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
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 adaptsto** 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

\[ u_j \]

\[ M_{ij} \]

\[ T_{ijk} \]
What is a tensor network?

Joining lines implies contraction, can omit names

\[ \sum_j M_{ij} v_j \]
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

\[ \approx \]

MPS approximation controlled by bond dimension "m"

Represent tensor with \( d^N \) parameters using only \( N d m^2 \) 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)
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 tensor product of "local feature vectors"

\[
x = [x_1, x_2, x_3, \ldots, x_N] \quad \text{raw inputs}
\]

\[
\Phi(x) = \begin{bmatrix} \phi_1(x_1) \\ \phi_2(x_1) \end{bmatrix} \otimes \begin{bmatrix} \phi_1(x_2) \\ \phi_2(x_2) \end{bmatrix} \otimes \begin{bmatrix} \phi_1(x_3) \\ \phi_2(x_3) \end{bmatrix} \otimes \cdots \otimes \begin{bmatrix} \phi_1(x_N) \\ \phi_2(x_N) \end{bmatrix} \quad \text{feature vector}
\]
Proposal for learning

Tensor notation

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

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

Local feature map, dimension $d=2$

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

Crucially, grayscale values not orthogonal
Decision function \( f(x) = W \cdot \Phi(x) \)

Weights \( W \) live in same space as \( \Phi(x) \)

\[
f(x) = \underbrace{W}_{\Phi(x)} = \sum_{s_1, s_2, \ldots, s_N} W_{s_1 s_2 \cdots s_N} \phi^{s_1}(x_1) \phi^{s_2}(x_2) \cdots \phi^{s_N}(x_N)
\]
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 $W^\ell$

(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 $W^\ell$

(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

\[ C = \frac{1}{2} \sum_{n=1}^{N_T} \sum_{\ell} (f^\ell(x_n) - \delta^\ell_{L_n})^2 \]

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

### Results

Only ~3 sweeps needed to converge

<table>
<thead>
<tr>
<th>Bond dimension</th>
<th>Test Set Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m = 10 )</td>
<td>(~5% ) ( (500/10,000 \text{ incorrect}) )</td>
</tr>
<tr>
<td>( m = 20 )</td>
<td>(~2% ) ( (200/10,000 \text{ incorrect}) )</td>
</tr>
<tr>
<td>( m = 120 )</td>
<td>( 0.97% ) ( (97/10,000 \text{ incorrect}) )</td>
</tr>
</tbody>
</table>

→ 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

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

Decision function becomes

$$f(x) = W \cdot \Phi(x)$$

$$= W_{111\ldots1}$$

$$+ W_{211\ldots1} x_1 + W_{121\ldots1} x_2 + W_{112\ldots1} x_3 + \ldots$$

$$+ W_{221\ldots1} x_1 x_2 + W_{212\ldots1} x_1 x_3 + \ldots$$

$$+ \ldots$$

$$+ W_{222\ldots2} x_1 x_2 x_3 \cdots x_N$$

Novikov, Trofimov, Oseledets, arxiv:1605.03795 (2016)
What is the form of the decision function?

Expand local d=2 feature map

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

Decision function becomes

$$f(x) = W \cdot \Phi(x)$$

$$= W_{111\ldots1}$$

$$+ W_{211\ldots1} x_1 + W_{121\ldots1} x_2 + W_{112\ldots1} x_3 + \ldots$$

$$+ W_{221\ldots1} x_1 x_2 + W_{212\ldots1} x_1 x_3 + \ldots$$

$$+ \ldots$$

$$+ W_{222\ldots2} x_1 x_2 x_3 \ldots x_N$$

Novikov, Trofimov, Oseledets, arxiv:1605.03795 (2016)
Consider extreme case of product decomposition (m=1 MPS)

\[ f(x) = \Phi(x) W = \begin{bmatrix} w_1^{(j)} \\ w_2^{(j)} \end{bmatrix} \]

\[ = w_1^{(1)} w_1^{(2)} w_1^{(3)} \cdots w_1^{(N)} + (w_2^{(1)} w_1^{(2)} w_1^{(3)} \cdots w_1^{(N)}) x_1 + (w_1^{(1)} w_2^{(2)} w_1^{(3)} \cdots w_1^{(N)}) x_2 + \ldots \]

\[ + (w_2^{(1)} w_2^{(2)} w_1^{(3)} \cdots w_1^{(N)}) x_1 x_2 + (w_2^{(1)} w_1^{(2)} w_2^{(3)} \cdots w_1^{(N)}) x_1 x_3 + \ldots \]
Interesting interpretation:

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

\[ 0 \ 1 \ 2 \ 3 \quad \text{fictitious "particle" number} \]

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

\[ f_1(x) = W_{100\ldots0} x_1 + W_{010\ldots0} x_2 + W_{001\ldots0} x_3 + \ldots \]

\[ f_2(x) = W_{200\ldots0} x_1^2 + W_{110\ldots0} x_1 x_2 + W_{101\ldots0} x_1 x_3 + \ldots \]

\[ f_3(x) = W_{300\ldots0} x_1^3 + W_{210\ldots0} x_1^2 x_2 + W_{111\ldots0} x_1 x_2 x_3 + \ldots \]
Helpful to view classification in two stages

\[ f^\ell(x) = \Phi(x) \quad W^\ell \]

\begin{align*}
\text{gauge} & \quad \text{transformation} \\
\text{use features to classify} & \quad \text{extract and select } m^2 \\
\text{important features} & \quad \text{features}
\end{align*}
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

$$W = \sum_n \alpha_n \Phi^\dagger(x_n)$$

the "*representer theorem""

Tensor network approx. can violate this condition

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

Tensor network model beyond just interpolation
Interesting consequences for generalization?

Blondel et al., arxiv:1607.08810 (2016)
Some Future Directions

• Useful to interpret $|W \cdot \Phi(x)|^2$ 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?