

Generative Restricted Kernel Machines

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 26, 2019

Abstract

We propose a novel method for estimating generative models based on the Restricted Kernel Machine (RKM) framework. This mechanism uses the shared representation of data from various sources where training only involves solving an eigenvalue problem. By defining an explicit feature map, we show that neural networks could be incorporated in the current framework. Experiments on various datasets demonstrate the potential of the model through qualitative evaluation of generated samples.

1 Introduction

Generative modeling is a rapidly advancing area of machine learning research finding applications in multiple fields such as, generated art, on-demand video, image denoising [1], exploration in reinforcement learning [2], collaborative filtering [3], inpainting [4] and many more.

In general, three approaches have been used in generative modeling tasks. First, graphical models based on a probabilistic framework with latent variables such as variational auto-encoders [5] and Restricted Boltzmann Machines (RBMs) [6, 7]. Then, more recently proposed models based on adversarial training such as Generative Adversarial Networks (GANs) [8] and its many variants. Furthermore, autoregressive models such as Pixel Recurrent Neural Networks (PixelRNNs) [9] that models the conditional distribution of every individual pixel given previous pixels and generation involves sequentially predicting the pixels in an image along the two spatial dimensions. All these approaches have their own advantages and disadvantages. For example, RBM allows us to perform both learning and Bayesian inference in graphical models with latent variables. However, such probabilistic models must be properly normalized, which requires evaluating intractable integrals over the space of all possible variable configurations, hence posing difficulties in maximum likelihood computations [7]. Currently GANs are state-of-the-art for generative modeling and produce sharpest images but they are more difficult to train due to unstable training dynamics, unless more sophisticated variants are applied. Few existing multi-view generative models exist which involve a combination of the above existing generative mechanisms [10, 11].

In this work, we propose an alternative generative mechanism based on the framework of RKMs [12] that yields a representation of kernel methods with visible and hidden units establishing links between Kernel Principal Component Analysis (Kernel PCA), Least-Squares Support Vector Machines (LS-SVM) [13] and RBMs. This framework renders the energy form of RBMs into a non-probabilistic setting thereby having no requirement for proper normalization. This allows their immediate applicability to regression and

classification problems. Recently, [14] used this framework to develop tensor-based multi-view classification models and [15] showed that Kernel PCA could be used to generate new data.

The contributions of this paper further extend the RKM framework to multi-view generative models where multiple data could be generated simultaneously using a single instance of a sampled latent variable. Learning data representations is based on common-subspace learning and training only involves solving an eigenvalue problem. The generative mechanism involves computing the pre-image of the feature vectors. Two methods are proposed: with the feature map explicitly known or unknown. We show that the mechanism is flexible to incorporate both the kernel-based and (deep) neural network based models in the same setting. Lastly, RKMs can be viewed as a form of non-probabilistic graphical model, and they provide high flexibility in the design of architectures and training criteria as would be shown in following sections.

This paper is organized as follows. In section 2 we discuss the training mechanism of Generative RKMs when multiple data sources are available. In section 3 we explain the generative mechanism when implicit or explicit feature maps are used during training. In section 4, we show experimental results of our mechanism on MNIST and small-norb datasets. Section 5 concludes the paper along with extensions to the future work.

2 Generative Restricted Kernel Machines

We assume a dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$, with $\mathbf{x}_i \in \mathbb{R}^d$, $\mathbf{y}_i \in \mathbb{R}^p$ comprising N data points. Here \mathbf{y}_i may represent an additional view of \mathbf{x}_i , for e.g., an additional image from a different angle of the same subject, or a label information, such as in case of MNIST digits. Starting from the RKM interpretation of Kernel PCA, which gives an upper bound on the equality constrained, least-squares Kernel PCA objective function [12], and applying the feature-map $\phi_1 : \mathbb{R}^d \mapsto \mathbb{R}^{d_f}$ and $\phi_2 : \mathbb{R}^p \mapsto \mathbb{R}^{p_f}$ to the input data points, the training objective function \mathcal{J}_t for generative RKM is given by:

$$\mathcal{J}_t = \sum_{i=1}^N \left(-\phi_1(\mathbf{x}_i)^\top \mathbf{V} \mathbf{h}_i - \phi_2(\mathbf{y}_i)^\top \mathbf{U} \mathbf{h}_i + \frac{\lambda_i}{2} \mathbf{h}_i^\top \mathbf{h}_i \right) + \frac{\eta_1}{2} \text{Tr}(\mathbf{V}^\top \mathbf{V}) + \frac{\eta_2}{2} \text{Tr}(\mathbf{U}^\top \mathbf{U}) \quad (1)$$

where $\mathbf{V} \in \mathbb{R}^{d_f \times s}$ and $\mathbf{U} \in \mathbb{R}^{p_f \times s}$ are the unknown interaction matrices, and $\mathbf{h}_i \in \mathbb{R}^s$ are the latent variables modeling a common subspace \mathcal{H} between the two input spaces \mathcal{X} and \mathcal{Y} (see Fig. 1). Similar to Energy-Based Models (EBMs), RKM objective functions capture dependencies between variables by associating a scalar energy to each configuration of the variables. Learning consists of finding an energy function in which observed configurations of the variables are given lower energies than unobserved ones. Note that the schematic representation is similar to discriminative RBMs [16] and the objective function \mathcal{J}_t has the energy form similar to RBMs with additional regularization terms. Given $\eta_1 > 0$ and $\eta_2 > 0$ as regularization parameters, the stationary points of \mathcal{J}_t are

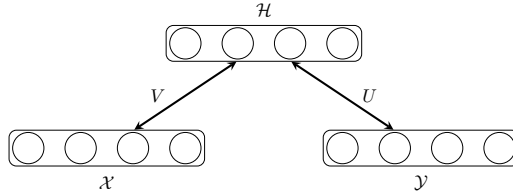


Figure 1: Restricted Kernel Machine modeling a common subspace between two data sources

given by:

$$\begin{cases} \frac{\partial \mathcal{J}_i}{\partial \mathbf{h}_i} = 0 \implies & \lambda_i \mathbf{h}_i = \mathbf{V}^\top \phi_1(\mathbf{x}_i) + \mathbf{U}^\top \phi_2(\mathbf{y}_i), \forall i = 1, \dots, N \\ \frac{\partial \mathcal{J}_i}{\partial \mathbf{V}} = 0 \implies & \mathbf{V} = \frac{1}{\eta_1} \sum_{i=1}^N \phi_1(\mathbf{x}_i) \mathbf{h}_i^\top, \\ \frac{\partial \mathcal{J}_i}{\partial \mathbf{U}} = 0 \implies & \mathbf{U} = \frac{1}{\eta_2} \sum_{i=1}^N \phi_2(\mathbf{y}_i) \mathbf{h}_i^\top, \end{cases} \quad (2)$$

where we have used vector and matrix derivatives [17]. Substituting \mathbf{V} and \mathbf{U} in the first equation above, denoting $\mathbf{\Lambda} = \text{diag}(\lambda_i) \in \mathbb{R}^{s \times s}$, yields the following eigenvalue problem:

$$\left[\frac{1}{\eta_1} \mathbf{K}_1 + \frac{1}{\eta_2} \mathbf{K}_2 \right] \mathbf{H}^\top = \mathbf{H}^\top \mathbf{\Lambda}, \quad (3)$$

where $\mathbf{H} = [\mathbf{h}_1 \dots \mathbf{h}_N] \in \mathbb{R}^{s \times N}$ with $s \leq N$ is the number of selected principal components and $\mathbf{K}_1, \mathbf{K}_2 \in \mathbb{R}^{N \times N}$ are kernel matrices corresponding to data sources. Based on Mercer's theorem [18], positive-definite kernel functions $k_1 : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$, $k_2 : \mathbb{R}^p \times \mathbb{R}^p \mapsto \mathbb{R}$ can be defined such that $k_1(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi_1(\mathbf{x}_i), \phi_1(\mathbf{x}_j) \rangle$, and $k_2(\mathbf{y}_i, \mathbf{y}_j) = \langle \phi_2(\mathbf{y}_i), \phi_2(\mathbf{y}_j) \rangle$, $\forall i, j = 1, \dots, N$ forms the elements of corresponding kernel matrices. The feature maps ϕ_1 and ϕ_2 , mapping the input data to high-dimensional feature space (possibly infinite) are implicitly defined by kernel functions such as Gaussian, polynomial and convolution kernels just to name a few [19]. However, one can also define explicit feature maps, which preserves the positive-definiteness of kernel function due to construction [13].

Remark on centering: For notational convenience, it is assumed that all the feature vectors defined above and in the following sections are centered, i.e. $\phi_c(\mathbf{x}_i) = \phi(\mathbf{x}_i) - \frac{1}{N} \sum_{j=1}^N \phi(\mathbf{x}_j)$, $\forall i = 1, \dots, N$. The case where the kernel matrix is not centered can be centered by replacing such matrices as follows:

$$\mathbf{K}_c = \mathbf{K} - N^{-1} \mathbf{1} \mathbf{1}^\top \mathbf{K} - N^{-1} \mathbf{K} \mathbf{1} \mathbf{1}^\top + N^{-2} \mathbf{1} \mathbf{1}^\top \mathbf{K} \mathbf{1} \mathbf{1}^\top, \quad (4)$$

where $\mathbf{1}$ denotes an N -dimensional vector of ones and \mathbf{K} is either \mathbf{K}_1 or \mathbf{K}_2 .

Remark on multiple data sources: While in the above section we have assumed that only two data sources (namely \mathcal{X} and \mathcal{Y}) are available for learning, the above procedure could be easily extended to multiple data-sources. Following the same arguments, this yields the following training problem:

$$\left[\sum_{\ell=1}^M \frac{1}{\eta_\ell} \mathbf{K}_\ell \right] \mathbf{H}^\top = \mathbf{H}^\top \mathbf{\Lambda},$$

where M is the number of views or data-sources.

3 Generating Data

In this section, we derive the equations for the generative mechanism. RKM being an Energy-Based Model, and the inference consists in clamping the value of observed variables and finding configurations of the remaining variables that minimizes the energy. For the learned interconnection matrices \mathbf{U} and \mathbf{V} , and latent variables \mathbf{h}^* , consider the following objective function:

$$\mathcal{J}_g = -\phi_1(\mathbf{x}^*)^\top \mathbf{V} \mathbf{h}^* - \phi_2(\mathbf{y}^*)^\top \mathbf{U} \mathbf{h}^* + \frac{1}{2} \phi_1(\mathbf{x}^*)^\top \phi_1(\mathbf{x}^*) + \frac{1}{2} \phi_2(\mathbf{y}^*)^\top \phi_2(\mathbf{y}^*), \quad (5)$$

with an additional regularization term on data sources. Here \mathcal{J}_g denotes the objective function for generation. The stationary points of \mathcal{J}_g are characterized by:

$$\begin{cases} \frac{\partial \mathcal{J}_g}{\partial \phi(\mathbf{x}^*)} = 0 \implies & \phi(\mathbf{x})^* = \mathbf{V} \mathbf{h}^*, \\ \frac{\partial \mathcal{J}_g}{\partial \phi(\mathbf{y}^*)} = 0 \implies & \phi(\mathbf{y})^* = \mathbf{U} \mathbf{h}^*. \end{cases} \quad (6)$$

Using \mathbf{V} and \mathbf{U} from Eq. (2), we obtain the generated feature vectors:

$$\phi_1(\mathbf{x}^*) = \left(\frac{1}{\eta_1} \sum_{i=1}^N \phi_1(\mathbf{x}_i) \mathbf{h}_i^\top \right) \mathbf{h}^*, \quad \phi_2(\mathbf{y}^*) = \left(\frac{1}{\eta_2} \sum_{i=1}^N \phi_2(\mathbf{y}_i) \mathbf{h}_i^\top \right) \mathbf{h}^*. \quad (7)$$

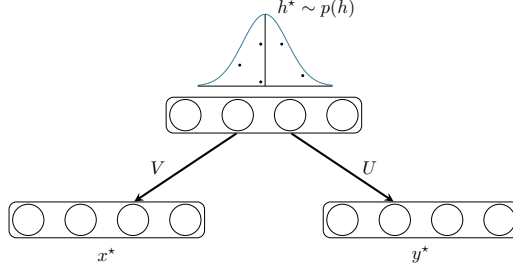


Figure 2: After learning, latent variables \mathbf{h}^* could be sampled from the fitted probability distribution $p(\mathbf{h})$ for generating \mathbf{x}^* and \mathbf{y}^* simultaneously.

Algorithm 1 Generative Restricted Kernel Machines

Input: $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$, η_1 , η_2 , s , choose feature map - explicit $\phi(\cdot)$ or implicit via kernel $k_1(\cdot, \cdot)$, $k_2(\cdot, \cdot)$

Output: generated data \mathbf{x}^* , \mathbf{y}^*

```

1: procedure TRAIN
2:   if feature map = implicit then
3:     choose kernel specific parameters
4:     construct centered kernel matrices  $\mathbf{K}_1$ ,  $\mathbf{K}_2$  (4)
5:     solve eigendecomposition problem (3)
6:     keep eigenvectors corresponding to  $s$  largest eigenvalues
7:   else if feature map = explicit then
8:     initialize parameters  $\theta_1$ ,  $\theta_2$ , choose n_epochs
9:     compute feature maps  $\phi_1(\mathbf{x}; \theta_1)$ ,  $\phi_2(\mathbf{y}; \theta_2)$ 
10:    for iter = 1 : n_epochs do
11:      repeat steps 4-6
12:      compute  $\mathcal{J}_t$  and  $\frac{\partial \mathcal{J}_t}{\partial \theta_1}$ ,  $\frac{\partial \mathcal{J}_t}{\partial \theta_2}$ 
13:      update  $\theta_1$ ,  $\theta_2$  using Gradient Descent algorithm
14:    end for
15:  end if
16: end procedure

17: procedure GENERATION
18:   sample  $\mathbf{h}^* \sim p(\mathbf{h})$ 
19:   if feature map = implicit then
20:     choose  $n_r$ 
21:     compute  $\mathbf{k}_{x^*}$ ,  $\mathbf{k}_{y^*}$  using Eq. (8)
22:      $\mathbf{x}^*$ ,  $\mathbf{y}^*$  using Eq. (9)
23:   else if feature map = explicit then
24:     compute  $\phi_1(\mathbf{x}^*)$ ,  $\phi_2(\mathbf{y}^*)$  using Eq. (7)
25:     compute pre-images  $\mathbf{x}^*$ ,  $\mathbf{y}^*$ 
26:   end if
27: end procedure

```

One now needs to find the inverse image of feature maps ϕ_1 and ϕ_2 in the respective input spaces i.e. solve the *pre-image problem*. When using kernel methods, feature maps are not known explicitly. Typically used kernels such as radial-basis function and polynomial kernels map the input data to very high dimensional feature space. Hence finding the pre-image, in general, is known to be an ill-conditioned [20] problem. However, various approximation techniques have been proposed [21, 22, 23, 24] which could be used to obtain the approximate pre-image $\hat{\mathbf{x}}$ of $\phi_1(\mathbf{x}^*)$. This problem could be avoided if one uses an explicit feature map, which is also invertible, during the training procedure and using the explicit expression for deducing \mathbf{x}^* from $\phi_1(\mathbf{x}^*)$. In the remaining section, we illustrate two pre-image methods for both cases when the feature map is explicitly known or unknown, and demonstrate corresponding experimental results in the following section.

Implicit Feature Map Since \mathbf{x}^* may not exist, we find an approximation $\hat{\mathbf{x}}$. As shown in [15], using the trick of left multiplying Eqs. (7) by $\phi_1(\mathbf{x}_i^*)^\top$ and $\phi_2(\mathbf{y}_i^*)^\top$, $\forall i = 1, \dots, N$, we obtain:

$$\mathbf{k}_{\mathbf{x}^*} = \frac{1}{\eta_1} \mathbf{K}_x \mathbf{H}^\top \mathbf{h}^*, \quad \mathbf{k}_{\mathbf{y}^*} = \frac{1}{\eta_2} \mathbf{K}_y \mathbf{H}^\top \mathbf{h}^*, \quad (8)$$

where, $\mathbf{k}_{\mathbf{x}^*} = [k(\mathbf{x}_1, \mathbf{x}^*), \dots, k(\mathbf{x}_N, \mathbf{x}^*)]^\top$ represents the similarities between \mathbf{x}^* and data points in the feature space, and \mathbf{K}_x represents the $N \times N$ centered kernel matrix of \mathcal{X} . Similar conventions follow for \mathcal{Y} respectively. Using the *kernel-smoother* method, the pre-images are given by:

$$\hat{\mathbf{x}} = \frac{\sum_{j=1}^{n_r} k_1(\mathbf{x}_j, \mathbf{x}^*) \mathbf{x}_j}{\sum_{j=1}^{n_r} k_1(\mathbf{x}_j, \mathbf{x}^*)}, \quad \hat{\mathbf{y}} = \frac{\sum_{j=1}^{n_r} k_2(\mathbf{y}_j, \mathbf{y}^*) \mathbf{y}_j}{\sum_{j=1}^{n_r} k_2(\mathbf{y}_j, \mathbf{y}^*)} \quad (9)$$

where n_r is the number of nearest neighbors.

Explicit Feature Map By using an appropriate feature map, Mercer’s theorem still holds due to the positive-definiteness of the kernel function by construction, thereby allowing the derivation of Eq. (3). In our experiments, we have used feed-forward neural networks with Parametric Rectified Linear unit (PReLU) activations in hidden layers as an explicit feature map [25]. Since, such networks are simply a composition of activation functions with matrix multiplication and addition of bias as arguments, we only require the activation functions to be invertible and weight matrices to be non-singular.

4 Experiments

To demonstrate the applicability of our framework, we trained the Generative RKM model on MNIST and small-NORB datasets using both the implicit feature map using a Gaussian kernel and an explicit feature map using the feed-forward neural network. In case of a kernel method, training only involves constructing the kernel and solving the eigenvalue problem (3), obtaining the latent variables \mathbf{H} . In principle, one could also use the latent variables directly for generation. However, in our experiments, we fit a normal distribution to the latent variables, and randomly sample a new point \mathbf{h}^* for generating views using the kernel smoother ($n_r = 3$) technique as explained above. The neural network architecture for constructing an explicit feature map consists of a hidden layer of 6 neurons with PReLU activation functions (parameter value = 0.001) and an output layer of 6 neurons with linear function. We chose a rather basic architecture since our aim is to show the applicability of the method. The training procedure in case of neural networks consisted of minimizing \mathcal{J}_t using the fmincon function in MATLAB 2018b. The weights and biases were initialized randomly from a Gaussian distribution $\mathcal{N}(0, 1)$ and within each iteration of the minimizer, Eq. (3) is solved to update the value of \mathbf{H} .

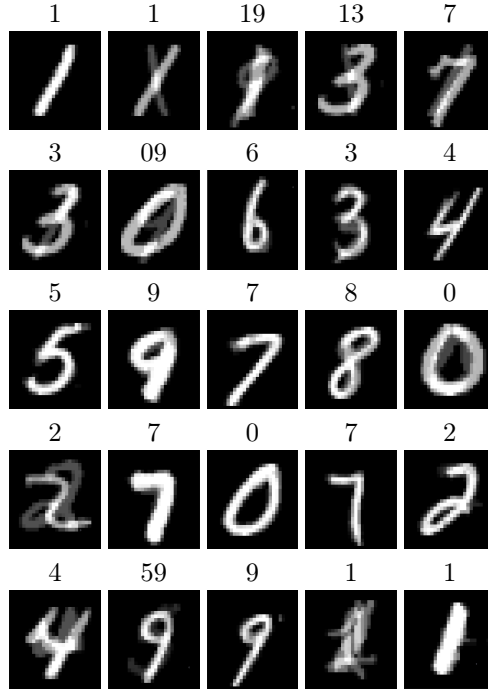


Figure 3: Image and label generation using the kernel smoother technique to construct the pre-image. The numbers above images show the generated label, where multiple labels above some images show the corresponding mixing of images.

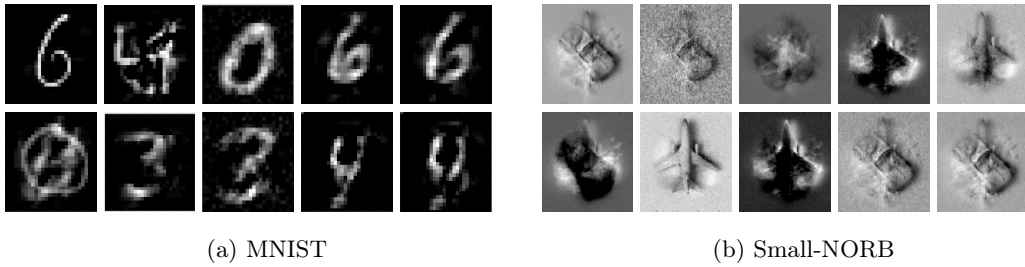


Figure 4: Image generation using neural networks as feature map.

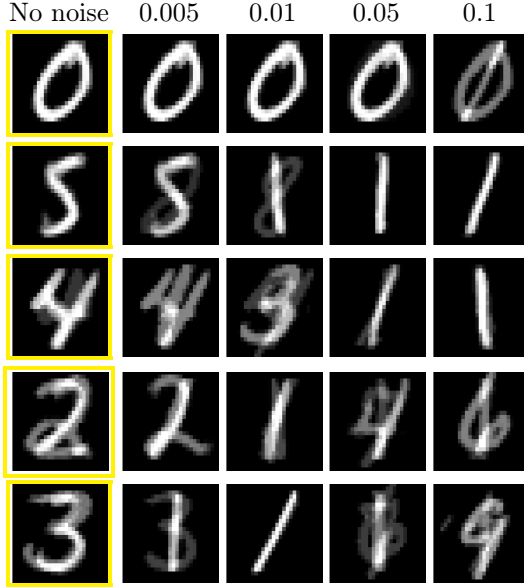


Figure 5: Targeted image generation through corresponding latent variable. First column (yellow bounding box) shows the generated images without noise addition to latent variable. The second, third, fourth and fifth columns show generated images when Gaussian noise $\mathcal{N}(0, \sigma^2)$ with $\sigma = 0.005, 0.01, 0.05$ and 0.1 respectively is added to the latent variable.

4.1 MNIST

The MNIST dataset contains 60,000 training and 10,000 testing images (28×28 pixels) of ten hand-written digits (0-9) along with the label information. We pre-processed the images by standardization, and original labels were transformed into a 10-dimensional array with 0-1 encoding. In our experiments, we use view \mathcal{X} as the images and view \mathcal{Y} as the label information. We trained the model on 5000 samples, with 500 samples from each class. Figure 3 shows the image and label generation using kernel smoother method. We can see that both the generated image and the generated label matches in most cases. Multiple digits on top of some images shows the mixing. For example, label 13 shows the mixing of images 1 and 3. Such label generation became possible thanks to 0-1 encoding scheme. Figure 4a shows the images generated when the feature map was a feed-forward neural network with architecture and parameters as described above.

4.2 Small-NORB

This dataset contains images and labels of 3D toys belonging to 5 generic categories: four-legged animals, human figures, airplanes, trucks, and cars. The images were taken by two cameras under 6 lighting conditions, 9 elevations (30 to 70 degrees every 5 degrees), and 18 azimuths (0 to 340 every 20 degrees) [26]. Each image is 96×96 pixels with integer greyscale values in range $[0, 255]$. We trained our models without any pre-processing on the original images of 2 classes i.e. cars and airplanes. Figure 4b shows the images generated when the neural network was used as the feature map.

4.3 Targeted Generation

Since every latent variable \mathbf{h}_i corresponds to a data-point, it can be selected to generate the corresponding data in the input space. This could be useful in critical applications

where data needs to be generated based on some prior-knowledge. Additionally, noise could be added to the latent variable and new data could be generated similar to the desired one. We demonstrate the targeted generation on the MNIST dataset using the kernel smoother technique (see Fig. 5). Latent variables corresponding to 0, 5, 4 and 3 were selected and the generated images are shown in the yellow bounding boxes. Then a Gaussian noise with mean 0 and variance 0.005, 0.01, 0.05 and 0.1 was added to the latent variables and the corresponding generated images are shown in the following columns. One can observe that as the noise variance is gradually increased, the generated images change from similar to a completely new image. The only exception is the case of 0, where the image generation is more robust towards noise.

5 Conclusion and future work

The paper shows a non-probabilistic framework that extends the RKM mechanism to multiple sources. This allows for a generative mechanism where the feature map can be defined using kernel functions or neural network based methods. When using kernel functions, the training consists of only solving an eigenvalue problem. In the case of a neural network based model, training involves an alternating minimization scheme. Experiments on the MNIST and Small-NORB datasets show that the model is capable of generating new images. Furthermore, a targeted generation mechanism is demonstrated using kernel-smoother method where adding noise to the latent variable generates similar images. Extensions of this work consists of training with more advanced models like Convolutional Neural Networks, secondly the effect of other pre-image methods would be investigated. This paper has demonstrated the applicability of the Generative Restricted Kernel Machine framework, suggesting the new research directions to be worth exploring.

Acknowledgments

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. The research is further supported by Research Council KUL: Optimization frameworks for deep kernel machines C14/18/068 and Flemish Government: FWO project GOA4917N (Deep Restricted Kernel Machines: Methods and Foundations), PhD/Postdoc grant.

References

- [1] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11:3371–3408, 2010.
- [2] Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1515–1528, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.
- [3] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine Learning - ICML '07*, pages 791–798, Corvallis, Oregon, 2007. ACM Press.
- [4] Raymond A. Yeh, Chen Chen, Teck Yian Lim, Alexander G. Schwing, Mark Hasegawa-Johnson, and Minh N. Do. Semantic image inpainting with deep generative models. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [5] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.
- [6] P. Smolensky. Parallel distributed processing: Explorations in the microstructure of cognition, vol. 1. chapter Information Processing in Dynamical Systems: Foundations of Harmony Theory, pages 194–281. MIT Press, Cambridge, MA, USA, 1986.
- [7] Ruslan Salakhutdinov and Geoffrey Hinton. Deep Boltzmann Machines. *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, Volume 5 of JMLR, 2009.
- [8] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014.
- [9] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16*, pages 1747–1756. JMLR.org, 2016.
- [10] Yunchen Pu, Zhe Gan, Ricardo Henao, Xin Yuan, Chunyuan Li, Andrew Stevens, and Lawrence Carin. Variational autoencoder for deep learning of images, labels and captions. In *Proceedings of the 30th International Conference on Neural Information Processing Systems, NIPS'16*, pages 2360–2368. Curran Associates Inc., USA, 2016.
- [11] Ming-Yu Liu and Oncel Tuzel. Coupled Generative Adversarial Networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 469–477. Curran Associates, Inc., 2016.
- [12] Johan A. K. Suykens. Deep Restricted Kernel Machines Using Conjugate Feature Duality. *Neural Computation*, 29(8):2123–2163, August 2017.

- [13] Johan A. K. Suykens, Tony Van Gestel, Jos De Brabanter, Bart De Moor, and Joos Vandewalle. *Least Squares Support Vector Machines*. World Scientific, River Edge, NJ, January 2002.
- [14] Lynn Houthuys and Johan A K Suykens. Tensor-based Restricted Kernel Machines for Multi-View Classification. In *27th International Conference on Artificial Neural Networks ICANN, Rhodes, Greece, October, 2018*, volume 11140, pages 205–215.
- [15] Joachim Schreurs and Johan A. K. Suykens. Generative Kernel PCA. In *ESANN*, 2018.
- [16] Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted Boltzmann machines. In *Proceedings of the 25th International Conference on Machine Learning - ICML '08*, pages 536–543, Helsinki, Finland, 2008. ACM Press.
- [17] K. B. Petersen and M. S. Pedersen. *The Matrix Cookbook*. Technical University of Denmark, nov 2012. Version 20121115.
- [18] James Mercer. Functions of positive and negative type, and their connection the theory of integral equations. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 209(441-458):415–446, January 1909.
- [19] Bernhard Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.
- [20] Sebastian Mika, Bernhard Schölkopf, Alex Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch. Kernel pca and de-noising in feature spaces. In *Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II*, pages 536–542, Cambridge, MA, USA, 1999. MIT Press.
- [21] Anh Tuan Bui, Joon-Ku Im, Daniel W. Apley, and George C. Runger. Projection-free kernel principal component analysis for denoising. *Neurocomputing*, 2019.
- [22] James T. Kwok and Ivor Wai-Hung Tsang. The pre-image problem in kernel methods. *IEEE Transactions on Neural Networks*, 15:1517–1525, 2003.
- [23] P. Honeine and C. Richard. Preimage Problem in Kernel-Based Machine Learning. *IEEE Signal Processing Magazine*, 28(2):77–88, March 2011.
- [24] Jason Weston, Bernhard Schölkopf, and Gökhan H. Bakir. Learning to Find Pre-Images. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 449–456. MIT Press, 2004.
- [25] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In Geoffrey J. Gordon, David B. Dunson, and Miroslav Dudík, editors, *AISTATS*, volume 15 of *JMLR Proceedings*, pages 315–323. JMLR.org, 2011.
- [26] Y. LeCun, Fu Jie Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–97–104 Vol.2, 2004.