 517–525 (1969)
7. Blumberg, D., Wang, Y., Telea, A., Keim, D.A., Dennig, F.L.: Inverting Multidimensional Scaling Projections Using Data Point Multilateration. In: EuroVis workshop on visual analytics (EuroVA). The Eurographics Association (2024)
8. Borg, I., Groenen, P.J.F.: Modern multidimensional scaling: theory and applications. Springer series in statistics, Springer, 2nd ed edn. (2005)
9. Breiman, L.: Random Forests. Mach. Learn. **45**(1), 5–32 (2001)
10. Camastra, F.: Data dimensionality estimation methods: a survey. Pattern Recognit. **36**(12), 2945–2954 (2003)
11. Campadelli, P., Casiraghi, E., Ceruti, C., Rozza, A.: Intrinsic dimension estimation: Relevant techniques and a benchmark framework. Math. Probl. Eng. **2015**, 1–21 (2015)
12. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
13. Cox, D.R.: Two further applications of a model for binary regression. Biometrika **45**(3/4), 562–565 (1958)
14. El Moudden, I., El Bernoussi, S., Benyacoub, B.: Modeling human activity recognition by dimensionality reduction approach. In: Proc. IBIMA. pp. 1800–1805 (2016)

15. Engel, D., Hüttenberger, L., Hamann, B.: A survey of dimension reduction methods for high-dimensional data analysis and visualization. In: Proc. IRTG workshop. vol. 27, pp. 135–149. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik (2012)
16. Espadoto, M., Martins, R., Kerren, A., Hirata, N., Telea, A.: Toward a quantitative survey of dimension reduction techniques. IEEE TVCG **27**(3), 2153–2173 (2019)
17. Espadoto, M., Rodrigues, F.C.M., Telea, A.: Visual analytics of multidimensional projections for constructing classifier decision boundary maps. In: Proc. IVAPP. SCITEPRESS (2019)
18. Espadoto, M., Appleby, G., Suh, A., Cashman, D., Li, M., Scheidegger, C.E., Anderson, E.W., Chang, R., Telea, A.C.: UnProjection: Leveraging Inverse-Projections for Visual Analytics of High-Dimensional Data. IEEE TVCG pp. 1–1 (2021)
19. Espadoto, M., Hirata, N., Telea, A.: Self-supervised Dimensionality Reduction with Neural Networks and Pseudo-labeling. In: Proc. IVAPP. pp. 27–37. SciTePress (2021)
20. Espadoto, M., Rodrigues, F.C.M., Hirata, N.S.T., Hirata Jr, R.: Deep Learning Inverse Multidimensional Projections. In: Proc. EuroVA. p. 5 (2019)
21. Facco, E., d'Errico, M., Rodriguez, A., Laio, A.: Estimating the intrinsic dimension of datasets by a minimal neighborhood information. Sci Rep **7**(1), 12140 (2017)
22. Fisher, R.A.: Iris Plants Database (1988), uCI Machine Learning Repository
23. Grosu, C., Wang, Y., Telea, A.: Computing fast and accurate decision boundary maps. In: EuroVis workshop on visual analytics (EuroVA). The Eurographics Association (2024)
24. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
25. Huang, X., Wu, L., Ye, Y.: A review on dimensionality reduction techniques. Int. J. Pattern Recognit. Artif. Intell. **33**(10), 1950017 (2019)
26. Joia, P., Coimbra, D., Cuminato, J.A., Paulovich, F.V., Nonato, L.G.: Local affine multidimensional projection. IEEE TVCG **17**(12), 2563–2571 (2011)
27. LeCun, Y., Cortes, C., Burges, C.: MNIST handwritten digit database (2010), http://yann.lecun.com/exdb/mnist
28. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**(11), 2579–2605 (2008)
29. Machado, A., Behrisch, M., Telea, A.: Exploring classifiers with differentiable decision boundary maps. Computer Graphics Forum (2024)
30. Marin, I., Gotovac, S., Russo, M., Božić-Štulić, D.: The effect of latent space dimension on the quality of synthesized human face images. Journal of Communications Software and Systems **17**(2), 124–133 (2021)
31. McInnes, L., Healy, J., Melville, J.: UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction (2018), arXiv:1802.03426 [cs, stat]
32. Nonato, L., Aupetit, M.: Multidimensional projection for visual analytics: Linking techniques with distortions, tasks, and layout enrichment. IEEE TVCG **25**, 2650–2673 (2018)
33. Oliveira, A.A.A.M., Espadoto, M., Hirata, R., Telea, A.C.: Stability Analysis of Supervised Decision Boundary Maps. SN COMPUT. SCI. **4**(3), 226 (2023)
34. Oliveira, A.A.A.M., Espadoto, M., Hirata Jr, R., Telea, A.C.: SDBM: Supervised Decision Boundary Maps for Machine Learning Classifiers. In: Proc. IVAPP. pp. 77–87 (2022)
35. Padala, M., Das, D., Gujar, S.: Effect of Input Noise Dimension in GANs. In: Neural Information Processing. pp. 558–569. Lecture Notes in Computer Science, Springer International Publishing (2021)

36. Paulsen, W.: A Peano-based space-filling surface of fractal dimension three. Chaos, Solitons & Fractals **168** (2023)
37. Peano, G.: Sur une courbe, qui remplit toute une aire plane. Mathematische Annalen **36**(1), 157–160 (1890)
38. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V.: Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
39. Rodrigues, F.C.M., Espadoto, M., Hirata, R., Telea, A.C.: Constructing and Visualizing High-Quality Classifier Decision Boundary Maps. Information **10**(9), 280 (2019)
40. Rodrigues, F.C.M., Hirata, R., Telea, A.C.: Image-based visualization of classifier decision boundaries. In: Proc. SIBGRAPI. pp. 353–360. IEEE (2018)
41. dos Santos Amorim, E.P., Brazil, E.V., Daniels, J., Joia, P., Nonato, L.G., Sousa, M.C.: iLAMP: Exploring high-dimensional spacing through backward multidimensional projection. In: Proc. IEEE VAST. pp. 53–62 (2012)
42. Schulz, A., Gisbrecht, A., Hammer, B.: Using discriminative dimensionality reduction to visualize classifiers. Neural Process. Lett. **42**, 27–54 (2015)
43. Schulz, A., Hinder, F., Hammer, B.: DeepView: Visualizing Classification Boundaries of Deep Neural Networks as Scatter Plots Using Discriminative Dimensionality Reduction. In: Proc. IJCAI. pp. 2305–2311 (2020)
44. Sohns, J.T., Garth, C., Leitte, H.: Decision Boundary Visualization for Counterfactual Reasoning. Comput. Graph. Forum **42**(1), 7–20 (2023)
45. Sorzano, C.O.S., Vargas, J., Montano, A.P.: A survey of dimensionality reduction techniques (2014), arXiv:1403.2877 [stat.ML]
46. Tian, Z., Zhai, X., van Driel, D., van Steenpaal, G., Espadoto, M., Telea, A.: Using multiple attribute-based explanations of multidimensional projections to explore high-dimensional data. Comput. Graph. **98**, 93–104 (2021)
47. Verveer, P.J., Duin, R.P.W.: An evaluation of intrinsic dimensionality estimators. IEEE PAMI **17**(1), 81–86 (1995)
48. Wang, Y., Yao, H., Zhao, S.: Auto-encoder based dimensionality reduction. Neurocomputing **184**, 232–242 (2016)
49. Wang, Y., Machado, A., Telea, A.: Quantitative and Qualitative Comparison of Decision-Map Techniques for Explaining Classification Models. Algorithms **16**(9), 438 (2023)
50. Wang, Y., Telea, A.: Fundamental Limitations of Inverse Projections and Decision Maps. In: Proc. IVAPP. vol. 1, pp. 571–582 (2024)