Understanding Neural Networks and its Roles in Prioritized Search

Yuandong Tian

Research Scientist and Manager
Facebook AI Research
Great Empirical Success from Deep Models
How do deep models work?

This is an apple

“Some Nonlinear Transformation”
Three Major Problems

Understanding how Deep Models work

Expressibility

"Neural Network is a universal approximator"
"Deep Models can express functions more efficiently than shallow ones"

Optimization

"Gradient vanishing/exploding"
"Gradient Descent might get stuck at saddle point / local minima"
"Can GD/SGD go to global optima? How fast?"

Generalization

"Does zero training error often lead to overfitting?"
"More parameters might lead to overfitting."
Supervised Learning

Dataset \[ \{(x_i, y_i)\} \]

Student Network
(Learnable Parameters)

Supervision

\( X \)
Student-Teacher Setting

By Network Expressibility

Teacher Network (Fixed parameters) $m_0$ → $X$ → Student Network (Learnable Parameters) $n_0$

Supervision

No direct supervision
Why Student-Teacher Setting?

Understanding how Deep Models work

- **Expressibility**
  - Provide a target function with bounded complexity

- **Optimization**
  - Study fine dynamics behaviors by comparing with teacher

- **Generalization**
  - Weight alignment with the teacher yields generalization
Old History of Teacher-Student Setting

\[ \epsilon(J) = \frac{1}{2} \langle |f(J, \xi) - f(B, \xi)|^2 \rangle_{\xi} \]
\[ f(J, \xi) = \sum_{i=1}^{K} \sigma(J_i \cdot \xi) \]

Study when the input dimension \( n_0 = m_0 \to +\infty \) (i.e., thermodynamics limits)

In some situations, student nodes are “specialized” to teacher node

One layer of trainable parameters
Nonlinear function \( \sigma(x) = \text{erf}(x / 2) \)
Locally linearized analysis around symmetry breaking plane and final solution

[On-line learning in soft committee machines, Saad & Solla, Phys. Rev 1995]
Proposed Setting

1. Finite $m_0$ and $n_0$
2. Works for $n_i \geq m_i$
(no crazy overparameterization)

$$\min_{\mathcal{W}} J(\mathcal{W}) = \frac{1}{2} \mathbb{E}_x \left[ \| f^*_L(x) - f_L(x) \|^2 \right]$$

No direct supervision

Main Question

**Question:** With over-parameterized student network:

- Small gradient during training
- Student aligns with the teacher

→ Small training error potentially leads to good generalization
Notation

Layer $l$ ($n_l$ nodes)

Layer $l - 1$ ($n_{l-1}$ nodes)

Weight update rule: $\dot{W}_l = \mathbb{E}_x [f_{l-1}(x)g_l^T(x)]$

Activation

$f_l(x) = \begin{bmatrix} f_{l,1}(x) \\ f_{l,2}(x) \end{bmatrix}$

Gradient

$g_l(x) = \begin{bmatrix} g_{l,1}(x) \\ g_{l,2}(x) \end{bmatrix}$

GD: expectation taken over the entire dataset

SGD: expectation taken over a batch
Lemma 1: Recursive Gradient Rule

For layer $l$, there exists $A_l(x)$ and $B_l(x)$ so that:

$$
g_l(x) = D_l(x) \left[ A_l(x)f_l^*(x) - B_l(x)f_l(x) \right]
$$

$A_l(x)$ and $B_l(x)$ are piece-wise constant.
Lemma 1: Recursive Gradient Rule

For layer \( l \), there exists \( A_l(x) \) and \( B_l(x) \) so that:

\[
g_l(x) = D_l(x) \left[ A_l(x)f_l^*(x) - B_l(x)f_l(x) \right]
\]

\( A_l(x) \) and \( B_l(x) \) are piece-wise constant.
Recursive Formula for $A_l(x)$ and $B_l(x)$

$$A_l(x) = V_l^T(x)V_l^*(x)$$

$$B_l(x) = V_l^T(x)V_l(x)$$

Recursive Formula for $V$:

$$V_{l-1}^*(x) = V_l^*(x)D_l^*(x)W_l^{*T}$$

$$V_{l-1}(x) = V_l(x)D_l(x)W_l^T$$

Base case:

$$V_L(x) = V_L^*(x) = I_{C \times C}$$
Main results: Alignment could happen!

student \( k' \)

teacher \( j \)
Definition of Alignment

Activation of node $j$  

Boundary of node $j$  

Boundary of node $k$  

Alignment in the lowest layer
Definition of “Observation”

Teacher \( j \) is observed by a student \( k \)

\[ \partial E_j^* \cap E_k \neq \emptyset \]

Teacher \( j \) is observed by a student \( k \)
Assumption of the dataset

\[ \rho(x) > 0 \]

Infinite dataset!
Assumption of the dataset

Infinite dataset! (Region needs to have interiors)

\[ \rho(x) > 0 \]

\[ R_0 \]
Assumptions on Teacher Network

- Cannot reconstruct arbitrary teachers
  - e.g., all ReLU nodes are dead

![Diagram showing distinct teacher nodes and teacher's boundary visible in the dataset](image-url)
Main results: Alignment could happen!

2-layer network

- **Layer 0**
  - Teacher $j$
  - Student $k'$
  - Observer $k$

- **Layer 1**
Main results: Alignment could happen!

At the lowest layer:

\[ g_1(x) = 0 \text{ for all } x \in R_0 \]

(all input gradients at layer 1 is zero everywhere)

Teacher node \( j \) is observed by a student node \( k \)

Teacher \( j \) is aligned with at least one student \( k' \)
Why?

The gradient of observer $k$ is 0:

From Lemma 1, $g_k(x) = \alpha_k^T f^*(x) - \beta_k^T f(x) = 0$

If $x \in E_k$
Why?

The gradient of observer $k$ is 0:

From Lemma 1, $g_k(x) = \alpha_k^T f^*(x) - \beta_k^T f(x) = 0$

If $x \in E_k$

RelUs are linear independent!

Coefficients for teacher $j$ direction must be 0
Why?

The gradient of observer $k$ is 0:

From Lemma 1, $g_k(x) = \alpha_k^T f^*(x) - \beta_k^T f(x) = 0$

If $x \in E_k$

RelUs are linear independent!

Coefficients for teacher $j$ direction must be 0

Teacher $j$ is aligned with at least one student $k'$

(sum of coefficients = 0)
Why Over-parameterization helps?

More observers!
What happens to unaligned students?

Aligned (can be one-to-many)
Simple 2D experiments

![Graph showing Student and Teacher Boundary with iterations 0 and 2.](image-url)
Simple 2D experiments
L-shape curve at convergence

\[ \|v_k\| \]

10x over-parameterization

10x, loss=0.00

Student nodes

Normalized correlation of a student node to its best correlated teacher
L-shape curve at convergence
Noisy Case \[ \| g_1 (x; \mathcal{W}) \|_\infty \leq \epsilon \]

For teacher $j$, there exists student $k'$:

weights \[ \sin \theta_{jk'} = O \left( \frac{\epsilon^{1-\delta}}{|\alpha_{k,j}|} \right) \]

bias \[ |b^*_j - b_{k'}| = O \left( \frac{\epsilon^{1-2\delta}}{|\alpha_{k,j}|} \right) \]
How to Prove?

Misalignment leads to small overlap
How to Prove?

Small overlap $\Rightarrow$ There exists a datapoint that is far away from all boundaries.
How to Prove?

Pick three points \( x_j, x_j^+, x_j^- \) and there will be one with \(|g_j(x)| > \epsilon\), which is a contradiction.
Multi-Layer case: Alignment could happen!

\[ \alpha_k^T(x)f^*(x) - \beta_k^T(x)f(x) = 0 \]

Piece wise constant, apply the same logic per region!
For 2-layer:
\[
\sqrt{\mathbb{E}_x [\beta_{kk}(x)]} = \|v_k\|
\]
Solutions can be connected by line segments

[Loss Surfaces, Mode Connectivity, and Fast Ensembling of DNNs, Garipov et al. NeurIPS 2018]
[Essentially No Barriers in Neural Network Energy Landscape, Draxler et al, 2018]
[Explaining Landscape Connectivity of Low-cost Solutions for Multilayer Nets, Kuditipudi et al, 2019]
Our Explanation

Student Solution 1

\[ \mathbf{v}_1 = 0 \]
\[ \mathbf{v}_3 = 0 \]

Student Solution 2

\[ \mathbf{v}_1 = 0 \]
Training Dynamics

Critical Points have nice properties!

*Can we achieve that via training with SGD?*

*Not Easy*
Strong/weak teacher nodes

\[ \|v_{j_1}\| \text{ large} \quad \|v_{j_2}\| \text{ small} \]

Strong teacher nodes are learned faster
1. Robust to Noise! 😊
2. Hard to learn weak teacher nodes 😞
Training Dynamics

Strong teacher node attracts many students!

Teacher $j$: $\|v_j^*\| \propto 1/j^p$
Training Dynamics

Teacher $j$: $\|v_j^*\| \propto 1/j^p$

Losing student node shifts focus.
Successful Rate of Teacher Node Reconstruction

$p = 0.5$

$p = 1$

$p = 1.5$

$p = 2$

---

5 epochs

100 epochs

Teacher $j$: $\|\mathbf{v}_j^*\| \propto 1/j^p$
Future Directions

• Training Dynamics
• Generalization Bound
• Landscape
• ResNet / DenseNet / Network with Attention
• Adversarial Samples
Understand the Role Played by Neural Network in Prioritized Search

Carrie Wu\textsuperscript{1}, Lexing Ying\textsuperscript{1}, Yuandong Tian\textsuperscript{2}

\textsuperscript{1}Stanford University, \textsuperscript{2}Facebook AI Research
AlphaGo Series

AlphaGo Lee (Mar. 2016)

AlphaGo Master (May. 2017)

AlphaGo Zero (Oct. 2017)
Monte Carlo Tree Search with Networks

Aggregate win rates, and search towards the good nodes.

$$22.23/40 = Q(s, a) = \frac{\tilde{Q}(s, a)}{N(s, a)}$$
Monte Carlo Tree Search with Networks

Policy Network $P(s, a)$

\[
a_t = \arg\max_a (Q(s_t, a) + u(s_t, a))
\]

\[
u(s, a) \propto \frac{P(s, a)}{1 + N(s, a)}
\]

PUCT
Monte Carlo Tree Search with Networks

Value Network $V(s)$
Monte Carlo Tree Search with Networks
Monte Carlo Tree Search with Networks

How Policy Network and Value Network improves Search Efficiency?

[Mastering the game of Go with deep neural networks and tree search, D. Silver et al. Nature 2016]
A Simple A* Model

- Expand node using policy model
- Prioritize node using value model
- Next node to expand

Priority Queue
\[ V(s_d) = V^* \]

**Notations**

- **Optimal path**
- **Sub-optimal**

**Definitions**

- \( K \): Branching factor
- \( V(s_d) \): True value of state \( s_d \) at depth \( d \)
- \( \Delta(s_d) = V^* - V(s_d) \): Gap to optimal value
- \( U(s_d) \): Predicted **deterministic** value of state \( s_d \) by value net
Notations

\[ X_d = V(s_d) - U(s_d): \]
\[ \text{i.i.d zero-mean random variable at depth } d \]
\[ \sigma_d: \text{ standard deviation} \]

\[ \sigma_d \text{ decays over depth} \]

Set \( c_d = 5\sqrt{d}\sigma_d \)
\[ |X_d| \leq c_d \text{ with high probability} \]

\[ U(s_d) + c_d: \text{ Priority value} \]
Value Network Only

A sub-optimal node is chosen if the heuristic value is over-estimated:

\[ U(s_d) + c_d \geq V^* \quad \text{or} \quad e(s_d) \equiv V^* - U(s_d) - c_d = \Delta(s_d) - X(s_d) - c_d \leq 0 \]

Expected Sample Complexity:

\[ \mathbb{E}[N] = K \left[ D + \sum_{s_d \notin \mathcal{L} \cup \mathcal{A}(l^*)} \mathbb{P} \left( e(s_d) \leq 0 \bigcap_{s_{d'} \in \mathcal{A}(s_d)} e(s_{d'}) \leq 0 \right) \right] \]
Neural Network Models

Constant Gap Models.

\[ V(s_d) = 0 \]

\[ V^* = \eta > 0 \]

\[ V^* - \Delta \]

Generative Models.

\[ \Delta \sim U[0, \eta] \]

\[ V^* - \Delta_1 \]

\[ V^* - \Delta_1 - \Delta_2 - \Delta_3 \]
Value Network Only (Constant Gap Model)

Sample Complexity (#calls of value functions):

\[ \mathbb{E}[N] = KD + D^2(K - 1)K^c \]

for some \( c \) so that \( \frac{\eta}{\sigma_c} - \sqrt{c} \geq \sqrt{2 \log K} \)

\[ \sigma_d = O(d^{-0.5-\delta}) \quad \Rightarrow \quad Polynomial \ sample \ complexity \]
Value Network Only (Generative Model)

Sample Complexity (#calls of value functions):

\[ \mathbb{E}[N] = KD + \sum_{d=1}^{D} K^{T(d)} \]

where \( T(d) = \frac{2}{\eta} \left( \sqrt{2\log K + 1} \right) \sqrt{d} \sigma_d \)

\[ \sigma_d = O(d^{-0.5-\delta}) \rightarrow \text{Polynomial sample complexity} \]
**Success Rate at 20k expansion**

**Constant Gap**

<table>
<thead>
<tr>
<th></th>
<th>Polynomially Decaying Noise</th>
<th>Exponentially Decaying Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\gamma = 1.3$</td>
<td>$\gamma = 1.5$</td>
</tr>
<tr>
<td></td>
<td>Alg 1 MCTS</td>
<td>Alg 1 MCTS</td>
</tr>
<tr>
<td>$\eta = 1$</td>
<td>1 0.51</td>
<td>1 0.695</td>
</tr>
<tr>
<td>$\eta = 0.5$</td>
<td>1 0.38</td>
<td>1 0.435</td>
</tr>
</tbody>
</table>

**Generative Model**

<table>
<thead>
<tr>
<th></th>
<th>Polynomially Decaying Noise</th>
<th>Exponentially Decaying Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\gamma = 1.3$</td>
<td>$\gamma = 1.5$</td>
</tr>
<tr>
<td></td>
<td>Alg 1 MCTS</td>
<td>Alg 1 MCTS</td>
</tr>
<tr>
<td>$\eta = 1$</td>
<td>1 0.895</td>
<td>1 0.895</td>
</tr>
<tr>
<td>$\eta = 0.5$</td>
<td>1 0.865</td>
<td>1 0.865</td>
</tr>
</tbody>
</table>

Polynomial: $X_d \sim N(0, d^{-2\gamma})$, Exponential: $X_d \sim N(0, \alpha^{-2d})$
Adding Policy Networks

\[ P(s, a_k) = \frac{\exp(U^\pi(s, a_k))}{\sum_{k=1}^{K} \exp(U^\pi(s, a_k))} \]

Assume \( U^\pi(s, a_k) = V(s'(s, a_k)) + X^\pi_d \):

\( X^\pi_d \) is i.i.d zero-mean random variable at depth \( d \)

\( \sigma^\pi_d \) : standard deviation
Adding Policy Networks

\[ P(s, a) \] One forward yields many values.

\[ U(s) \] One forward yields a single value

Sort \( P(s, a) \) so that \( P(s, a_1) \geq P(s, a_2) \geq \cdots \geq P(s, a_K) \)

If \( \log P(s, a_1) - \log P(s, a_k) \geq 2c_{\pi_d} \), stop expanding now.
Value and Policy Networks

Sample Complexity (#calls of neural networks):

\[
\mathbb{E}[N] \leq \sum_{s_d \notin \mathcal{L}} \left( 2 + \sum_{k=2}^{K-1} \mathbb{P}\left( U^\pi(s_d, a_1) - U^\pi(s_d, a_k) \leq 2c_{d+1}^\pi \right) \right).
\]

\[
\mathbb{P}\left( e(s_d) \leq 0 \bigcap_{s_{d'} \in \mathcal{A}(s_d)} e(s_{d'}) \leq 0 \right)
\]

No fixed \( K \) expansions anymore
## Value + Policy (Success Rate at 20k expansion)

### Constant Gap

<table>
<thead>
<tr>
<th></th>
<th>Polynomially Decaying Noise</th>
<th>Exponentially Decaying Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\gamma = 1.3$</td>
<td>$\gamma = 1.5$</td>
</tr>
<tr>
<td></td>
<td>Alg 2</td>
<td>PUCT</td>
</tr>
<tr>
<td>$\eta = 1$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\eta = 0.5$</td>
<td>1</td>
<td>0.885</td>
</tr>
</tbody>
</table>

### Generative Model

<table>
<thead>
<tr>
<th></th>
<th>Polynomially Decaying Noise</th>
<th>Exponentially Decaying Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\gamma = 1.3$</td>
<td>$\gamma = 1.5$</td>
</tr>
<tr>
<td></td>
<td>Alg 2</td>
<td>PUCT</td>
</tr>
<tr>
<td>$\eta = 1$</td>
<td>1</td>
<td>0.94</td>
</tr>
<tr>
<td>$\eta = 0.5$</td>
<td>1</td>
<td>0.935</td>
</tr>
</tbody>
</table>
Future Work

- PUCT (MCTS + Policy Network) becomes much more efficient, why?
- Visitation counts (memory)
- Max versus Average, which one is better in which situations
- Test it in real games/environment.
Thanks!