On the Expressive Power of Deep Learning: A Tensor Analysis

Nadav Cohen Or Sharir Amnon Shashua

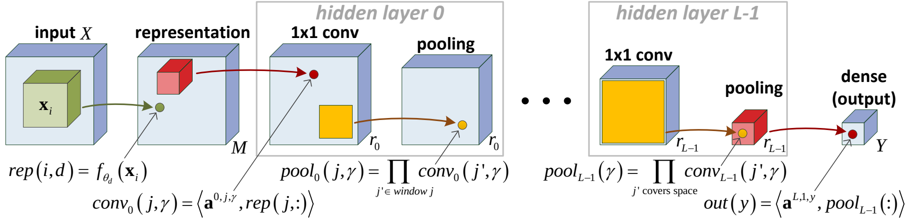
The Hebrew University of Jerusalem

Conference on Learning Theory (COLT) 2016

The Expressive Power of Deep Learning

- Expressive power of depth – the driving force behind Deep Learning
- **Depth efficiency**: when a polynomially sized deep network realizes a function that requires shallow networks to have super-polynomial size
- Prior works on depth efficiency:
  - Show its *existence*, without discussing how frequent it is
  - Do not apply to convolutional networks (locality+sharing+pooling), the most successful deep learning architecture to date

Convolutional Arithmetic Circuits

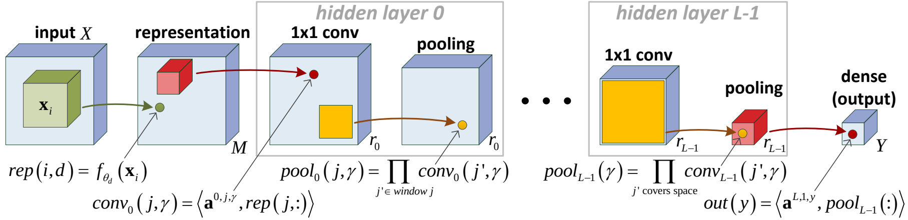


Convolutional networks:

- locality
- sharing (optional)
- product pooling

Computation in log-space leads to **SimNets** – new deep learning architecture showing promising empirical performance [1]

[1] *Deep SimNets, CVPR'16*

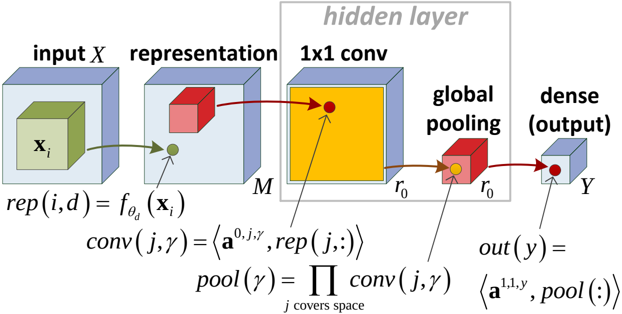


Function realized by output y:

$$h_y(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \sum_{d_1 \ldots d_N = 1}^{M} \mathcal{A}^y_{d_1, \ldots, d_N} \prod_{i=1}^{N} f_{\theta_{d_i}}(\mathbf{x}_i)$$

- $\mathbf{x}_1 \ldots \mathbf{x}_N$ – input patches
- $f_{\theta_1} \ldots f_{\theta_M}$ – representation layer functions
- $\mathcal{A}^y$ – **coefficient tensor** (M^N entries, polynomials in weights $\mathbf{a}^{l,j,\gamma}$)

Shallow network (single hidden layer, global pooling):



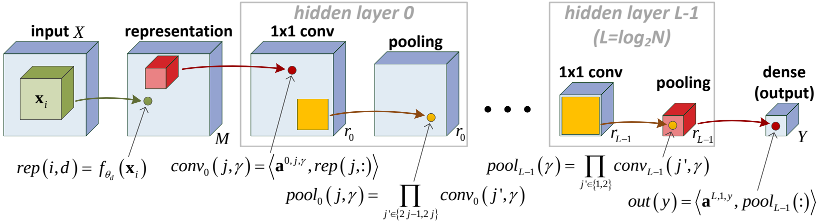
Coefficient tensor $\mathcal{A}^y$ given by classic **CP decomposition**:

$$\mathcal{A}^y = \sum_{\gamma=1}^{r_0} a_\gamma^{1,1,y} \cdot \underbrace{\mathbf{a}^{0,1,\gamma} \otimes \mathbf{a}^{0,2,\gamma} \otimes \cdots \otimes \mathbf{a}^{0,N,\gamma}}_{\text{rank-1 tensor}}$$

$$(rank(\mathcal{A}^y) \leq r_0)$$

Deep Network $\leftrightarrow$ Hierarchical Tucker Decomposition

Deep network ($L = \log_2 N$ hidden layers, size-2 pooling windows):



Coefficient tensor $\mathcal{A}^y$ given by **Hierarchical Tucker decomposition**:

$$
\begin{aligned}
\phi^{1,j,\gamma} &= \sum\nolimits_{\alpha=1}^{r_0} a_\alpha^{1,j,\gamma} \cdot \mathbf{a}^{0,2j-1,\alpha} \otimes \mathbf{a}^{0,2j,\alpha} \\
&\cdots \\
\phi^{l,j,\gamma} &= \sum\nolimits_{\alpha=1}^{r_{l-1}} a_\alpha^{l,j,\gamma} \cdot \phi^{l-1,2j-1,\alpha} \otimes \phi^{l-1,2j,\alpha} \\
&\cdots \\
\mathcal{A}^y &= \sum\nolimits_{\alpha=1}^{r_{L-1}} a_\alpha^{L,1,y} \cdot \phi^{L-1,1,\alpha} \otimes \phi^{L-1,2,\alpha}
\end{aligned}
$$

Theorem of Network Capacity

Theorem

The rank of tensor $\mathcal{A}^y$ given by Hierarchical Tucker decomposition is at least $\min\{r_0, M\}^{N/2}$ almost everywhere w.r.t. decomposition parameters.

Since rank of $\mathcal{A}^y$ generated by CP decomposition is no more than the number of terms (# of hidden channels in shallow network):

Corollary

*Randomizing linear weights of deep network by a continuous distribution gives functions that **with probability one**, cannot be approximated by shallow network with less than $\min\{r_0, M\}^{N/2}$ hidden channels.*

Depth efficiency holds almost always!

Theorem of Network Capacity – Proof Sketch

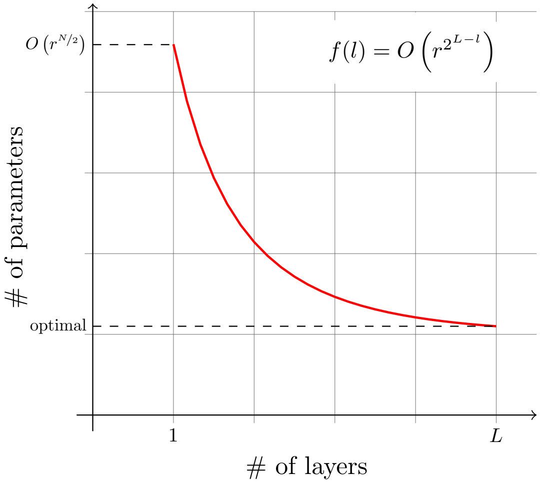
- $[\![\mathcal{A}]\!]$ – arrangement of tensor $\mathcal{A}$ as matrix (*matricization*)
- $\odot$ – Kronecker product for matrices. Holds: $rank(A \odot B) = rank(A) \cdot rank(B)$
- Relation between tensor and Kronecker products: $[\![\mathcal{A} \otimes \mathcal{B}]\!] = [\![\mathcal{A}]\!] \odot [\![\mathcal{B}]\!]$
- Implies: $\mathcal{A} = \sum_{z=1}^{Z} \lambda_z \mathbf{v}_1^{(z)} \otimes \cdots \otimes \mathbf{v}_{2^L}^{(z)} \implies rank[\![\mathcal{A}]\!] \leq Z$
- By induction over $l = 1 \ldots L$, almost everywhere w.r.t. $\{\mathbf{a}^{l,j,\gamma}\}_{l,j,\gamma}$:

$$\forall j \in [{}^{N}/_{2^l}], \gamma \in [r_l] : rank[\![\phi^{l,j,\gamma}]\!] \geq (\min\{r_0, M\})^{2^l/2}$$

  - Base: "SVD has maximal rank almost everywhere"
  - Step: $rank[\![\mathcal{A} \otimes \mathcal{B}]\!] = rank([\![\mathcal{A}]\!] \odot [\![\mathcal{B}]\!]) = rank[\![\mathcal{A}]\!] \cdot rank[\![\mathcal{B}]\!]$, and "linear combination preserves rank almost everywhere"

Generalization

Comparison between arbitrary depths shows penalty in resources grows *double exponentially* w.r.t. number of layers cut off.



Conclusion

Through tensor decompositions, we showed that ***depth efficiency holds almost always with convolutional arithmetic circuits***

Equivalence between convolutional networks and tensor decompositions has many other applications, for example:

- Expressiveness of convolutional ReLU networks: [1]
  - Average pooling leads to loss of universality
  - Depth efficiency exists but does *not* hold almost always
- Inductive bias of convolutional arithmetic circuits: [2]
  - Deep networks can model strong correlation between input elements, shallow networks can't
  - Pooling geometry of a deep network selects supported correlations

[1] *Convolutional Rectifier Networks as Generalized Tensor Decompositions, ICML'16*
[2] *Inductive Bias of Deep Convolutional Networks through Pooling Geometry, arXiv*

Thank You