Robust Generative Restricted Kernel Machines using Weighted Conjugate Feature Duality

Arun Pandey*, Joachim Schreurs* & Johan A. K. Suykens
Department of Electrical Engineering ESAT-STADIUS, KU Leuven
Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
{arun.pandey, joachim.schreurs, johan.suykens}@esat.kuleuven.be

June 24, 2020

Abstract

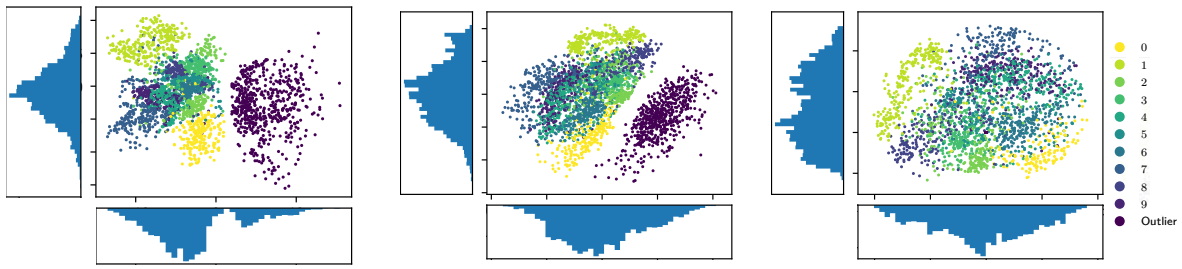
Interest in generative models has grown tremendously in the past decade. However, their training performance can be adversely affected by contamination, where outliers are encoded in the representation of the model. This results in the generation of noisy data. In this paper, we introduce weighted conjugate feature duality in the framework of Restricted Kernel Machines (RKMs). The RKM formulation allows for an easy integration of methods from classical robust statistics. This formulation is used to fine-tune the latent space of generative RKMs using a weighting function based on the Minimum Covariance Determinant, which is a highly robust estimator of multivariate location and scatter. Experiments show that the weighted RKM is capable of generating clean images when contamination is present in the training data. We further show that the robust method also preserves uncorrelated feature learning through qualitative and quantitative experiments on standard datasets.

1 Introduction

A popular choice for generative models in machine learning are latent variable models such as Variational Auto-Encoders (VAE) [1], Restricted Boltzmann Machines (RBM) [2,3] and Generative Adversarial Networks (GAN) [4–6]. These latent spaces provide a representation of the input data by embedding into an underlying vector space. Exploring these spaces allows for deeper insights in the structure of the data distribution, as well as understanding relationships between data points. The interpretability of the latent space is enhanced when the model learns a disentangled representation [7, 8]. In a disentangled representation, a single latent feature is sensitive to changes in a single generative factor, while being relatively invariant to changes in other factors [9]. For example hair color, lighting conditions or orientation of faces.

In generative modelling, training data is often assumed to be ground truth, therefore outliers can severely degrade the learned representations and performance of trained models. The same issue arises in generative modelling where contamination of the training data results in encoding of the outliers. Consequently, the network generates noisy images when reconstructing out-of-sample extensions. To solve this problem, multiple robust variants of generative models were proposed in [10–12]. However, these generative models require clean training data or only consider the case where there is label noise. In this paper, we address the problem of *contamination on the training data itself*. This is a common problem in real-life datasets, which are often contaminated by human error, measurement errors or changes in system behaviour. To the best of our knowledge, this specific problem is not addressed in other generative methods. The Restricted Kernel Machine (RKM) formulation [13] allows for a straightforward integration of methods from classical robust statistics to the RKM framework. The RKM framework yields a representation of kernel methods with visible and hidden units establishing links between kernel methods [14] and RBMs. [15] showed how kernel PCA fits into the RKM framework. A tensor-based multi-view classification model was developed in [16]. In [17], a multi-view generative model called Generative RKM (Gen-RKM) is introduced which uses explicit feature-maps for joint feature-selection and subspace learning. Gen-RKM learns the basis of the latent space, yielding uncorrelated latent variables. This allows to generate data with specific features, i.e. a disentangled representation.

*Authors contributed equally to this work.



(a) VAE (b) Gen-RKM (c) Robust Gen-RKM

Figure 1: Illustration of robustness against outliers on the MNIST dataset. 20% of the training data is contaminated with noise. The models are trained with a 2-dimensional latent space in the standard setup, see Section 4. The presence of outliers distorts the distribution of the latent variables for the Gen-RKM and VAE, where the histogram of the latent variables is skewed. By down-weighting the outliers, the histogram resembles a Gaussian distribution again.

Contributions: This paper introduces a weighted Gen-RKM model that detects and penalizes the outliers to regularize the latent space. Thanks to the introduction of weighted conjugate feature duality, a RKM formulation for weighted kernel PCA is derived. This formulation is used within the Gen-RKM training procedure to fine-tune the latent space using different weighting schemes. A weighting function based on Minimum Covariance Determinant (MCD) [18] is proposed. Qualitative and quantitative experiments on standard datasets show that the proposed model is unaffected by large contamination and can learn meaningful representations.

2 Weighted Restricted Kernel Machines

2.1 Weighted Conjugate Feature Duality

For a comprehensive overview of the RKM framework, the reader is encouraged to refer [13, 17]. In this section, we extend the notion of conjugate feature duality by introducing a weighting matrix. Assuming $\boldsymbol{D} \succ 0$ to be a positive-definite diagonal weighting matrix, the following holds for any two vectors $\boldsymbol{e}, \boldsymbol{h} \in \mathbb{R}^n$, $\lambda > 0$:

$$\frac{1}{2\lambda}\boldsymbol{e}^\top \boldsymbol{D}\boldsymbol{e} + \frac{\lambda}{2}\boldsymbol{h}^\top \boldsymbol{D}^{-1}\boldsymbol{h} \geq \boldsymbol{e}^\top \boldsymbol{h}. \tag{1}$$

The inequality could be verified using the Schur complement by writing the above in its quadratic form:

$$\frac{1}{2}\begin{bmatrix}\boldsymbol{e}^\top & \boldsymbol{h}^\top\end{bmatrix}\begin{bmatrix}\frac{1}{\lambda}\boldsymbol{D}\boldsymbol{I} & -\boldsymbol{I} \\ -\boldsymbol{I} & \lambda \boldsymbol{D}^{-1}\boldsymbol{I}\end{bmatrix}\begin{bmatrix}\boldsymbol{e} \\ \boldsymbol{h}\end{bmatrix} \geq 0. \tag{2}$$

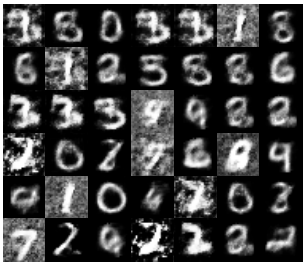
It states that for a matrix $\boldsymbol{Q} = \left[\begin{smallmatrix}\boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{B}^\top & \boldsymbol{C}\end{smallmatrix}\right]$, one has $\boldsymbol{Q} \succeq 0$ if and only if $\boldsymbol{A} \succ 0$ and the Schur complement $\boldsymbol{C} - \boldsymbol{B}^\top \boldsymbol{A}^{-1}\boldsymbol{B} \succeq 0$ [19], which proves the above inequality. This is also known as the Fenchel-Young inequality for quadratic functions [20].

We assume a dataset $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^N$, with $\boldsymbol{x}_i \in \mathbb{R}^d$ consisting of N data points. For a feature-map $\phi : \mathbb{R}^d \mapsto \mathbb{R}^{d_f}$ defined on input data points, the weighted kernel PCA objective [21] in the Least-Squares Support Vector Machine (LS-SVM) setting is given by [22]:

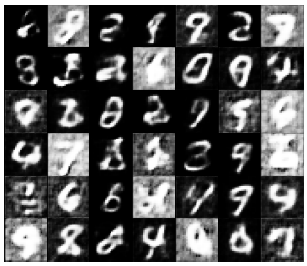
$$\min_{\boldsymbol{U},\boldsymbol{e}} J(\boldsymbol{U}, \boldsymbol{e}) = \frac{\eta}{2}\operatorname{Tr}(\boldsymbol{U}^\top \boldsymbol{U}) - \frac{1}{2\lambda}\boldsymbol{e}^\top \boldsymbol{D}\boldsymbol{e} \text{ s.t. } \boldsymbol{e}_i = \boldsymbol{U}^\top \phi(\boldsymbol{x}_i), \forall i = 1, \ldots, N, \tag{3}$$

where $\boldsymbol{U} \in \mathbb{R}^{d_f \times s}$ is the unknown interconnection matrix. By using (1), the error variables $\boldsymbol{e}_i$ are conjugated to latent variables $\boldsymbol{h}_i \in \mathbb{R}^s$ and substituting the constraints into the objective function yields

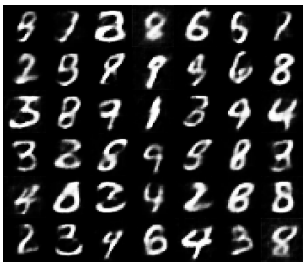
$$J \leq \mathcal{J}_t^{\boldsymbol{D}} := \sum_{i=1}^N \left\{-\phi(\boldsymbol{x}_i)^\top \boldsymbol{U}\boldsymbol{h}_i + \frac{\lambda}{2}\boldsymbol{D}_{ii}^{-1}\boldsymbol{h}_i^\top \boldsymbol{h}_i\right\} + \frac{\eta}{2}\operatorname{Tr}(\boldsymbol{U}^\top \boldsymbol{U}). \tag{4}$$



(a) VAE



(b) Gen-RKM



(c) Robust Gen-RKM

Figure 2: Illustration of robust generation on the MNIST dataset. 20% of the training data is contaminated with noise. The images are generated by random sampling from a fitted Gaussian distribution on the learned latent variables. When using a robust training procedure, the model does not encode the noisy images. As a consequence, no noisy images are generated.

The stationary points of $\mathcal{J}_t^{\boldsymbol{D}}$ are given by:

$$\begin{cases} \frac{\partial \mathcal{J}_t^{\boldsymbol{D}}}{\partial \boldsymbol{h}_i} = 0 \implies \lambda \boldsymbol{D}_{ii}^{-1} \boldsymbol{h}_i = \boldsymbol{U}^\top \phi(\boldsymbol{x}_i), \ \ \forall i = 1, \ldots, N \\ \frac{\partial \mathcal{J}_t^{\boldsymbol{D}}}{\partial \boldsymbol{U}} = 0 \implies \boldsymbol{U} = \frac{1}{\eta} \sum_{i=1}^N \phi(\boldsymbol{x}_i) \boldsymbol{h}_i^\top. \end{cases} \tag{5}$$

Eliminating $\boldsymbol{U}$ and denoting the kernel matrix $\boldsymbol{K} := [k(x_i, x_j)]_{ij}$ with kernel function $k(x, y) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle$, the eigenvectors $\boldsymbol{H} := [\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N]$, $\boldsymbol{\Lambda} := \mathrm{diag}\{\lambda_1, \ldots, \lambda_s\} \in \mathbb{R}^{s \times s}$ such that $\lambda_1 \geq \ldots \geq \lambda_s$ with s the dimension of the latent space, we get the weighted eigenvalue problem:

$$\frac{1}{\eta} [\boldsymbol{D}\boldsymbol{K}] \boldsymbol{H}^\top = \boldsymbol{H}^\top \boldsymbol{\Lambda}. \tag{6}$$

One can verify that each eigenvalue-eigenvector pair lead to the value $\mathcal{J}_t^{\boldsymbol{D}} = 0$. Using the weighted kernel PCA potential outliers can be penalized, which is discussed in more detail in Section 3.

2.2 Generation

Given the learned interconnection matrix $\boldsymbol{U}$, and a latent variable $\boldsymbol{h}^\star$, consider the following objective function

$$\mathcal{J}_g = -\phi(\boldsymbol{x}^\star)^\top \boldsymbol{U} \boldsymbol{h}^\star + \frac{1}{2} \phi(\boldsymbol{x}^\star)^\top \phi(\boldsymbol{x}^\star), \tag{7}$$

with a regularization term on the input data. Here $\mathcal{J}_g$ denotes the objective function for generation. To reconstruct or denoise a training point, $\boldsymbol{h}^\star$ can be one-of-the corresponding hidden units of the training point. Random generation is done by fitting a normal distribution on the learned latent variables, afterwards a random $\boldsymbol{h}^\star$ is sampled from the distribution which is put through the decoder network. Note that in the training objective (ref. (4)) we are imposing a soft-Gaussian prior over latent variables through quadratic regularization on $\{\boldsymbol{h}_i\}_{i=1}^N$. The stationary points of (7) yields the *generated feature vector* [15, 17] $\varphi(\boldsymbol{x}^\star)$, given by the corresponding $\boldsymbol{h}^\star$. With slight abuse of notation, we denote the generated feature-vector by $\varphi(\boldsymbol{x}^\star) = [\frac{1}{\eta} \sum_{i=1}^N \phi(\boldsymbol{x}_i) \boldsymbol{h}_i^\top] \boldsymbol{h}^\star$, which is a point in the feature-space corresponding to an unknown $\boldsymbol{x}^\star$ in data space. To obtain the generated data in the input space, the inverse image of the feature map $\phi(\cdot)$ should be computed. In kernel methods, this is known as the pre-image problem. We seek to find the function $\psi \colon \mathbb{R}^{d_f} \mapsto \mathbb{R}^d$, such that $(\psi \circ \varphi)(\boldsymbol{x}^\star) \approx \boldsymbol{x}^\star$, where $\varphi(\boldsymbol{x}^\star)$ is calculated from above. The pre-image problem is known to be ill-conditioned [23], and consequently various approximation techniques have been proposed [24]. Another approach is to explicitly define pre-image maps and learn the parameters in the training procedure [17]. In the experiments, we use (convolutional) neural networks as the feature maps $\phi_{\boldsymbol{\theta}}(\cdot)$, where the notation extends to $\varphi_{\boldsymbol{\theta}}(\cdot)$. Another (transposed convolutional) neural network is used for the pre-image map $\psi_{\boldsymbol{\zeta}}(\cdot)$ [25]. The parameters $\boldsymbol{\theta}$ and $\boldsymbol{\zeta}$ correspond to the network parameters. These parameters are learned by minimizing the reconstruction error in combination with the weighted RKM objective function. The training algorithm is described in more detail in section 3.2.

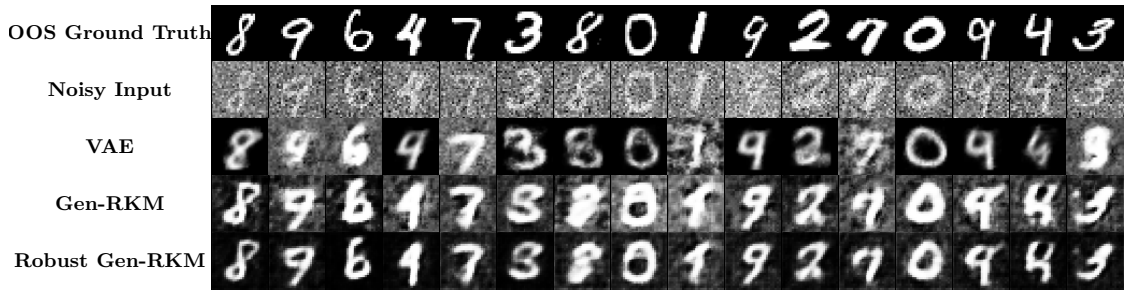


Figure 3: Illustration of robust denoising on the MNIST dataset. 20 % of the training data is contaminated with noise. The first and second row show the clean and noisy test images respectively. The third, fourth and fifth row show the denoised image using the VAE, Gen-RKM and robust Gen-RKM respectively.

Remark on Out-of-Sample extension: To reconstruct or denoise an out-of-sample test point $\boldsymbol{x}^\star$, the data is projected on the latent space using:

$$\boldsymbol{h}^\star = \lambda^{-1}\boldsymbol{U}^\top\phi(\boldsymbol{x}^\star) = \frac{1}{\lambda\eta}\sum_{i=1}^{N}\boldsymbol{h}_i k(\boldsymbol{x}_i, \boldsymbol{x}^\star). \tag{8}$$

The latent point is reconstructed by projecting back to the input space by first computing the generated feature vector followed by its pre-image map $\psi_{\boldsymbol{\zeta}}(\cdot)$.

3 Robust estimation of the latent variables

3.1 Robust Weighting Scheme

In this paper, we propose a weighting scheme to make the estimation of the latent variables more robust against contamination. The weighting matrix is a diagonal matrix with a weight $\boldsymbol{D}_{ii}$ corresponding to every $\boldsymbol{h}_i$ such that:

$$\boldsymbol{D}_{ii} = \begin{cases} 1 & \text{if } d_i^2 \leq \chi^2_{s,\alpha} \\ 10^{-4} & \text{otherwise,} \end{cases} \tag{9}$$

with s the dimension of the latent space, α the significance level of the Chi-squared distribution and d_i^2 the Mahalanobis distance for the corresponding $\boldsymbol{h}_i$:

$$d_i^2 = \left(\boldsymbol{h}_i - \hat{\boldsymbol{\mu}}\right)^\top \hat{\boldsymbol{S}}^{-1}\left(\boldsymbol{h}_i - \hat{\boldsymbol{\mu}}\right), \tag{10}$$

with $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{S}}$ the robustly estimated mean and covariance matrix respectively. In this paper, we propose to use the Minimum Covariance Determinant (MCD) [18]. The MCD is a highly robust estimator of multivariate location and scatter which has been used in many robust multivariate statistical methods [26, 27]. Given a data matrix of N rows with s columns, the objective is to find the $N_{\text{MCD}} < N$ observations whose sample covariance matrix has the lowest determinant. Its influence function is bounded [28] and has the highest possible breakdown value when $N_{\text{MCD}} = \lfloor (N + s + 1)/2 \rfloor$. In the experiments, we typically take $N_{\text{MCD}} = \lfloor N \times 0.75 \rfloor$ and $\alpha = 0.975$ for the Chi-squared distribution. The user could further tune these parameters according to the estimated contamination degree in the dataset. Eventually, the reweighting procedure can be repeated iteratively, but in practice one single additional weighted step will often be sufficient. Kernel PCA can take the interpretation of a one-class modeling problem with zero target value around which one maximizes the variance [29]. The same holds in the Gen-RKM framework. This is a natural consequence of the regularization term $\frac{\lambda}{2}\sum_{i=1}^{N}\boldsymbol{h}_i^\top\boldsymbol{h}_i$ in the training objective (see (4)), which implicitly puts a Gaussian prior on the hidden units. When the training of feature map is done correctly, one expects the latent variables to be normally distributed around zero [14]. Gaussian distributed latent variables are essential for having a *continuous* and *smooth* latent space, allowing easy interpolation. This property is also essential for VAEs and was studied in [1], where a regularization term, in the form of the Kullback-Leibler divergence between the encoder's distribution and a unit Gaussian as a prior on the latent variables was used. When training a non-robust generative model in the presence of outliers, the contamination can severely distort the distribution of the latent variables. This effect is seen in Figure 1, where a discontinuous and skewed distribution is visible.

3.2 Algorithm

We propose to use the above described reweighting step within the Gen-RKM framework [17]. The algorithm is flexible to incorporate both kernel-based, (deep) neural network and Convolutional based models within the same setting, and is capable of jointly learning the feature maps and latent representations. The Gen-RKM algorithm consists out of two phases: a training phase and a generation phase which occurs one after another. In the case of explicit feature maps, the training phase consists of determining the parameters of the explicit feature and pre-image map together with the hidden units $\{\boldsymbol{h}_i\}_{i=1}^N$.

We propose an adapted algorithm of [17] with an extra re-weighting step wherein the system in (6) is solved. Furthermore, the reconstruction error is weighted to reduce the effect of potential outliers on the pre-image maps. The loss function now becomes:

$$\min_{\boldsymbol{\theta},\boldsymbol{\zeta}} \ \mathcal{J}_c^{\boldsymbol{D}}(\boldsymbol{\theta},\boldsymbol{\zeta}) = \ \mathcal{J}_t^{\boldsymbol{D}} + \frac{c_{\text{stab}}}{2}(\mathcal{J}_t^{\boldsymbol{D}})^2 + \frac{c_{\text{acc}}}{N}\sum_{i=1}^{N} \boldsymbol{D}_{ii}\mathcal{L}(\boldsymbol{x}_i, \psi_{\boldsymbol{\zeta}}(\varphi_{\boldsymbol{\theta}}(\boldsymbol{x}_i))), \tag{11}$$

where $c_{stab} \in \mathbb{R}^+$ is a stability constant [13] and $c_{\text{acc}} \in \mathbb{R}^+$ is a regularization constant to control the stability with reconstruction accuracy. In the experiments, the loss function is equal to the mean squared error (MSE), however other loss functions are possible. The generation algorithm is the same as in [17].

Table 1: FID Scores [30] over 10 iterations for 4000 randomly generated samples when the training data is contaminated with 20% outliers. (smaller is better).

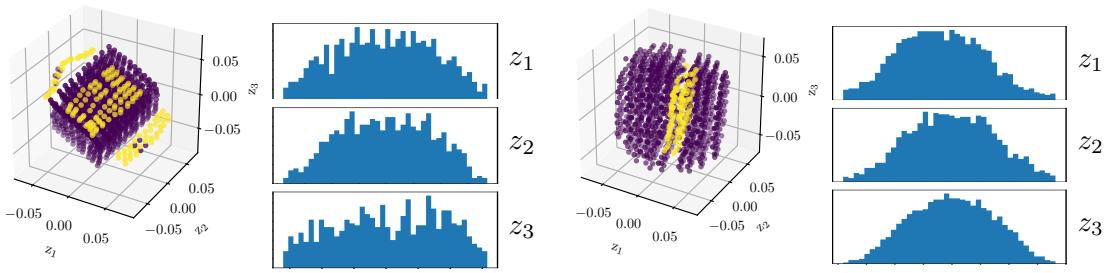
Dataset	**FID score**			
	VAE	**β-VAE** ($\beta = 3$)	**RKM**	**Rob Gen-RKM**
MNIST	142.54±0.73	187.21±0.11	134.95±1.61	**87.32±1.92**
F-MNIST	245.84±0.43	291.11±1.6	163.51±1.24	**153.32±0.05**
SVHN	168.21±0.23	234.87±1.45	112.45±1.4	**98.14 ±1.2**
CIFAR-10	201.21±0.71	241.23±0.34	187.08±0.58	**132.6±0.21**
Dsprites	234.51±1.10	298.21±1.5	182.65±0.57	**160.56±0.96**
3Dshapes	233.18±0.94	252.41±0.38	177.29±1.60	**131.18±1.45**

4 Experiments

In this section, we evaluate the robustness of the weighted Gen-RKM on the MNIST, Fashion-MNIST (F-MNIST), CIFAR-10, SVHN, Dsprites and 3Dshapes dataset[1]. The last two datasets will be used in disentanglement experiments since they include the ground truth generating factors which are necessary to quantify the performance. Training of the robust Gen-RKM is done using the algorithm proposed in Section 3.2, where we take $N_{\text{MCD}} = \lfloor N \times 0.75 \rfloor$ and $\alpha = 0.975$ for the Chi-squared distribution (see (9)). Afterwards we compare with the standard Gen-RKM [17], VAE and β-VAE. The models have the same encoder/decoder architecture, optimization parameters and are trained until convergence. Information on the training settings and model architectures is given in the Appendix.

Generation and Denoising: Figure 2 shows the generation of random images when models were trained on the contaminated MNIST dataset. The contamination consists of adding Gaussian noise $\mathcal{N}(0.5, 0.5)$ to 20% of the data. The images are generated by random sampling from a fitted Gaussian distribution on the learned latent variables. As we can see, when using a robust training procedure, the model does not encode the noisy images. As a consequence, no noisy images are generated and the generation quality is significantly better. This is also confirmed by the Fréchet Inception Distance (FID) scores [30] in Table 1, which quantifies the quality of generation. The robust Gen-RKM clearly outperforms the other methods when the data has contamination. Moreover VAEs are known to generate samples closer to the mean of dataset. This negatively affects the FID scores which also takes into account the diversity within the generated images. The scores for β-VAE are worst due to the inherent emphasis on imposing a Gaussian distribution on latent variables trading-off with the reconstruction quality [7]. The classical RKM performs

[1] http://yann.lecun.com/exdb/mnist/, https://github.com/zalandoresearch/fashion-mnist, https://github.com/deepmind/dsprites-dataset, https://github.com/deepmind/3d-shapes, https://www.cs.toronto.edu/~kriz/cifar.html, http://ufldl.stanford.edu/housenumbers/



(a) Gen-RKM (b) Robust Gen-RKM

Figure 4: Illustration of disentanglement on the 3DShapes dataset. Clean data is depicted in purple, outliers in yellow. The training subset is contaminated with a third generating factor (20% of the data is considered as outliers). The outliers are down-weighted in the robust Gen-RKM, which moves them to the center.

slightly better than VAE and its variant. This is attributed to the presence of kernel PCA during training, which is often used in denoising applications and helps to some extent in dealing with contamination in the dataset.

Next, we use generative models in the denoising experiment. Image denoising is accomplished by projecting the noisy test set observations on the latent space, afterwards projecting back to the input space. Because there is a latent bottleneck, the most important features of the images are retained while insignificant features like noise are removed. Figure 3 shows an illustration of robust denoising on the MNIST dataset. The robust Gen-RKM does not encode the noisy images within the training procedure. Consequently, the model is capable of denoising the out-of-sample test images. When comparing the denoising quality on the full test set (5000 images sampled uniformly at random), the Mean Absolute Error (MAE) of Gen-RKM: $\mathrm{MAE} = 0.415$ and VAE: $\mathrm{MAE} = 0.434$ is much higher than the robust version: $\mathrm{MAE} = 0.206$. The experiments show that basic generative models like Gen-RKM and VAE are highly affected by outliers, while the robust counterpart can cope with a significant fraction of contamination. **Effect on**

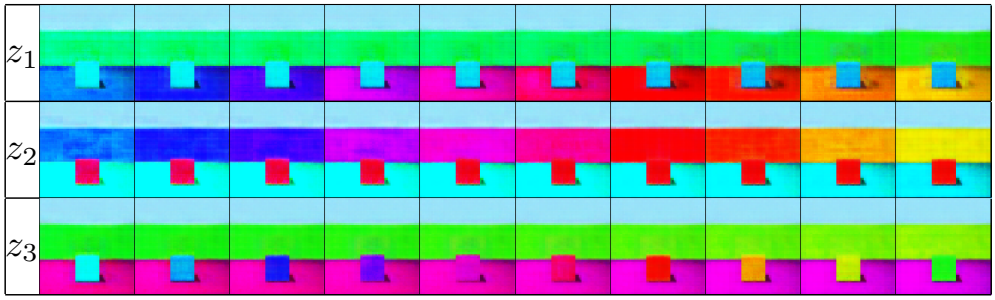


Figure 5: Illustration of latent traversals along the 3 latent dimensions for 3DShapes dataset using the robust Gen-RKM model. The first, second and third row distinctly captures the floor-hue, wall-hue and object-hue respectively while keeping other generative factors constant.

Disentanglement: In this experiment, contamination is an extra generating factor which is not present in the majority of the data. The goal is to train a disentangled representation, where the robust model only focuses on the most prominent generating factors. We subsample a 'clean' training subset which consists of cubes with different floor, wall and object hue. The scale and orientation are kept constant with minimum scale and 0° orientation respectively. Afterwards, the training data is contaminated by cylinders with maximum scale at 30° orientation (20% of the data is considered as outliers). The training data now consist out of 3 'true' generating factors (floor, wall and object hue) which appear in the majority of the data and 3 'noisy' generating factors (object, scale and orientation) which only occur in a small fraction. To illustrate the effect of the weighting scheme, Figure 4 visualizes the latent space of the (robust) Gen-RKM model. The classical Gen-RKM encodes the outliers in the representation, which results in a distorted Gaussian distribution of the latent variables. This is not the case for the robust Gen-RKM, where the outliers are downweighted. An illustration of latent traversals along the 3 latent dimensions using the robust Gen-RKM model is given in Figure 5, where the robust model is capable of disentangling the 3 'clean' generating factors.

Table 2: Disentanglement Metric on DSprites and 3D Shapes dataset. The training subset is contaminated with extra generating factors (20% of the data is considered as outliers). The framework of [31] with Lasso and Random Forest regressor [31] is used to evaluate the learned representation. For disentanglement and completeness higher score is better, for informativeness, lower is better.

Dataset	h_{dim}	**Algorithm**	**Lasso**			**Random Forest**		
			Disent.	**Comple.**	**Inform.**	**Disent.**	**Comple.**	**Inform.**
DSprites	2	β-VAE ($\beta = 3$)	0.19	0.16	6.42	0.13	0.32	**1.39**
		Gen-RKM	0.07	0.07	**5.82**	0.25	0.27	5.91
		Rob Gen-RKM	**0.21**	**0.21**	9.13	**0.36**	**0.38**	5.95
3DShapes	3	β-VAE ($\beta = 3$)	0.24	0.28	**2.72**	0.12	0.13	2.15
		Gen-RKM	0.14	0.14	3.03	0.15	0.15	1.09
		Rob Gen-RKM	**0.47**	**0.49**	3.13	**0.44**	**0.45**	**1.02**

To quantify the performance of disentanglement, we use the proposed framework[2] of [31], which consists of 3 metrics: disentanglement, completeness and informativeness. The framework could be used when the *ground-truth latent structure is available*, which is the case for 3Dshapes and DSprites dataset. The results are shown in Table 2, where the robust method outperforms the Gen-RKM. The above experiment is repeated on the DSprites dataset. The 'clean' training subset consists of ellipse shaped datapoints with minimal scale and 0° angle at different x and y positions. Afterwards, the training data is contaminated with a random sample of different objects at larger scales, different angles at different x and y positions. The training data now consist out of 2 'true' generating factors (x and y positions) which appear in the majority of the data and 3 'noisy' generating factor (orientation, scale and shape) which only occur in a small fraction. In addition to RKM, the results of β-VAE are shown in Table 2.

5 Conclusion

Using a weighted conjugate feature duality, a RKM formulation for weighted kernel PCA is proposed. This formulation is used within the Gen-RKM training procedure to fine-tune the latent space using a weighting function based on the MCD. Experiments show that the weighted RKM is capable of generating denoised images in spite of contamination in the training data. Furthermore, being a latent variable model, robust Gen-RKM preserves the disentangled representation. Future work consists of exploring various robust estimators and other weighting schemes to control the effect of sampling bias in the data.

Acknowledgments

EU: The research leading to these results has received funding from the European Research Council under the European Union's Horizon 2020 research and innovation program / ERC Advanced Grant E-DUALITY (787960). This paper reflects only the authors' views and the Union is not liable for any use that may be made of the contained information. Research Council KUL: Optimization frameworks for deep kernel machines C14/18/068 Flemish Government: FWO: projects: GOA4917N (Deep Restricted Kernel Machines: Methods and Foundations), PhD/Postdoc grant Impulsfonds AI: VR 2019 2203 DOC.0318/1QUATER Kenniscentrum Data en Maatschappij Ford KU Leuven Research Alliance Project KUL0076 (Stability analysis and performance improvement of deep reinforcement learning algorithms).

Appendix

Table 3 shows the details on training settings used in this paper. The PyTorch library in Python was used with a 8GB NVIDIA QUADRO P4000 GPU.

[2]Code and dataset available at `https://github.com/cianeastwood/qedr`

Table 3: Model architectures. All convolutions and transposed-convolutions are with stride 2 and padding 1. Unless stated otherwise, layers have Parametric-RELU ($\alpha = 0.2$) activation functions, except output layers of the pre-image maps which have sigmoid activation functions. $N_{\text{sub}} \leq N$ is the training subset size, s the latent space dimension and m the minibatch size.

Dataset	**Optimizer** (Adam)	**Architecture**		**Parameters**	
MNIST/F-MNIST/ CIFAR-10/SVHN	1e-4	Feature-map (fm)	Conv 32×4×4; Conv 64×4×4; FC 228 (Linear)	N_{sub}	3000
		Pre-image map	reverse of fm	s	10
		Latent space dim.	10	m	200
Dsprites/3DShapes	1e-4	Feature-map (fm)	Conv 20×4×4; Conv 40×4×4; Conv 80×4×4; FC 228 (Linear)	N_{sub}	1024/1200
		Pre-image map	reverse of fm	s	2
		Latent space dim.	$2/3$	m	200

References

[1] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *2nd International Conference on Learning Representations, 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[2] P. Smolensky, "Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1," ch. Information Processing in Dynamical Systems: Foundations of Harmony Theory, pp. 194–281, Cambridge, MA, USA: MIT Press, 1986.

[3] R. Salakhutdinov and G. Hinton, "Deep Boltzmann Machines," *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, 2009.

[4] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative Adversarial Nets," in *Advances in Neural Information Processing Systems 27*, pp. 2672–2680, 2014.

[5] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *ICML*, pp. 214–223, 2017.

[6] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Advances in Neural Information Processing Systems*, pp. 469–477, 2016.

[7] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "Beta-vae: Learning basic visual concepts with a constrained variational framework.," *International Conference on Learning Representations*, 2017.

[8] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Advances in Neural Information Processing Systems*, pp. 2172–2180, 2016.

[9] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[10] F. Futami, I. Sato, and M. Sugiyama, "Variational inference based on robust divergences," *21st International Conference on Artificial Intelligence and Statistics*, 2018.

[11] G. G. Chrysos, J. Kossaifi, and S. Zafeiriou, "Robust conditional generative adversarial networks," *International Conference on Learning Representations*, 2019.

[12] Y. Tang, R. Salakhutdinov, and G. Hinton, "Robust boltzmann machines for recognition and denoising," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2264–2271, IEEE, 2012.

[13] J. A.K. Suykens, "Deep Restricted Kernel Machines using Conjugate Feature Duality," *Neural Computation*, vol. 29, pp. 2123–2163, Aug. 2017.

[14] J. A.K. Suykens, T. Van Gestel, J. De Brabanter, B. De Moor, and J. Vandewalle, *Least Squares Support Vector Machines*. Jan. 2002.

[15] J. Schreurs and J. A.K. Suykens, "Generative Kernel PCA," in *European Symposium on Artificial Neural Networks*, pp. 129–134, 2018.

[16] L. Houthuys and J. A.K. Suykens, "Tensor learning in multi-view kernel PCA ," in *27th International Conference on Artificial Neural Networks ICANN, Rhodes, Greece*, vol. 11140, pp. 205–215, 2018.

[17] A. Pandey, J. Schreurs, and J. A.K. Suykens, "Generative restricted kernel machines," *arXiv preprint arXiv:1906.08144*, 2019.

[18] P. J. Rousseeuw and K. V. Driessen, "A fast algorithm for the minimum covariance determinant estimator," *Technometrics*, vol. 41, no. 3, pp. 212–223, 1999.

[19] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

[20] R. T. Rockafellar, *Conjugate Duality and Optimization*. SIAM, 1974.

[21] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *International Conference on Artificial Neural Networks*, pp. 583–588, Springer, 1997.

[22] C. Alzate and J. A.K. Suykens, "Multiway spectral clustering with out-of-sample extensions through weighted kernel PCA," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 2, pp. 335–347, 2008.

[23] S. Mika, B. Schölkopf, A. Smola, K.-R. Müller, M. Scholz, and G. Rätsch, "Kernel PCA and De-noising in Feature Spaces," in *Advances in Neural Information Processing Systems*, pp. 536–542, 1999.

[24] P. Honeine and C. Richard, "Preimage Problem in Kernel-Based Machine Learning," *IEEE Signal Processing Magazine*, vol. 28, pp. 77–88, Mar. 2011.

[25] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv preprint arXiv:1603.07285*, 2016.

[26] M. Hubert, P. J. Rousseeuw, and K. Vanden Branden, "ROBPCA: a new approach to robust principal component analysis," *Technometrics*, vol. 47, no. 1, pp. 64–79, 2005.

[27] M. Hubert, P. J. Rousseeuw, and T. Verdonck, "A deterministic algorithm for robust location and scatter," *Journal of Computational and Graphical Statistics*, vol. 21, no. 3, pp. 618–637, 2012.

[28] C. Croux and G. Haesbroeck, "Influence function and efficiency of the minimum covariance determinant scatter matrix estimator," *Journal of Multivariate Analysis*, vol. 71, no. 2, pp. 161–190, 1999.

[29] J. A.K. Suykens, T. Van Gestel, J. Vandewalle, and B. De Moor, "A support vector machine formulation to pca analysis and its kernel version," *IEEE Transactions on Neural Networks*, vol. 14, no. 2, pp. 447–450, 2003.

[30] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Proceedings of the 31st NIPS*, pp. 6629–6640, 2017.

[31] C. Eastwood and C. K. I. Williams, "A framework for the quantitative evaluation of disentangled representations," in *International Conference on Learning Representations*, 2018.