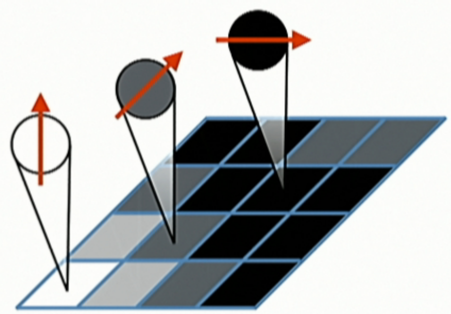
Title: Learning with Quantum-Inspired Tensor Networks

Date: Aug 09, 2016 11:45 AM

URL: http://pirsa.org/16080007

Abstract: We propose a family of models with an exponential number of parameters, but which are approximated by a tensor network. Tensor networks are used to represent quantum wavefunctions, and powerful methods for optimizing them can be extended to machine learning applications as well. We use a matrix product state to classify images, and find that a surprisingly small bond dimension yields state-of-the-art results. Tensor networks offer many advantages for machine learning, such as better scaling for existing machine learning approaches and the ability to adapt hyperparameters during training. We will also propose a generative interpretation of the trained models.

Learning with Quantum-Inspired Tensor Networks



E.M. Stoudenmire and David J. Schwab

arxiv:1605.05775

Perimeter Institute - August 2016

UCIrvine

SIMONS FOUNDATION

Take Home Message

Tensor networks used for classical simulations of quantum many-body systems

Tensor networks enable training M.L. model with *exponentially* many formal parameters

Related to non-linear kernel learning

Benefits include:

- linear cost in training set size
- model adapts to structure of data set
- opportunities to understand what model learns

What is a tensor network?

Tensor network = approximation of N-index tensor as contraction of O(N) smaller tensors

What is a tensor network?

Tensor diagrams have rigorous meaning

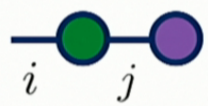
v_j

M_{ij}

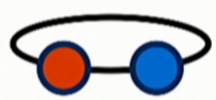
T_{ijk}

What is a tensor network?

Joining lines implies contraction, can omit names



$$\longleftrightarrow \quad \sum_j M_{ij} v_j$$



Matrix Product State (MPS)

Matrix product state (MPS) well understood, powerful class of tensor network

Matrix Product State (MPS)

Matrix product state (MPS) well understood, powerful class of tensor network

MPS approximation controlled by *bond dimension* "m"

Represent tensor with d^N parameters using only Ndm^2 parameters

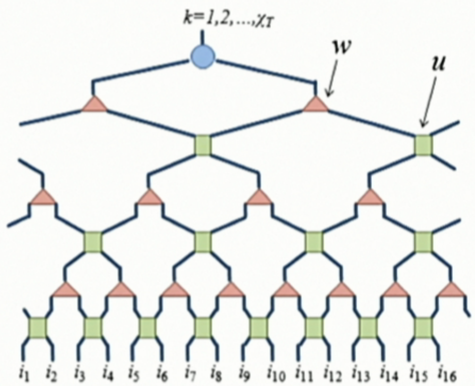
Matrix Product State (MPS)

Matrix product state (MPS) well understood, powerful class of tensor network

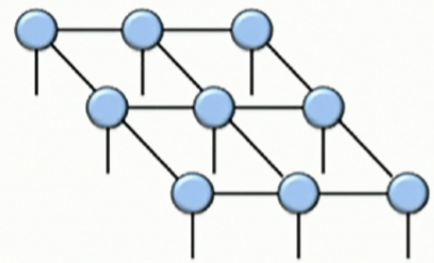
MPS approximation controlled by *bond dimension* "m"

Represent tensor with d^N parameters using only Ndm^2 parameters

Other successful tensor networks include MERA and PEPS



MERA



PEPS

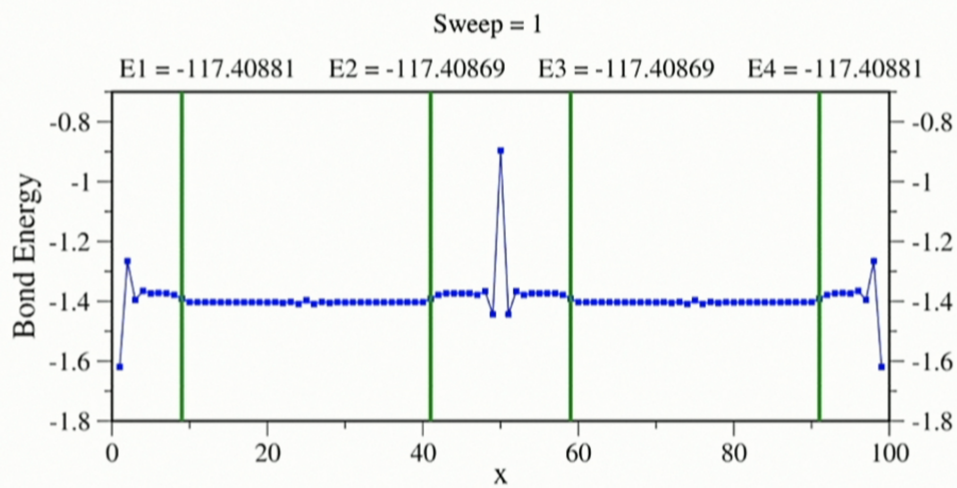
Evenbly, Vidal, PRB **79**, 144108 (2009)

Verstraete, Cirac, cond-mat/0407066 (2004)

Orus, Ann. Phys. **349**, 117 (2014)

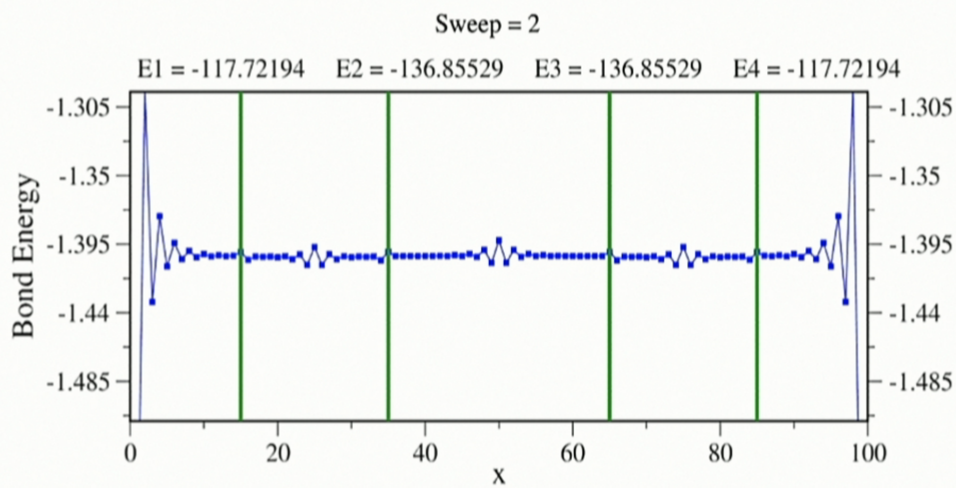
Uses of tensor networks in physics:

- MPS leads to DMRG method for 1d, 2d systems
- Tensor networks useful for classifying exotic phases of matter (e.g. topological phases)
- Rapid, continuing development of powerful numerical methods



Uses of tensor networks in physics:

- MPS leads to DMRG method for 1d, 2d systems
- Tensor networks useful for classifying exotic phases of matter (e.g. topological phases)
- Rapid, continuing development of powerful numerical methods



Proposal for learning

In physics, tensor networks approximate exponentially large vectors (e.g. quantum states)

To mimic for learning, map data vector to <u>tensor product</u> of "local feature vectors"

$$\mathbf{x} = [x_1, \quad x_2, \quad x_3, \quad \ldots \quad , \quad x_N] \qquad \textit{raw inputs}$$

$$\Downarrow$$

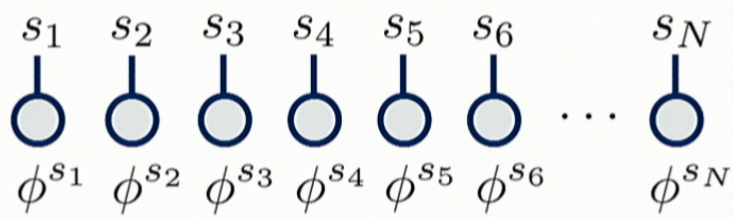
$$\Phi(\mathbf{x}) = \begin{bmatrix} \phi_1(x_1) \\ \phi_2(x_1) \end{bmatrix} \otimes \begin{bmatrix} \phi_1(x_2) \\ \phi_2(x_2) \end{bmatrix} \otimes \begin{bmatrix} \phi_1(x_3) \\ \phi_2(x_3) \end{bmatrix} \otimes \cdots \otimes \begin{bmatrix} \phi_1(x_N) \\ \phi_2(x_N) \end{bmatrix} \qquad \textit{feature vector}$$

Proposal for learning

Tensor notation

$$\Phi^{s_1 s_2 \cdots s_N}(\mathbf{x}) = \phi^{s_1}(x_1) \otimes \phi^{s_2}(x_2) \otimes \cdots \otimes \phi^{s_N}(x_N)$$

$=$

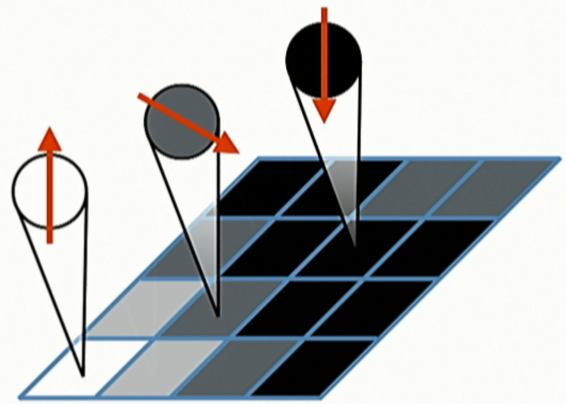


- Each local map $\phi^{s_j}(x_j)$ normalized, d-component vector
- $\Phi(\mathbf{x})$ maps N-dimensional to d^N-dimensional space
- Resembles product state wavefunction

Example: grayscale image

Local feature map, dimension d=2

$$\phi(x_j) = \left[\cos\left(\frac{\pi}{2}x_j\right), \sin\left(\frac{\pi}{2}x_j\right)\right] \qquad x_j \in [0,1]$$

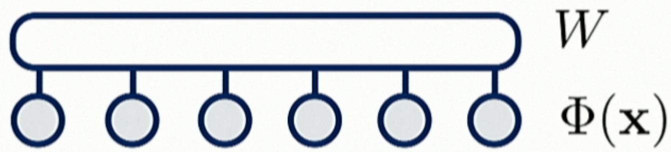


Crucially, grayscale values not orthogonal

Decision function $f(\mathbf{x}) = W \cdot \Phi(\mathbf{x})$

Weights W live in same space as $\Phi(\mathbf{x})$

$$f(\mathbf{x}) =$$



In traditional notation

$$f(\mathbf{x}) = \sum_{s_1, s_2, \ldots, s_N} W_{s_1 s_2 \cdots s_N} \; \phi^{s_1}(x_1) \phi^{s_2}(x_2) \cdots \phi^{s_N}(x_N)$$

MNIST Experiment

MNIST is a benchmark data set of grayscale handwritten digits (labels $\ell = 0,1,2,\ldots,9$)

60,000 labeled training images
10,000 labeled test images

MNIST Experiment

Approach:

(1) Shrink 28x28 images to 14x14, more manageable

(2) Order 2d pixels in a "zig-zag" pattern for MPS decomposition of W^ℓ

(3) Optimize with DMRG-like algorithm but with core step gradient descent

MNIST Experiment

Approach:

(1) Shrink 28x28 images to 14x14, more manageable

(2) Order 2d pixels in a "zig-zag" pattern for MPS decomposition of W^ℓ

(3) Optimize with DMRG-like algorithm but with core step gradient descent

MNIST Experiment

Approach:

(3) Optimize with DMRG-like algorithm
but with core step gradient descent

Used quadratic cost function

$$C = \frac{1}{2} \sum_{n=1}^{N_T} \sum_{\ell} (f^{\ell}(\mathbf{x}_n) - \delta^{\ell}_{L_n})^2$$

Wants $f^{L_n}(x_n) = 1$, other $f^{\ell}(x_n) = 0$

MNIST Experiment

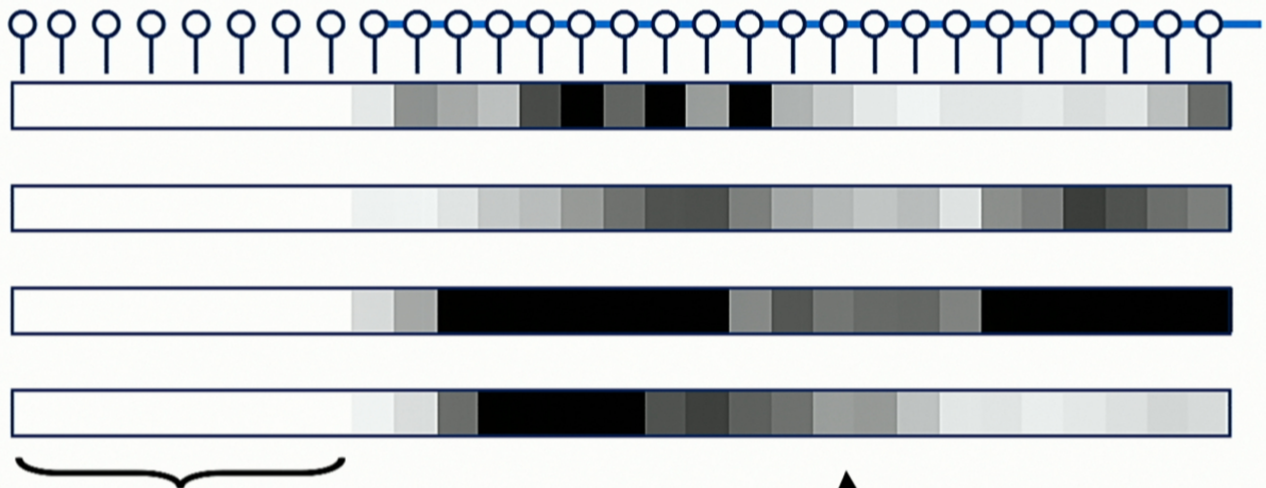
Results

Only ~3 sweeps needed to converge

Bond dimension	Test Set Error	
m = 10	~5%	(500/10,000 incorrect)
m = 20	~2%	(200/10,000 incorrect)
m = 120	0.97%	(97/10,000 incorrect)

⟶ ***Demo***

Tensor network is adaptive



boundary pixels not useful for learning

grayscale training data

What is the form of the decision function?

Expand local d=2 feature map

$$\phi^s(x) = [\cos(\pi/2\,x), \sin(\pi/2\,x)] \sim [1, x] \qquad\qquad x \ll 1$$

Decision function becomes

$$
\begin{aligned}
f(\mathbf{x}) &= W \cdot \Phi(\mathbf{x}) \\
&= W_{111\cdots1} \\
&\quad + W_{211\cdots1}\, x_1 + W_{121\cdots1}\, x_2 + W_{112\cdots1}\, x_3 + \ldots \\
&\quad + W_{221\cdots1}\, x_1 x_2 + W_{212\cdots1}\, x_1 x_3 + \ldots \\
&\quad + \ldots \\
&\quad + W_{222\cdots2}\, x_1 x_2 x_3 \cdots x_N
\end{aligned}
$$

Novikov, Trofimov, Oseledets, arxiv:1605.03795 (2016)

What is the form of the decision function?

Expand local d=2 feature map

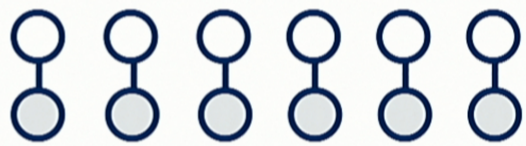
$$\phi^s(x) = [\cos(\pi/2\,x), \sin(\pi/2\,x)] \sim [1, x] \qquad\qquad x \ll 1$$

Decision function becomes

$$\begin{aligned} f(\mathbf{x}) = W \cdot \Phi(\mathbf{x}) \\ &= W_{111\cdots1} \\ &\quad + W_{211\cdots1}\,x_1 + W_{121\cdots1}\,x_2 + W_{112\cdots1}\,x_3 + \ldots \\ &\quad + W_{221\cdots1}\,x_1x_2 + W_{212\cdots1}\,x_1x_3 + \ldots \\ &\quad + \ldots \\ &\quad + W_{222\cdots2}\,x_1x_2x_3\cdots x_N \end{aligned}$$

Novikov, Trofimov, Oseledets, arxiv:1605.03795 (2016)

Consider extreme case of product decomposition (m=1 MPS)

$f(\mathbf{x}) =$  W $\Phi(\mathbf{x})$

 $= \begin{bmatrix} w_1^{(j)} \\ w_2^{(j)} \end{bmatrix}$

$$
\begin{aligned}
&= w_1^{(1)} w_1^{(2)} w_1^{(3)} \cdots w_1^{(N)} \\
&+ \left(w_2^{(1)} w_1^{(2)} w_1^{(3)} \cdots w_1^{(N)}\right)\, x_1 + \left(w_1^{(1)} w_2^{(2)} w_1^{(3)} \cdots w_1^{(N)}\right)\, x_2 + \ldots \\
&+ \left(w_2^{(1)} w_2^{(2)} w_1^{(3)} \cdots w_1^{(N)}\right)\, x_1 x_2 + \left(w_2^{(1)} w_1^{(2)} w_2^{(3)} \cdots w_1^{(N)}\right)\, x_1 x_3 \\
&+ \ldots
\end{aligned}
$$

Interesting interpretation:

$$\phi(x) = [1,\ x,\ x^2,\ x^3, \ldots]$$

$0 \quad 1 \quad 2 \quad 3$ *fictitious "particle" number*

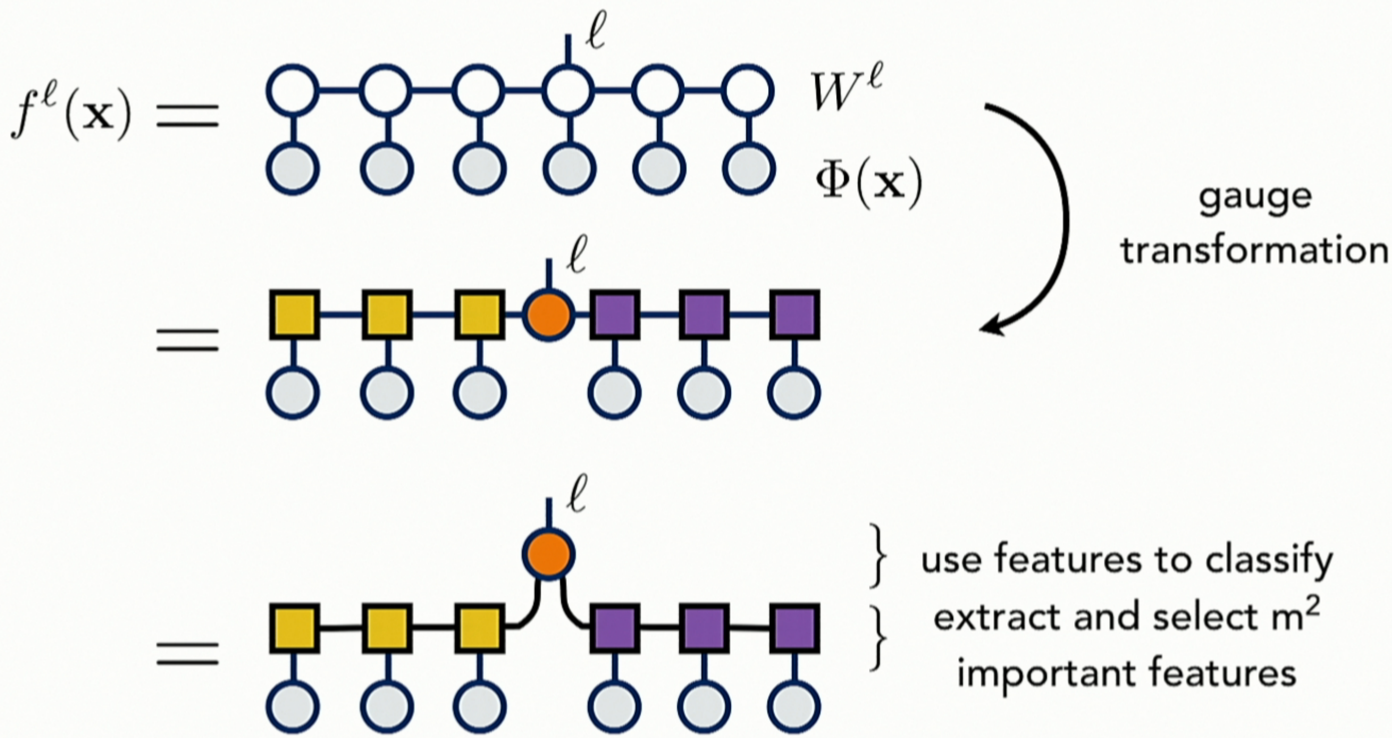
Then fixed-particle sectors of W tensor correspond to homogeneous polynomial kernels

$$f_1(\mathbf{x}) = W_{100\cdots0}\, x_1 + W_{010\cdots0}\, x_2 + W_{001\cdots0}\, x_3 + \ldots$$

$$f_2(\mathbf{x}) = W_{200\cdots0}\, x_1^2 + W_{110\cdots0}\, x_1 x_2 + W_{101\cdots0}\, x_1 x_3 + \ldots$$

$$f_3(\mathbf{x}) = W_{300\cdots0}\, x_1^3 + W_{210\cdots0} x_1^2 x_2 + W_{111\cdots0}\, x_1 x_2 x_3 + \ldots$$

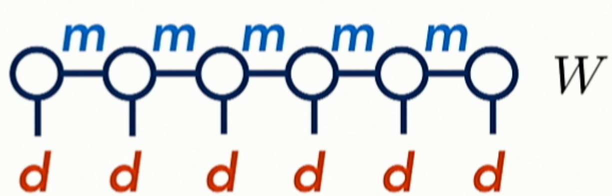
Helpful to view classification in two stages



Regularization

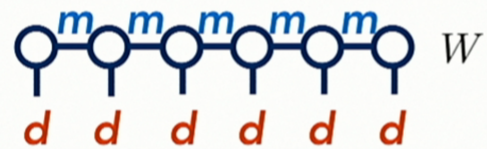
Two types of regularization:

- size **d** of local feature map: controls sensitivity to changes in individual x_j
- tensor network approximation (e.g. MPS bond dimension ***m***)



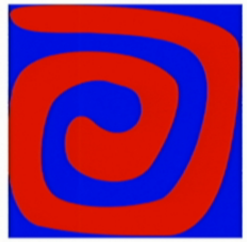
At minimum, these limit number of parameters

Regularization



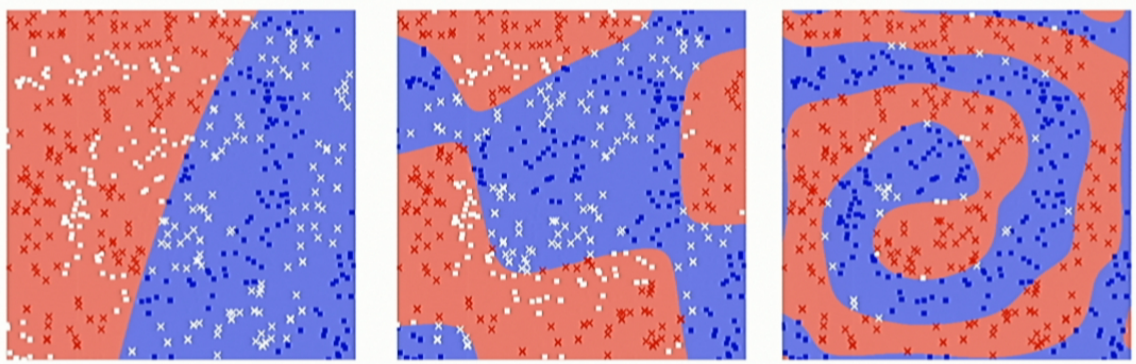
Test size *d* of local feature map on toy 2d data

Hidden distribution

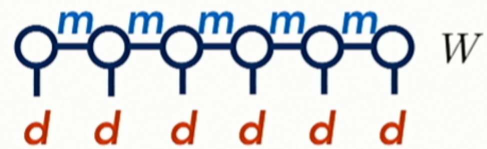


Learn from samples, taking $\phi(x_j) = [1, x_j, x_j^2, \ldots, x_j^{(d-1)}]$

Results: d=2 d=4 d=9



Regularization



Role of bond dimension *m*?

Can show *optimal* weight tensor has form

$$W = \sum_n \alpha_n \Phi^\dagger(\mathbf{x}_n)$$

the "*representer theorem*"

Tensor network approx. can violate this condition

$$W \approx \tilde{W} \neq \sum_n \tilde{\alpha}_n \Phi^\dagger(\mathbf{x}_n) \quad \text{for any} \quad \{\tilde{\alpha}_n\}$$

Tensor network model beyond just interpolation

Interesting consequences for generalization?

Blondel et al., arxiv:1607.08810 (2016)

Some Future Directions

- Useful to interpret $|W \cdot \Phi(\mathbf{x})|^2$ as probability density? Could import even more physics insights.

- For which data sets is entanglement of W small? (i.e. efficient tensor network approximation) Implications for quantum algorithms?

- Best tensor networks for various classes of data?

- Features extracted by elements of tensor network?