Understanding Deep Learning via Physics:

The use of Quantum Entanglement for studying the Inductive Bias of Convolutional Networks

Nadav Cohen

Institute for Advanced Study

Symposium on Physics and Machine Learning

The City University of New York Graduate Center

15 December 2017
Sources

Deep SimNets
  C, Sharir and Shashua
  *Computer Vision and Pattern Recognition (CVPR) 2016*

On the Expressive Power of Deep Learning: A Tensor Analysis
  C, Sharir and Shashua
  *Conference on Learning Theory (COLT) 2016*

Convolutional Rectifier Networks as Generalized Tensor Decompositions
  C and Shashua
  *International Conference on Machine Learning (ICML) 2016*

Inductive Bias of Deep Convolutional Networks through Pooling Geometry
  C and Shashua
  *International Conference on Learning Representations (ICLR) 2017*

Tensorial Mixture Models
  Sharir, Tamari, C and Shashua
  *arXiv preprint 2017*

Boosting Dilated Convolutional Networks with Mixed Tensor Decompositions
  C, Tamari and Shashua
  *arXiv preprint 2017*

Deep Learning and Quantum Entanglement: Fundamental Connections with Implications to Network Design
  Levine, Yakira, C and Shashua
  *arXiv preprint 2017*
Collaborators

- Or Sharir
- Amnon Shashua
- Ronen Tamari
- Yoav Levine
- David Yakira
- Nadav Cohen (IAS)
Outline

1. Perspective: Understanding Deep Learning

2. Convolutional Networks as Hierarchical Tensor Decompositions

3. Expressive Efficiency
   - Efficiency of Depth
   - Efficiency of Interconnectivity

4. Inductive Bias via Quantum Entanglement

5. Conclusion
Statistical Learning Setup

\( \mathcal{X} \) – instance space (e.g. \( \mathbb{R}^{100 \times 100} \) for 100-by-100 grayscale images)

\( \mathcal{Y} \) – label space (e.g. \( \mathbb{R} \) for regression or \( [k] := \{1, \ldots, k\} \) for classification)

\( \mathcal{D} \) – distribution over \( \mathcal{X} \times \mathcal{Y} \) (unknown)

\( \ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_{\geq 0} \) – loss func (e.g. \( \ell(y, \hat{y}) = (y - \hat{y})^2 \) for \( \mathcal{Y} = \mathbb{R} \))

**Task**

Given training sample \( S = \{(X_1, y_1), \ldots, (X_m, y_m)\} \) drawn i.i.d. from \( \mathcal{D} \), return hypothesis (predictor) \( h : \mathcal{X} \rightarrow \mathcal{Y} \) that minimizes population loss:

\[
L_D(h) := \mathbb{E}_{(X, y) \sim \mathcal{D}}[\ell(y, h(X))]
\]

**Approach**

Predetermine hypotheses space \( \mathcal{H} \subset \mathcal{Y}^{\mathcal{X}} \), and return hypothesis \( h \in \mathcal{H} \) that minimizes empirical loss:

\[
L_S(h) := \mathbb{E}_{(X, y) \sim S}[\ell(y, h(X))] = \frac{1}{m} \sum_{i=1}^{m} \ell(y_i, h(X_i))
\]
Three Pillars of Statistical Learning Theory: Expressiveness, Generalization and Optimization

$f^*_D$ – ground truth $\left( \arg\min_{f \in \mathcal{Y}^X} L_D(f) \right)$

$h^*_D$ – optimal hypothesis $\left( \arg\min_{h \in \mathcal{H}} L_D(h) \right)$

$h^*_S$ – empirically optimal hypothesis $\left( \arg\min_{h \in \mathcal{H}} L_S(h) \right)$

$h$ – returned hypothesis
**Optimization**

Empirical loss minimization is a convex program:

$$\overline{h} \approx h^*_S \ (\text{training err} \approx 0)$$

**Expressiveness & Generalization**

Bias-variance trade-off:

<table>
<thead>
<tr>
<th>H</th>
<th>approximation err</th>
<th>estimation err</th>
</tr>
</thead>
<tbody>
<tr>
<td>expands</td>
<td>↓</td>
<td>↗</td>
</tr>
<tr>
<td>shrinks</td>
<td>↑</td>
<td>↓</td>
</tr>
</tbody>
</table>
Deep Learning

Optimization

Empirical loss minimization is a non-convex program:
- $h_s^*$ is not unique – many hypotheses have low training err
- Stochastic Gradient Descent somehow reaches one of these

Expressiveness & Generalization

Vast difference from classical ML:
- Some low training err hypotheses generalize well, others don’t
- W/typical data, solution returned by SGD often generalizes well
- Expanding $\mathcal{H}$ reduces approximation err, but also estimation err!
Outline

1. Perspective: Understanding Deep Learning

2. Convolutional Networks as Hierarchical Tensor Decompositions

3. Expressive Efficiency
   - Efficiency of Depth
   - Efficiency of Interconnectivity

4. Inductive Bias via Quantum Entanglement

5. Conclusion
Convolutional Networks

Most successful deep learning arch to date!

**Classic structure:**

![Classic structure diagram](image)

**Modern variants:**

![Modern variants diagrams](image)

Traditionally used for images/video, nowadays for audio and text as well
Convolutional Networks as Hierarchical Tensor Decompositions

**Tensor Product of $L^2$ Spaces**

ConvNets realize **func over many local elements** (e.g. pixels, audio samples)

Let $\mathbb{R}^s$ be the space of such elements (e.g. $\mathbb{R}^3$ for RGB pixels)

Consider:

- $L^2(\mathbb{R}^s)$ – space of func over single element
- $L^2((\mathbb{R}^s)^N)$ – space of func over $N$ elements

**Fact**

$L^2((\mathbb{R}^s)^N)$ is equal to the **tensor product** of $L^2(\mathbb{R}^s)$ with itself $N$ times:

$$L^2((\mathbb{R}^s)^N) = L^2(\mathbb{R}^s) \otimes \cdots \otimes L^2(\mathbb{R}^s)$$

**Implication**

If $\{f_d(x)\}_{d=1}^{\infty}$ is a basis\(^1\) for $L^2(\mathbb{R}^s)$, the following is a basis for $L^2((\mathbb{R}^s)^N)$:

$$\left\{(x_1, \ldots, x_N) \mapsto \prod_{i=1}^{N} f_{d_i}(x_i)\right\}_{d_1 \ldots d_N = 1}^{\infty}$$

\(^1\)Set of linearly independent func w/dense span
**Coefficient Tensor**

For practical purposes, restrict $L^2(\mathbb{R}^s)$ basis to a finite set: $f_1(x) \ldots f_M(x)$

We call $f_1(x) \ldots f_M(x)$ **descriptors**

General func over $N$ elements can now be written as:

$$h(x_1, \ldots, x_N) = \sum_{d_1 \ldots d_N = 1}^{M} A_{d_1 \ldots d_N} \prod_{i=1}^{N} f_{d_i}(x_i)$$

w/func fully determined by the **coefficient tensor**:

$$A \in \mathbb{R}^{M \times \cdots \times M}$$

**Example**

- 100-by-100 images ($N = 10^4$)
- pixels represented by 256 descriptors ($M = 256$)

Then, func over images correspond to coeff tensors of:

- order $10^4$
- dim 256 in each mode
Decomposing Coefficient Tensor

→ Convolutional Arithmetic Circuit

\[ h(x_1, \ldots, x_N) = \sum_{d_1 \ldots d_N=1}^M A_{d_1 \ldots d_N} \prod_{i=1}^N f_{d_i}(x_i) \]

Coeff tensor \( A \) is exponential (in \# of elements \( N \))

→ directly computing a general func is intractable

**Observation**

Applying **hierarchical decomposition** to coeff tensor gives ConvNet w/linear activation and product pooling (**Convolutional Arithmetic Circuit**)!

- decomposition type (mode tree, internal ranks etc) ↔ network structure (depth, width, pooling etc)
- decomposition parameters ↔ network weights
Example 1: CP Decomposition \(\rightarrow\) Shallow Network

\[
h(x_1, \ldots, x_N) = \sum_{d_1 \ldots d_N=1}^{M} \mathcal{A}_{d_1 \ldots d_N} \prod_{i=1}^{N} f_{d_i}(x_i)
\]

W/CP decomposition applied to coeff tensor:

\[
\mathcal{A} = \sum_{\gamma=1}^{r_0} a_{\gamma,1,1,y} \cdot a_{0,1,\gamma} \otimes a_{0,2,\gamma} \otimes \ldots \otimes a_{0,N,\gamma}
\]

func is computed by shallow network (single hidden layer, global pooling):

\[
\text{input } X \quad \text{representation} \quad 1\times1 \text{ conv} \quad \text{global pooling} \quad \text{dense (output)}
\]

\[
\text{rep}(i,d) = f_{\theta_d}(x_i)
\]

\[
\text{conv}(j,\gamma) = \langle a_{0,j,\gamma}, \text{rep}(j,:)) \rangle
\]

\[
\text{pool}(\gamma) = \prod_{j \text{ covers space}} \text{conv}(j,\gamma)
\]

\[
\text{out}(y) = \langle a_{1,1,y}, \text{pool}(:) \rangle
\]
Example 2: HT Decomposition \(\rightarrow\) Deep Network

\[
h(x_1, \ldots, x_N) = \sum_{d_1 \ldots d_N=1}^{M} \mathcal{A}_{d_1 \ldots d_N} \prod_{i=1}^{N} f_{d_i}(x_i)
\]

W/Hierarchical Tucker (HT) decomposition applied to coeff tensor:

\[
\phi^{1,j,\gamma} = \sum_{\alpha=1}^{r_0} a^{1,j,\gamma}_{\alpha} \cdot a^{0,2j-1,\alpha} \otimes a^{0,2j,\alpha}
\]

\[
\phi^{l,j,\gamma} = \sum_{\alpha=1}^{r_{l-1}} a^{l,j,\gamma}_{\alpha} \cdot \phi^{l-1,2j-1,\alpha} \otimes \phi^{l-1,2j,\alpha}
\]

\[
\mathcal{A} = \sum_{\alpha=1}^{r_L-1} a^{L,1,y}_{\alpha} \cdot \phi^{L-1,1,\alpha} \otimes \phi^{L-1,2,\alpha}
\]

func is computed by deep network w/size-2 pooling windows:
Generalization to Other Types of Convolutional Networks

We established equivalence:

hierarchical tensor decompositions $\leftrightarrow$ conv arith circuits (ConvACs)

ConvACs deliver promising empirical results, but other types of ConvNets (e.g. w/ ReLU activation and max/ave pooling) are much more common.

The equivalence extends to other types of ConvNets if we generalize the notion of tensor product:

\[
\text{Tensor product:} \quad (A \otimes B)_{d_1\ldots d_{P+Q}} = A_{d_1\ldots d_P} \cdot B_{d_{P+1}\ldots d_{P+Q}}
\]

\[
\text{Generalized tensor product:} \quad (A \otimes_g B)_{d_1\ldots d_{P+Q}} := g(A_{d_1\ldots d_P}, B_{d_{P+1}\ldots d_{P+Q}})
\]

(same as $\otimes$ but w/general $g : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ instead of mult)

---

1. *Deep SimNets, CVPR’16*; *Tensorial Mixture Models, arXiv’17*
2. *Convolutional Rectifier Networks as Generalized Tensor Decompositions, ICML’16*
Outline

1. Perspective: Understanding Deep Learning
2. Convolutional Networks as Hierarchical Tensor Decompositions
3. Expressive Efficiency
   - Efficiency of Depth
   - Efficiency of Interconnectivity
4. Inductive Bias via Quantum Entanglement
5. Conclusion
Expressive Efficiency

Expressiveness

$f_D^*$ – ground truth ($\arg\min_{f \in \mathcal{Y}^X} L_D(f)$)

$h_D^*$ – optimal hypothesis ($\arg\min_{h \in \mathcal{H}} L_D(h)$)

$h_S^*$ – empirically optimal hypothesis ($\arg\min_{h \in \mathcal{H}} L_S(h)$)

$h$ – returned hypothesis
Expressive efficiency compares network arch in terms of their ability to compactly represent func

Let:
- $\mathcal{H}_A$ – space of func compactly representable by network arch $A$
- $\mathcal{H}_B$ – network arch $B$

A is **efficient** w.r.t. $B$ if $\mathcal{H}_A$ is a strict superset of $\mathcal{H}_B$

A is **completely efficient** w.r.t. $B$ if $\mathcal{H}_B$ has zero “volume” inside $\mathcal{H}_A$
Expressive Efficiency – Formal Definition

Network arch $A$ is **efficient** w.r.t. network arch $B$ if:

1. $\forall$ func realized by $B$ w/ size $r_B$ can be realized by $A$ w/ size $r_A \in O(r_B)$
2. $\exists$ func realized by $A$ w/ size $r_A$ requiring $B$ to have size $r_B \in \Omega(f(r_A))$, where $f(\cdot)$ is super-linear

$A$ is **completely efficient** w.r.t. $B$ if (2) holds for all its func but a set of Lebesgue measure zero (in weight space)
Outline

1. Perspective: Understanding Deep Learning
2. Convolutional Networks as Hierarchical Tensor Decompositions
3. Expressive Efficiency
   - Efficiency of Depth
   - Efficiency of Interconnectivity
4. Inductive Bias via Quantum Entanglement
5. Conclusion
Longstanding conjecture

**Efficiency of depth**: deep ConvNets realize func that require shallow ConvNets to have exponential size (width)
Expressive Efficiency

Efficiency of Depth

Tensor Decomposition Viewpoint

\[ h(x_1, \ldots, x_N) = \sum_{d_1 \ldots d_N = 1}^{M} A_{d_1 \ldots d_N} \prod_{i=1}^{N} f_{d_i}(x_i) \]

Shallow Network

\[ \leftrightarrow \quad \mathcal{A} = \sum_{\gamma=1}^{r_0} a^{1,1,y}_{\gamma} \cdot a^{0,1,\gamma} \otimes \cdots \otimes a^{0,N,\gamma} \]

Deep Network

\[ \leftrightarrow \quad \mathcal{A} = \sum_{\alpha=1}^{r_0} a^{L,1,y}_{\alpha} \cdot \phi^{L-1,1,\alpha} \otimes \phi^{L-1,2,\alpha} \]

Efficiency of depth

HT decomposition realizes tensors that require CP decomposition to have exponential rank \( (r_0 \text{ exponential in } N) \)
HT vs. CP Analysis

Theorem

Besides a negligible (zero measure) set, all parameter settings for HT decomposition lead to tensors with CP-rank exponential in $N$.

**HT Decomposition**

$$
\phi_{1,j,\gamma} = \sum_{\alpha=1}^{r_0} a_{\alpha}^{1,j,\gamma} \cdot a^{0,2j-1,\alpha} \otimes a^{0,2j,\alpha}
$$

$$
\phi_{l,j,\gamma} = \sum_{\alpha=1}^{r_{l-1}} a_{\alpha}^{l,j,\gamma} \cdot \phi^{l-1,2j-1,\alpha} \otimes \phi^{l-1,2j,\alpha}
$$

$$
\mathcal{A} = \sum_{\alpha=1}^{r_L-1} a_{\alpha}^{L,1,\gamma} \cdot \phi^{L-1,1,\alpha} \otimes \phi^{L-1,2,\alpha}
$$

**CP Decomposition**

$$
\mathcal{A} = \sum_{\gamma=1}^{r_0} a_{\gamma}^{1,1,\gamma} \cdot a^{0,1,\gamma} \otimes \ldots \otimes a^{0,N,\gamma}
$$
HT vs. CP Analysis (cont’d)

Theorem proof sketch

- \([\mathcal{A}]\) – matricization of \(\mathcal{A}\) (arrangement of tensor as matrix)
- \(\odot\) – Kronecker product for matrices. Holds: \(\text{rank}(A \odot B) = \text{rank}(A) \cdot \text{rank}(B)\)
- Relation between tensor and Kronecker products: \([\mathcal{A} \otimes \mathcal{B}] = [\mathcal{A}] \odot [\mathcal{B}]\)
- Implies: \(\text{rank}[\mathcal{A}] \leq \text{CP-rank}(\mathcal{A})\)
- By induction over levels of HT, \(\text{rank}[\mathcal{A}]\) is exponential almost always:
  - Base: “SVD has maximal rank almost always”
  - Step: \(\text{rank}[\mathcal{A} \otimes \mathcal{B}] = \text{rank}([\mathcal{A}] \odot [\mathcal{B}]) = \text{rank}[\mathcal{A}] \cdot \text{rank}[\mathcal{B}]\), and “linear combination preserves rank almost always”

<table>
<thead>
<tr>
<th>HT Decomposition</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\phi^1_{j,\gamma}) = (\sum_{\alpha=1}^{r_0} a^{1,j,\gamma}_{\alpha} \cdot a^{0,2j-1,\alpha} \otimes a^{0,2j,\alpha})</td>
</tr>
<tr>
<td>(\phi^l_{j,\gamma}) = (\sum_{\alpha=1}^{r_{l-1}} a^{l,j,\gamma}_{\alpha} \cdot \phi^{l-1,2j-1,\alpha} \otimes \phi^{l-1,2j,\alpha})</td>
</tr>
<tr>
<td>(\mathcal{A} = \sum_{\alpha=1}^{r_L-1} a^L_{\alpha} \cdot \phi^{L-1,1,\alpha} \otimes \phi^{L-1,2,\alpha})</td>
</tr>
</tbody>
</table>
Randomizing \textbf{weights} of deep ConvAC by a cont distribution leads, w.p. 1, to func that require shallow ConvAC to have exponential \# of channels
HT vs. CP Analysis – Generalizations

HT vs. CP analysis may be generalized in various ways, e.g.:

- **Comparison between arbitrary depths**
  Penalty in resources is double-exponential w.r.t. # of layers cut-off

\[
\# \text{ of layers} \quad \# \text{ of parameters} \quad \text{optimal} \quad O\left(\frac{r^N}{2}\right) \quad L_1 \quad f(l) = O\left(\frac{r^{2L-l}}{2^{L-1}}\right)
\]

- **Adaptation to other types of ConvNets**
  W/ReLU activation and max pooling, deep nets realize func requiring shallow nets to be exponentially large, but not almost always

**Efficiency of depth is incomplete w/ReLU ConvNets!**
Outline

1. Perspective: Understanding Deep Learning

2. Convolutional Networks as Hierarchical Tensor Decompositions

3. Expressive Efficiency
   - Efficiency of Depth
   - Efficiency of Interconnectivity

4. Inductive Bias via Quantum Entanglement

5. Conclusion
Efficiency of Interconnectivity

Classic ConvNets have feed-forward (chain) structure:

Modern ConvNets employ elaborate connectivity schemes:

Inception (GoogLeNet)  ResNet  DenseNet

Question
Can such connectivities lead to expressive efficiency?
We focus on dilated ConvNets (D-ConvNets) for sequence data:

- 1D ConvNets
- No pooling
- Dilated (gapped) conv windows

Underlie Google’s WaveNet & ByteNet – state of the art for audio & text!
Dilations and Mode Trees

W/D-ConvNet, mode tree underlying corresponding tensor decomposition determines dilation scheme
Expressive Efficiency

Efficiency of Interconnectivity

Mixed Tensor Decompositions

Let:  \( T, \bar{T} \) – mode trees; \( \text{mix}(T, \bar{T}) \) – set of nodes present in both trees

A **mixed tensor decomposition** blends together \( T \) and \( \bar{T} \) by running their decompositions in parallel, exchanging tensors in each node of \( \text{mix}(T, \bar{T}) \)
Mixed tensor decomposition corresponds to **mixed D-ConvNet**, formed by interconnecting the networks of $T$ and $\bar{T}$:
Theorem

Mixed tensor decomposition of $T$ and $\bar{T}$ can generate tensors that require individual decompositions to grow quadratically (in terms of their ranks).

Corollary

Mixed D-ConvNet can realize func that require individual networks to grow quadratically (in terms of layer widths).

Experiment

Interconnectivity can lead to expressive efficiency!
Outline

1. Perspective: Understanding Deep Learning

2. Convolutional Networks as Hierarchical Tensor Decompositions

3. Expressive Efficiency
   - Efficiency of Depth
   - Efficiency of Interconnectivity

4. Inductive Bias via Quantum Entanglement

5. Conclusion
Inductive Bias

Networks of reasonable size can only realize a fraction of all possible functions. Expressive efficiency does not explain why this fraction is effective.

Why are these functions interesting?

To explain the effectiveness, one must consider the inductive bias:

- Not all functions are equally useful for a given task.
- The network only needs to represent useful functions.
Unlike expressive efficiency, inductive bias can’t be studied via math alone – it requires reasoning about nature of real-world tasks.

Physics bears the potential to bridge this gap!
Modeling Interactions

ConvNets realize func over many local elements (e.g. pixels, audio samples)

Key property of such func:
interactions modeled between different sets of elements

Questions

- What kind of interactions do ConvNets model?
- How do these depend on network structure?
In quantum physics, state of particle is represented as vec in Hilbert space:

\[
|\text{particle state}\rangle = \sum_{d=1}^{M} a_d \cdot |\psi_d\rangle \in H
\]

System of \( N \) particles is represented as vec in tensor product space:

\[
|\text{system state}\rangle = \sum_{d_1 \ldots d_N=1}^{M} A_{d_1 \ldots d_N} \cdot |\psi_{d_1}\rangle \otimes \cdots \otimes |\psi_{d_N}\rangle \in H \otimes \cdots \otimes H
\]

Quantum entanglement measures quantify interactions that a system state models between sets of particles
Quantum Entanglement (cont’d)

\[ |\text{system state}\rangle = \sum_{d_1 \ldots d_N=1}^M A_{d_1 \ldots d_N} \cdot |\psi_{d_1}\rangle \otimes \cdots \otimes |\psi_{d_N}\rangle \]

Consider partition of the \( N \) particles into sets \( \mathcal{I} \) and \( \mathcal{I}^c \)

\[ [A]_\mathcal{I} \text{ – matricization of coeff tensor } A \text{ w.r.t. } \mathcal{I}: \]

- arrangement of \( A \) as matrix
- rows/cols correspond to modes indexed by \( \mathcal{I}/\mathcal{I}^c \)
Quantum Entanglement (cont’d)

Let $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_R)$ be the singular vals of $[A]_I$

Entanglement measures between particles of $I$ and of $I^c$ are based on $\sigma$:

- **Entanglement Entropy**: entropy of $(\sigma_1^2, \ldots, \sigma_R^2)/\|\sigma\|^2_2$
- **Geometric Measure**: $1 - \sigma_1^2/\|\sigma\|^2_2$
- **Schmidt Number**: $\|\sigma\|_0 = rank([A]_I)$
Entanglement with ConvACs

Structural equivalence:

\[
|\text{system state}\rangle = \sum_{d_1 \ldots d_N = 1}^{M} \mathcal{A}_{d_1 \ldots d_N} \cdot |\psi_{d_1}\rangle \otimes \cdots \otimes |\psi_{d_N}\rangle
\]

func realized by ConvAC

\[
h(x_1, \ldots, x_N) = \sum_{d_1 \ldots d_N = 1}^{M} \mathcal{A}_{d_1 \ldots d_N} \cdot f_{d_1}(x_1) \cdots f_{d_N}(x_N)
\]

We may quantify interactions ConvAC models between input sets by applying entanglement measures to its coeff tensor!
When func $h$ realized by ConvAC is separable w.r.t. input sets $\mathcal{I}/\mathcal{I}^c$:

$$\exists g, g' \text{ s.t. } h(x_1, \ldots, x_N) = g((x_i)_{i \in \mathcal{I}}) \cdot g'((x_i')_{i' \in \mathcal{I}^c})$$

it does not model any interaction between the input sets

In a stat setting, this corresponds to independence of $(x_i)_{i \in \mathcal{I}}$ and $(x_i')_{i' \in \mathcal{I}^c}$

Entanglement measures on coeff tensor of $h$ quantify dist from separability:

- $\mathcal{A}$ has high (low) entanglement w.r.t. $\mathcal{I}/\mathcal{I}^c$  
  $\implies h$ is far from (close to) separability w.r.t. $\mathcal{I}/\mathcal{I}^c$

- Choice of entanglement measure determines distance metric
Quantum Tensor Networks

Coeff tensors of quantum many-body states are simulated via:

**Tensor Networks**

Tensor Networks (TNs):

- Graphs in which: vertices $\leftrightarrow$ tensors edges $\leftrightarrow$ modes

- Edge (mode) connecting two vertices (tensors) represents contraction

*inner-product between vectors*

*matrix multiplication*
ConvACs as Tensor Networks

Coeff tensor of ConvAC may be represented via TN:

- **input**
- **rep**
- **conv**
- **pool**

**Tree structure** corresponds to ConvAC **pooling geometry**

- **Open nodes** correspond to ConvAC **inputs** (e.g., pixels)
- **Edge weights** correspond to ConvAC **layer widths**

*Understanding Deep Learning via Physics*
Theorem ("Quantum Max Flow/Min Cut")

Maximal Schmidt entanglement ConvAC models between input sets $\mathcal{I}/\mathcal{I}^c$ is equal to min cut in respective TN separating nodes of $\mathcal{I}/\mathcal{I}^c$
Controlling entanglement (interactions) modeled by ConvAC is equivalent to controlling min cuts in respective TN.

Open nodes correspond to ConvAC inputs (e.g. pixels), edge weights correspond to ConvAC layer widths, and tree structure corresponds to ConvAC pooling geometry.

Two sources of control: layer widths, pooling geometry.

We may analyze the effect of ConvAC arch on the interactions (entanglement) it can model!
Claim

*Deep (early) layer widths are important for long (short)-range interactions*

Experiment

![Graph showing the relationship between accuracy and the number of channels parameter r, with different lines representing wide-base and wide-tip test and train data for global and local tasks.](image)
Claim

*Input elements pooled together early have stronger interaction*

Experiment

- **Data**
  - closedness: low
  - symmetry: low
  - closedness: high
  - symmetry: low
  - closedness: low
  - symmetry: high
  - closedness: high
  - symmetry: high

- **Archs**
  - square pooling (local interactions)
  - mirror pooling (interactions between reflections)

- **Results**
  - Closedness task
  - Symmetry task

Nadav Cohen (IAS)

Understanding Deep Learning via Physics

Physics-ML, CUNY, Dec'17
Outline

1. Perspective: Understanding Deep Learning
2. Convolutional Networks as Hierarchical Tensor Decompositions
3. Expressive Efficiency
   • Efficiency of Depth
   • Efficiency of Interconnectivity
4. Inductive Bias via Quantum Entanglement
5. Conclusion
Conclusion

- Three pillars of statistical learning theory:
  - Expressiveness
  - Generalization
  - Optimization
    - Well developed theory for classical ML
    - Limited understanding for Deep Learning

- We derive equivalence:
  \[ \text{ConvNets} \leftrightarrow \text{hierarchical tensor decompositions} \]
  and use it to analyze expressive efficiency

- To understand expressiveness, efficiency is not enough – one must consider the inductive bias:
  - This cannot be done via math alone
  - **Physics bears the potential to bridge this gap!**

- We use **Quantum Entanglement and Tensor Networks** to study ConvNets’ ability to model interactions
Future Possibilities

Further studying inductive bias of ConvNets via Quantum Physics:
- Understanding overlapping operations via MERA disentanglers
- Characterizing correlations (interactions) in natural data-sets

Transfer of computational tools between Deep Learning and Physics:
- Training ConvNets w/Tensor Network algorithms (e.g. DMRG)
- Quantum Computation (wave function reconstruction) w/SGD

AND MUCH MORE...
Recent Development \textit{(Levine et al)}: From ConvNets to Recurrent Neural Networks (RNNs)

RNNs – most successful Deep Learning arch for sequence processing

Start-End Entanglement quantifies long-term memory of a network

Authors analyze this via TNs (MPS and generalizations):

Show Start-End Entanglement increases exponentially w/depth!
1. Perspective: Understanding Deep Learning

2. Convolutional Networks as Hierarchical Tensor Decompositions

3. Expressive Efficiency
   - Efficiency of Depth
   - Efficiency of Interconnectivity

4. Inductive Bias via Quantum Entanglement

5. Conclusion
Thank You