EuroVis Workshop on Visual Analytics (2023)
M. Angelini and M. El-Assady (Editors)

ShaRP: Shape-Regularized Multidimensional Projections

A. Machado[1], A. Telea[1], M. Behrisch[1]

[1] Department of Information and Computing Science, Utrecht University, Netherlands

Abstract

Projections, or dimensionality reduction methods, are techniques of choice for the visual exploration of high-dimensional data. Many such techniques exist, each one of them having a distinct visual signature — i.e.*, a recognizable way to arrange points in the resulting scatterplot. Such signatures are implicit consequences of algorithm design, such as whether the method focuses on local* vs *global data pattern preservation; optimization techniques; and hyperparameter settings. We present a novel projection technique — ShaRP — that provides users explicit control over the visual signature of the created scatterplot, which can cater better to interactive visualization scenarios. ShaRP scales well with dimensionality and dataset size, generically handles any quantitative dataset, and provides this extended functionality of controlling projection shapes at a small, user-controllable cost in terms of quality metrics.*

CCS Concepts

• ***Human-centered computing*** → *Visualization techniques;* • ***Mathematics of computing*** → *Dimensionality reduction;*

1. Introduction

Projection, also called Dimensionality Reduction (DR), methods are popular tools for exploring high-dimensional datasets. They transform the task of discovering data patterns in high-dimensional spaces into a perceptually-driven search and inspection task of *visual patterns* in 2D or 3D through scatterplots. Prior research has shown that such scatterplots help uncovering topological aspects, such as groupings, outliers, and correlations in the data [BBK*18, PKF*16, WFC*18].

However, visual patterns in a projection depend not only on the underlying *data*, but also on how the DR technique is designed. For example, for the same dataset, t-SNE tends to create organic, round, structures; Auto-Encoders create starburst-like clusters; and UMAP creates very dense, round, clusters, to mention just a few [NA18, EMK*21]. We further call such aspects the *visual signature* of a projection technique.

We believe that users can benefit from having *direct* control over the visual signatures of a projection technique. For instance, when performing interactive data labeling using rectangular selections or displaying image thumbnails over data clusters (see Figure 2), a projection whose clusters resemble rectangles would be more suitable than one creating various-shaped clusters (modulo other aspects of the two projections, *e.g.*, quality). However, controlling such visual signatures is typically hard with current projection methods.

To fill that gap, we present *ShaRP* (standing for **Sha**pe **R**egularized Neural **P**rojection), to the best of our knowledge the first algorithm that provides users with direct control over cluster shapes in their projection scatterplots. We next describe the technique, illustrate this new shape regularization ability, show that it comes at a user-controlled penalty to standard quality metrics, and point towards avenues for further exploration.

2. Background and Related Work

We first introduce a few notations: A dataset $\mathbf{X} = \{\mathbf{x}_i\}_{i=1,\dots,m}$ has m samples $\mathbf{x}_i = [x_{i1}, \dots, x_{in}]^T$, where $\mathbf{x}_i$ is a point in $\mathbb{R}^n$ with components x_{ij}, $1 \leq j \leq n$ and an optional label $y_i \in \{1, \dots, K\}$. We use capitals to denote the set of all elements for the corresponding small letter, *e.g.*, $\bar{Y} = \{\bar{y}_i\}_{i=1,\dots,m}$. We denote the Euclidean norm by $\|\mathbf{x}\|_2 = \sqrt{\mathbf{x}^T\mathbf{x}}$ and the expected value of a function of a random variable $\mathbf{z}$ distributed according to p by $\mathbb{E}_{\mathbf{z}\sim p}[f(\mathbf{z})]$. Further, we use θ to denote probability distribution parameters, for example $\theta = (\vec{\mu} \in \mathbb{R}^2, \vec{\sigma}^2 \in \mathbb{R}^2)$ for a 2D Diagonal Gaussian distribution.

Dimensionality reduction: Projection algorithms are formally functions $P_\eta : \mathbb{R}^n \to \mathbb{R}^q$ where $q \ll n$ and η denote (hyper)parameters. In this work we focus on 2D scatterplots ($q = 2$) and use the term "projections" to refer to both such 2D scatterplots and the DR algorithms that create them.

Many projection algorithms are available nowadays. These are described from technical perspectives (how they differ design-wise) in several surveys [vP09, HG02, Yin07, BBH12, SVPM14, CG15, LMW*15]; and from the perspective of *local* quality metrics [NA18, EMK*21]. Well-known projection methods include Principal Component Analysis (PCA) — a simple, easy to code,

DOI: 10.2312/eurova.20231088

but qualitatively limited method especially for complex non-planar data structures embedded in high dimensions [F.R01]; Isomap — a technique which works well if the data resides on a (single) high-dimensional manifold [TSL00]; t-SNE, which works well for arbitrary high-dimensional data distributions but has challenges in controlling (and predicting) the shapes of the emerging visual clusters [vdMH08, vdM14, Uly16]; and UMAP, similar to t-SNE in terms of ease and of visual cluster control [MH18].

Visual signatures: Significant work aimed to develop ways to control or adapt the visual signatures of projections and studying whether it is *possible* to do so for existing projection methods, as follows. Cutura et al. [CMC*22] use space-filling curves to adapt the position of data points in image thumbnail scatterplots such that they are non-overlapping. This idea is effective but limited to image datasets, while our proposed technique is designed to be generic. The perplexity parameter in t-SNE is responsible for, among other things, the visual appearance of the projection. Its effect is intricately enmeshed within t-SNE, leading to cluster shapes, sizes, and distances that do not necessarily convey meaning [WVJ16]. Our technique minimizes the variability of this visual appearance. Since fully doing away with hyperparameters might be infeasible, techniques such as HyperNP [AEC*21] learn to simulate their effects on the resulting projection. Our technique instead works well for several datasets using a single setting of hyperparameters, reducing the need for such simulation. Makhzani *et al.* [MSJG15] propose an approach similar to ours. They adapt an auto-encoder into an adversarial setting so as to increase the quality of the projection (*e.g.*, better cluster separability) that would have been created by the vanilla auto-encoder. Our method instead aims mainly at putting cluster shapes under the user's control.

3. ShaRP: Shape-Regularized Neural Projection

We now introduce ShaRP, our novel DR technique, which is based on deep neural networks. Such networks are, in general, able to approximate complex non-linear functions and have several desirable features that ShaRP inherits:

Scalable: ShaRP scales linearly in the number of samples because it avoids precomputing pairwise distances or covariance matrices, like in PCA or t-SNE, and lends itself to hardware-acceleration through GPUs or TPUs tailored for fast deep learning.

Parametric: ShaRP operates in a "learn once, project as needed" fashion. It learns to parameterize a projection function instead of only outputting the projected points, such as t-SNE or UMAP. This allows ShaRP to project data it was not trained on along with existing data (out-of-sample ability).

Generic: ShaRP handles any dataset comprised of numeric features and can be applied to a wide range of datasets using the same or only slightly adapted hyperparameter settings.

Sound: ShaRP scores comparably to state of the art techniques in relevant projection quality metrics.

To these, ShaRP adds two flavors of **Shape Regularization**:

- *Intra-projection*: ShaRP creates point clusters having shapes coming from the same *family*: ellipses, rectangles, triangles.
- *Inter-projection*: Running ShaRP over different datasets produces a consistent visual signature where differences in the projections are driven mainly by data patterns.

ShaRP is implemented in Python using Keras (Tensorflow backend) [C*15], Tensorflow Probability [DLT*17] for sampling and calculating log-probabilities under different distributions and is publicly available at https://github.com/amreis/sharp.

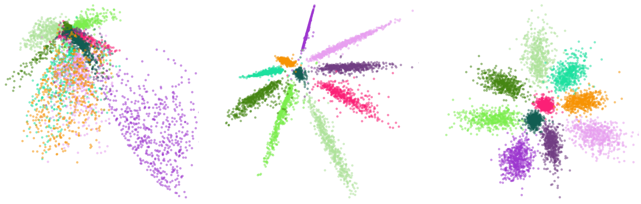
3.1. Method description

ShaRP belongs to the family of Representation Learning [BCV13] techniques, *i.e.*, it learns a *latent encoding* for input data. A latent encoding is a vector $\mathbf{r} \in \mathbb{R}^q$, where $\mathbf{r} = f(\mathbf{x})$ is a low-dimensional representation of the input $\mathbf{x} \in \mathbb{R}^n$ that enables a reconstruction of $\mathbf{x}$ with minimal errors. As said earlier, we aim to create 2D projections, so $q = 2$.

ShaRP builds atop of the recent DR method SSNP [EHT21]. SSNP extends a vanilla auto-encoder with loss $\mathcal{L}_{\text{AE}}$ with a classifier head (with an accompanying loss $\mathcal{L}_{\text{class}}$), yielding the total loss to be optimized as

$$\mathcal{L}_{\text{SSNP}}(\mathbf{X}, \hat{\mathbf{X}}, \bar{Y}, \hat{Y}) = \mathcal{L}_{\text{AE}}(\mathbf{X}, \hat{\mathbf{X}}) + \rho \mathcal{L}_{\text{class}}(\bar{Y}, \hat{Y}). \tag{1}$$

The projection $\mathbf{r}_i \in \mathbb{R}^2$ of each input $\mathbf{x}_i$ is generated by the bottleneck layer of the network. The classification loss $\mathcal{L}_{\text{class}}$, together with target labels or *pseudolabels* generated by a clustering algorithm, enables SSNP to separate data clusters better than plain autoencoders (see Figure 1). Yet, as the figure shows, SSNP collapses some clusters into elongated shapes, which we argue is (a) unnatural, as it suggests some anisotropy in the sample distribution; (b) space-inefficient, as much white space is not used to depict data; and (c) suboptimal for visualization as we cannot *e.g.* easily select a cluster by rubberband tools or annotate it with a square-like icon.



(a) *AE (0.54)* **(b)** *SSNP (0.87)* **(c)** *ShaRP (0.97)*

Figure 1: *Comparison of projections of the MNIST dataset learned using (a) Auto-encoders, (b) SSNP [EHT21], and (c) ShaRP. SSNP and ShaRP were trained using the* ground truth *labels as class information — encoded, here and next, by colors. Values in brackets are Distance Consistency scores (DSC [SNLH09]), a quality metric that measures separability of clusters, with 1 being a perfect score.*

ShaRP overcomes these shortcomings of SSNP by an explicit user-controlled shape regularization mechanism, described next (see also Section 3.2 for examples). ShaRP replaces SSNP's Auto-Encoder (AE) with a Variational AE (VAE) [KW14]. The key AE-VAE difference is the latter's use of a *sampling* process in the network's bottleneck layer. This, coupled with a necessary KL-Divergence regularization term

$$\mathcal{L}_{\text{reg}}(\theta) = D_{\text{KL}}(q_\theta || p) \doteq \mathbb{E}_{\mathbf{z} \sim q_\theta}[\log(q_\theta(\mathbf{z})/p(\mathbf{z}))] \tag{2}$$

has as an immediate effect on the regularization of the learned latent space: Using $\mathcal{L}_{\text{reg}}$ pushes the learned probability distributions q_θ toward a standard form p defined a priori (e.g., a standard Gaussian distribution) which prevents learning degenerate distributions. Also, crucially for our goals, this loss can be *exploited* to model different shape regularization constraints (see next Section 3.2). The complete loss function for ShaRP then reads as

$$\begin{aligned}\mathcal{L}_{\text{ShaRP}}(\mathbf{X},\hat{\mathbf{X}},\bar{Y},\hat{Y},\Theta) =& \mathcal{L}_{\text{recon}}(\mathbf{X},\hat{\mathbf{X}}) + \rho\mathcal{L}_{\text{class}}(\bar{Y},\hat{Y}) + \beta\mathcal{L}_{\text{reg}}(\Theta) \qquad (3)\\ =& \mathcal{L}_{\text{SSNP}}(\mathbf{X},\hat{\mathbf{X}},\bar{Y},\hat{Y}) + \beta\mathcal{L}_{\text{reg}}(\Theta),\end{aligned}$$

where we make the connection to the SSNP loss explicit.

By using a suitable *sampling* process, the clusters emerging in the projection will be shape-regularized. For instance, using a 2D Gaussian sampling distribution yields *elliptical* shapes (see Figure 1c) because the equidensity contours of a 2D Gaussian are ellipses. This is dependent on $\mathcal{L}_{\text{reg}}$ preventing the degenerate learning of low (respectively, high) variances, which would give rise to point-like (resp., line-like) shapes in the projection.

3.2. Controlling cluster shapes

We use as regularization targets the following shapes.

Ellipses. Consider a diagonal Multivariate Normal distribution, *i.e.*, $\mathbf{z}_i \sim \mathcal{N}(\vec{\mu}, \text{diag}(\vec{\sigma}^2))$. The natural prior to use here is the standard Multivariate Normal distribution $\mathcal{N}(\vec{\mathbf{0}}, \mathbf{I})$ which simplifies sampling, propagating gradients, and calculating the KL-Divergence loss [KW14]. By using this prior, we encourage learned probability distributions to be as close as possible to a standard Gaussian. Hence, the learned projection will output data clusters that resemble circles or ellipses (see Figure 4).

Using a Gaussian sampling distribution is standard for VAEs. For our projection goals, tweaking the sampling distribution and using suitable priors allows favoring different cluster shapes. We can use *any* distribution, as long as we have (i) access to log-probabilities of samples under the learned distribution and the prior; (ii) a way to propagate gradients through the sampling procedure (using a reparametrization trick or otherwise).

Access to the log-probabilities of samples under learned distributions and the prior removes the (constraining) need to analytically calculate the KL-Divergence since we can re-express it as a sample-based computation as

$$D_{\text{KL}}(q_\theta||p) \approx \frac{1}{m}\sum_{i=1}^{m}\left(\log q_\theta(\mathbf{z}_i) - \log p(\mathbf{z}_i)\right), \qquad (4)$$

where the approximation holds if $\mathbf{z}_i \sim q_\theta(\cdot)$. Our next examples of regularization shape targets use Equation 4 of computing the (approximate) KL-Divergence.

Rectangles. To create rectangular clusters, we use a generalized Normal ($\mathcal{GN}$) probability distribution. It introduces an additional shape parameter to the Gaussian (here denoted ω) and has a density function of the form

$$p(x|\mu,\alpha,\omega) \propto \exp\left(-(|x-\mu|/\alpha)^{\omega}\right).$$

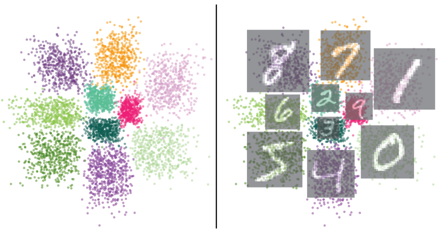
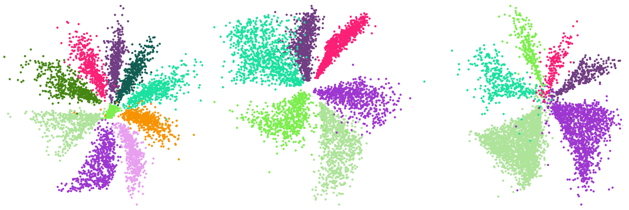


Figure 2: *Rectangular shaping can be convenient for data labeling tasks, as illustrated by the right image where class representatives are overlaid atop their clusters. We achieve this using a Generalized Normal distribution, here shown on the MNIST dataset for* $\omega = 10$ *(left).*

Tuning ω makes the tails of the distribution heavier or lighter. This is similar to the Minkowski p-norm where higher p values (analogous to ω) make sets of equidistant points approach axis-aligned squares as $p \to \infty$ instead of circles ($p = 2$). Using this distribution for sampling, with a high ω value, yields cluster shapes that resemble squares/rectangles instead of ellipses (see Figure 2).



(a) *MNIST (0.905)* **(b)** *HAR (0.965)* **(c)** *Reuters (0.959)*

Figure 3: *The results of our Triangular shaping sampling scheme over 3 different datasets. DSC values (in brackets) are close to the best value possible, indicating good class separability.*

Convex polygons. If $\mathbf{V} \in \mathbb{R}^{2\times v}$ is a matrix of a base convex polygon's v vertices in $\mathbb{R}^2$ and $\mathbf{w} \in [0,1]^v$ is a vector such that $w_i \geq 0\ \forall i, \sum_{i=1}^{v} w_i = 1$, then $\mathbf{p} = \mathbf{Vw}$ is a point inside the base polygon with barycentric coordinates w_i. To sample points inside this polygon, we use the Dirichlet probability distribution

$$\mathbf{w} \sim \text{Dir}(\alpha_1, \alpha_2, \ldots, \alpha_v) \Rightarrow \mathbf{w} \in [0,1]^v, \quad \sum_{i=1}^{v} w_i = 1 \quad (\alpha_i > 0,\ \forall i)$$

which generates vectors with the same properties as $\mathbf{w}$ above.

This sampling scheme alone is not enough to learn a useful embedding since all data points will draw samples from the same region in space. Hence, we augment this scheme with rotation, scaling, and translation. Figure 3 shows this scheme for triangles, *i.e.* $v = 3$, using as prior the "uniform" distribution on the triangle, which corresponds to $\text{Dir}(1,1,1)$.

Table 1 summarizes our proposed mechanisms for controlling cluster shapes by sampling distributions. For more technical details, see the supplemental material.

4. Evaluation

We next discuss how ShaRP gives direct control over cluster shapes while learning to project data (Section 4.1), the quality of ShaRP

Proceedings published by Eurographics - The European Association for Computer Graphics.

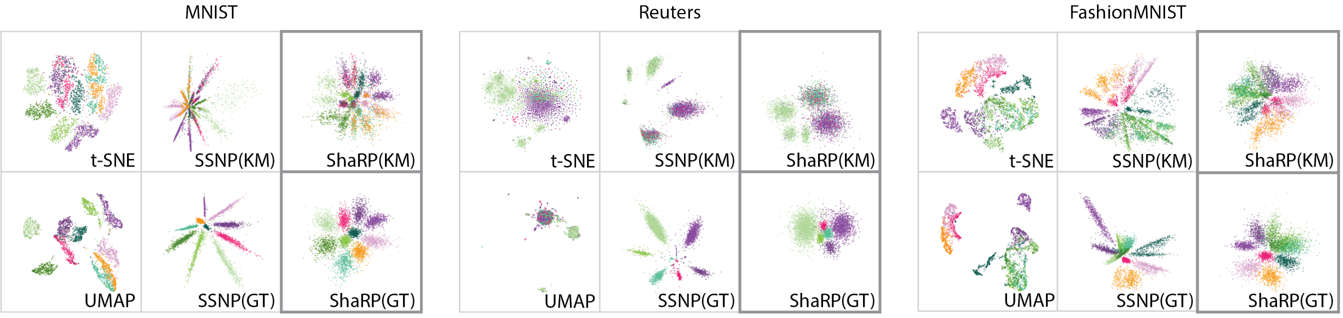


Figure 4: *Our ShaRP method produces cluster shapes regularized towards a user-chosen target — here, ellipses — and can handle diverse data distributions. We demonstrate this here for the cases where we use ground truth labels (GT) or K-Means-generated pseudolabels (KM). We compare our results to SSNP (GT, KM) and to t-SNE and UMAP. More comparisons are present in the supplemental material.*

Table 1: *Correspondences between sampling schemes and shapes.*

Sampling	Prior	Shape
$\mathbf{z} \sim \mathcal{N}(\mu, \mathrm{diag}(\sigma^2))$	$\mathcal{N}(0, I)$	○
$\mathbf{z} \sim \mathcal{GN}(\mu, \alpha, \omega)$	$\mathcal{GN}(0, 1, \omega)$	□
$\mathbf{z} \sim \mathrm{Dir}(\alpha_1, \alpha_2, \alpha_3)$ $\mathbf{z} \mapsto \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \mathbf{Vz} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$	$\mathrm{Dir}(1, 1, 1)$	△

projections (Section 4.2), and how tuning a single hyperparameter controls the shape regularization strength (Section 4.3). Finally, we discuss ShaRP's computational scalability (Section 4.4).

Datasets. We use 5 datasets for evaluation (Table 2) which have different levels of classification difficulty, dimensionality, data type (images, motion data, text), and are often used in DR evaluations [EMK*21].

Table 2: *Datasets used in our evaluation.*

Dataset	Dimensionality (n)	# classes (K)
USPS [Hul94]	256	10
HAR [AGO*12]	561	6
MNIST [LBBH98]	784	10
FashionMNIST [XRV17]	784	10
Reuters [Tho17]	5000	6

Techniques. We compare ShaRP with t-SNE, UMAP, and Isomap, due to their wide adoption in the DR arena. We also compare with Auto-Encoders since they are a key building block of our technique; with SSNP since we are extending it; and with NNP [EHT20], a technique that learns to imitate projections, here trained to imitate t-SNE. For both ShaRP and SSNP, we use three different label sources: (1) from the ground truth of the dataset (GT); and pseudolabels created by the K-Means (2, KM) [Llo82] and Agglomerative (3, AG) [KR90] clustering techniques.

Hyperparameter settings. We train ShaRP with the Adam optimizer using default parameter settings. We add L2 regularization to the bottleneck layer of the network with a coefficient of 0.5. We use $\rho = 1$ and $\beta = 0.1$ and mini-batches of 256 data points.

4.1. Generating shape-regularized projections

Figure 4 shows examples of how ShaRP can regularize projections. Instead of producing scatterplots where cluster shapes, sizes, and intercluster spacing are widely different (as with t-SNE and UMAP), ShaRP generates a more similar representation of the high-dimensional data in each 2D projection (intra-projection regularization). Also, the visual signature obtained is consistent throughout datasets (inter-projection regularization). The learned projections do well with respect to quality metrics (see Table 3 and its discussion in Section 4.2). All images were generated using the same hyperparameter values, showing the robustness of ShaRP to different datasets.

4.2. Measuring the projection quality

We evaluate ShaRP by a set of established projection quality metrics (trustworthiness, continuity, Shepard correlation, normalized stress, neighborhood hit, and distance consistency) following Espadoto et al. [EMK*21, EHT21]. Precise metric definitions are listed in the supplemental material. We compute metrics over all datasets using a Gaussian sampling layer which produces ellipse-like clusters. Table 3 shows these mean and standard deviations of the metrics over all datasets for ShaRP and the other six evaluated techniques. We see that ShaRP avoids very high Stress values (present in t-SNE, *all* studied datasets; UMAP and AEs, some datasets). We do, however, have higher Stress than SSNP, since we *force* clusters into desired shapes, which can require projected (2D) distances to be quite different from data-space distances. Given that our Stress is still lower than t-SNE, UMAP, and AE, we believe this is a reasonable trade-off. For the other metrics, ShaRP performs comparably to t-SNE and UMAP. Overall, we claim that ShaRP offers its capability of shape regularization without negatively impacting quality. It is worth noting that, for their AG and KM versions, both SSNP and ShaRP can be held back by the clustering algorithm's ability to properly group the dataset into classes.

To test how ShaRP's support of different regularization shapes affects projection quality, we asked ShaRP to produce clusters in five shapes – ellipses (using Gaussian sampling); rectangles ($\omega = 5$ and $\omega = 15$, see Section 3.2); and triangles (translated in projected space and respectively forced to $t_x = t_y = 0$), for all 5 tested datasets. Table 4 shows the mean and standard deviation of quality metrics per dataset. We see little variation in these metrics. This points to the robustness of ShaRP and further supports our claim that controlling the visual signatures of projections can be done without (strongly) influencing quality metric values.

Table 3: *Means and (standard deviations) of metrics computed for ShaRP and six other projection methods over 5 datasets. See Section 4.2. For non-aggregated results, see supplemental material.*

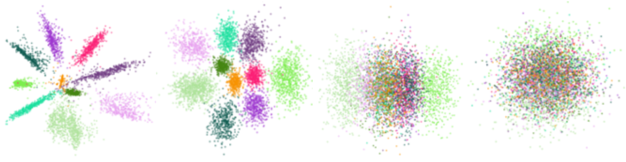
Method	Trustworthiness	Continuity	Shepard Corr.	Stress	Neigh. Hit	Dist. Consistency
Isomap	0.817 (0.126)	0.927 (0.096)	0.540 (0.410)	4.355 (4.359)	0.737 (0.091)	0.587 (0.095)
t-SNE	0.940 (0.110)	0.968 (0.032)	0.436 (0.250)	25.951 (18.373)	0.914 (0.064)	0.754 (0.141)
UMAP	0.910 (0.139)	0.960 (0.055)	0.424 (0.336)	1.389 (1.451)	0.874 (0.088)	0.741 (0.166)
NNP[t-SNE]	0.903 (0.122)	0.967 (0.033)	0.441 (0.254)	0.872 (0.044)	0.859 (0.075)	0.742 (0.141)
AE	0.878 (0.136)	0.917 (0.091)	0.342 (0.331)	1.236 (1.075)	0.793 (0.043)	0.622 (0.082)
SSNP (AG)	0.849 (0.139)	0.922 (0.080)	0.461 (0.180)	0.301 (0.067)	0.812 (0.058)	0.674 (0.090)
SSNP (KM)	0.862 (0.142)	0.928 (0.065)	0.451 (0.221)	0.334 (0.134)	0.777 (0.046)	0.620 (0.104)
SSNP (GT)	0.797 (0.126)	0.902 (0.078)	0.454 (0.117)	0.500 (0.061)	0.977 (0.025)	0.930 (0.043)
ShaRP (AG)	0.816 (0.136)	0.886 (0.108)	0.382 (0.311)	0.770 (0.089)	0.771 (0.047)	0.661 (0.073)
ShaRP (KM)	0.832 (0.143)	0.897 (0.082)	0.426 (0.268)	0.782 (0.079)	0.747 (0.045)	0.658 (0.051)
ShaRP (GT)	0.755 (0.118)	0.864 (0.098)	0.343 (0.221)	0.783 (0.085)	0.939 (0.056)	0.890 (0.074)

Table 4: *Means and (standard deviations) of metrics for ShaRP computed for 5 datasets and 5 different settings for shape regularization. See Section 4.2. For non-aggregated results, see supplemental material.*

Dataset	Trustworthiness	Continuity	Shepard Correlation	Stress	Neigh. Hit	Dist. Consistency
MNIST	0.735 (0.007)	0.878 (0.035)	0.167 (0.071)	0.809 (0.073)	0.965 (0.024)	0.909 (0.068)
FashionMNIST	0.823 (0.014)	0.888 (0.035)	0.468 (0.079)	0.837 (0.065)	0.830 (0.018)	0.761 (0.043)
HAR	0.826 (0.001)	0.841 (0.039)	0.546 (0.053)	0.690 (0.087)	0.959 (0.021)	0.925 (0.025)
Reuters	0.555 (0.002)	0.691 (0.011)	0.302 (0.050)	0.631 (0.083)	0.969 (0.007)	0.903 (0.035)
USPS	0.805 (0.006)	0.899 (0.036)	0.275 (0.105)	0.700 (0.104)	0.965 (0.012)	0.904 (0.053)

4.3. Control of shape regularization intensity

We adjust the amount of shape regularization through the β multiplier in the loss function (Equation 3). Figure 5 shows this: Larger β values force clusters to conform to the shape generated in the sampling layer — ellipses, in this case. Exaggerated shape regularization (high β), however, makes ShaRP favor 'shape over data' too much and creates projections which cannot properly depict data — sampling from a distribution similar to the prior overshadows producing a sensible embedding. In our tests, we have found a value of $\beta = 0.1$ to give consistently good results.



(a) $\beta = 0.0$ (b) $\beta = 0.1$ (c) $\beta = 0.5$ (d) $\beta = 1.0$

Figure 5: *The β coefficient (Equation 3) controls the shape regularization strength, shown here on the USPS dataset. $\beta = 0$ approximately reproduces SSNP (a). Increasing it (b, c) progressively forces the learned clusters into circular shapes, up to the point where the projection is of low quality (d).*

β =	0.0	0.05	0.1	0.25	0.5	1.0
Trustworthiness	0.87	0.81	0.79	0.77	0.66	0.54
Continuity	0.95	0.93	0.85	0.74	0.70	0.56
Shepard Corr.	0.39	0.30	0.27	0.26	0.21	0.09
Stress	0.93	0.84	0.78	0.66	0.57	0.54
Neighborhood Hit	0.99	0.99	0.97	0.91	0.60	0.45

(a) *Quality Metrics for varying values of* β.

4.4. Computational performance

Figure 6 shows how ShaRP fares compared to other projection techniques *vs.* computational time. Tests were run on a PC with an AMD Ryzen 9 5900HX 3.3GHz 8-core processor and an NVIDIA RTX 3080 GPU. ShaRP is much faster than t-SNE (50-80% speedup) and Isomap (50-60%). It is also faster than AE, UMAP, and only slightly slower than SSNP, its predecessor. The used batch size (256 data points) is largely responsible for ShaRP's speed.

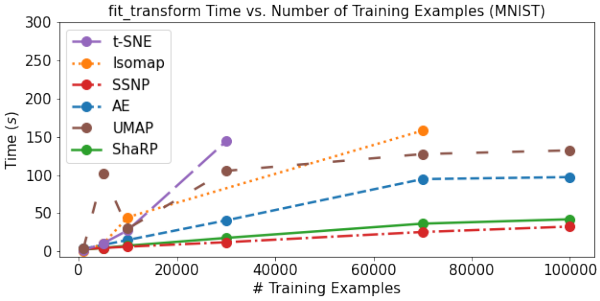


Figure 6: *Run times for five projection techniques and ShaRP, MNIST dataset. Runs were stopped after 5 minutes. See discussion on Section 4.4. For more results, see supplemental material.*

5. Discussion and Future Work

ShaRP introduces a novel level of pattern steerability for a projection algorithm, all while performing comparably to state-of-the-art methods in relevant (visual) quality metrics. However, ShaRP also has some limitations which frame our future work directions. Currently, we only support numerical features as these work directly with Auto-Encoders. One-hot encoding or Categorical Variational Auto-Encoders [MST*21] can overcome this limitation with only slight adaptions to our network architecture. Also, the sampling schemes we devised support shaping clusters into ellipses and convex polygons. A wider variety of shapes can be obtained by devising new sampling schemes. However, obtaining log-probabilities for samples of complex sampling schemes can be computationally intensive. We next aim to study sampling schemes that naturally encourage further visual aspects of interest of projections, *e.g.*, cluster separability.

Shape-regularized projections also can help interaction and visualization tasks. For example, rectangular shaped clusters can help as a clutter-reduction mechanism whenever (annotation) overlays with text and images should be added to projection (cf Fig. 2). We aim to further study this aspect, including the ease of interactive hierarchical navigation of thumbnail-annotated squarified projections to support various analysis tasks of high-dimensional data.

References

[AEC*21] Appleby G., Espadoto M., Chen R., Goree S., Telea A. C., Anderson E. W., Chang R.: HyperNP: Interactive visual exploration of multidimensional projection hyperparameters. *CoRR abs/2106.13777* (2021). arXiv:2106.13777. 2

[AGO*12] 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*. Ambient Assisted Living and Home Care. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 216–223. 4

[BBH12] Bunte K., Biehl M., Hammer B.: A general framework for dimensionality reducing data visualization mapping. *Neural Computation 24*, 3 (2012), 771–804. 1

[BBK*18] Behrisch M., Blumenschein M., Kim N. W., Shao L., El-Assady M., Fuchs J., Seebacher D., Diehl A., Brandes U., Pfister H., Schreck T., Weiskopf D., Keim D. A.: Quality metrics for information visualization. *Computer graphics forum 37*, 3 (Jun 2018), 625–662. 1

[BCV13] Bengio Y., Courville A. C., Vincent P.: Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell. 35*, 8 (2013), 1798–1828. doi:10.1109/TPAMI.2013.50. 2

[C*15] Chollet F., et al.: Keras. https://keras.io, 2015. 2

[CG15] Cunningham J., Ghahramani Z.: Linear dimensionality reduction: Survey, insights, and generalizations. *JMLR 16* (2015), 2859–2900. 1

[CMC*22] Cutura R., Morariu C., Cheng Z., Wang Y., Weiskopf D., Sedlmair M.: Hagrid: using hilbert and gosper curves to gridify scatterplots. *Journal of Visualization* (July 2022). doi:10.1007/s12650-022-00854-7. 2

[DLT*17] Dillon J. V., Langmore I., Tran D., Brevdo E., Vasudevan S., Moore D., Patton B., Alemi A., Hoffman M., Saurous R. A.: Tensorflow distributions, 2017. doi:10.48550/ARXIV.1711.10604. 2

[EHT20] Espadoto M., Hirata N. S. T., Telea A. C.: Deep learning multidimensional projections. *Information Visualization 19*, 3 (2020), 247–269. doi:10.1177/1473871620909485. 4

[EHT21] Espadoto M., Hirata N. S. T., Telea A. C.: Self-supervised dimensionality reduction with neural networks and pseudo-labeling. In *Proc. IVAPP (2021)* (2021), Hurter C., Purchase H. C., Braz J., Bouatouch K., (Eds.), SCITEPRESS, pp. 27–37. doi:10.5220/0010184800270037. 2, 4

[EMK*21] Espadoto M., Martins R. M., Kerren A., Hirata N. S. T., Telea A. C.: Toward a quantitative survey of dimension reduction techniques. *IEEE transactions on visualization and computer graphics 27*, 3 (Mar 01, 2021), 2153–2173. 1, 4

[F.R01] F.R.S. K. P.: Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 2*, 11 (1901), 559–572. doi:10.1080/14786440109462720. 2

[HG02] Hoffman P., Grinstein G.: A survey of visualizations for high-dimensional data mining. *Information Visualization in Data Mining and Knowledge Discovery 104* (2002), 47–82. 1

[Hul94] Hull J. J.: A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence 16*, 5 (1994), 550–554. doi:10.1109/34.291440. 4

[KR90] Kaufman L., Rousseeuw P. J.: *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley, 1990. doi:10.1002/9780470316801. 4

[KW14] Kingma D. P., Welling M.: Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014* (2014), Bengio Y., LeCun Y., (Eds.). 2, 3

[LBBH98] LeCun Y., Bottou L., Bengio Y., Haffner P.: Gradient-based learning applied to document recognition. *Proceedings of the IEEE 86*, 11 (1998), 2278–2324. doi:10.1109/5.726791. 4

[Llo82] Lloyd S. P.: Least squares quantization in PCM. *IEEE Trans. Inf. Theory 28*, 2 (1982), 129–136. doi:10.1109/TIT.1982.1056489. 4

[LMW*15] Liu S., Maljovec D., Wang B., Bremer P.-T., Pascucci V.: Visualizing high-dimensional data: Advances in the past decade. *IEEE TVCG 23*, 3 (2015), 1249–1268. 1

[MH18] McInnes L., Healy J.: UMAP: uniform manifold approximation and projection for dimension reduction. *CoRR abs/1802.03426* (2018). arXiv:1802.03426. 2

[MSJG15] Makhzani A., Shlens J., Jaitly N., Goodfellow I. J.: Adversarial autoencoders. *CoRR abs/1511.05644* (2015). arXiv:1511.05644. 2

[MST*21] Morton J. T., Silverman J., Tikhonov G., Lähdesmäki H., Bonneau R.: Scalable estimation of microbial co-occurrence networks with variational autoencoders. *bioRxiv* (2021). doi:10.1101/2021.11.09.467939. 5

[NA18] Nonato L., Aupetit M.: Multidimensional projection for visual analytics: Linking techniques with distortions, tasks, and layout enrichment. *IEEE TVCG* (2018). doi:10.1109/TVCG.2018.2846735. 1

[PKF*16] Pandey A. V., Krause J., Felix C., Boy J., Bertini E.: Towards understanding human similarity perception in the analysis of large sets of scatter plots. In *Conference on Human Factors in Computing Systems CHI* (2016), Kaye J., Druin A., Lampe C., Morris D., Hourcade J. P., (Eds.), ACM, pp. 3659–3669. doi:10.1145/2858036.2858155. 1

[SNLH09] Sips M., Neubert B., Lewis J. P., Hanrahan P.: Selecting good views of high-dimensional data using class consistency. *Computer graphics forum 28*, 3 (Jun 2009), 831–838. 2

[SVPM14] Sorzano C., Vargas J., Pascual-Montano A.: A survey of dimensionality reduction techniques, 2014. arXiv:1403.2877 [stat.ML]. 1

[Tho17] Thoma M.: The reuters dataset, July 2017. 4

[TSL00] Tenenbaum J. B., Silva V. d., Langford J. C.: A global geometric framework for nonlinear dimensionality reduction. *Science 290*, 5500 (Dec 22, 2000), 2319–2323. 2

[Uly16] Ulyanov D.: Multicore-TSNE. https://github.com/DmitryUlyanov/Multicore-TSNE, 2016. 2

[vdM14] van der Maaten L.: Accelerating t-sne using tree-based algorithms. *J. Mach. Learn. Res. 15*, 1 (2014), 3221–3245. doi:10.5555/2627435.2697068. 2

[vdMH08] van der Maaten L., Hinton G. E.: Visualizing high-dimensional data using t-SNE. *Journal of Machine Learning Research 9* (2008), 2579–2605. 2

[vP09] van der Maaten L., Postma E.: *Dimensionality Reduction: A Comparative Review*. Tech. rep., Tilburg University, Netherlands, 2009. Tech. report TiCC TR 2009-005. 1

[WFC*18] Wang Y., Feng K., Chu X., Zhang J., Fu C., Sedlmair M., Yu X., Chen B.: A perception-driven approach to supervised dimensionality reduction for visualization. *IEEE Trans. Vis. Comput. Graph. 24*, 5 (2018), 1828–1840. doi:10.1109/TVCG.2017.2701829. 1

[WVJ16] Wattenberg M., Viégas F., Johnson I.: How to Use t-SNE Effectively. *Distill* (2016). doi:10.23915/distill.00002. 2

[XRV17] Xiao H., Rasul K., Vollgraf R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR abs/1708.07747* (2017). arXiv:1708.07747. 4

[Yin07] Yin H.: Nonlinear dimensionality reduction and data visualization: A review. *Intl. Journal of Automation and Computing 4*, 3 (2007), 294–303. 1