ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

Generative Kernel PCA

Joachim Schreurs and Johan A.K. Suykens *

KU Leuven - Department of Electrical Engineering ESAT-STADIUS
Kasteelpark Arenberg 10 B-3001 Leuven - Belgium

Abstract. Kernel PCA has shown to be a powerful feature extractor within many applications. Using the Restricted Kernel Machine formulation, a representation using visible and hidden units is obtained. This enables the exploration of new insights and connections between Restricted Boltzmann machines and kernel methods. This paper explores these connections, introducing a generative kernel PCA which can be used to generate new data, as well as denoise a given training dataset. This in a non-probabilistic setting. Moreover, relations with linear PCA and a pre-image reconstruction method are introduced in this paper.

1 Introduction

Generative models have seen a rise in popularity the past decades, being used in applications as image generation [1], collaborative filtering [2] and denoising [3]. A commonly used method in these applications is the Restricted Boltzmann Machine (RBM) [4, 5], which is a specific type of Markov random field. RBM's are generative stochastic artificial neural networks that learn the probability distribution over a training dataset. Similar to RBM's, kernel PCA is a nonlinear feature extractor which is trained in a unsupervised way [6]. Probabilistic approaches to kernel PCA are [7, 8]. Suykens proposed a new framework of Restricted Kernel Machines (RKM) [9], which yields a representation of kernel methods with visible and hidden units. This is related to the energy form of RBM's, only in a non-probabilistic setting. By leveraging the RKM representation and the similarity between RBM's and kernel PCA, a generative mechanism is proposed in this paper.

2 Generative Kernel PCA

In this section, the generative kernel PCA formulation is deduced. We start from the RKM representation of kernel PCA (see equation (3.24) in [9]), that gives an upper bound on the original kernel PCA objective function. Given the training data $\{v_i\}_{i=1}^N$:

$$\bar{J}_{\text{train}}(h_i, W) = -\sum_{i=1}^{N} v_i^{\text{T}} W h_i + \frac{\lambda}{2}\sum_{i=1}^{N} h_i^{\text{T}} h_i + \frac{\eta}{2}\text{Tr}(W^{\text{T}} W),$$

*Research supported by Research Council KUL: CoE PFV/10/002 (OPTEC), PhD/Postdoc grants Flemish Government; FWO: projects: G0A4917N (Deep restricted kernel machines), G.088114N (Tensor based data similarity) ; Belgian Federal Science Policy Office: IUAP P7/19 (DYSCO, Dynamical systems, control and optimization, 2012-2017).

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

where $v_i \in \mathbb{R}^d$ represents the visible unit and $h_i \in \mathbb{R}^s$ the corresponding hidden unit for the ith datapoint, $W \in \mathbb{R}^{d \times s}$ the interaction matrix. Similar to RBM's, the input patterns are clamped to the visible units in order to learn the hidden units and interaction matrix. Stationary points of the training function $\bar{J}_{\text{train}}(h_i, W)$ are given by:

$$\frac{\partial \bar{J}_{\text{train}}}{\partial h_i} = 0 \Rightarrow W^{\text{T}} v_i = \lambda h_i, \ \forall i \tag{1}$$

$$\frac{\partial \bar{J}_{\text{train}}}{\partial W} = 0 \Rightarrow W = \frac{1}{\eta} \sum_{i=1}^{N} v_i h_i^{\text{T}}. \tag{2}$$

Elimination of W results in the following eigenvalue problem, which corresponds to the original linear kernel PCA formulation [6]:

$$\frac{1}{\eta} K H^{\text{T}} = H^{\text{T}} \Delta,$$

where $H = [h_1, \ldots, h_N] \in \mathbb{R}^{s \times N}$, $\Delta = \text{diag}\{\lambda_1, \ldots, \lambda_s\}$ with $s \leq N$ the number of selected components and $K_{ij} = v_i^{\text{T}} v_j$ the kernel matrix elements. This can easily be extended to the nonlinear case by using the feature map and kernel trick by replacing v_i by $\varphi(v_i)$ [9].

After training the model, we should be able to re-generate the visible units of the training dataset. The hidden units h_i and interaction matrix W are assumed to be known from training the model and correspond to equations (1) and (2). We propose the following generating objective function, by introducing a regularization term on the visible units:

$$\bar{J}_{\text{gen}}(v_i) = -\sum_{i=1}^{N} v_i^{\text{T}} W h_i + \frac{1}{2} \sum_{i=1}^{N} v_i^{\text{T}} v_i.$$

Stationary points of the generating objective function $\bar{J}_{\text{gen}}(v_i)$ are given by:

$$\frac{\partial \bar{J}_{\text{gen}}}{\partial v_i} = 0 \Rightarrow W h_i = v_i, \ \forall i.$$

One can easily see that filling in the hidden features h_i and W of the training phase, results in the original visible units v_i.

A clear link with RBM's is visible [4, 5]. Similar as in RBM's, there first occurs a training phase to find the hidden units, weights and biases. Using the conditional distributions $p(h|v, \theta)$ and $p(v|h, \theta)$, the contrastive divergence algorithm is used to optimize the model parameters θ [10]. The algorithm makes use of Gibbs sampling inside a gradient descent procedure to compute weight updates. After the RBM is trained, the model can be used to generate new samples. Given a visible unit v, the model returns a hidden unit h and vice-versa. This mechanism is made possible by the energy function of the RBM [4, 5]:

$$E(v, h; \theta) = -v^{\text{T}} W h - c^{\text{T}} v - a^{\text{T}} h,$$

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

with model parameters $\theta = \{W, c, a\}$. The same property is present in the RKM objective function, which is a combination of $\bar{J}_{\text{train}}$ and $\bar{J}_{\text{gen}}$:

$$\bar{J}(v, h, W) = -v^{\mathrm{T}} W h + \frac{\lambda}{2} h^{\mathrm{T}} h + \frac{1}{2} v^{\mathrm{T}} v + \frac{\eta}{2} \mathrm{Tr}(W^{\mathrm{T}} W), \tag{3}$$

which can be seen as a non-probabilistic variant of the RBM energy function. The training phase however consists of solving an eigenvalue problem. For generating the visible units, a matrix multiplication is needed in the RKM case.

Generative kernel PCA can be used to generate new data. Instead of using the hidden units of the training set, one could use a new hidden unit $h^\star$. Similar to RBM's, a new hidden unit is clamped to the model. We propose generating new hidden units by fitting a normal distribution through the trained hidden units, with afterwards sampling from this distribution $p(h)$. As shown by Suykens et al. [11], kernel PCA corresponds to a one-class LS-SVM problem with zero target value around which one maximizes the variance. This property results in the hidden variables most typically having a normal distribution around zero (however for generating new hidden units other distributions are possible). The optimization problem, where W is obtained by the training phase in equation (2) and $h^\star$ is sampled from a normal distribution, corresponds to:

$$\bar{J}_{\text{gen}}(v^\star) = -{v^\star}^{\mathrm{T}} W h^\star + \frac{1}{2} {v^\star}^{\mathrm{T}} v^\star,$$

with $v^\star$ generated by equation:

$$v^\star = W h^\star. \tag{4}$$

3 Dimensionality reduction and denoising

3.1 Linear case

In the linear case, let us take the visible units equal to the training points $v_i = x_i$. When using a subset of the trained hidden features $H \in \mathbb{R}^{s \times N}$ and trained interaction matrix W (see equations (1) and (2)) to re-generate the original dataset:

$$\hat{X} = WH = (\frac{1}{\eta} \sum_{i=1}^{N} x_i h_i^{\mathrm{T}}) H = \frac{1}{\eta} X H^{\mathrm{T}} H, \tag{5}$$

with training dataset $X \in \mathbb{R}^{d \times N}$. This corresponds to minimizing the reconstruction error $\|X - \hat{X}\|^2$. The above equation is also equal to reconstruction or denoising using linear PCA [12].

3.2 Nonlinear case

In the nonlinear case, the visible units are equal to the feature map of the data points $v_i = \varphi(x_i)$ where $\varphi(x_i) : \mathbb{R}^d \to \mathbb{R}^{n_f}$ is assumed to be a centered feature

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

map [6]. A new datapoint $x^\star$ is generated using the known corresponding hidden unit $h^\star$. The generative equation (4) becomes:

$$\varphi(x^\star) = Wh^\star = (\frac{1}{\eta}\sum_{i=1}^{N}\varphi(x_i)h_i^{\mathrm{T}})h^\star,$$

where W is the trained interaction matrix of equation (2) and h_i the trained hidden unit of equation (1). However the above equation requires $\varphi(x_i)$ in its explicit form. Finding the value original datapoint based on the mapping in the feature space is known as the pre-image problem [13].

To solve this problem, we propose to multiply both sides of the equation with the feature map of every training-point $\varphi(x_j)$: $(\varphi(x_j)\cdot\varphi(x^\star)) = \frac{1}{\eta}(\varphi(x_j)\cdot\sum_{i=1}^{N}\varphi(x_i)h_i^{\mathrm{T}})h^\star$, where $j = 1,\ldots,N$. This results in the following equation:

$$K(x_j, x^\star) = \frac{1}{\eta}(\sum_{i=1}^{N}K(x_j, x_i)h_i^{\mathrm{T}})h^\star, \tag{6}$$

where $K(x_j, x^\star) = (\varphi(x_j)\cdot\varphi(x^\star))$ is a centered kernel function. Instead of explicitly calculating the feature map of the point $x^\star$, the kernel or similarity to the training-points is calculated. Using above equation to re-generate the kernel matrix K of the training dataset, the denoised similarities $\hat{K}$ are calculated:

$$\hat{K} = \frac{1}{\eta}KH^{\mathrm{T}}H, \tag{7}$$

where $H \in \mathbb{R}^{s\times N}$ is a subset of the trained hidden units with $s \leq N$. A similar pattern occurs as in equation (5).

We propose to use these similarities in a kernel smoother approach [14], however other mechanisms are possible. The estimated value $\hat{x}$ for $x^\star$ is now equal to:

$$\hat{x} = \frac{\sum_{j=1}^{S}\tilde{K}(x_j, x^\star)x_j}{\sum_{j=1}^{S}\tilde{K}(x_j, x^\star)}, \tag{8}$$

where $\tilde{K}(x_j, x^\star)$ is the scaled similarity between 0 and 1 calculated in equation (6) and a design parameter $S \leq N$, the S closest points based on the similarity $\tilde{K}(x_j, x^\star)$. Kernel smoothing often works with a localized kernel like the RBF kernel, where the second design parameter is the bandwidth $\tilde{\sigma}$.

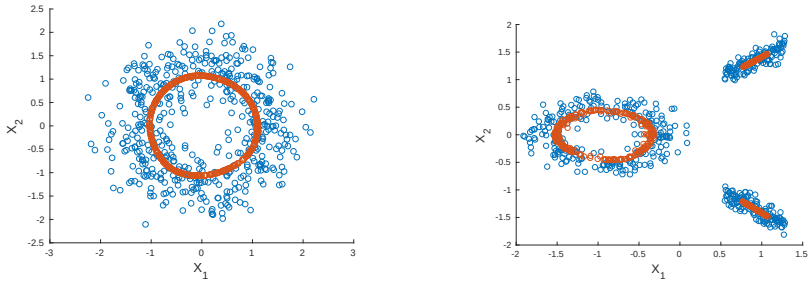
4 Illustrative examples

Denoising example (Figure 1). In a first experiment, we consider the dataset $X \in \mathbb{R}^{2\times 500}$ of a unit circle with Gaussian noise $\sigma = 0.3$. Kernel PCA is applied to the dataset, using an RBF kernel with $\tilde{\sigma}^2 = 1$. Using the first 2 principal components, the similarities with other points of the dataset are calculated using equation (7). The pre-image is determined using the kernel smoother of

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

equation (8), where the $S = 150$ closed points are used. The same procedure is repeated on a second dataset $X \in \mathbb{R}^{2\times 500}$ of a unit circle and two lines with Gaussian noise $\sigma = 0.2$ using an RBF kernel with $\tilde{\sigma}^2 = 0.2$, $S = 100$ and reconstruction with the first 8 principal components.

Generating new data example (Figure 2). In a first experiment, we try to generate a new digit using the MNIST handwritten digits dataset. 50 images each are taken for the digits 0 and 1, afterwards kernel PCA using an RBF kernel with $\tilde{\sigma}^2 = 50$ is performed on this small subsample. A normal distribution was fitted through the hidden units of the training data and used to generate a new hidden unit $h^\star$. Afterwards, the similarities of the new datapoint $x^\star$ with the digits 0 and 1 where calculated using equation (6) with the first 20 principal components. The kernel smoother uses these similarities to generate a new digit using equation (8), where $S = 10$. By choosing only the 10 most similar images, only zeros are used in the smoothing. In a second experiment, the same procedure is repeated with a subsample of 50 images for every digit using kernel PCA with an RBF kernel with $\tilde{\sigma}^2 = 0.01$, $S = 100$ and the first 50 principal components. Figure 2 shows the newly generated digit, that most resembles the digit 0 and 8. This corresponds with the average scaled similarity that is the highest for digits 0 and 8. By using a higher $S = 100$, images of all digits are used in the smoothing procedure.



(a) (b)

Fig. 1: Denoising method of section 3.2: (a) the first 2 principal components are used; (b) the first 8 principal components are used.

5 Conclusion

In this paper, a generative kernel PCA is introduced. The method is based on the RKM formulation [9]. Under this framework, kernel PCA is related to the energy form of RBM. This paper builds upon this premise, by presenting a similar generative mechanism. We consider two different cases. Firstly denoising, where the hidden units of the training dataset are used. Secondly generating new data, where new hidden units are sampled from a normal distribution. To solve the pre-image problem, a kernel smoothing method is proposed. In future work, we

ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6.
Available from http://www.i6doc.com/en/.

(a) (b) (c)

Fig. 2: Generative method of section 3.2: (a) shows a newly generated digit 0; (b) shows a mixing between all digits; (c) displays the average scaled similarity to every digit of the newly generated digit in Figure (b).

want to expand the method to deep generative kernel PCA. Secondly, we aim to extend this generative mechanism to RKM classification and regression.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[2] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th ICML*, pages 791–798, 2007.

[3] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2013.

[4] Geoffrey Hinton. A practical guide to training restricted Boltzmann machines. *Momentum*, 9(1):926, 2010.

[5] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[6] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *ICANN*, pages 583–588. Springer, 1997.

[7] Neil Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *Journal of machine learning research*, 6:1783–1816, 2005.

[8] Zhihua Zhang, Gang Wang, Dit-Yan Yeung, and James T Kwok. Probabilistic kernel principal component analysis. *Department of Compputer Science, The Hong Kong University of Science and Technology, Tech. Rep*, 2004.

[9] Johan A.K. Suykens. Deep Restricted Kernel Machines using Conjugate Feature Duality. *Neural Computation*, 29(8):2123–2163, 2017.

[10] Asja Fischer and Christian Igel. Training restricted Boltzmann machines: An introduction. *Pattern Recognition*, 47(1):25–39, 2014.

[11] Johan A.K. Suykens, Tony Van Gestel, Joos Vandewalle, and Bart De Moor. A support vector machine formulation to PCA analysis and its kernel version. *IEEE Transactions on neural networks*, 14(2):447–450, 2003.

[12] Ian T Jolliffe. *Principal component analysis*. Springer, 1986.

[13] Paul Honeine and Cedric Richard. Preimage problem in kernel-based machine learning. *IEEE Signal Processing Magazine*, 28(2):77–88, 2011.

[14] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.