Deep Learning, Neural Networks and Kernel Machines

Johan Suykens

KU Leuven, ESAT-Stadius & Leuven.AI Institute
Kasteelpark Arenberg 10, B-3001 Leuven, Belgium
Email: johan.suykens@esat.kuleuven.be
http://www.esat.kuleuven.be/stadius/

Deeplearn 2021, Gran Canaria, July 2021

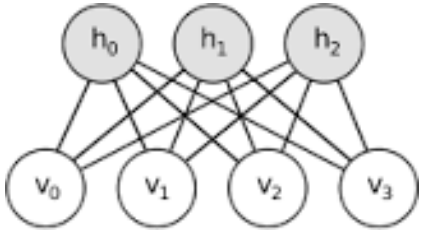


Part II: RBMs, Gen-RKM and deep learning

- Restricted Boltzmann Machines (RBM)
- Deep Boltzmann Machines (Deep BM)
- Restricted Kernel Machines (RKM)
- Deep RKM (see Part III)
- Generative RKM (Gen-RKM)

Generative models: RBM, GAN and deep learning

Restricted Boltzmann Machines (RBM)



- Markov random field, bipartite graph, stochastic binary units
  Layer of visible units v and layer of hidden units h
  No hidden-to-hidden connections
- Energy:

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

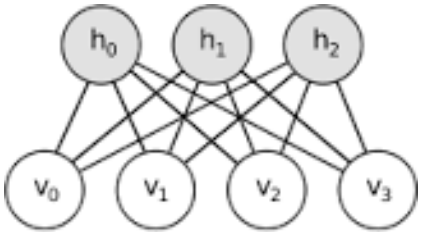
Joint distribution:

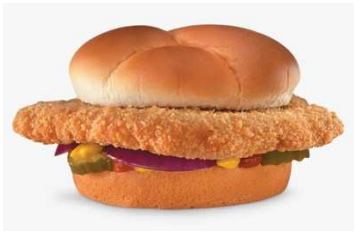
$$P(v, h; \theta) = \frac{1}{Z(\theta)} \exp(-E(v, h; \theta))$$

with partition function $Z(\theta) = \sum_v \sum_h \exp(-E(v, h; \theta))$

[Hinton, Osindero, Teh, Neural Computation 2006]

Restricted Boltzmann Machines (RBM)





- Markov random field, bipartite graph, stochastic binary units
  Layer of visible units v and layer of hidden units h
  No hidden-to-hidden connections
- Energy:

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

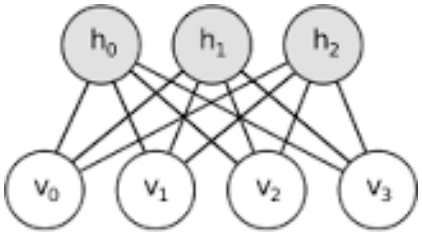
Joint distribution:

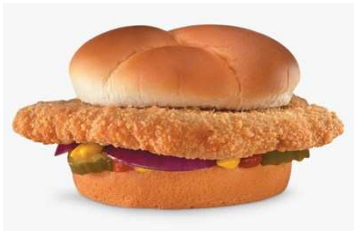
$$P(v,h;\theta) = \frac{1}{Z(\theta)} \exp(-E(v,h;\theta))$$

with partition function $Z(\theta) = \sum_v \sum_h \exp(-E(v,h;\theta))$

[Hinton, Osindero, Teh, Neural Computation 2006]

Restricted Boltzmann Machines (RBM)





- Markov random field, bipartite graph, stochastic binary units
  Layer of visible units v and layer of hidden units h
  No hidden-to-hidden connections
- Energy:

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

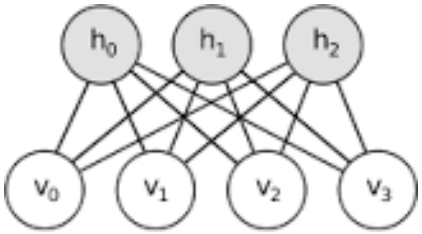
Joint distribution:

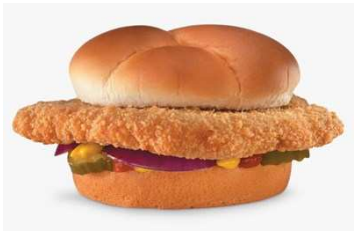
$$P(v,h;\theta) = \frac{1}{Z(\theta)} \exp(-E(v,h;\theta))$$

with partition function $Z(\theta) = \sum_v \sum_h \exp(-E(v,h;\theta))$

[Hinton, Osindero, Teh, Neural Computation 2006]

Restricted Boltzmann Machines (RBM)





- Markov random field, bipartite graph, stochastic binary units
  Layer of visible units v and layer of hidden units h
  No hidden-to-hidden connections
- Energy:

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

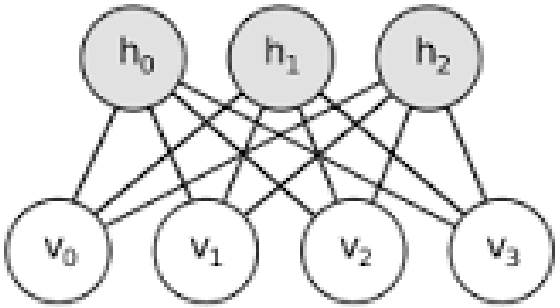
Joint distribution:

$$P(v,h;\theta) = \frac{1}{Z(\theta)} \exp(-E(v,h;\theta))$$

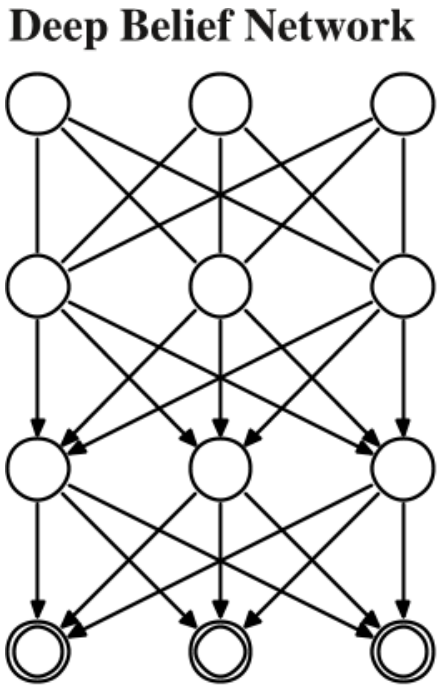
with partition function $Z(\theta) = \sum_v \sum_h \exp(-E(v,h;\theta))$

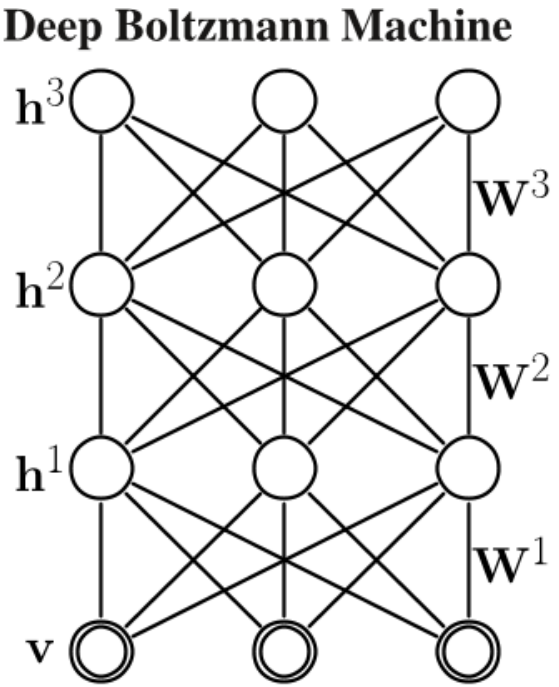
[Hinton, Osindero, Teh, Neural Computation 2006]

RBM and deep learning



$p(v, h)$

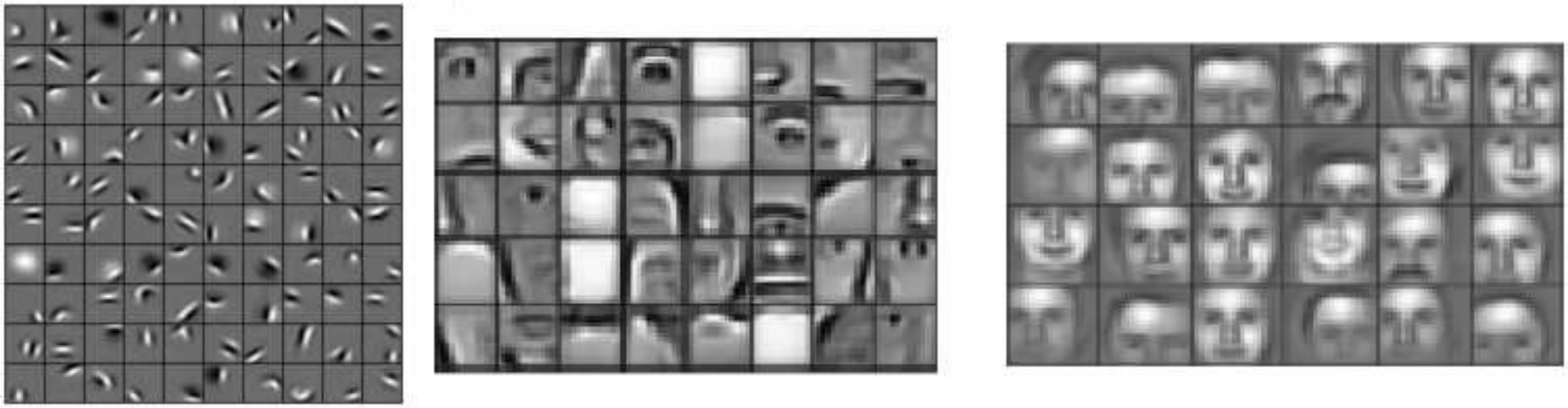




$p(v, h^1, h^2, h^3, ...)$

[Hinton et al., 2006; Salakhutdinov, 2015]

Convolutional Deep Belief Networks



Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks [Lee et al. 2011]

Energy function

- RBM:

$$E = -v^T W h$$

- Deep Boltzmann machine (two layers):

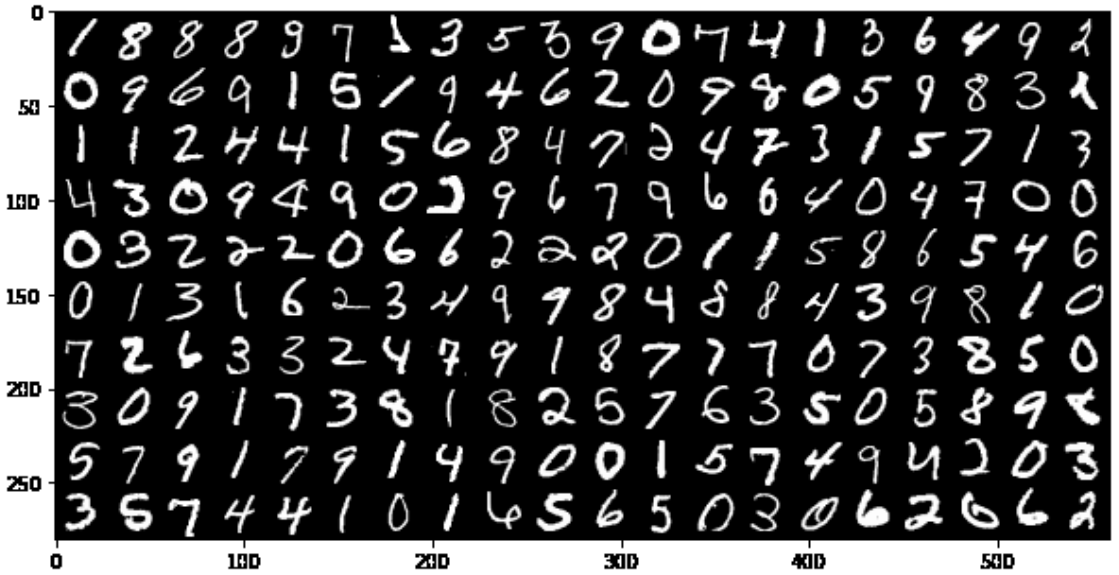
$$E = -v^T W^1 h^1 - {h^1}^T W^2 h^2$$

- Deep Boltzmann machine (three layers):

$$E = -v^T W^1 h^1 - {h^1}^T W^2 h^2 - {h^2}^T W^3 h^3$$

RBM: example on MNIST

MNIST training data:



Generating new images:



source: https://www.kaggle.com/nicw102168/restricted-boltzmann-machine-rbm-on-mnist

RBM training (1)

Thanks to the special bipartite structure, explicit **marginalization** is possible:

$$P(v;\theta) = \frac{1}{Z(\theta)} \sum_h \exp(-E(v,h;\theta)) = \frac{1}{Z(\theta)} \exp(b^T v) \prod_j (1+\exp(a_j + \sum_i W_{ij} v_j))$$

with $v_i \in \{0,1\}$, $h_i \in \{0,1\}$.

Conditional distributions:

$$P(h|v;\theta) = \prod_j p(h_j|v) \text{ with } p(h_j = 1|v) = \sigma(\sum_i W_{ij} v_i + a_j)$$

and

$$P(v|h;\theta) = \prod_i p(v_i|h) \text{ with } p(v_i = 1|h) = \sigma(\sum_j W_{ij} h_j + b_i)$$

with σ the sigmoid activation.

RBM training (2)

Given observations $\{v_n\}_{n=1}^N$, the **derivative of the log-likelihood** is

$$\begin{aligned}
\frac{1}{N}\sum_n \frac{\partial \log P(v_n;\theta)}{\partial W_{ij}} &= \mathbb{E}_{P_{\text{data}}}[v_i h_j] - \mathbb{E}_{P_{\text{model}}}[v_i h_j] \\
\frac{1}{N}\sum_n \frac{\partial \log P(v_n;\theta)}{\partial a_j} &= \mathbb{E}_{P_{\text{data}}}[h_j] - \mathbb{E}_{P_{\text{model}}}[h_j] \\
\frac{1}{N}\sum_n \frac{\partial \log P(v_n;\theta)}{\partial b_i} &= \mathbb{E}_{P_{\text{data}}}[v_i] - \mathbb{E}_{P_{\text{model}}}[v_i]
\end{aligned}$$

with

- **Data-dependent expectation** $\mathbb{E}_{P_{\text{data}}}[\cdot]$ (*form of Hebbian learning*):
  an expectation with respect to the data distribution $P_{\text{data}}(h, v; \theta) = P(h|v;\theta)P_{\text{data}}(v)$ with $P_{\text{data}}(v) = \frac{1}{N}\sum_n \delta(v - v_n)$ the empirical distribution.

- **Model's expectation** $\mathbb{E}_{P_{\text{model}}}[\cdot]$ (*unlearning*):
  an expectation with respect to the distribution defined by the model $P(v, h; \theta) = \frac{1}{Z(\theta)}\exp(-E(v, h; \theta))$.

RBM training (3)

Exact maximum likelihood learning is intractable (due to computation of $\mathbb{E}_{P_{\text{model}}}[\cdot]$). In practice, **Contrastive Divergence** (CD) algorithm [Hinton 2002]:

$$\Delta W = \alpha(\mathbb{E}_{P_{\text{data}}}[vh^T] - \mathbb{E}_{P_T}[vh^T])$$

with α learning rate and P_T a distribution defined by running a Gibbs chain initialized at the data for T full steps ($T = 1$, i.e. CD1 often in practice).

CD1 scheme:

1. Start Gibbs sampler $v^{(1)} := v_n$ and generate $h^{(1)} \sim P(h|v^{(1)})$

2. After obtaining $h^{(1)}$, generate $v^{(2)} \sim P(v|h^{(1)})$ (called fantasy data)

3. After obtaining $v^{(2)}$, generate $h^{(2)} \sim P(h|v^{(2)})$

with

$$\Delta W \propto (v_n {h^{(1)}}^T - v^{(2)} {h^{(2)}}^T)$$

Deep Boltzmann machine training (1)

Consider 3-layer Deep BM with **energy function** [Salakhutdinov 2015]:

$$E(v, h^1, h^2, h^3; \theta) = -v^T W^1 h^1 - {h^1}^T W^2 h^2 - {h^2}^T W^3 h^3$$

with unknown model parameters $\theta = \{W^1, W^2, W^3\}$.

The model assigns the following probability to a visible vector v:

$$P(v; \theta) = \frac{1}{Z(\theta)} \sum_{h^1, h^2, h^3} \exp(-E(v, h^1, h^2, h^3; \theta))$$

Deep Boltzmann machine training (2)

For training:

$$\begin{aligned}
\frac{\partial \log P(v;\theta)}{\partial W^1} &= \mathbb{E}_{P_{\text{data}}}[vh^{1^T}] - \mathbb{E}_{P_{\text{model}}}[vh^{1^T}] \\
\frac{\partial \log P(v;\theta)}{\partial W^2} &= \mathbb{E}_{P_{\text{data}}}[h^1 h^{2^T}] - \mathbb{E}_{P_{\text{model}}}[h^1 h^{2^T}] \\
\frac{\partial \log P(v;\theta)}{\partial W^3} &= \mathbb{E}_{P_{\text{data}}}[h^2 h^{3^T}] - \mathbb{E}_{P_{\text{model}}}[h^2 h^{3^T}]
\end{aligned}$$

Problem: the conditional distribution over the states of the hidden variables conditioned on the data is **no longer factorial**. For simplicity and speed one can **assume and impose a fully factorized distribution**, corresponding to a naive mean-field approximation [Salakhutdinov 2015].

Multimodal Deep Boltzmann Machine

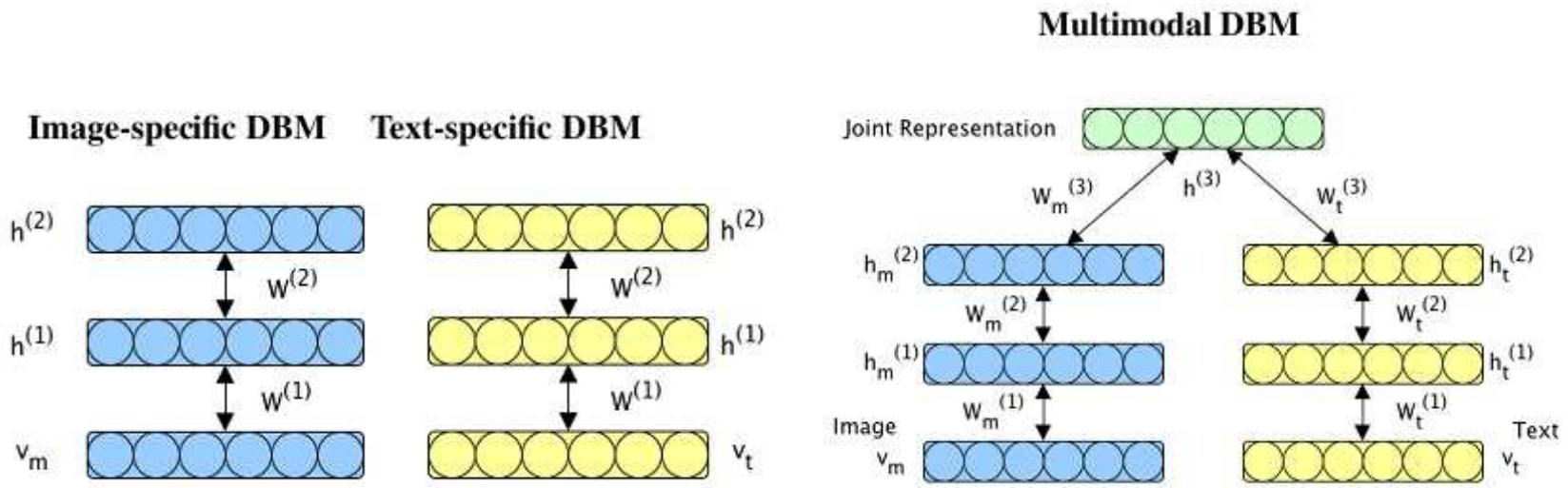


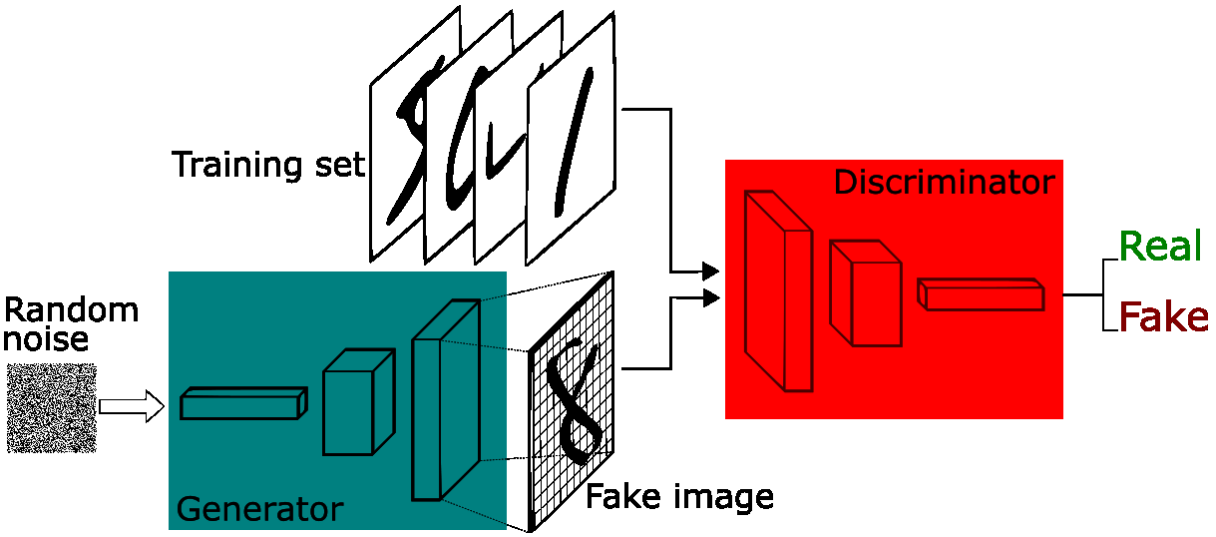
Figure 2: **Left:** Image-specific two-layer DBM that uses a Gaussian model to model the distribution over real-valued image features. **Middle:** Text-specific two-layer DBM that uses a Replicated Softmax model to model its distribution over the word count vectors. **Right:** A Multimodal DBM that models the joint distribution over image and text inputs.

From [Srivastava & Salakhutdinov 2014]

Generative Adversarial Network (GAN)

Generative Adversarial Network (GAN) [Goodfellow et al., 2014]
Training of two competing models in a zero-sum game:

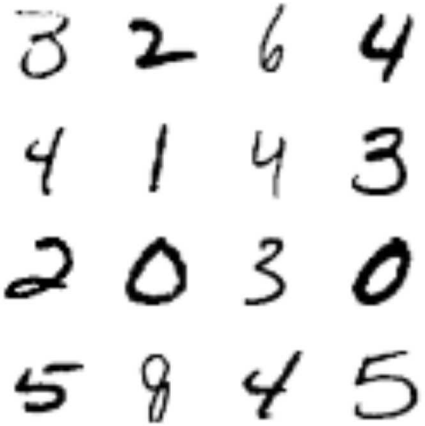
(Generator) generate fake output examples from random noise
(Discriminator) discriminate between fake examples and real examples.



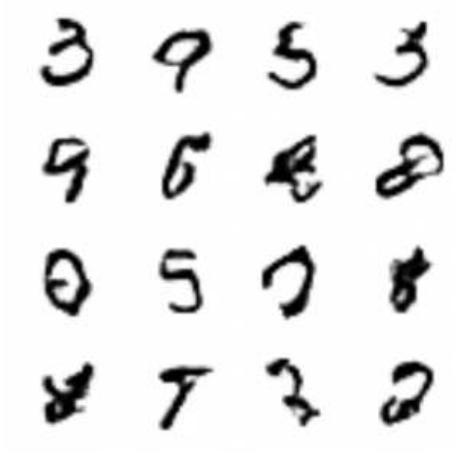
source: https://deeplearning4j.org/generative-adversarial-network

GAN: example on MNIST

MNIST training data:



GAN generated examples:



source: https://www.kdnuggets.com/2016/07/mnist-generative-adversarial-model-keras.html

Kernel methods and deep learning

Kernel machines & deep learning

previous approaches:

- kernels for deep learning [Cho & Saul, 2009]
- mathematics of the neural response [Smale et al., 2010]
- deep gaussian processes [Damianou & Lawrence, 2013]
- convolutional kernel networks [Mairal et al., 2014]
- multi-layer support vector machines [Wiering & Schomaker, 2014]
- other

Kernel machines & deep learning: New Challenges

- *new synergies* and *new foundations* between support vector machines & kernel methods and deep learning architectures?
- possible to extend primal and dual model representations (as occuring in SVM and LS-SVM models) *from shallow to deep architectures*?
- possible to handle *deep feedforward neural networks* and *deep kernel machines* within a common setting?

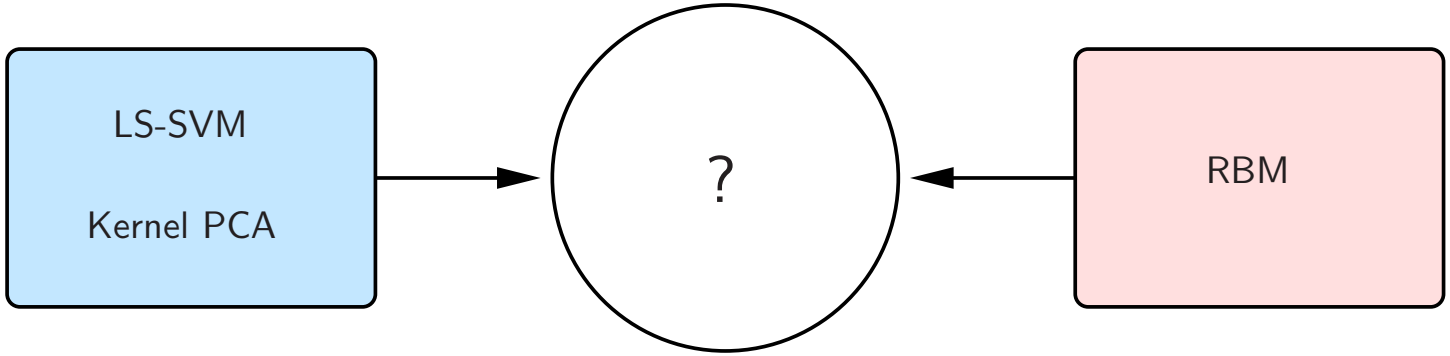
Kernel machines & deep learning: New Challenges

- *new synergies* and *new foundations* between support vector machines & kernel methods and deep learning architectures?
- possible to extend primal and dual model representations (as occuring in SVM and LS-SVM models) *from shallow to deep architectures*?
- possible to handle *deep feedforward neural networks* and *deep kernel machines* within a common setting?

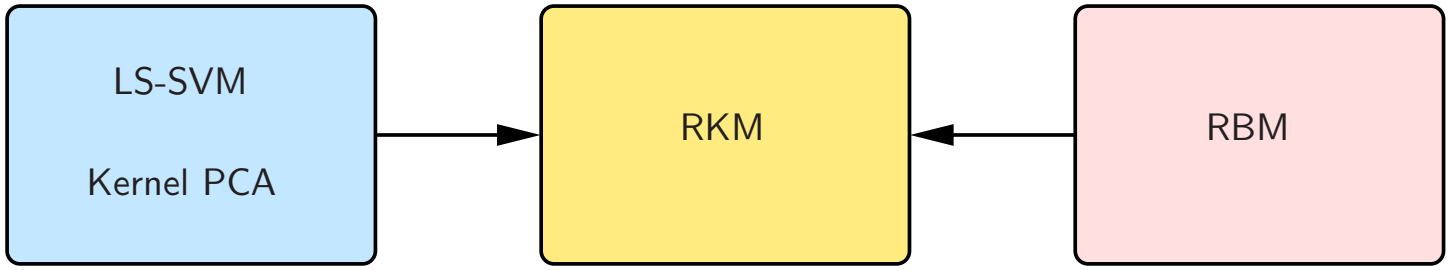
$\rightarrow$ new framework:

"Deep Restricted Kernel Machines" [Suykens, Neural Computation, 2017]
https://www.mitpressjournals.org/doi/pdf/10.1162/neco_a_00984

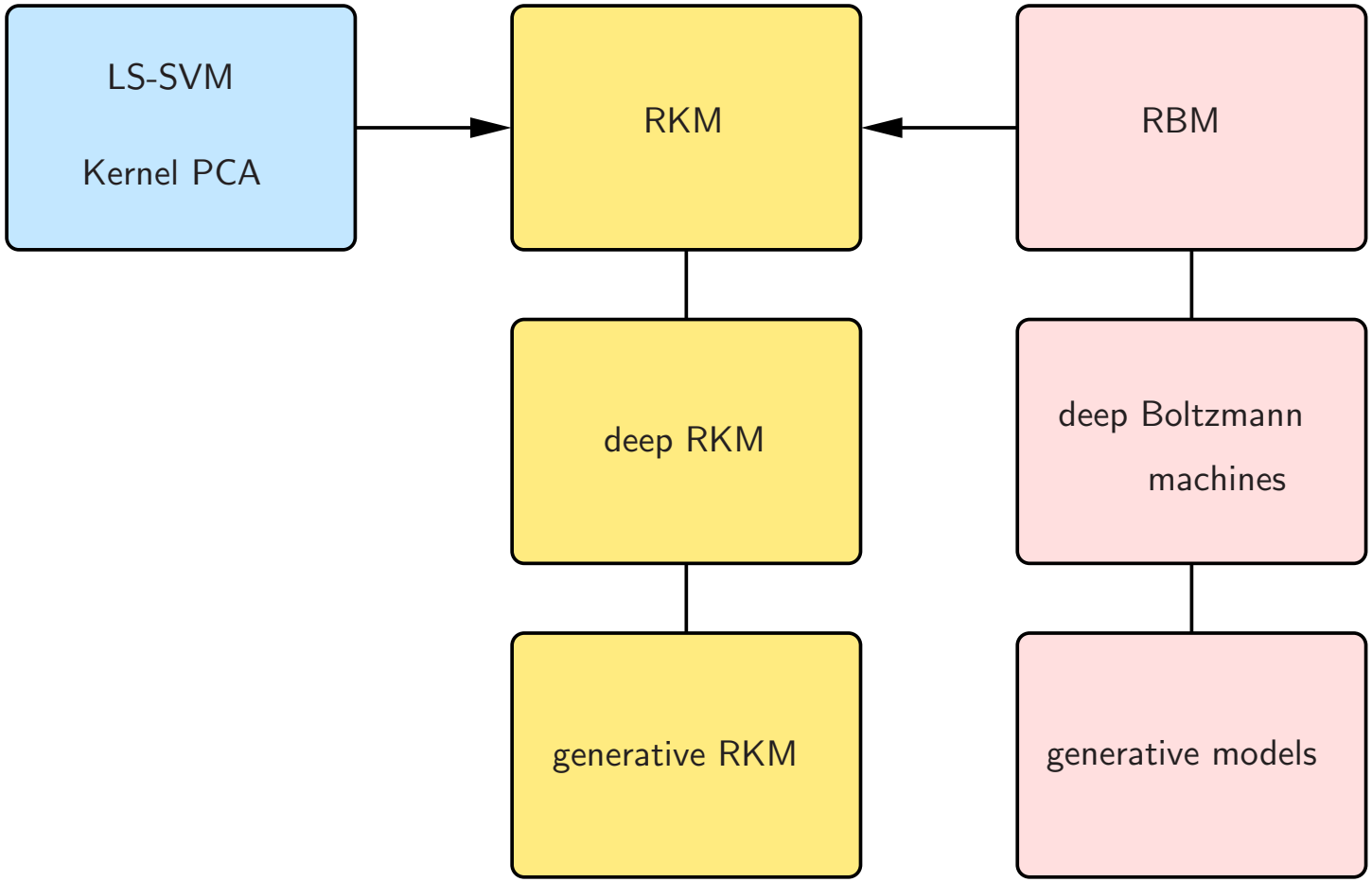
New connections



New connections



New connections



Restricted Kernel Machines

Restricted Kernel Machines (RKM)

Main characteristics:

- Kernel machine interpretations in terms of **visible and hidden units** (similar to Restricted Boltzmann Machines (**RBM**))
- Restricted Kernel Machine (**RKM**) representations for
  - LS-SVM regression/classification
  - Kernel PCA
  - Matrix SVD
  - Parzen-type models
  - other
- Based on principle of **conjugate feature duality** (with hidden features corresponding to dual variables)

LS-SVM regression model: classical approach

LS-SVM regression model, given input & output data $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$

$$\begin{array}{ll} \min\limits_{w,b,e_i} & \frac{1}{2}w^T w + \frac{\gamma}{2}\sum\limits_{i=1}^{N} e_i^2 \\ \text{subject to} & y_i = w^T\varphi(x_i) + b + e_i,\ i = 1, ..., N. \end{array}$$

Solution in Lagrange multipliers α_i:

$$\left[\begin{array}{c|c} K + I/\gamma & 1_N \\ \hline 1_N^T & 0 \end{array}\right] \left[\begin{array}{c} \alpha \\ \hline b \end{array}\right] = \left[\begin{array}{c} y_{1:N} \\ \hline 0 \end{array}\right]$$

with $K(x_i, x_j) = \varphi(x_i)^T\varphi(x_j)$, $y_{1:N} = [y_1; ...; y_N]$
and $\hat{y} = \sum_i \alpha_i K(x, x_i) + b$.

LS-SVM regression model: classical approach

LS-SVM regression model, given input & output data $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$

$$\begin{array}{ll} \min\limits_{w,b,e_i} & \frac{1}{2}w^T w + \frac{\gamma}{2}\sum\limits_{i=1}^{N} e_i^2 \\ \text{subject to} & y_i = w^T\varphi(x_i) + b + e_i,\ i = 1, ..., N. \end{array}$$

Solution in Lagrange multipliers α_i:

$$\left[\begin{array}{c|c} K + I/\gamma & 1_N \\ \hline 1_N^T & 0 \end{array}\right] \left[\begin{array}{c} \alpha \\ \hline b \end{array}\right] = \left[\begin{array}{c} y_{1:N} \\ \hline 0 \end{array}\right]$$

with $K(x_i, x_j) = \varphi(x_i)^T\varphi(x_j)$, $y_{1:N} = [y_1; ...; y_N]$
and $\hat{y} = \sum_i \alpha_i K(x, x_i) + b$.

$\rightarrow$ **How to achieve a representation with visible and hidden units?**

Conjugate feature duality

Property. For $\lambda > 0$, the following quadratic form property holds:

$$\frac{1}{2\lambda} e^T e \geq e^T h - \frac{\lambda}{2} h^T h, \quad \forall e, h \in \mathbb{R}^p$$

Conjugate feature duality

Property. For $\lambda > 0$, the following quadratic form property holds:

$$\frac{1}{2\lambda} e^T e \geq e^T h - \frac{\lambda}{2} h^T h, \quad \forall e, h \in \mathbb{R}^p$$

Proof: This is verified by writing the quadratic form as

$$\frac{1}{2} \begin{bmatrix} e^T & h^T \end{bmatrix} \begin{bmatrix} \frac{1}{\lambda} I & I \\ I & \lambda I \end{bmatrix} \begin{bmatrix} e \\ h \end{bmatrix} \geq 0, \ \forall e, h \in \mathbb{R}^p.$$

It is known that

$$Q = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \geq 0$$

if and only if $A > 0$ and the Schur complement $C - B^T A^{-1} B \geq 0$. This results into the condition $\frac{1}{2}(\lambda I - I(\lambda I) I) \geq 0$, which holds.

Conjugate feature duality

Property. For $\lambda > 0$, the following quadratic form property holds:

$$\frac{1}{2\lambda} e^T e \geq e^T h - \frac{\lambda}{2} h^T h, \quad \forall e, h \in \mathbb{R}^p$$

Proof: This is verified by writing the quadratic form as

$$\frac{1}{2} \begin{bmatrix} e^T & h^T \end{bmatrix} \begin{bmatrix} \frac{1}{\lambda} I & I \\ I & \lambda I \end{bmatrix} \begin{bmatrix} e \\ h \end{bmatrix} \geq 0, \ \forall e, h \in \mathbb{R}^p.$$

It is known that

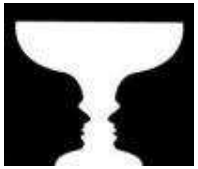
$$Q = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \geq 0$$

if and only if $A > 0$ and the Schur complement $C - B^T A^{-1} B \geq 0$. This results into the condition $\frac{1}{2}(\lambda I - I(\lambda I) I) \geq 0$, which holds.

Note. One has

$$\frac{1}{2\lambda} e^T e = \max_h (e^T h - \frac{\lambda}{2} h^T h)$$

Model: living in two worlds ...

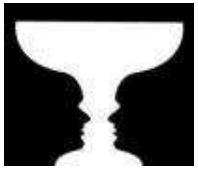


Original model:

$$\hat{y} = W^T x + b,\ e = y - \hat{y}$$

objective J
$=$ regularization term $\mathrm{Tr}(W^T W)$
$+$ $(\frac{1}{\lambda})$ error term $\sum_i e_i^T e_i$

Model: living in two worlds ...



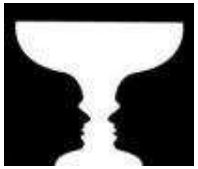
Original model:

$\hat{y} = W^T x + b,\ e = y - \hat{y}$

objective J
$=$ regularization term $\mathrm{Tr}(W^T W)$
$+$ $(\frac{1}{\lambda})$ error term $\sum_i e_i^T e_i$

$\downarrow \quad \frac{1}{2\lambda} e^T e \geq e^T h - \frac{\lambda}{2} h^T h$

Model: living in two worlds ...



Original model:

$\hat{y} = W^T x + b,\ e = y - \hat{y}$

objective J
$=$ regularization term $\mathrm{Tr}(W^T W)$
$+$ $(\frac{1}{\lambda})$ error term $\sum_i e_i^T e_i$

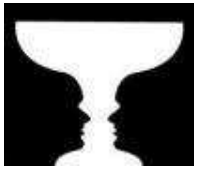
$\downarrow \quad \frac{1}{2\lambda} e^T e \geq e^T h - \frac{\lambda}{2} h^T h$

New representation:

$\hat{y} = \sum_j h_j x_j^T x + b$

obtain $J \geq \underline{J}(h_i, W, b)$
solution from stationary points of $\underline{J}$:
$\frac{\partial \underline{J}}{\partial h_i} = 0,\ \frac{\partial \underline{J}}{\partial W} = 0,\ \frac{\partial \underline{J}}{\partial b} = 0$

Model: living in two worlds ...



Original model:

$\hat{y} = W^T \varphi(x) + b,\ e = y - \hat{y}$

objective J
$=$ regularization term $\mathrm{Tr}(W^T W)$
$+$ $(\frac{1}{\lambda})$ error term $\sum_i e_i^T e_i$

$\downarrow \quad \frac{1}{2\lambda} e^T e \geq e^T h - \frac{\lambda}{2} h^T h$

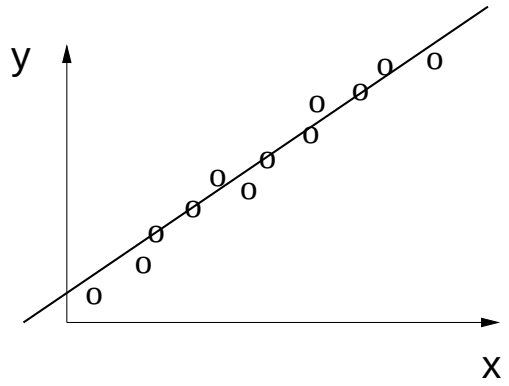
New representation:

$\hat{y} = \sum_j h_j K(x_j, x) + b$

obtain $J \geq \underline{J}(h_i, W, b)$
solution from stationary points of $\underline{J}$:
$\frac{\partial \underline{J}}{\partial h_i} = 0,\ \frac{\partial \underline{J}}{\partial W} = 0,\ \frac{\partial \underline{J}}{\partial b} = 0$

Simplest example: line fitting

Given data: $\{(x_i, y_i)\}_{i=1}^N$, $x_i, y_i \in \mathbb{R}$

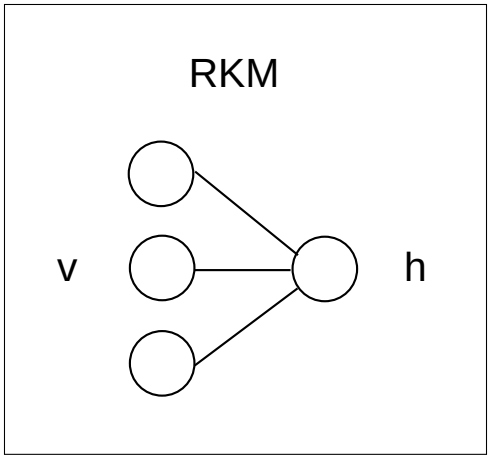


Linear model:

$$\hat{y} = wx + b, \;\; e = y - \hat{y}$$

RKM representation:

$$\hat{y} = \sum_i h_i x_i x + b$$



3 visible units: $v = [x; 1; -y]$
1 hidden unit: $h \in \mathbb{R}$

From LS-SVM to the RKM representation

Multi-output model $\hat{y} = W^T x + b$, $e = y - \hat{y}$

Objective in LS-SVM regression (linear case)

$$J = \frac{\eta}{2}\mathrm{Tr}(W^T W) + \frac{1}{2\lambda}\sum_{i=1}^{N} e_i^T e_i \text{ s.t. } e_i = y_i - W^T x_i - b, \forall i$$

From LS-SVM to the RKM representation

Multi-output model $\hat{y} = W^T x + b$, $e = y - \hat{y}$

Objective in LS-SVM regression (linear case)

$$\begin{aligned} J &= \frac{\eta}{2}\mathrm{Tr}(W^T W) + \frac{1}{2\lambda}\sum_{i=1}^{N} e_i^T e_i \text{ s.t. } e_i = y_i - W^T x_i - b, \forall i \\ &\geq \sum_{i=1}^{N} e_i^T h_i - \frac{\lambda}{2}\sum_{i=1}^{N} h_i^T h_i + \frac{\eta}{2}\mathrm{Tr}(W^T W) \text{ s.t. } e_i = y_i - W^T x_i - b, \forall i \end{aligned}$$

From LS-SVM to the RKM representation

Multi-output model $\hat{y} = W^T x + b$, $e = y - \hat{y}$

Objective in LS-SVM regression (linear case)

$$
\begin{aligned}
J &= \frac{\eta}{2}\mathrm{Tr}(W^T W) + \frac{1}{2\lambda}\sum_{i=1}^{N} e_i^T e_i \ \text{ s.t. } e_i = y_i - W^T x_i - b, \forall i \\
&\geq \sum_{i=1}^{N} e_i^T h_i - \frac{\lambda}{2}\sum_{i=1}^{N} h_i^T h_i + \frac{\eta}{2}\mathrm{Tr}(W^T W) \ \text{ s.t. } e_i = y_i - W^T x_i - b, \forall i \\
&= \sum_{i=1}^{N} (y_i^T - x_i^T W - b^T) h_i - \frac{\lambda}{2}\sum_{i=1}^{N} h_i^T h_i + \frac{\eta}{2}\mathrm{Tr}(W^T W) \triangleq \underline{J}(h_i, W, b) \\
&= R_{\mathrm{RKM}}^{\mathrm{train}} - \frac{\lambda}{2}\sum_{i=1}^{N} h_i^T h_i + \frac{\eta}{2}\mathrm{Tr}(W^T W)
\end{aligned}
$$

Connection between RKM and RBM

- RKM & RBM: interpretation in terms of **visible and hidden units**

- RKM: **energy form** as in RBM:

$$\begin{aligned} R_{\mathrm{RKM}}^{\mathrm{train}} &= \sum_{i=1}^{N} R_{\mathrm{RKM}}(v_i, h_i) \\ &= -\sum_{i=1}^{N}(x_i^T W h_i + b^T h_i - y_i^T h_i) = \sum_{i=1}^{N} e_i^T h_i \end{aligned}$$

  with $R_{\mathrm{RKM}}(v,h) = -v^T \tilde{W} h = -(x^T W h + b^T h - y^T h) = e^T h.$

- **Conjugate feature duality:** hidden features h_i are conjugated to the e_i and serve as dual variables.

From LS-SVM to RKM representation (2)

- Stationary points of $\underline{J}(h_i, W, b)$ (nonlinear case, feature map $\varphi(\cdot)$)

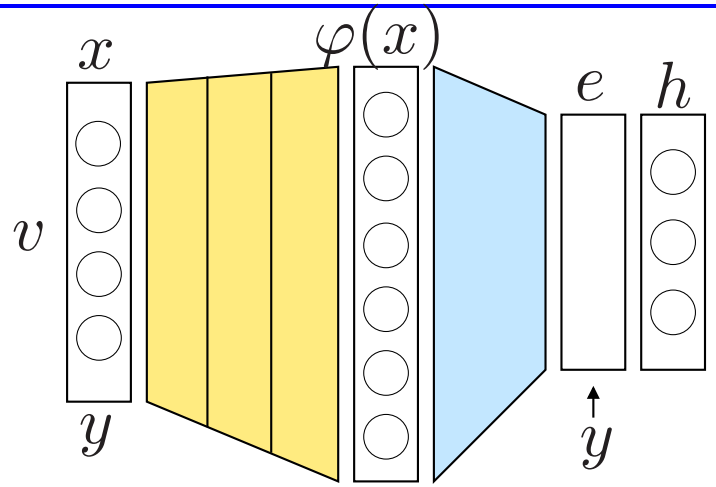
$$\begin{cases} \frac{\partial \underline{J}}{\partial h_i} = 0 & \Rightarrow \quad y_i = W^T \varphi(x_i) + b + \lambda h_i, \ \forall i \\ \frac{\partial \underline{J}}{\partial W} = 0 & \Rightarrow \quad W = \frac{1}{\eta} \sum_i \varphi(x_i) h_i^T \\ \frac{\partial \underline{J}}{\partial b} = 0 & \Rightarrow \quad \sum_i h_i = 0. \end{cases}$$

- Solution in h_i and b with positive definite kernel $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$

$$\left[\begin{array}{c|c} \frac{1}{\eta} K + \lambda I_N & 1_N \\ \hline 1_N^T & 0 \end{array}\right] \left[\begin{array}{c} H^T \\ \hline b^T \end{array}\right] = \left[\begin{array}{c} Y^T \\ \hline 0 \end{array}\right]$$

with $K = [K(x_i, x_j)]$, $H = [h_1 ... h_N]$, $Y = [y_1 ... y_N]$.

From LS-SVM to RKM representation (3)

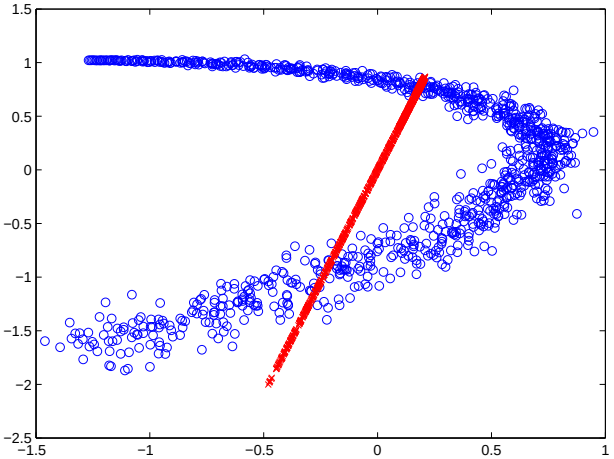


Note: $\varphi(x)$ can be multi-layered, visible units: $[\varphi(x); 1; -y]$
Conjugate feature duality: primal and dual model representations:

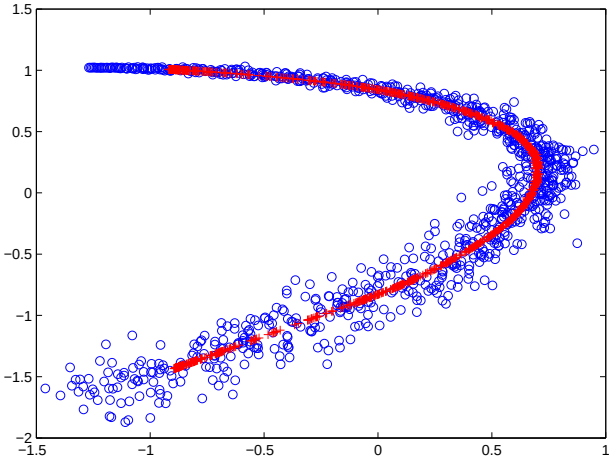
$$\mathcal{M} \begin{cases} \nearrow & (P)_{\text{RKM}}: \quad \hat{y} = W^T \varphi(x) + b \\ \searrow & (D)_{\text{RKM}}: \quad \hat{y} = \frac{1}{\eta} \sum_j h_j K(x_j, x) + b. \end{cases}$$

(large N, small d) versus (large d, small N)

Kernel principal component analysis (KPCA)



linear PCA



kernel PCA (RBF kernel)

Kernel PCA [Schölkopf et al., 1998]:
take eigenvalue decomposition of the kernel matrix

$$\begin{bmatrix} K(x_1, x_1) & \dots & K(x_1, x_N) \\ \vdots & & \vdots \\ K(x_N, x_1) & \dots & K(x_N, x_N) \end{bmatrix}$$

(applications in dimensionality reduction and denoising)

Kernel PCA: classical LS-SVM approach

- Primal problem: [Suykens et al., 2002]

$$\min_{w,b,e} \ \frac{1}{2}w^T w - \frac{1}{2}\gamma \sum_{i=1}^{N} e_i^2 \ \text{ s.t. } \ e_i = w^T \varphi(x_i) + b, \ i = 1, ..., N.$$

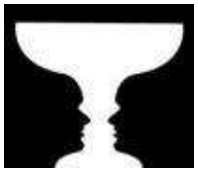
- Dual problem corresponds to kernel PCA

$$\Omega^{(c)}\alpha = \lambda\alpha \ \text{ with } \ \lambda = 1/\gamma$$

with $\Omega_{ij}^{(c)} = (\varphi(x_i) - \hat{\mu}_\varphi)^T(\varphi(x_j) - \hat{\mu}_\varphi)$ the *centered kernel matrix* and $\hat{\mu}_\varphi = (1/N)\sum_{i=1}^{N} \varphi(x_i)$.

- Interpretation:
  1. pool of candidate components (objective function equals zero)
  2. select relevant components

From KPCA to RKM representation



Model:

$e = W^T \varphi(x)$

objective J
$=$ regularization term $\mathrm{Tr}(W^T W)$
- $(\frac{1}{\lambda})$ variance term $\sum_i e_i^T e_i$

$\downarrow \quad - \frac{1}{2\lambda} e^T e \leq -e^T h + \frac{\lambda}{2} h^T h$

RKM representation:

$e = \sum_j h_j K(x_j, x)$

obtain $J \leq \overline{J}(h_i, W)$
solution from stationary points of $\overline{J}$:
$\frac{\partial \overline{J}}{\partial h_i} = 0$, $\frac{\partial \overline{J}}{\partial W} = 0$

From KPCA to RKM representation (2)

- Objective

$$
\begin{aligned}
J &= \frac{\eta}{2}\mathrm{Tr}(W^TW) - \frac{1}{2\lambda}\sum_{i=1}^{N} e_i^T e_i \text{ s.t. } e_i = W^T\varphi(x_i), \forall i \\
&\leq -\sum_{i=1}^{N} e_i^T h_i + \frac{\lambda}{2}\sum_{i=1}^{N} h_i^T h_i + \frac{\eta}{2}\mathrm{Tr}(W^TW) \text{ s.t. } e_i = W^T\varphi(x_i), \forall i \\
&= -\sum_{i=1}^{N} \varphi(x_i)^T W h_i + \frac{\lambda}{2}\sum_{i=1}^{N} h_i^T h_i + \frac{\eta}{2}\mathrm{Tr}(W^TW) \triangleq \overline{J}
\end{aligned}
$$

- Stationary points of $\overline{J}(h_i, W)$:

$$
\begin{cases}
\dfrac{\partial \overline{J}}{\partial h_i} = 0 \quad \Rightarrow \quad W^T\varphi(x_i) = \lambda h_i, \ \forall i \\
\dfrac{\partial \overline{J}}{\partial W} = 0 \quad \Rightarrow \quad W = \dfrac{1}{\eta}\displaystyle\sum_i \varphi(x_i) h_i^T
\end{cases}
$$

From KPCA to RKM representation (3)

- Elimination of W gives the eigenvalue decomposition:

$$\frac{1}{\eta} K H^T = H^T \Lambda$$

where $H = [h_1 ... h_N] \in \mathbb{R}^{s \times N}$ and $\Lambda = \mathrm{diag}\{\lambda_1, ..., \lambda_s\}$ with $s \leq N$

- Primal and dual model representations

$$\mathcal{M} \begin{cases} \nearrow & (P)_{\mathrm{RKM}}: \quad \hat{e} = W^T \varphi(x) \\ \searrow & (D)_{\mathrm{RKM}}: \quad \hat{e} = \frac{1}{\eta} \sum_j h_j K(x_j, x). \end{cases}$$

Singular value decomposition

- Objective: given x_i, z_j row and column data of (non-square) matrix

$$J = -\frac{\eta}{2}\mathrm{Tr}(V^T W) + \frac{1}{2\lambda}\sum_{i=1}^{N} e_i^T e_i + \frac{1}{2\lambda}\sum_{j=1}^{M} r_j^T r_j \quad \text{s.t.} \quad e_i = W^T\varphi(x_i), \forall i$$
$$r_j = V^T\psi(z_j), \forall j$$

- primal and dual representations (relates to non-symmetric kernels)

$$\mathcal{M} \nearrow \quad (P)_{\mathrm{RKM}}: \quad \hat{e} = W^T\varphi(x), \quad \hat{r} = V^T\psi(z)$$

$$\mathcal{M} \searrow \quad (D)_{\mathrm{RKM}}: \quad \hat{e} = \frac{1}{\eta}\sum_j h_{r_j}\psi(z_j)^T\varphi(x), \quad \hat{r} = \frac{1}{\eta}\sum_i h_{e_i}\varphi(x_i)^T\psi(z)$$

Kernel probability mass function estimation

- Objective:

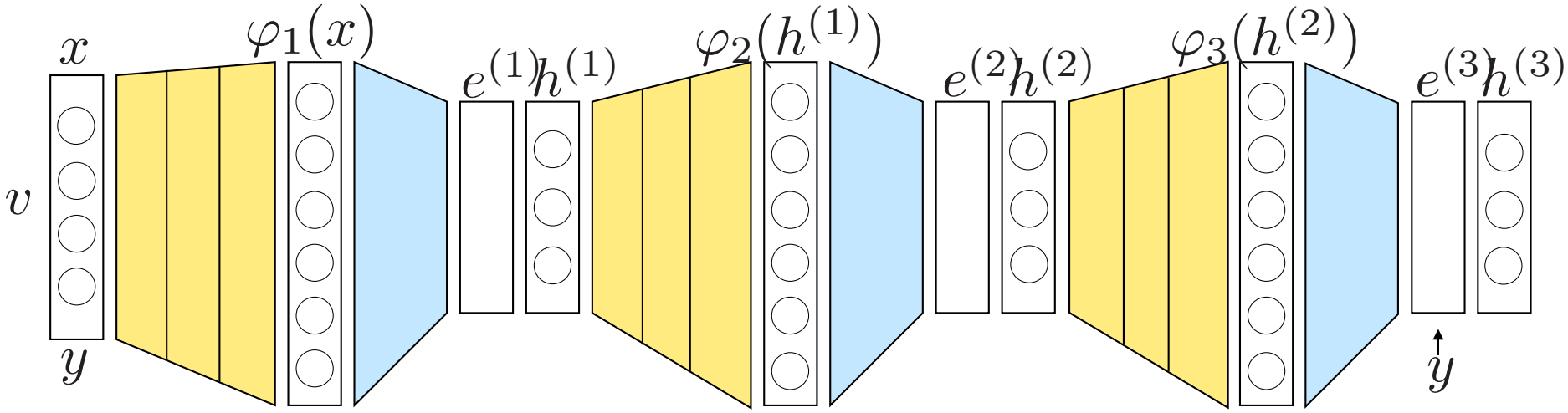
$$J = \sum_{i=1}^{N} (p_i - \varphi(x_i)^T w) h_i - \sum_{i=1}^{N} p_i + \frac{\eta}{2} w^T w$$

- primal and dual representations

$$\mathcal{M} \begin{cases} \nearrow & (P)_{\mathrm{RKM}}: \quad p_i = w^T \varphi(x_i) \\ \searrow & (D)_{\mathrm{RKM}}: \quad p_i = \frac{1}{\eta} \sum_j K(x_j, x_i) \end{cases}$$

Deep Restricted Kernel Machines

Deep RKM: example



Deep RKM: KPCA + KPCA + LSSVM

Coupling of RKMs by taking sum of the objectives

$$J_{\text{deep}} = \overline{J}_1 + \overline{J}_2 + \underline{J}_3$$

Generative kernel PCA

RKM objective for training and generating (1)

- RBM energy function

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

with model parameters $\theta = \{W, c, a\}$

- RKM objective function

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

Training:	clamp v	$\rightarrow$	$\bar{J}_{\mathrm{train}}(h, W)$
Generating:	clamp h, W	$\rightarrow$	$\bar{J}_{\mathrm{gen}}(v)$

[Schreurs & Suykens, ESANN 2018]

RKM objective for training and generating (2)

- **Training:** (clamp v)

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

Stationary points:

$$\begin{aligned} \frac{\partial \bar{J}_{\text{train}}}{\partial h_i} = 0 \quad &\Rightarrow W^{\mathrm{T}} v_i = \lambda h_i, \ \forall i \\ \frac{\partial \bar{J}_{\text{train}}}{\partial W} = 0 \quad &\Rightarrow W = \tfrac{1}{\eta}\textstyle\sum_{i=1}^{N} v_i h_i^{\mathrm{T}} \end{aligned}$$

Elimination of W:

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

where $H = [h_1, \ldots, h_N] \in \mathbb{R}^{s \times N}$, $\Delta = \mathrm{diag}\{\lambda_1, \ldots, \lambda_s\}$ with $s \leq N$ the number of selected components and $K_{ij} = v_i^{\mathrm{T}} v_j$ the kernel matrix elements.

RKM objective for training and generating (3)

- **Generating:** (clamp h, W)

  Estimate distribution $p(h)$ from $h_i, i = 1, ..., N$ (or assumed normal).
  Obtain a new value $h^\star$.
  Generate in this way $v^\star$ from

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

  Stationary points:

$$\frac{\partial \bar{J}_{\text{gen}}}{\partial v^\star} = 0$$

  This gives

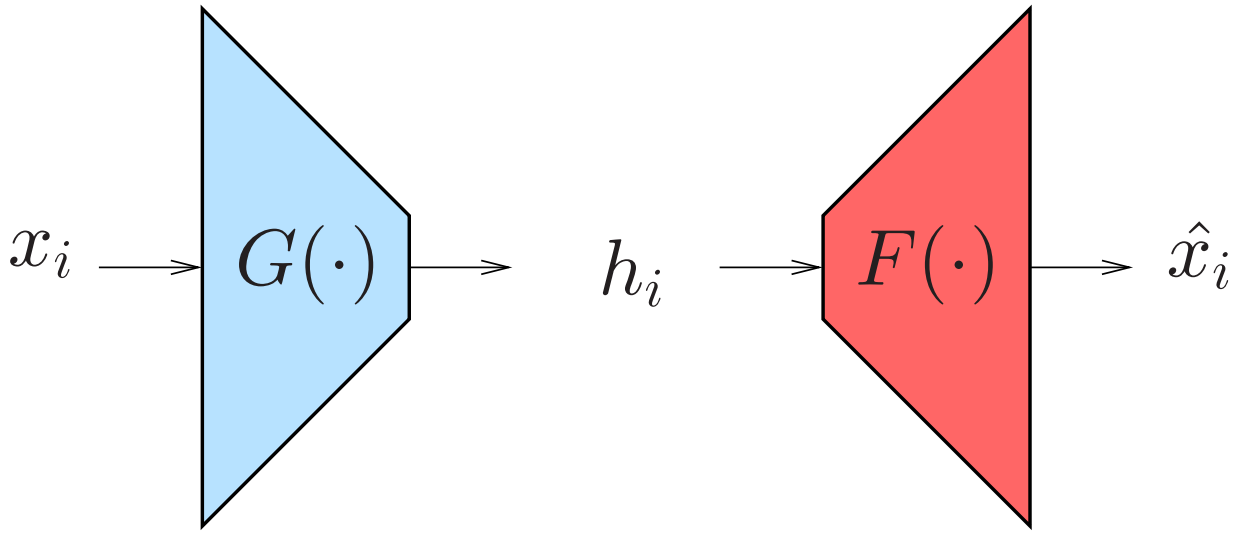
$$v^\star = W h^\star$$

Dimensionality reduction and denoising: linear case

- Given training data $v_i = x_i$ with $X \in \mathbb{R}^{d \times N}$, obtain hidden features $H \in \mathbb{R}^{s \times N}$:

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

- Reconstruction error: $\|X - \hat{X}\|^2$



Dimensionality reduction and denoising: nonlinear case (1)

- New datapoint $x^\star$ generated from $h^\star$ by

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

- Multiplying both sides by $\varphi(x_j)$ gives:

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

On training data:

$$\hat{\Omega} = \frac{1}{\eta}\Omega H^{\mathrm{T}}H$$

with $H \in \mathbb{R}^{s\times N}, \Omega_{ij} = K(x_i, x_j) = \varphi(x_i)^T\varphi(x_j)$.

Dimensionality reduction and denoising: nonlinear case (2)

- Estimated value $\hat{x}$ for $x^\star$ by kernel smoother:

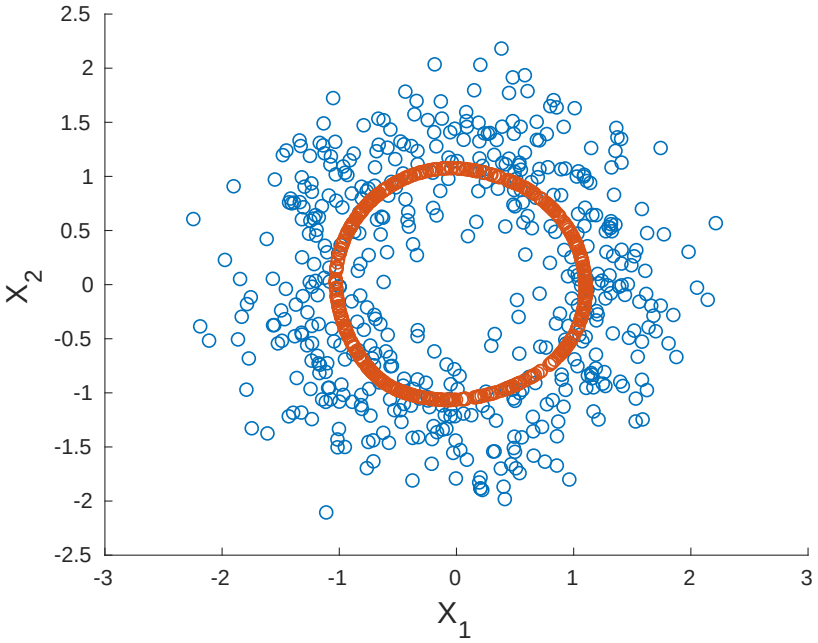
$$\hat{x} = \frac{\sum_{j=1}^{S} \tilde{K}(x_j, x^\star) x_j}{\sum_{j=1}^{S} \tilde{K}(x_j, x^\star)}$$

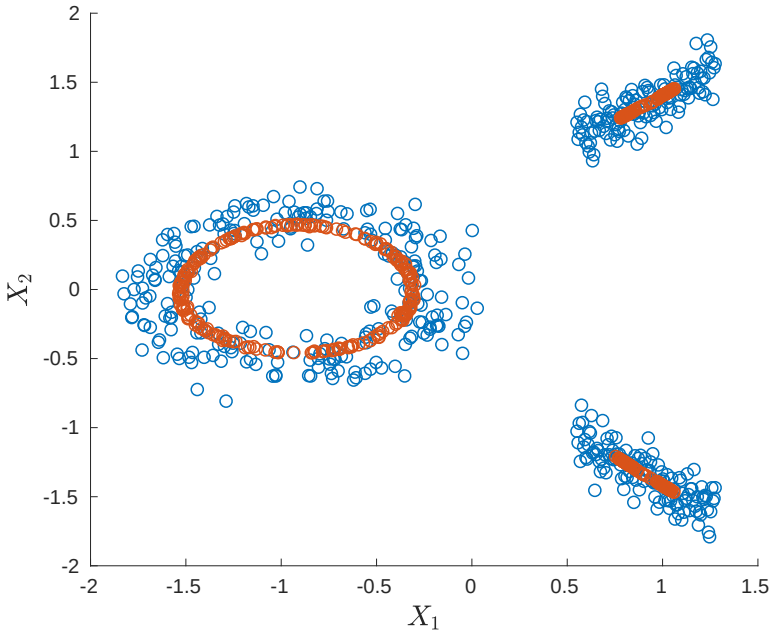
with $\tilde{K}(x_j, x^\star)$ (e.g. RBF kernel) the scaled similarity between 0 and 1, a design parameter $S \leq N$ (S closest points based on the similarity $\tilde{K}(x_j, x^\star)$).

[Schreurs & Suykens, ESANN 2018]

Example: denoising

Synthetic data sets:





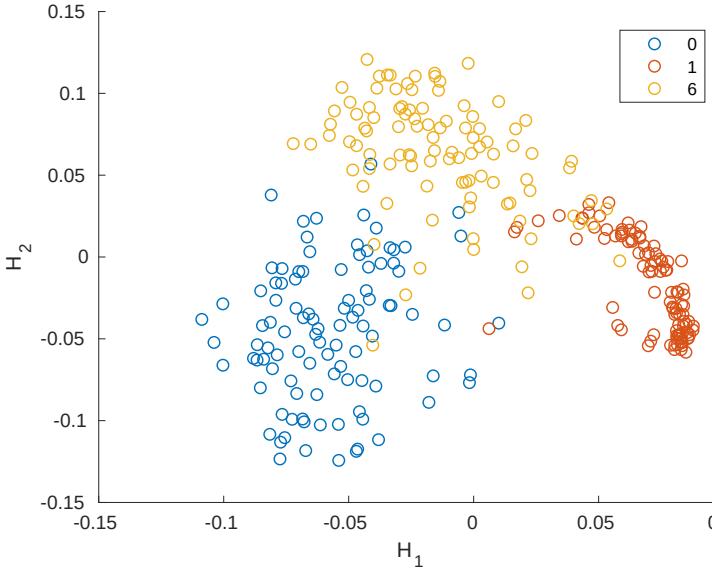
$X \in \mathbb{R}^{2\times 500}$ $(d = 2, N = 500)$

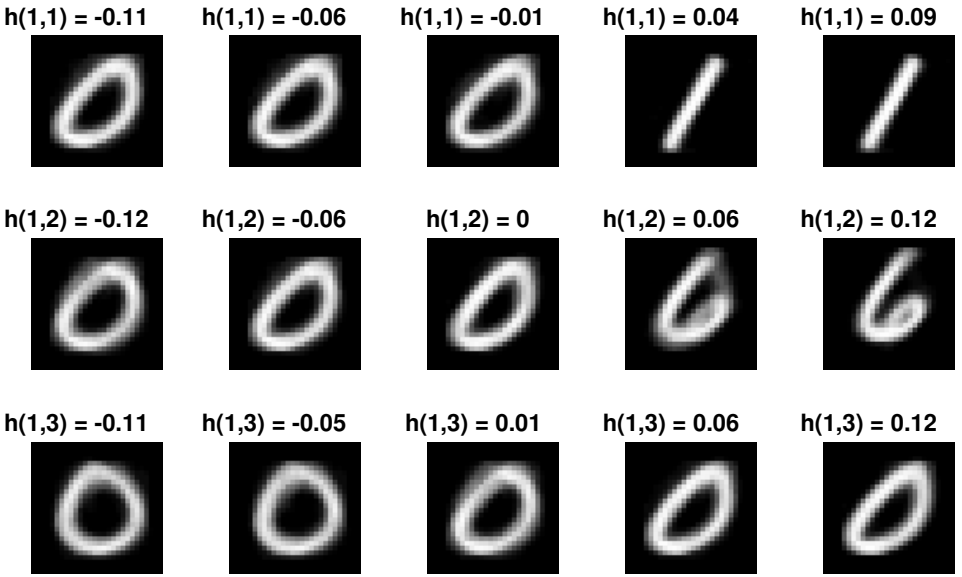
Kernel PCA using RBF kernel with $\tilde{\sigma}^2 = 1$ (left: $s = 2$; right: $s = 8$)

Kernel smoother: $S = 100$ closed points, $\tilde{\sigma}^2 = 0.2$

Explainable AI: latent space exploration

hidden units: exploring the **whole continuum**:

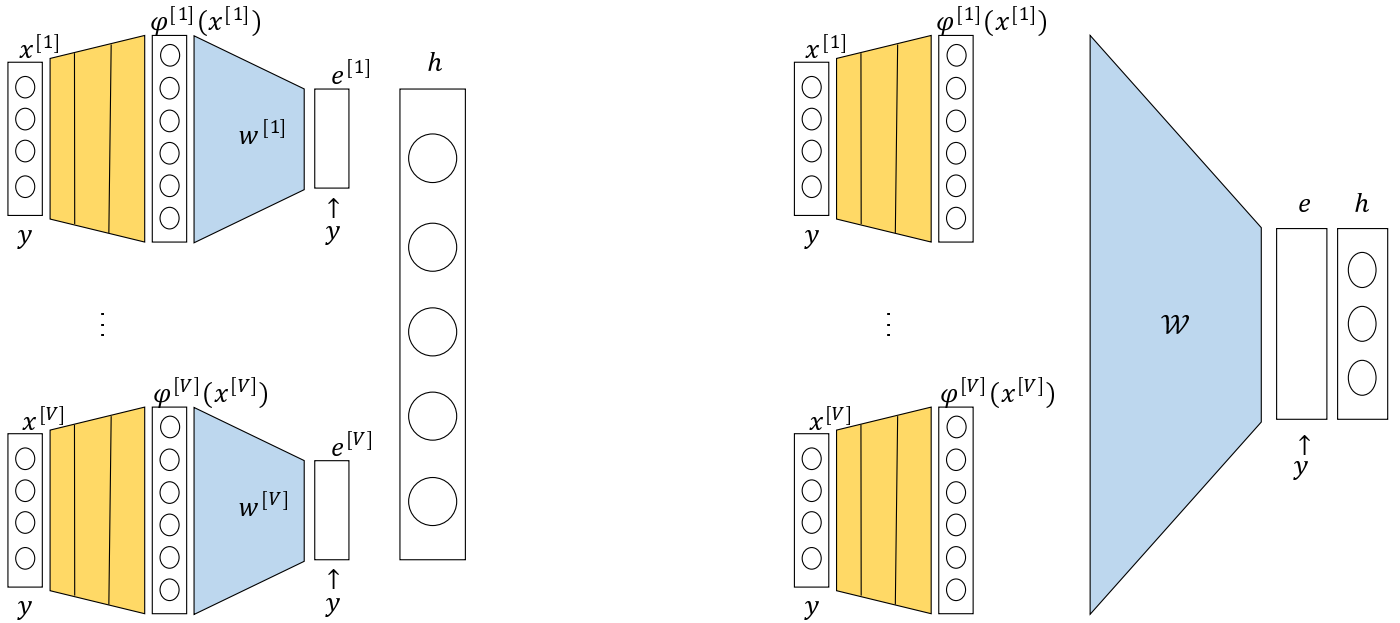




[figures by Joachim Schreurs]

Tensor-based RKM for Multi-view KPCA

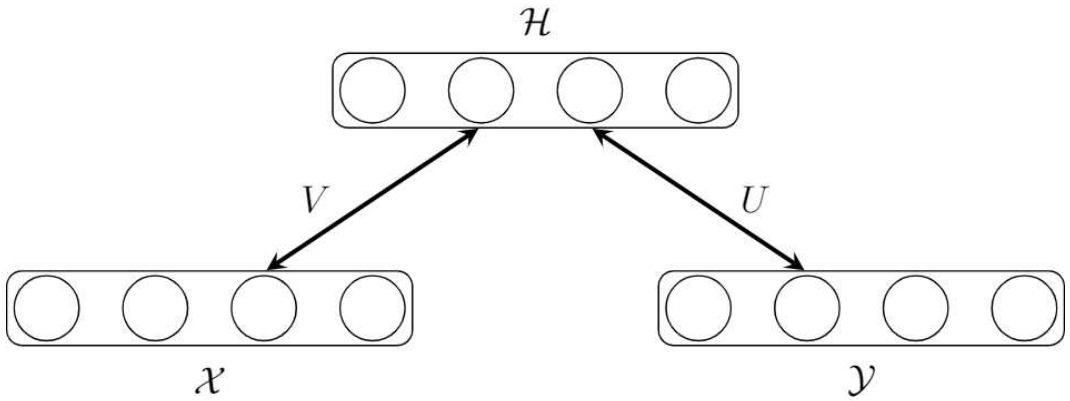
$$\min \langle \mathcal{W}, \mathcal{W} \rangle - \sum_{i=1}^{N} \left\langle \Phi_{(i)}, \mathcal{W} \right\rangle h_i + \lambda \sum_{i=1}^{N} h_i^2 \quad \text{with} \quad \Phi_{(i)} = \varphi^{[1]}(x_i^{[1]}) \otimes \ldots \otimes \varphi^{[V]}(x_i^{[V]})$$



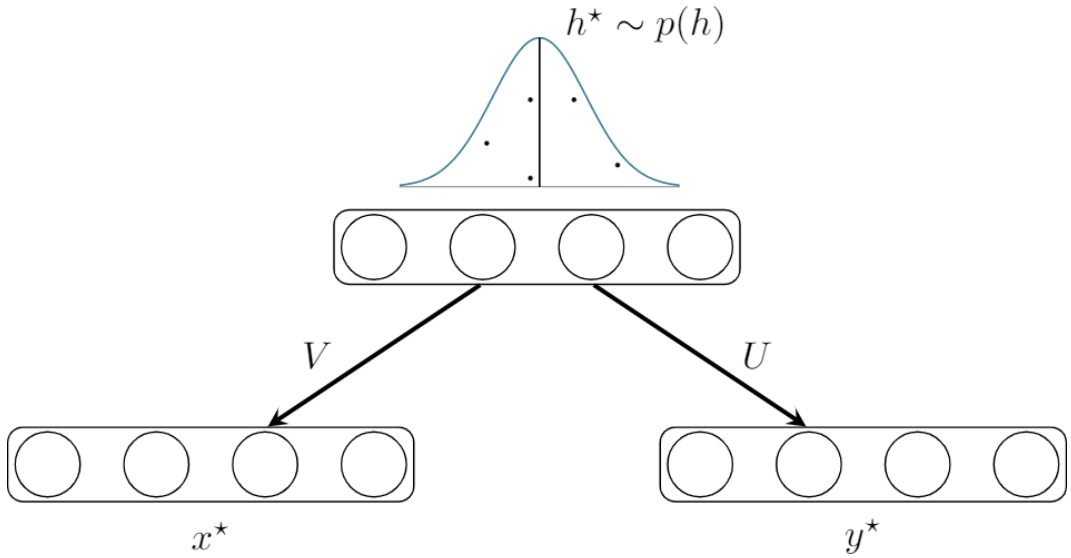
[Houthuys & Suykens, ICANN 2018]

Generative RKM (Gen-RKM) (1)

Train:

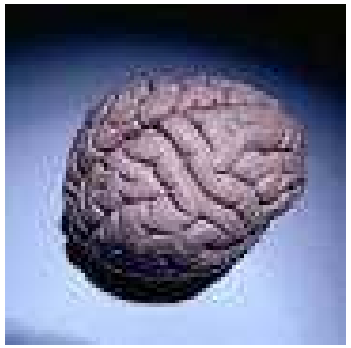


Generate:

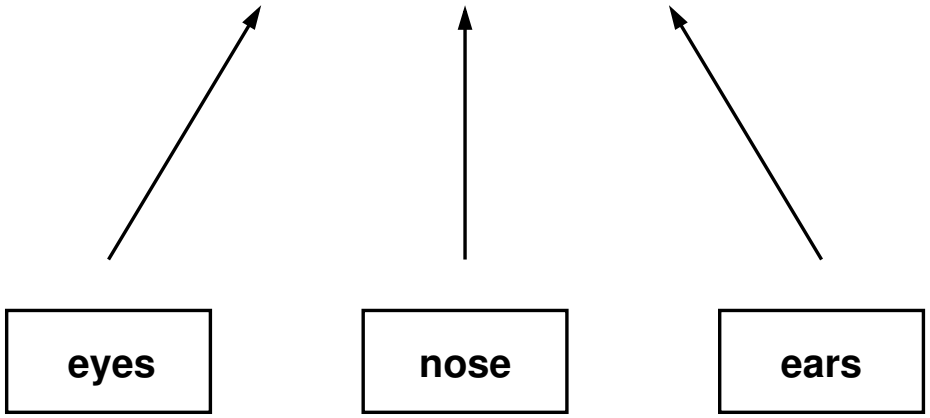


[Pandey, Schreurs & Suykens, 2019, arXiv:1906.08144]

Analogy with the human brain



shared feature representation



multimodal data

Gen-RKM (2)

The objective

$$J_{\text{train}}(\boldsymbol{h}_i, \boldsymbol{U}, \boldsymbol{V}) = \sum_{i=1}^{N} \left(-\phi_1(\boldsymbol{x}_i)^T \boldsymbol{U} \boldsymbol{h}_i - \phi_2(\boldsymbol{y}_i)^T \boldsymbol{V} \boldsymbol{h}_i + \frac{\lambda}{2} \boldsymbol{h}_i^T \boldsymbol{h}_i\right) + \frac{\eta_1}{2}\text{Tr}(\boldsymbol{U}^T \boldsymbol{U}) + \frac{\eta_2}{2}\text{Tr}(\boldsymbol{V}^T \boldsymbol{V})$$

results for **training** into the eigenvalue problem

$$(\frac{1}{\eta_1}\boldsymbol{K}_1 + \frac{1}{\eta_2}\boldsymbol{K}_2)\boldsymbol{H}^T = \boldsymbol{H}^T \boldsymbol{\Lambda}$$

with $\boldsymbol{H} = [\boldsymbol{h}_1 ... \boldsymbol{h}_N]$ and kernel matrices $\boldsymbol{K}_1, \boldsymbol{K}_2$ related to ϕ_1, ϕ_2.

[Pandey, Schreurs & Suykens, 2019, arXiv:1906.08144]

Gen-RKM (3)

Generating data is based on a newly generated $\boldsymbol{h}^\star$ and the objective

$$J_{\text{gen}}(\phi_1(\boldsymbol{x}^\star), \varphi_2(\boldsymbol{y}^\star)) = -\phi_1(\boldsymbol{x}^\star)^T \boldsymbol{V} \boldsymbol{h}^\star - \phi_2(\boldsymbol{y}^\star)^T \boldsymbol{U} \boldsymbol{h}^\star + \frac{1}{2}\phi_1(\boldsymbol{x}^\star)^T \phi_1(\boldsymbol{x}^\star) + \frac{1}{2}\phi_2(\boldsymbol{y}^\star)^T \phi_2(\boldsymbol{y}^\star)$$

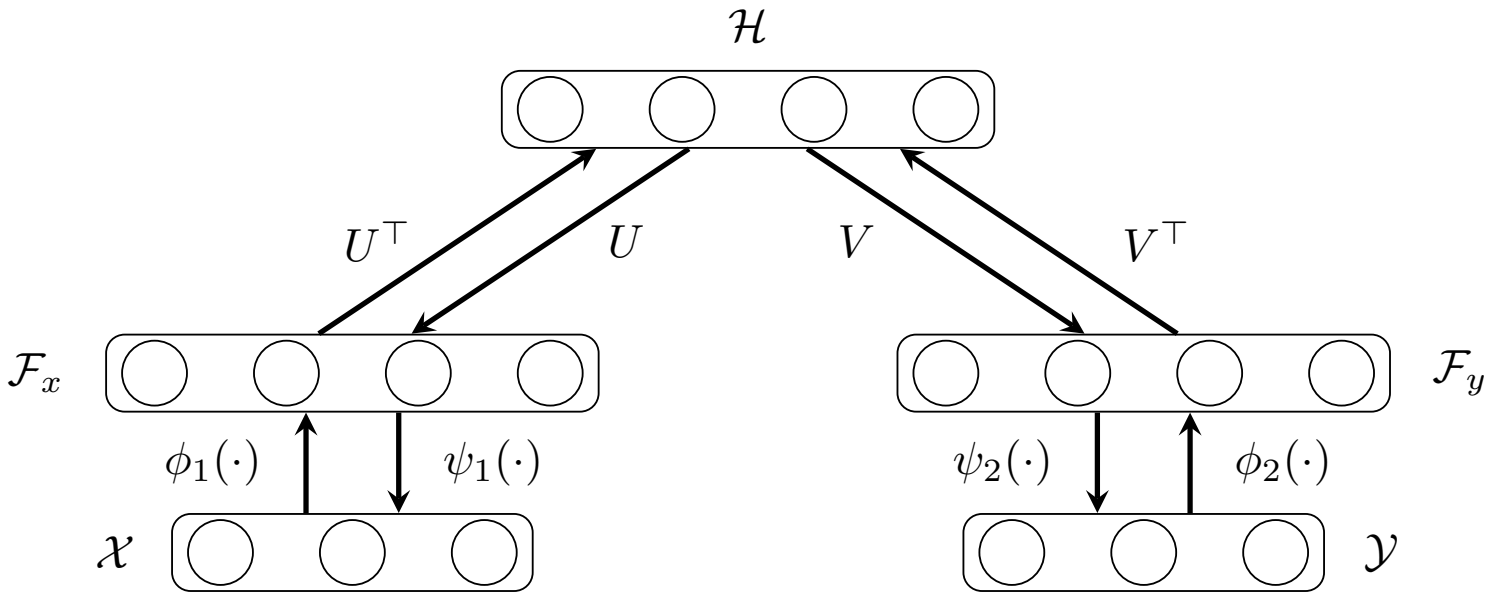
giving

$$\phi_1(\boldsymbol{x}^\star) = \frac{1}{\eta_1}\sum_{i=1}^{N} \phi_1(\boldsymbol{x}_i)\boldsymbol{h}_i^T \boldsymbol{h}^\star, \;\; \phi_2(\boldsymbol{y}^\star) = \frac{1}{\eta_2}\sum_{i=1}^{N} \phi_2(\boldsymbol{y}_i)\boldsymbol{h}_i^T \boldsymbol{h}^\star.$$

For generating $\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}$ one can either work with the kernel smoother or work with an explicit feature map using a (deep) neural network or CNN.

[Pandey, Schreurs & Suykens, 2019, arXiv:1906.08144]

Gen-RKM (4)



Gen-RKM schematic representation modeling a common subspace $\mathcal{H}$ between two data sources $\mathcal{X}$ and $\mathcal{Y}$. The ϕ_1, ϕ_2 are the feature maps ($\mathcal{F}_x$ and $\mathcal{F}_y$ represent the feature-spaces) corresponding to the two data sources. While ψ_1, ψ_2 represent the pre-image maps. The interconnection matrices U, V model dependencies between latent variables and the mapped data sources.

[Pandey, Schreurs & Suykens, 2019, arXiv:1906.08144]

Gen-RKM: implicit feature map (choose kernel)

Obtain

$$\boldsymbol{k}_{\boldsymbol{x}^\star} = \frac{1}{\eta_1}\boldsymbol{K}_1\boldsymbol{H}^\top\boldsymbol{h}^\star, \quad \boldsymbol{k}_{\boldsymbol{y}^\star} = \frac{1}{\eta_2}\boldsymbol{K}_2\boldsymbol{H}^\top\boldsymbol{h}^\star,$$

with $\boldsymbol{k}_{\boldsymbol{x}^\star} = [k(\boldsymbol{x}_1, \boldsymbol{x}^\star), \ldots, k(\boldsymbol{x}_N, \boldsymbol{x}^\star)]^\top$.

Using the kernel-smoother:

$$\hat{\boldsymbol{x}} = \psi_1\left(\phi_1(\boldsymbol{x}^\star)\right) = \frac{\sum_{j=1}^{n_r}\tilde{k}_1(\boldsymbol{x}_j, \boldsymbol{x}^\star)\boldsymbol{x}_j}{\sum_{j=1}^{n_r}\tilde{k}_1(\boldsymbol{x}_j, \boldsymbol{x}^\star)}, \quad \hat{\boldsymbol{y}} = \psi_2\left(\phi_2(\boldsymbol{y}^\star)\right) = \frac{\sum_{j=1}^{n_r}\tilde{k}_2(\boldsymbol{y}_j, \boldsymbol{y}^\star)\boldsymbol{y}_j}{\sum_{j=1}^{n_r}\tilde{k}_2(\boldsymbol{y}_j, \boldsymbol{y}^\star)},$$

with $\tilde{k}_1(\boldsymbol{x}_i, \boldsymbol{x}^\star)$ and $\tilde{k}_2(\boldsymbol{y}_i, \boldsymbol{y}^\star)$ the scaled similarities between 0 and 1.
n_r is the number of closest points based on the similarity defined by kernels $\tilde{k}_1$ and $\tilde{k}_2$.

Gen-RKM: choose explicit (CNN) feature map

Parametrized feature maps: $\phi_{\boldsymbol{\theta}}(\cdot)$, $\psi_{\boldsymbol{\zeta}}(\cdot)$ (e.g. CNN and transposed CNN).

Overall objective function, using a stabilization mechanism [Suykens, 2017]:

$$\begin{aligned}\min_{\boldsymbol{\theta}_1,\boldsymbol{\theta}_2,\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2} \mathcal{J}_c \;=\; & \mathcal{J}_{\text{train}} + \tfrac{c_{\text{stab}}}{2}\mathcal{J}^2_{\text{train}} \\ & + \tfrac{c_{\text{acc}}}{2N}\left(\sum_{i=1}^{N}\left[\mathcal{L}_1(\boldsymbol{x}_i^\star, \psi_{1_{\boldsymbol{\zeta}_1}}(\phi_{1_{\boldsymbol{\theta}_1}}(\boldsymbol{x}_i^\star))) + \mathcal{L}_2(\boldsymbol{y}_i^\star, \psi_{2_{\boldsymbol{\zeta}_2}}(\phi_{2_{\boldsymbol{\theta}_2}}(\boldsymbol{y}_i^\star)))\right]\right)\end{aligned}$$

with reconstruction errors

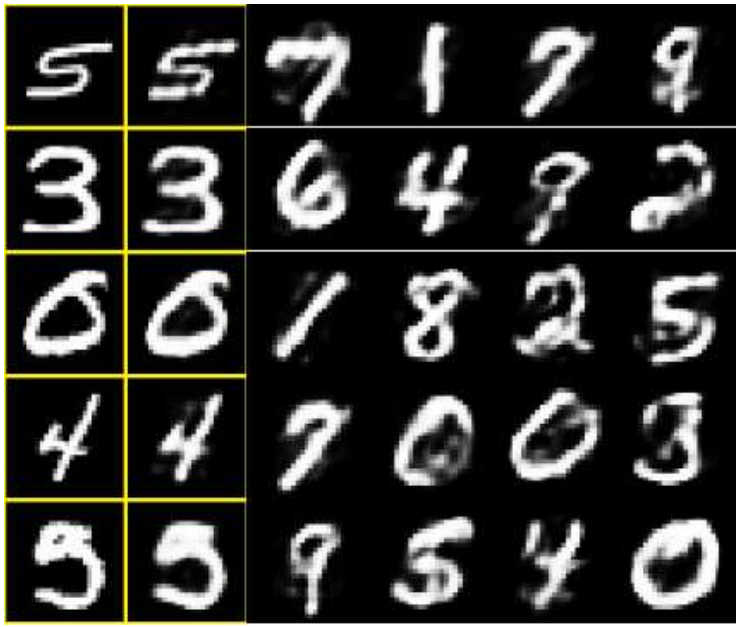
$$\begin{aligned}\mathcal{L}_1(\boldsymbol{x}_i^\star, \psi_{1_{\boldsymbol{\zeta}_1}}(\phi_{1_{\boldsymbol{\theta}_1}}(\boldsymbol{x}_i^\star))) &= \tfrac{1}{N}\|\boldsymbol{x}_i^\star - \psi_{1_{\boldsymbol{\zeta}_1}}(\phi_{1_{\boldsymbol{\theta}_1}}(\boldsymbol{x}_i^\star))\|_2^2, \\ \mathcal{L}_2(\boldsymbol{y}_i^\star, \psi_{2_{\boldsymbol{\zeta}_2}}(\phi_{2_{\boldsymbol{\theta}_2}}(\boldsymbol{y}_i^\star))) &= \tfrac{1}{N}\|\boldsymbol{y}_i^\star - \psi_{2_{\boldsymbol{\zeta}_2}}(\phi_{2_{\boldsymbol{\theta}_2}}(\boldsymbol{y}_i^\star))\|_2^2\end{aligned}$$

and with $\Phi_{\boldsymbol{x}} = [\phi_1(\boldsymbol{x}_1), \ldots, \phi_1(\boldsymbol{x}_N)], \Phi_{\boldsymbol{y}} = [\phi_2(\boldsymbol{y}_1), \ldots, \phi_2(\boldsymbol{y}_N)]$, U, V from

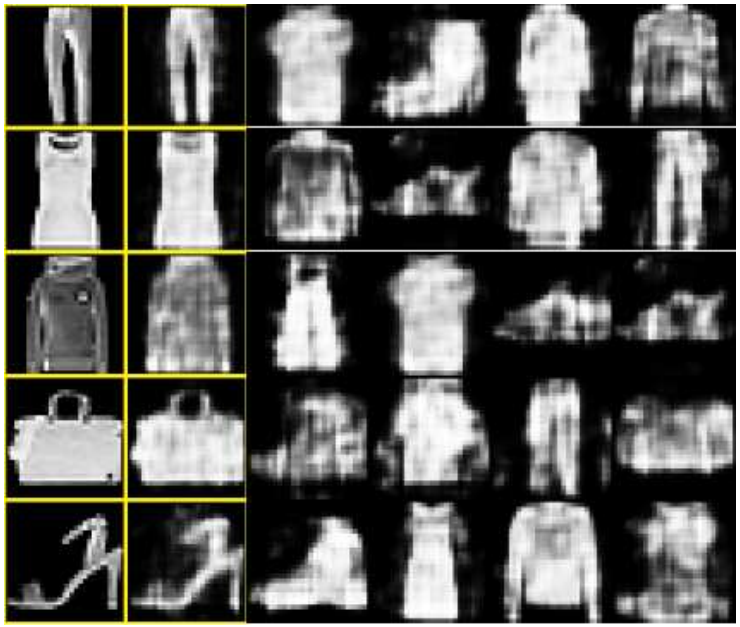
$$\begin{bmatrix} \frac{1}{\eta_1}\Phi_{\boldsymbol{x}}\Phi_{\boldsymbol{x}}^\top & \frac{1}{\eta_1}\Phi_{\boldsymbol{x}}\Phi_{\boldsymbol{y}}^\top \\ \frac{1}{\eta_2}\Phi_{\boldsymbol{y}}\Phi_{\boldsymbol{x}}^\top & \frac{1}{\eta_2}\Phi_{\boldsymbol{y}}\Phi_{\boldsymbol{y}}^\top \end{bmatrix} \begin{bmatrix} U \\ V \end{bmatrix} = \begin{bmatrix} U \\ V \end{bmatrix} \Lambda.$$

Hence, joint feature learning and subspace learning.

Gen-RKM: examples (1)



MNIST

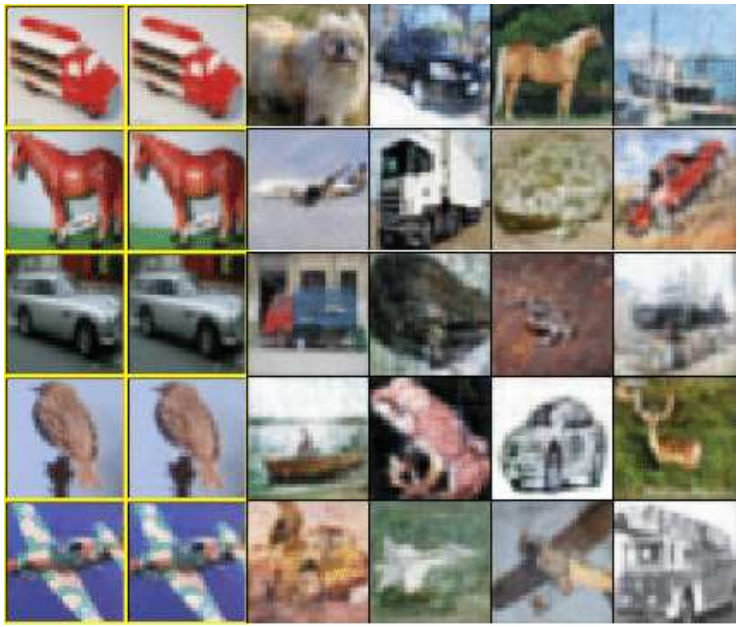


Fashion-MNIST

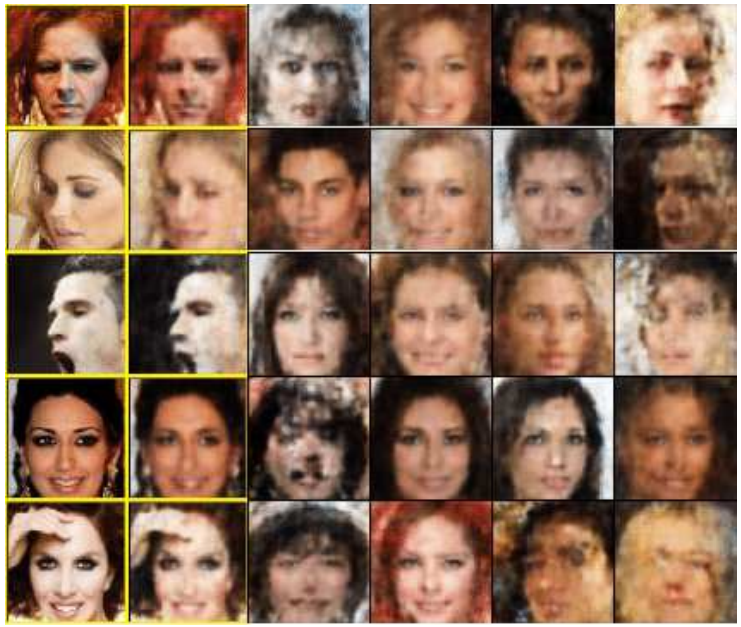
Generated samples from the model using CNN as explicit feature map in the kernel function. The yellow boxes show training examples and the adjacent boxes show the reconstructed samples. The other images (columns 3-6) are generated by random sampling from the fitted distribution over the learned latent variables.

[Pandey, Schreurs & Suykens, Neural Networks 2021]

Gen-RKM: examples (2)



CIFAR-10



CelebA

Generated samples from the model using CNN as explicit feature map in the kernel function. The yellow boxes show training examples and the adjacent boxes show the reconstructed samples. The other images (columns 3-6) are generated by random sampling from the fitted distribution over the learned latent variables.

[Pandey, Schreurs & Suykens, Neural Networks 2021]

Gen-RKM: multi-view generation (1)

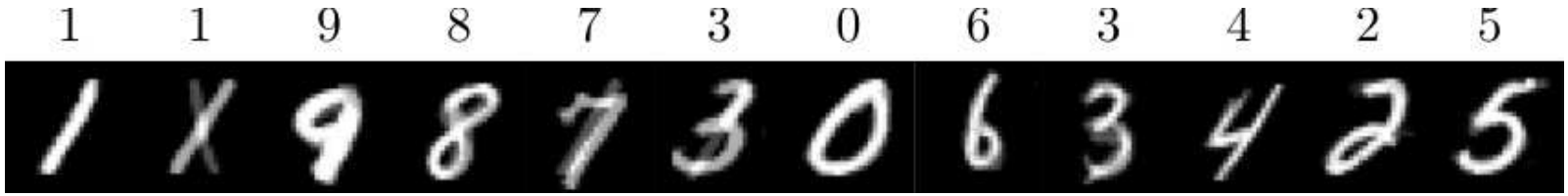


CelebA

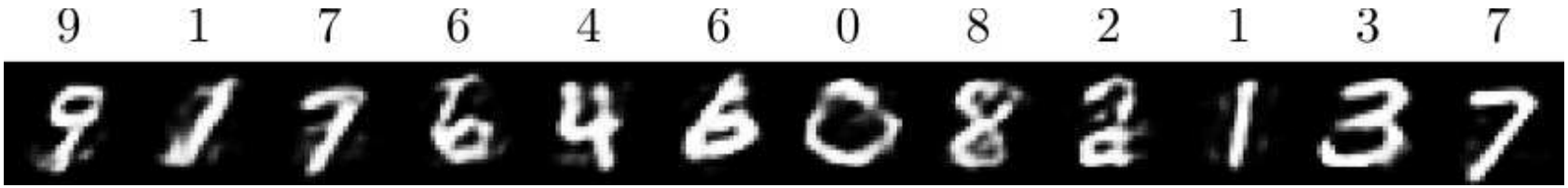
Multi-view generation on CelebA dataset showing images and attributes

[Pandey, Schreurs & Suykens, Neural Networks 2021]

Gen-RKM: multi-view generation (2)



MNIST: Implicit feature maps with Gaussian kernel + generation by kernel-smoother



MNIST: Explicit feature maps using Convolutional Neural Networks

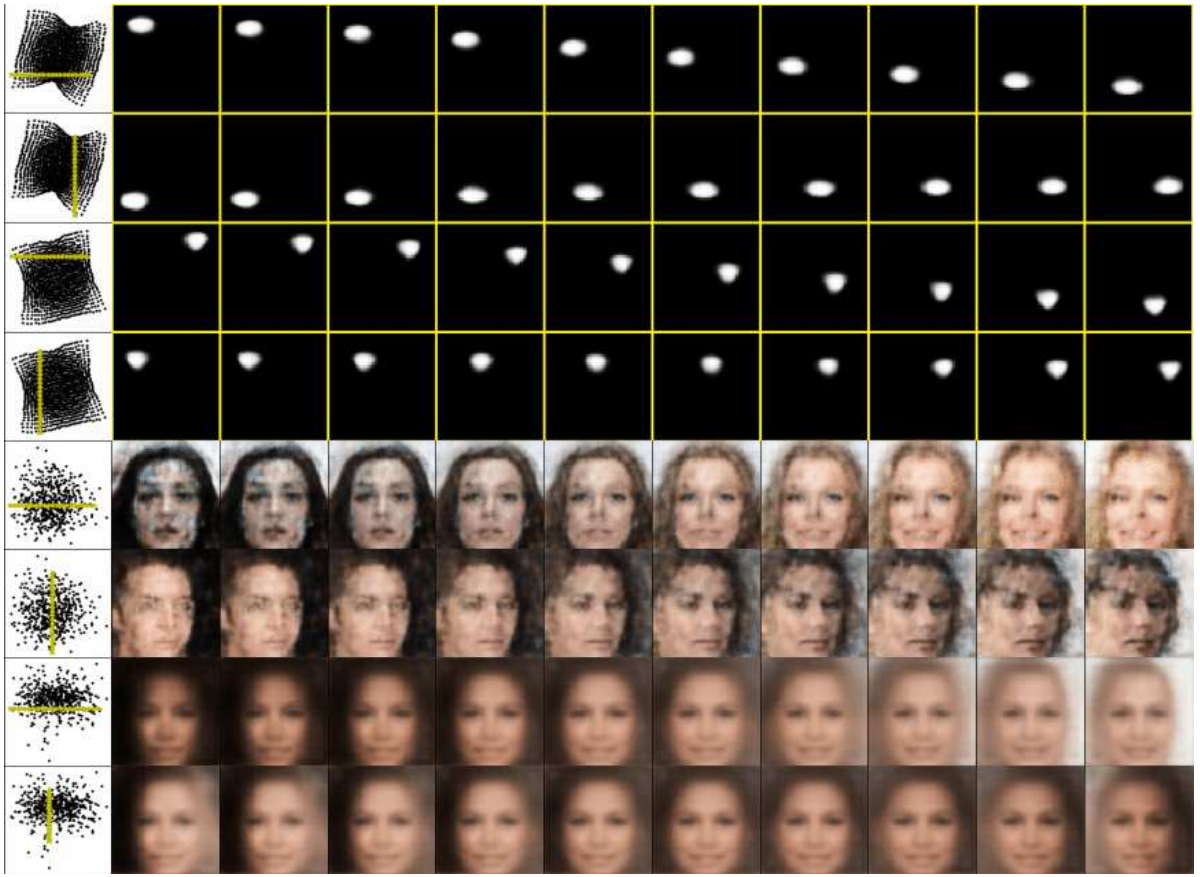


CIFAR-10: Explicit feature maps using CNNs + Transposed CNNs

Multi-view Generation (images and labels) using implicit and explicit feature maps

[Pandey, Schreurs & Suykens, Neural Networks 2021]

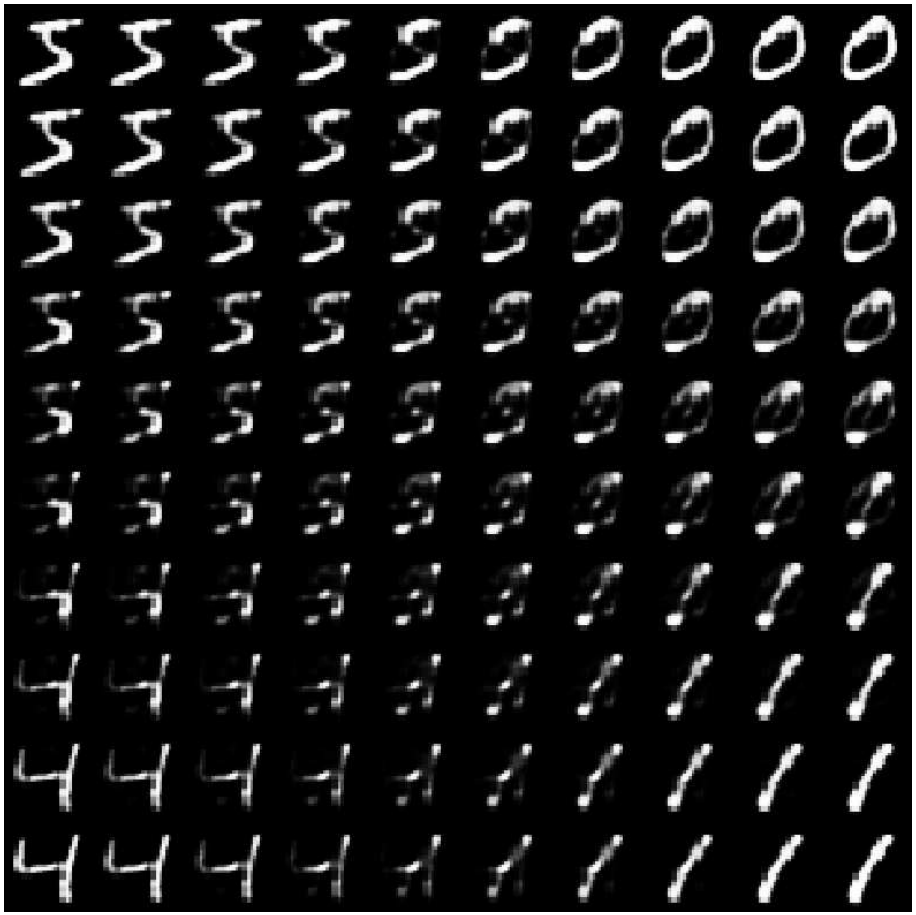
Gen-RKM: latent space exploration (1)



Exploring the learned **uncorrelated-features** by traversing along the eigenvectors

Explainability: changing one single neuron's hidden feature changes the hair color while preserving face structure! [Pandey, Schreurs & Suykens, Neural Networks 2021]

Gen-RKM: latent space exploration (2)



MNIST reconstructed images by bilinear-interpolation in latent space
[Pandey, Schreurs & Suykens, Neural Networks 2021]

Gen-RKM: latent space exploration (3)



CelebA reconstructed images by bilinear-interpolation in latent space

[Pandey, Schreurs & Suykens, Neural Networks 2021]

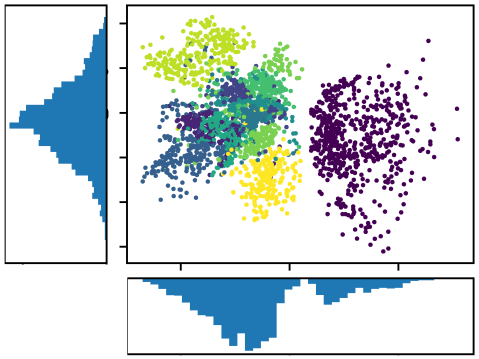
Gen-RKM - traversals along principal components



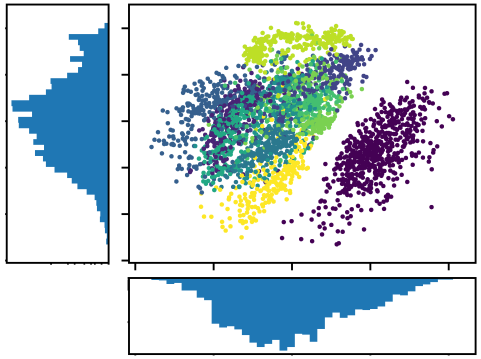
Disentangled Representation Learning and Generation with Manifold Optimization

[Pandey, Fanuel, Schreurs & Suykens, 2019, arXiv:2006.07046]

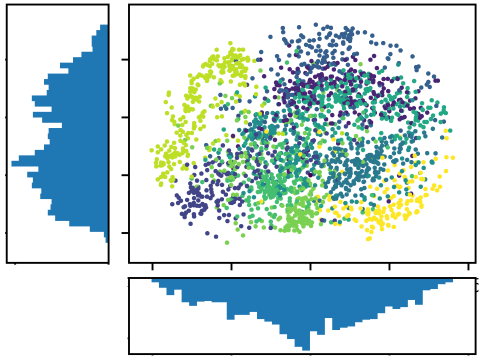
Robust Gen-RKM



VAE



Gen-RKM



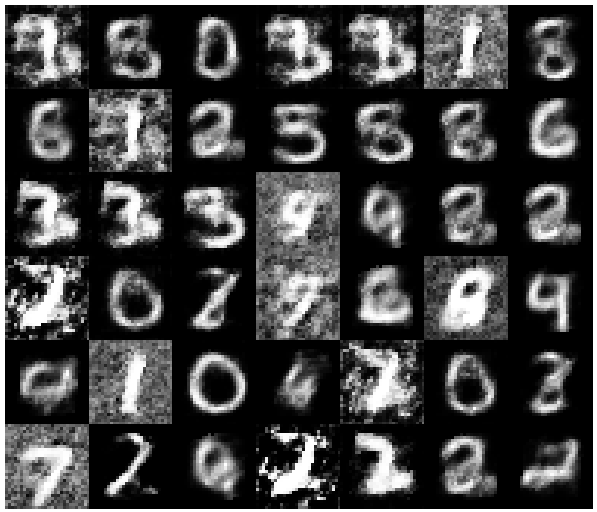
Robust Gen-RKM

- VAE and Gen-RKM: presence of outliers distorts the distribution of the latent variables
- Robust Gen-RKM: down-weighting of the outliers makes close to Gaussian distribution

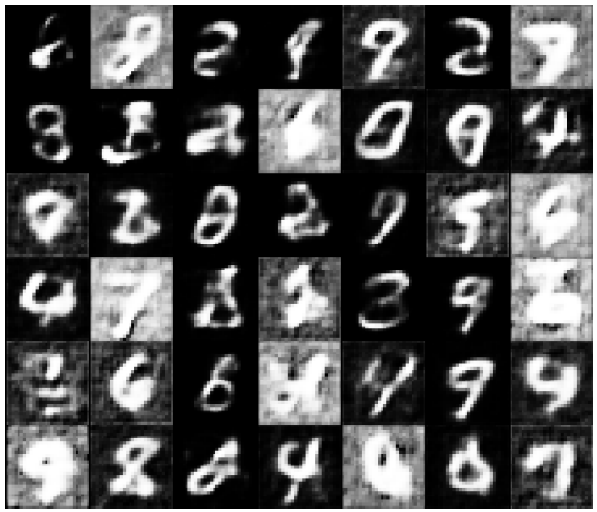
Weighted conjugate feature duality: $\frac{1}{2\lambda}e^T De + \frac{\lambda}{2}h^T D^{-1}h \geq e^T h$

[Pandey, Schreurs & Suykens, 2019, arXiv:2002.01180, LOD 2020]

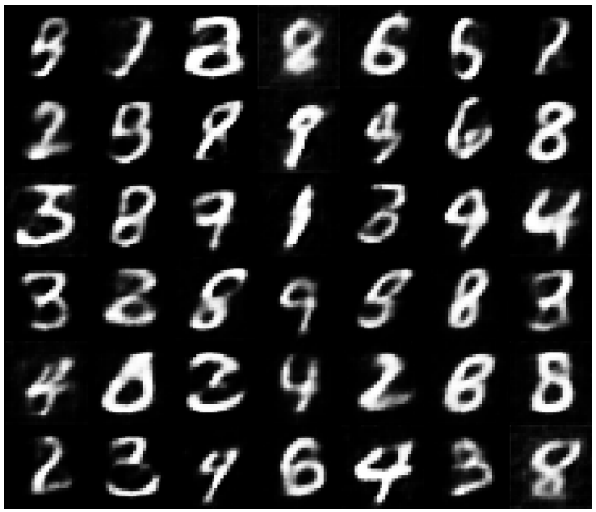
Robust Gen-RKM - Robust generation



VAE



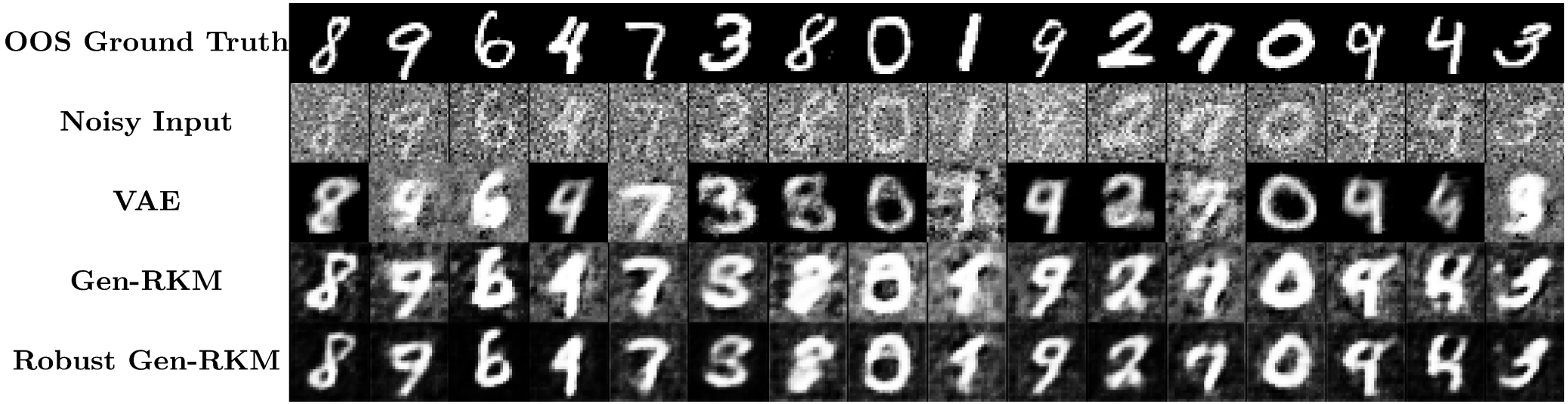
Gen-RKM



Robust Gen-RKM

[Pandey, Schreurs & Suykens, 2019, arXiv:2002.01180, LOD 2020]

Robust Gen-RKM - Robust denoising



[Pandey, Schreurs & Suykens, 2019, arXiv:2002.01180, LOD 2020]

RKM references and software

- Suykens J.A.K., "Deep Restricted Kernel Machines using Conjugate Feature Duality", *Neural Computation*, vol. 29, no. 8, pp. 2123-2163, Aug. 2017. https://www.mitpressjournals.org/doi/pdf/10.1162/neco_a_00984
- Houthuys L., Suykens J.A.K., "Tensor Learning in Multi-View Kernel PCA",*International Conference on Artificial Neural Networks 2018 (ICANN 2018)*, Rhodes, Greece, Oct. 2018, pp. 205-215.
- Schreurs J., Suykens J.A.K., "Generative Kernel PCA", *European Symposium on Artificial Neural Networks (ESANN 2018)*, Bruges, Belgium, Apr. 2018, pp. 129-134.
- Pandey A., Schreurs J., Suykens J.A.K., "Generative Restricted Kernel Machines: A framework for multi-view generation and disentangled feature learning", *Neural Networks*, Vol. 135, 177-191, 2021
- Pandey A., Fanuel M., Schreurs J., Suykens J.A.K., Disentangled Representation Learning and Generation with Manifold Optimization, arXiv:2006.07046
- Pandey A., Schreurs J., Suykens J.A.K., Robust Generative Restricted Kernel Machines using Weighted Conjugate Feature Duality, arXiv:2002.01180, *LOD 2020*.

Software: see https://www.esat.kuleuven.be/stadius/E/software.php

Conclusions

- From RBM to deep BM
- From RKM to deep RKM
- RKM and RBM representation: visible and hidden units
- RKM representation for LS-SVM, KPCA, SVD and others
- RKM representation obtained by conjugate feature duality
- Generative RKM
- Explicit or implicit feature maps (synergies deep learning and kernel machines)

Acknowledgements (1)

- Current and former co-workers at ESAT-STADIUS:

  C. Alzate, Y. Chen, J. De Brabanter, K. De Brabanter, B. De Cooman, L. De Lathauwer, H. De Meulemeester, B. De Moor, H. De Plaen, Ph. Dreesen, M. Espinoza, T. Falck, M. Fanuel, Y. Feng, B. Gauthier, X. Huang, L. Houthuys, V. Jumutc, Z. Karevan, R. Langone, F. Liu, R. Mall, S. Mehrkanoon, G. Nisol, M. Orchel, A. Pandey, P. Patrinos, K. Pelckmans, S. RoyChowdhury, S. Salzo, J. Schreurs, M. Signoretto, Q. Tao, F. Tonin, J. Vandewalle, T. Van Gestel, S. Van Huffel, C. Varon, Y. Yang, and others

- Many other people for joint work, discussions, invitations, organizations

- Support from ERC AdG E-DUALITY, ERC AdG A-DATADRIVE-B, KU Leuven, OPTEC, IUAP DYSCO, FWO projects, IWT, Flanders AI, Leuven.AI

Acknowledgements (2)



KATHOLIEKE UNIVERSITEIT
LEUVEN

FWO
VLAANDEREN

Acknowledgements (3)



ERC Advanced Grant E-DUALITY

Exploring duality for future data-driven modelling

Thank you