Estimating the Information Flow in Deep Neural Networks

Ziv Goldfeld

MIT
Complex System: Multi-component system driven by local interactions
**Complex System:** Multi-component system driven by local interactions

- Human Brain
- Economic Networks
- Physical Matter
- Social Networks
- Earth’s Climate
**Complex System:** Multi-component system driven by local interactions
Complex System: Multi-component system driven by local interactions
Complex System: Multi-component system driven by local interactions

Emerging Technologies:
Shrink magnetic region per bit
**Complex System:** Multi-component system driven by local interactions

**Emerging Technologies:**
Shrink magnetic region per bit

**Challenges:**
Stabilization of written data
Complex System: Multi-component system driven by local interactions

Emerging Technologies:
Shrink magnetic region per bit

Challenges:
Stabilization of written data

Model & Study:
Interacting particle sys.

↓
Storage capacity & HDD designs
**Complex System:** Multi-component system driven by local interactions
**Complex System:** Multi-component system driven by local interactions

**Emerging Networks (IoT):**
1) Decentralized & ad hoc
2) Varying connectivity
3) Cooperative components
Complex System: Multi-component system driven by local interactions

Emerging Networks (IoT):
1) Decentralized & ad hoc
2) Varying connectivity
3) Cooperative components

Building Blocks:
Rich IT literature
Complex System: Multi-component system driven by local interactions

Emerging Networks (IoT):
1) Decentralized & ad hoc
2) Varying connectivity
3) Cooperative components

Building Blocks:
Rich IT literature

Network Modeling:
Random graphs &
Actions based on primitives
Complex System: Multi-component system driven by local interactions
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
- Lacking theory:
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
- Lacking theory:
  - What drives the evolution of hidden representations?
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
- Lacking theory:
  - What drives the evolution of hidden representations?
  - What are properties of learned representations?
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
- Lacking theory:
  - What drives the evolution of hidden representations?
  - What are properties of learned representations?
  - How fully trained networks process information?
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
- Lacking theory:
  - What drives the evolution of hidden representations?
  - What are properties of learned representations?
  - How fully trained networks process information?
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
- Lacking theory:
  - What drives the evolution of hidden representations?
  - What are properties of learned representations?
  - How fully trained networks process information?
- Some past attempts to understand effectiveness of deep learning
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
- Lacking theory:
  - What drives the evolution of hidden representations?
  - What are properties of learned representations?
  - How fully trained networks process information?
- Some past attempts to understand effectiveness of deep learning
  - Optimization dynamics in parameter space
    [Saxe-McClelland-Ganguli’14, Choromanska-et al’15, Wei-Lee-Ma’18]
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
- Lacking theory:
  - What drives the evolution of hidden representations?
  - What are properties of learned representations?
  - How fully trained networks process information?

Some past attempts to understand effectiveness of deep learning

- Optimization dynamics in parameter space
  [Saxe-McClelland-Ganguli'14, Choromanska-et al’15, Wei-Lee-Ma’18]

- Classes of efficiently representable functions
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
- Lacking theory:
  - What drives the evolution of hidden representations?
  - What are properties of learned representations?
  - How fully trained networks process information?

Some past attempts to understand effectiveness of deep learning

- Optimization dynamics in parameter space
  [Saxe-McClelland-Ganguli'14, Choromanska-et al’15, Wei-Lee-Ma’18]

- Classes of efficiently representable functions

- Information Bottleneck Theory
  [Tishby-Zaslavsky1'15, Shwartz-Tishby'17, Saxe et al.’18]
Deep Neural Networks

- Unprecedented practical success in hosts of tasks

- Lacking theory:
  - What drives the evolution of hidden representations?
  - What are properties of learned representations?
  - How fully trained networks process information?

- Some past attempts to understand effectiveness of deep learning
  - Optimization dynamics in parameter space
    [Saxe-McClelland-Ganguli'14, Choromanska-et al’15, Wei-Lee-Ma’18]
  - Classes of efficiently representable functions
  - Information Bottleneck Theory
    [Tishby-Zaslavsky1'15, Shwartz-Tishby’17, Saxe et al.’18]
Deep Neural Networks

- Unprecedented practical success in hosts of tasks
- Lacking theory:
  - What drives the evolution of hidden representations?
  - What are properties of learned representations?
  - How fully trained networks process information?
- Some past attempts to understand effectiveness of deep learning
  - Optimization dynamics in parameter space
    [Saxe-McClelland-Ganguli'14, Choromanska-et al’15, Wei-Lee-Ma’18]
  - Classes of efficiently representable functions
  - **Information Bottleneck Theory**
    [Tishby-Zaslavsky1'15, Shwartz-Tishby’17, Saxe et al.’18]

★ **Goal:** Explain ‘compression’ in Information Bottleneck framework
Setup and Preliminaries

Feedforward DNN for Classification:

\[ Y \quad (\text{Label}) \]

\[ X \quad (\text{Feature/Image}) \]

\[ T_0 = X \quad (\text{Input Layer}) \]

\[ T_1 \quad (\text{Hidden Layer 1}) \]

\[ T_2 \quad (\text{Hidden Layer 1}) \]

\[ T_3 \quad (\text{Hidden Layer 1}) \]

\[ T_4 \rightarrow \hat{Y} \quad (\text{Output Layer}) \]
Setup and Preliminaries

Feedforward DNN for Classification:

- **Deterministic DNN:** \( T_\ell = f_\ell(T_{\ell-1}) \)  
  - (MLP: \( T_\ell = \sigma(W_\ell T_{\ell-1} + b_\ell) \))
Feedforward DNN for Classification:

- **Deterministic DNN:** \( T_\ell = f_\ell(T_{\ell-1}) \)  
  - **MLP:** \( T_\ell = \sigma(W_\ell T_{\ell-1} + b_\ell) \)

- **Joint Distribution:** \( P_{X,Y} \)
Setup and Preliminaries

Feedforward DNN for Classification:

- **Deterministic DNN:** $T_\ell = f_\ell(T_{\ell-1})$  
  (MLP: $T_\ell = \sigma(W_\ell T_{\ell-1} + b_\ell)$)

- **Joint Distribution:** $P_{X,Y} \Rightarrow P_{X,Y} \cdot P_{T_1,\ldots,T_L|X}$
Setup and Preliminaries

Feedforward DNN for Classification:

- **Deterministic DNN:** \( T_\ell = f_\ell(T_{\ell-1}) \) (MLP: \( T_\ell = \sigma(W_\ell T_{\ell-1} + b_\ell) \))
- **Joint Distribution:** \( P_{X,Y} \implies P_{X,Y} \cdot P_{T_1,\ldots,T_L|X} \)
- **IB Theory:** Track MI pairs \( (I(X;T_\ell), I(Y;T_\ell)) \) (information plane)
Feedforward DNN for Classification:

IB Theory Claim: Training comprises 2 phases
Feedforward DNN for Classification:

<table>
<thead>
<tr>
<th>Y (Label)</th>
<th>X (Feature/Image)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cat</td>
<td>![Cat Image]</td>
</tr>
<tr>
<td>Dog</td>
<td>![Dog Image]</td>
</tr>
</tbody>
</table>

\[
T_0 = X \quad (\text{Input Layer}) \quad T_1 \quad (\text{Hidden Layer 1}) \quad T_2 \quad (\text{Hidden Layer 1}) \quad T_3 \quad (\text{Hidden Layer 1})
\]

IB Theory Claim: Training comprises 2 phases

- **Fitting:** \( I(Y; T_\ell) \) & \( I(X; T_\ell) \) rise (short)
Setup and Preliminaries

Feedforward DNN for Classification:

IB Theory Claim: Training comprises 2 phases

- **Fitting:** $I(Y; T_\ell) \& I(X; T_\ell)$ rise (short)
- **Compression:** $I(X; T_\ell)$ slowly drops (long)
Setup and Preliminaries

Feedforward DNN for Classification:

IB Theory Claim: Training comprises 2 phases

- **Fitting:** $I(Y; T_\ell)$ & $I(X; T_\ell)$ rise (short)
- **Compression:** $I(X; T_\ell)$ slowly drops (long)

[Shwartz-Tishby'17]
**Vacuous Mutual Information**

<table>
<thead>
<tr>
<th>Observation</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)</em></td>
</tr>
</tbody>
</table>
Observation

Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)

\[ I(X; T_\ell) \text{ is independent of the DNN parameters} \]
Vacuous Mutual Information

Observation

Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid) \( \implies I(X;T_\ell) \) is independent of the DNN parameters

Why?
Vacuous Mutual Information

Observation

\[ \text{Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)} \implies I(X; T_\ell) \text{ is independent of the DNN parameters} \]

Why?

- Continuous \( X \):
Vacuous Mutual Information

Observation

Det. DNNs with strictly monotone nonlinearities (e.g., \texttt{tanh} or \texttt{sigmoid})

\[ \implies I(X; T_\ell) \text{ is independent of the DNN parameters} \]

Why?

- **Continuous** \( X \):
  \[ I(X; T_\ell) = h(T_\ell) - h(T_\ell | X) \]
**Vacuous Mutual Information**

<table>
<thead>
<tr>
<th>Observation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)</strong></td>
</tr>
<tr>
<td>$\implies I(X; T_\ell)$ is independent of the DNN parameters</td>
</tr>
</tbody>
</table>

**Why?**

- **Continuous $X$:**
  $$I(X; T_\ell) = h(T_\ell) - h(T_\ell | X)$$
Observation

Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)

\[ I(X; T_\ell) \] is independent of the DNN parameters

Why?

- Continuous \( X \): \[ I(X; T_\ell) = h(T_\ell) - h(\tilde{f}_\ell(X) \mid X) \]
Vacuous Mutual Information

**Observation**

Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid) \[ \implies I(X; T_\ell) \text{ is independent of the DNN parameters} \]

**Why?**

- **Continuous** \( X \):
  \[
  I(X; T_\ell) = h(T_\ell) - h(\tilde{f}_\ell(X)|X) = -\infty
  \]
Vacuous Mutual Information

Observation

Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)

⇒ \( I(X; T_\ell) \) is independent of the DNN parameters

Why?

- **Continuous \( X \):**
  \[
  I(X; T_\ell) = h(T_\ell) - h(\tilde{f}_\ell(X)|X) = \infty
  \]
Vacuous Mutual Information

Observation

Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)

\[ \Rightarrow I(X; T_\ell) \text{ is independent of the DNN parameters} \]

Why?

- Continuous \( X \): 
  \[ I(X; T_\ell) = h(T_\ell) - h(\tilde{f}_\ell(X)|X) = \infty \]

- Discrete \( X \):
Observation

Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid) \( \implies I(X; T_\ell) \) is independent of the DNN parameters

Why?

- **Continuous** \( X \): \[
I(X; T_\ell) = h(T_\ell) - h(\tilde{f}_\ell(X)|X) = \infty
\]
- **Discrete** \( X \): The map \( X \mapsto T_\ell \) is injective*  

* For almost all weight matrices and bias vectors
Vacuous Mutual Information

**Observation**

*Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)*

\[ \implies I(X; T_\ell) \text{ is independent of the DNN parameters} \]

**Why?**

- **Continuous** \(X\): \[ I(X; T_\ell) = h(T_\ell) - h(\tilde{f}_\ell(X)|X) = \infty \]
- **Discrete** \(X\): The map \(X \mapsto T_\ell\) is injective* \(\implies I(X; T_\ell) = H(X)\)

* For almost all weight matrices and bias vectors
Vacuous Mutual Information

Observation

Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)

$\Rightarrow I(X; T_\ell)$ is independent of the DNN parameters

Why?

- **Continuous $X$:**
  \[ I(X; T_\ell) = h(T_\ell) - h(f_\ell(X) | X) = \infty \]

- **Discrete $X$:**
  The map $X \mapsto T_\ell$ is injective$^\star$ $\Rightarrow I(X; T_\ell) = H(X)$
Vacuous Mutual Information

**Observation**

Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)

\[ \implies I(X; T_\ell) \text{ is independent of the DNN parameters} \]

**Why?**

- **Continuous** \( X \):
  \[ I(X; T_\ell) = h(T_\ell) - h(\tilde{f}_\ell(X)|X) = \infty \]

- **Discrete** \( X \): The map \( X \mapsto T_\ell \) is injective* \[ \implies I(X; T_\ell) = H(X) \]

**Past Works:**

[Shwartz-Tishby’17, Saxe et al.’18]
Plots via binning-based estimator of $I(X; T_\ell)$, for $X \sim \text{Unif}(\text{dataset})$
What is going on here?

- Plots via binning-based estimator of $I(X; T_\ell)$, for $X \sim \text{Unif}(\text{dataset})$
  
  $\implies$ Plotted values are $I(X; \text{Bin}(T_\ell))$
What is going on here?

- Plots via binning-based estimator of $I(X; T_\ell)$, for $X \sim \text{Unif}(\text{dataset})$
  
  $\implies$ Plotted values are $I(X; \text{Bin}(T_\ell)) \approx I(X; T_\ell)$
What is going on here?

- Plots via binning-based estimator of $I(X; T_\ell)$, for $X \sim \text{Unif}($dataset$)$

  $\implies$ Plotted values are $I(X; \text{Bin}(T_\ell)) \approx I(X; T_\ell)$  No!
What is going on here?

- Plots via binning-based estimator of $I(X; T_\ell)$, for $X \sim \text{Unif}(\text{dataset})$

  $\implies$ Plotted values are $I(X; \text{Bin}(T_\ell)) \approx I(X; T_\ell)$ No!
Plots via binning-based estimator of $I(X; T_\ell)$, for $X \sim \text{Unif}(\text{dataset})$

$\implies$ Plotted values are $I(X; \text{Bin}(T_\ell)) \approx I(X; T_\ell)$ \textbf{No!}

Smaller bins $\implies$ Closer to truth: $I(X; T_\ell) = \ln(2^{12}) \approx 8.31$
What is going on here?

- Plots via binning-based estimator of $I(X; T_\ell)$, for $X \sim \text{Unif}(\text{dataset})$
  \[\implies \text{Plotted values are } I(X; \text{Bin}(T_\ell)) \approx I(X; T_\ell) \quad \text{No!}\]

- Smaller bins $\implies$ Closer to truth: $I(X; T_\ell) = \ln(2^{12}) \approx 8.31$

- Binning introduces “noise” into estimator (not present in the DNN)
What is going on here?

- Plots via binning-based estimator of $I(X; T_\ell)$, for $X \sim \text{Unif}(\text{dataset})$

  $\implies$ Plotted values are $I(X; \text{Bin}(T_\ell)) \approx I(X; T_\ell) \quad \text{No!}$

- Smaller bins $\implies$ Closer to truth: $I(X; T_\ell) = \ln(2^{12}) \approx 8.31$

- Binning introduces “noise” into estimator (not present in the DNN)

- Plots showing estimation errors
What is going on here?

- Plots via binning-based estimator of $I(X; T_\ell)$, for $X \sim \text{Unif}(\text{dataset})$
  
  $\implies$ Plotted values are $I(X; \text{Bin}(T_\ell)) \approx I(X; T_\ell)$  
  
  No!

- Smaller bins $\implies$ Closer to truth: $I(X; T_\ell) = \ln(2^{12}) \approx 8.31$

- Binning introduces “noise” into estimator (not present in the DNN)

- Plots showing estimation errors

- **Real Problem:** $I(X; T_\ell)$ is meaningless in det. DNNs
Modification: Inject (small) Gaussian noise to neurons’ output

[G.-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy’18]
**Modification:** Inject (small) Gaussian noise to neurons’ output

[Germain-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy’18]

Formally:  \[ T_\ell = f_\ell(T_{\ell-1}) + Z_\ell, \text{ where } Z_\ell \sim \mathcal{N}(0, \beta^2 I) \]

\[
\begin{align*}
T_{\ell-1} &\quad \xrightarrow{\sigma(W^{(k)}_{\ell}T_{\ell-1} + b_{\ell}(k))} \\ S_\ell(k) &\quad \xrightarrow{+} \\ T_\ell(k) &\quad \xrightarrow{Z_\ell(k) \sim \mathcal{N}(0, \beta^2)}
\end{align*}
\]
**Modification:** Inject (small) Gaussian noise to neurons’ output

[G.-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy’18]

- **Formally:** \( T_\ell = f_\ell(T_{\ell-1}) + Z_\ell, \) where \( Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{align*}
T_{\ell-1} &\xrightarrow{\sigma(W^{(k)}_{\ell}T_{\ell-1} + b_{\ell}(k))} S_{\ell}(k) \\
&\xrightarrow{+} T_\ell(k) \\
Z_\ell(k) &\sim \mathcal{N}(0, \beta^2)
\end{align*}
\]

\( \implies X \mapsto T_\ell \) is a **parametrized channel** that depends on DNN param.
**Modification:** Inject (small) Gaussian noise to neurons’ output

[G.-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy’18]

- **Formally:** \( T_\ell = f_\ell(T_{\ell-1}) + Z_\ell \), where \( Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{align*}
T_{\ell-1} &\xrightarrow{\sigma(W^{(k)}_\ell T_{\ell-1} + b_\ell(k))} S_\ell(k) \\
&\xrightarrow{+} T_\ell(k) \\
&\xrightarrow{Z_\ell(k)} \sim \mathcal{N}(0, \beta^2)
\end{align*}
\]

\( \Rightarrow \) \( X \mapsto T_\ell \) is a **parametrized channel** that depends on DNN param.!

\( \Rightarrow \) \( I(X; T_\ell) \) is a **function** of weights and biases!
Modification: Inject (small) Gaussian noise to neurons’ output

[G.-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy’18]

Formally: $T_\ell = f_\ell(T_{\ell-1}) + Z_\ell$, where $Z_\ell \sim \mathcal{N}(0, \beta^2 \mathbf{I})$

\[ T_{\ell-1} \xrightarrow{\sigma(W^{(k)}_{\ell}T_{\ell-1} + b_\ell(k))} S_\ell(k) \xrightarrow{+} T_\ell(k) \]

$Z_\ell(k) \sim \mathcal{N}(0, \beta^2)$

$\Rightarrow X \mapsto T_\ell$ is a **parametrized channel** that depends on DNN param.!

$\Rightarrow I(X; T_\ell)$ is a **function** of weights and biases!

Operational Perspective:
Modification: Inject (small) Gaussian noise to neurons’ output

[G.-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy’18]

Formally: \( T_\ell = f_\ell(T_{\ell-1}) + Z_\ell \), where \( Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[ T_{\ell-1} \xrightarrow{\sigma(W_\ell^{(k)}T_{\ell-1} + b_\ell(k))} S_\ell(k) \xrightarrow{+} T_\ell(k) \]

\[ Z_\ell(k) \sim \mathcal{N}(0, \beta^2) \]

\( X \mapsto T_\ell \) is a parametrized channel that depends on DNN param.!

\( I(X; T_\ell) \) is a function of weights and biases!

Operational Perspective:

Performance & learned representations similar to det. DNNs (\( \beta \approx 10^{-1} \))
Mutual Information in Noisy DNNs
Mutual Information Estimation in Noisy DNNs

Noisy DNN:

\[
X \xrightarrow{f_1} S_1 \xrightarrow{T_1} f_2 \xrightarrow{S_2} T_2 \ldots
\]

\[Z_1\] and \[Z_2\] are noise terms.
Mutual Information Estimation in Noisy DNNs

Noisy DNN:

\[
X \rightarrow f_1 \rightarrow S_1 \rightarrow T_1 \rightarrow f_2 \rightarrow S_2 \rightarrow T_2 \ldots
\]
Mutual Information Estimation in Noisy DNNs

Noisy DNN:

\[
\begin{align*}
X & \xrightarrow{f_1} S_1 \xrightarrow{T_1} f_2 \xrightarrow{S_2} T_2 & \cdots \\
& \quad \uparrow Z_1 \\
& \quad \uparrow Z_2
\end{align*}
\]
Noisy DNN: \( S_\ell \triangleq f_\ell(T_{\ell-1}) \)
Mutual Information Estimation in Noisy DNNs

**Noisy DNN:**

\[ S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \]
Mutual Information Estimation in Noisy DNNs

Noisy DNN: \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

- Assume: \( X \sim \text{Unif}(\mathcal{X}), \) where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset
Mutual Information Estimation in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

![Diagram of Noisy DNN](image)

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\[ I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell|X = x_i) \]
**Mutual Information Estimation in Noisy DNNs**

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \ Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[ X \xrightarrow{f_1} S_1 \xrightarrow{\oplus} T_1 \xrightarrow{f_2} S_2 \xrightarrow{\oplus} T_2 \cdots \]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}), \) where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

- **Mutual Information:**
  \[ I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \]

- **Structure:** \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P * \varphi \)
Mutual Information Estimation in Noisy DNNs

**Noisy DNN:**  \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{array}{c}
X \xrightarrow{f_1} S_1 \xrightarrow{Z_1} T_1 \xrightarrow{f_2} S_2 \xrightarrow{Z_2} T_2 \cdots
\end{array}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^{m} h(T_\ell | X = x_i) \)

- **Structure:** \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P * \varphi \)
Mutual Information Estimation in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}), \) where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

- ** Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \)

- **Structure:** \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \varphi \)
Mutual Information Estimation in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \quad \implies \quad T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[ \begin{array}{c}
X \xrightarrow{f_1} S_1 \xrightarrow{+} T_1 \xrightarrow{f_2} S_2 \xrightarrow{+} T_2 \quad \cdots
\end{array} \]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}), \) where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\[ \implies \text{Mutual Information:} \quad I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \]

- **Structure:** \( S_\ell \perp Z_\ell \quad \implies \quad T_\ell = S_\ell + Z_\ell \sim P \ast \varphi \)

- **Know** the distribution \( \varphi \) of \( Z_\ell \) (noise injected by design)
Mutual Information Estimation in Noisy DNNs

Noisy DNN: \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \ Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \)

- **Structure:** \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \varphi \)

- **Know** the distribution \( \varphi \) of \( Z_\ell \) (noise injected by design)
Assume: \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\[ I(X;T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell|X=x_i) \]

Structure: \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \varphi \)

Know the distribution \( \varphi \) of \( Z_\ell \) (noise injected by design)

\( P_{S_\ell} \) and \( P_{S_\ell|X} \) are extremely complicated to compute/evaluate
Mutual Information Estimation in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

| \begin{array}{c}
X \\
\rightarrow \\
f_1 \\
\rightarrow \\
S_1 \\
\rightarrow \\
T_1 \\
\rightarrow \\
f_2 \\
\rightarrow \\
S_2 \\
\rightarrow \\
T_2 \\
\rightarrow \\
\end{array} | \quad \begin{array}{c}
Z_1 \\
\downarrow \\
S_\ell \\
\downarrow \\
Z_\ell \\
\downarrow \\
T_\ell \\
\downarrow \\
\end{array}

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \)

- **Structure:** \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \varphi \)

- **Know** the distribution \( \varphi \) of \( Z_\ell \) (noise injected by design)

- **\( P_{S_\ell} \) and \( P_{S_\ell | X} \)** are **extremely complicated** to compute/evaluate

- **But both are easily sampled** via the DNN forward pass
Mutual Information Estimation in Noisy DNNs

Noisy DNN: \[ S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \]

\[ X \xrightarrow{f_1} S_1 \xrightarrow{+} T_1 \xrightarrow{f_2} S_2 \xrightarrow{+} T_2 \cdots \]

Differential Entropy Estimation under Gaussian Convolutions

Estimate \( h(P * \varphi) \) based on \( n \) i.i.d. samples from \( P \in \mathcal{F}_d \) (nonparametric class) and knowledge of \( \varphi \) (PDF of \( \mathcal{N}(0, \beta^2 I_d) \)).
Mutual Information Estimation in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{array}{cccccccc}
X & \rightarrow & f_1 & \rightarrow & S_1 & + & T_1 & \rightarrow & f_2 & \rightarrow & S_2 & + & T_2 & \cdots \\
& & & & Z_1 & & & & Z_2 & & & & & \\
\end{array}
\]

**Differential Entropy Estimation under Gaussian Convolutions**

Estimate \( h(P \ast \varphi) \) based on \( n \) i.i.d. samples from \( P \in \mathcal{F}_d \) (nonparametric class) and knowledge of \( \varphi \) (PDF of \( \mathcal{N}(0, \beta^2 I_d) \)).

**Estimation Results** [G.-Greenewald-Polyanskiy’18]:

Mutual Information Estimation in Noisy DNNs

**Noisy DNN:**  \( S_\ell \overset{\Delta}{=} f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[X \xrightarrow{f_1} S_1 \xrightarrow{f_2} S_2 \xrightarrow{f_2} \cdots \]

**Differential Entropy Estimation under Gaussian Convolutions**

Estimate \( h(\hat{P} * \phi) \) based on \( n \) i.i.d. samples from \( P \in \mathcal{F}_d \) (nonparametric class) and knowledge of \( \phi \) (PDF of \( \mathcal{N}(0, \beta^2 I_d) \)).

**Estimation Results** [G.-Greenewald-Polyanskiy’18]:

- Efficient & parallelizable estimator \( h(\hat{P}_n * \phi) \approx h(P * \phi) \)
Mutual Information Estimation in Noisy DNNs

**Noisy DNN:** \( S_\ell \overset{\Delta}{=} f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{array}{ccccccc}
X & \rightarrow & f_1 & \rightarrow & S_1 & \rightarrow & T_1 & \rightarrow & f_2 & \rightarrow & S_2 & \rightarrow & \ldots \\
& & & & \Updownarrow & & \Updownarrow & & \Updownarrow & & \Updownarrow & & \\
& & & & Z_1 & & Z_2 & & & & & \\
\end{array}
\]

**Differential Entropy Estimation under Gaussian Convolutions**

Estimate \( h(P * \varphi) \) based on \( n \) i.i.d. samples from \( P \in \mathcal{F}_d \) (nonparametric class) and knowledge of \( \varphi \) (PDF of \( \mathcal{N}(0, \beta^2 I_d) \)).

**Estimation Results** [G.-Greenewald-Polyanskiy’18]:

- Efficient & parallelizable estimator \( h(\hat{P}_n * \varphi) \approx h(P * \varphi) \)
- **Guarantees:** Estimation risk is \( O(1/\sqrt{n}) \) (all constants explicit)*

* Exponentially large in \( d \) though constants, which is provably necessary.
Mutual Information Estimation in Noisy DNNs

Noisy DNN: \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[ \begin{align*}
X & \rightarrow f_1 \rightarrow S_1 \rightarrow T_1 \rightarrow f_2 \rightarrow S_2 \rightarrow T_2 \rightarrow \cdots \\
Z_1 & \uparrow \\
Z_2 & \uparrow
\end{align*} \]

Differential Entropy Estimation under Gaussian Convolutions

Estimate \( h(P \ast \varphi) \) based on \( n \) i.i.d. samples from \( P \in \mathcal{F}_d \) (nonparametric class) and knowledge of \( \varphi \) (PDF of \( \mathcal{N}(0, \beta^2 I_d) \)).

Estimation Results [G.-Grenewald-Polyanskiy’18]:

- Efficient & parallelizable estimator \( h(\hat{P}_n \ast \varphi) \approx h(P \ast \varphi) \)
- **Guarantees:** Estimation risk is \( O\left(\frac{1}{\sqrt{n}}\right) \) (all constants explicit)*
- **Faster Rate:** kNN/KDE est. via ‘noisy’ samples attain \( O \left( n^{-\frac{a}{b+d}} \right) \)
Back to Noisy DNNs
Single Neuron Classification:

\[ I(X; T_\ell) \text{ Dynamics - Illustrative Minimal Example} \]

\[
\begin{align*}
X & \xrightarrow{\tanh(wX + b)} S_{w,b} \xrightarrow{+} T \\
Z & \sim \mathcal{N}(0, \beta^2)
\end{align*}
\]
Single Neuron Classification:

**Input:** \( X \sim \text{Unif}\{\pm 1, \pm 3\} \)

\[ \mathcal{X}_{y=-1} \triangleq \{-3, -1, 1\}, \quad \mathcal{X}_{y=1} \triangleq \{3\} \]
**Single Neuron Classification:**

**Input:** \(X \sim \text{Unif}\{\pm 1, \pm 3\}\)

\(\mathcal{X}_{y=-1} \triangleq \{-3, -1, 1\}, \mathcal{X}_{y=1} \triangleq \{3\}\)
**Single Neuron Classification:**

**Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$

$\mathcal{X}_{y=-1} \triangleq \{-3, -1, 1\}$, $\mathcal{X}_{y=1} \triangleq \{3\}$

$S_{1,0}$

Center & sharpen transition ( $\iff$ increase $w$ and keep $b = -2w$)
Single Neuron Classification:

**Input:** \( X \sim \text{Unif}\{\pm 1, \pm 3\} \)

\[ \mathcal{X}_{y=1} \triangleq \{-3, -1, 1\}, \quad \mathcal{X}_{y=1} \triangleq \{3\} \]
Single Neuron Classification:

**Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$

$\mathcal{X}_{y=-1} \triangleq \{-3, -1, 1\}$, $\mathcal{X}_{y=1} \triangleq \{3\}$

\[
X \xrightarrow{\text{tanh}(wX + b)} S_{w,b} \xrightarrow{\mathcal{Z} \sim \mathcal{N}(0, \beta^2)} T
\]
Single Neuron Classification:

- **Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$
  
  $\mathcal{X}_{y=-1} \triangleq \{-3, -1, 1\}$, $\mathcal{X}_{y=1} \triangleq \{3\}$

- **Mutual Information:**

$$\begin{align*}
&X \\ &\xrightarrow{\text{tanh}(wX + b)} S_{w,b} \\ &\text{Mutual Information:} \\ &Z \sim \mathcal{N}(0, \beta^2)
\end{align*}$$
Single Neuron Classification:

**Input:** \( X \sim \text{Unif}\{\pm 1, \pm 3\} \)

\[ \mathcal{X}_{y=-1} \triangleq \{-3, -1, 1\} , \, \mathcal{X}_{y=1} \triangleq \{3\} \]

**Mutual Information:** \( I(X; T) = I(S_{w,b}; S_{w,b} + Z) \)
Single Neuron Classification:

- **Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$
  
  \[ X_{y=-1} \triangleq \{-3, -1, 1\}, \quad X_{y=1} \triangleq \{3\} \]

- **Mutual Information:** $I(X; T) = I(S_{w,b}; S_{w,b} + Z)$
  
  $I(X; T)$ is \# bits (nats) transmittable over AWGN with symbols

  \[ S_{w,b} \triangleq \{\tanh(-3w+b), \tanh(-w+b), \tanh(w+b), \tanh(3w+b)\} \]
Single Neuron Classification:

- **Input:** \( X \sim \text{Unif}\{\pm 1, \pm 3\} \)
  \[ X_{y=-1} \triangleq \{-3, -1, 1\} \text{, } X_{y=1} \triangleq \{3\} \]

- **Mutual Information:** \( I(X; T) = I(S_{w,b}; S_{w,b} + Z) \)

\[ \Rightarrow \quad I(X; T) \text{ is } \# \text{ bits (nats) transmittable over AWGN with symbols} \]

\[ S_{w,b} \triangleq \{\tanh(-3w+b), \tanh(-w+b), \tanh(w+b), \tanh(3w+b)\} \longrightarrow \{\pm 1\} \]
**Single Neuron Classification:**

- **Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$
  
  $\mathcal{X}_{y=-1} \triangleq \{-3, -1, 1\}$, $\mathcal{X}_{y=1} \triangleq \{3\}$

- **Mutual Information:** $I(X; T) = I(S_{w,b}; S_{w,b} + Z)$

$\Rightarrow I(X; T)$ is $\# \text{ bits}$ (nats) transmittable over AWGN with symbols $S_{w,b} \triangleq \{\tanh(-3w+b), \tanh(-w+b), \tanh(w+b), \tanh(3w+b)\} \rightarrow \{\pm 1\}$
**Single Neuron Classification:**

- **Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$
  
  $\mathcal{X}_{y=-1} \triangleq \{-3, -1, 1\}$, $\mathcal{X}_{y=1} \triangleq \{3\}$

- **Mutual Information:** $I(X; T) = I(S_{w,b}; S_{w,b} + Z)$

$\implies I(X; T)$ is $\#$ bits (nats) transmittable over AWGN with symbols

$S_{w,b} \triangleq \{\tanh(-3w+b), \tanh(-w+b), \tanh(w+b), \tanh(3w+b)\} \longrightarrow \{\pm 1\}$
Clustering of Representations - Larger Networks

Noisy version of DNN from [Shwartz-Tishby’17]:
Clustering of Representations - Larger Networks

Noisy version of DNN from [Shwartz-Tishby’17]:

- **Binary Classification**: 12-bit input & 12–10–7–5–4–3–2 MLP arch.
Clustering of Representations - Larger Networks

Noisy version of DNN from [Shwartz-Tishby’17]:

- **Binary Classification**: 12-bit input & 12–10–7–5–4–3–2 MLP arch.
- **Noise std.**: Set to $\beta = 0.01$
Clustering of Representations - Larger Networks

Noisy version of DNN from [Shwartz-Tishby’17]:

- **Binary Classification**: 12-bit input & 12–10–7–5–4–3–2 MLP arch.
- **Noise std.**: Set to $\beta = 0.01$
Clustering of Representations - Larger Networks

Noisy version of DNN from [Shwartz-Tishby’17]:

- **Binary Classification**: 12-bit input & 12–10–7–5–4–3–2 MLP arch.
- **Noise std.**: Set to $\beta = 0.01$
Clustering of Representations - Larger Networks

Noisy version of DNN from [Shwartz-Tishby’17]:

- **Binary Classification**: 12-bit input & 12–10–7–5–4–3–2 MLP arch.
- **Noise std.**: Set to $\beta = 0.01$

(weight orthonormality regularization)
Noisy version of DNN from [Shwartz-Tishby’17]:

- **Binary Classification**: 12-bit input & 12–10–7–5–4–3–2 MLP arch.
- **Noise std.**: Set to $\beta = 0.01$
- Verified in multiple additional experiments
Clustering of Representations - Larger Networks

Noisy version of DNN from [Shwartz-Tishby’17]:

- **Binary Classification:** 12-bit input & 12–10–7–5–4–3–2 MLP arch.
- **Noise std.:** Set to $\beta = 0.01$
- Verified in multiple additional experiments

$\implies$ Compression of $I(X; T_\ell)$ driven by clustering of representations
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering

- Alternative measures for clustering (det. and noisy DNNs):
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering
- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\Rightarrow$ Doesn’t measure clustering

- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
  - Binned entropy $H(\text{Bin}(T_\ell))$
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering

- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
  - Binned entropy $H(\text{Bin}(T_\ell))$
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering

- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
  - Binned entropy $H(\text{Bin}(T_\ell))$
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering
- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
  - Binned entropy $H(\text{Bin}(T_\ell)) \uparrow$
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering

- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
  - Binned entropy $H(\text{Bin}(T_\ell)) \downarrow$
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering

Alternative measures for clustering (det. and noisy DNNs):

- Scatter plots (up to 3D layers)
- Binned entropy $H(\text{Bin}(T_\ell))$

Noisy DNNs: $I(X; T_\ell)$ and $H(\text{Bin}(T_\ell))$ highly correlated!*

* When bin size chosen $\propto$ noise std.
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering

- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
  - Binned entropy $H(\text{Bin}(T_\ell))$

- **Noisy DNNs**: $I(X; T_\ell)$ and $H(\text{Bin}(T_\ell))$ highly correlated!*

* When bin size chosen $\propto$ noise std.
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering
- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
  - Binned entropy $H(\text{Bin}(T_\ell))$
- **Noisy DNNs:** $I(X; T_\ell)$ and $H(\text{Bin}(T_\ell))$ highly correlated!*
- **Det. DNNs:** $H(\text{Bin}(T_\ell)) = I(X; \text{Bin}(T_\ell))$ compresses
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering

- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
  - Binned entropy $H(\text{Bin}(T_\ell))$

- **Noisy DNNs:** $I(X; T_\ell)$ and $H(\text{Bin}(T_\ell))$ highly correlated!* 

- **Det. DNNs:** $H(\text{Bin}(T_\ell)) = I(X; \text{Bin}(T_\ell))$ compresses

  ✗ Incapable of accurately estimating MI values
Circling back to Deterministic DNNs

- \( I(X; T_\ell) \) is constant \( \implies \) Doesn’t measure clustering

- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
  - Binned entropy \( H(\text{Bin}(T_\ell)) \)

- **Noisy DNNs:** \( I(X; T_\ell) \) and \( H(\text{Bin}(T_\ell)) \) highly correlated!

- **Det. DNNs:** \( H(\text{Bin}(T_\ell)) = I(X; \text{Bin}(T_\ell)) \) compresses
  - \( \times \) Incapable of accurately estimating MI values
  - \( \checkmark \) Does track clustering!
Circling back to Deterministic DNNs

- $I(X; T_\ell)$ is constant $\implies$ Doesn’t measure clustering

- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)
  - Binned entropy $H(\text{Bin}(T_\ell))$

- **Noisy DNNs:** $I(X; T_\ell)$ and $H(\text{Bin}(T_\ell))$ highly correlated!*  

- **Det. DNNs:** $H(\text{Bin}(T_\ell)) = I(X; \text{Bin}(T_\ell))$ compresses

  ✗ Incapable of accurately estimating MI values

  ✓ Does track clustering!

$\implies$ Past works were not showing MI but clustering (via binned-MI)!
Summary

- Reexamined Information Bottleneck Compression:
Summary

- **Reexamined Information Bottleneck Compression:**
  - $I(X;T)$ fluctuations in det. DNNs are theoretically impossible
Reexamined Information Bottleneck Compression:

- $I(X; T)$ fluctuations in det. DNNs are theoretically impossible
- Yet, past works presented (binned) $I(X; T)$ dynamics during training
Summary

- **Reexamined Information Bottleneck Compression:**
  - $I(X; T)$ fluctuations in det. DNNs are theoretically impossible
  - Yet, past works presented (binned) $I(X; T)$ dynamics during training

- **Noisy DNN Framework:** Studying IT quantities over DNNs
Summary

- **Reexamined Information Bottleneck Compression:**
  - $I(X;T)$ fluctuations in det. DNNs are theoretically impossible
  - Yet, past works presented (binned) $I(X;T)$ dynamics during training

- **Noisy DNN Framework:** Studying IT quantities over DNNs
  - Developed estimator for accurate MI estimation over this framework
Reexamined Information Bottleneck Compression:

- $I(X;T)$ fluctuations in det. DNNs are theoretically impossible
- Yet, past works presented (binned) $I(X;T)$ dynamics during training

Noisy DNN Framework: Studying IT quantities over DNNs

- Developed estimator for accurate MI estimation over this framework
- Clustering of the learned representations is the source of compression
Summary

- **Reexamined Information Bottleneck Compression:**
  - $I(X;T)$ fluctuations in det. DNNs are theoretically impossible
  - Yet, past works presented (binned) $I(X;T)$ dynamics during training

- **Noisy DNN Framework:** Studying IT quantities over DNNs
  - Developed estimator for accurate MI estimation over this framework
  - Clustering of the learned representations is the source of compression

- **Clarify Past Observations of Compression:** in fact show clustering
Summary

- **Reexamined Information Bottleneck Compression:**
  - $I(X;T)$ fluctuations in det. DNNs are theoretically impossible
  - Yet, past works presented (binned) $I(X;T)$ dynamics during training

- **Noisy DNN Framework:** Studying IT quantities over DNNs
  - Developed estimator for accurate MI estimation over this framework
  - Clustering of the learned representations is the source of compression

- **Clarify Past Observations of Compression:** in fact show clustering

  ➞ **Clustering** is the common phenomenon of interest!
Future/Ongoing Clustering Inspired Research

- Track Clustering in High-Dimensions:
Track Clustering in High-Dimensions:

- Lower-dimensional embedding
Future/Ongoing Clustering Inspired Research

- **Track Clustering in High-Dimensions:**
  - Lower-dimensional embedding
  - Summarizing statistics
Track Clustering in High-Dimensions:

- Lower-dimensional embedding
- Summarizing statistics
- Graph clusterability [Czumaj-Peng-Sohler’15]
Future/Ongoing Clustering Inspired Research

- **Track Clustering in High-Dimensions:**
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability [Czumaj-Peng-Sohler’15]

- **Role of Compression/Clustering:**
Future/Ongoing Clustering Inspired Research

- **Track Clustering in High-Dimensions:**
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability [Czumaj-Peng-Sohler’15]

- **Role of Compression/Clustering:**
  - Is it necessary? Desirable?
Future/Ongoing Clustering Inspired Research

- **Track Clustering in High-Dimensions:**
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability [Czumaj-Peng-Sohler’15]

- **Role of Compression/Clustering:**
  - Is it necessary? Desirable?
Future/Ongoing Clustering Inspired Research

- **Track Clustering in High-Dimensions:**
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability [Czumaj-Peng-Sohler’15]

- **Role of Compression/Clustering:**
  - Is it necessary? Desirable?
  - Design tool for DNN architectures
Future/Ongoing Clustering Inspired Research

- **Track Clustering in High-Dimensions:**
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability [Czumaj-Peng-Sohler’15]

- **Role of Compression/Clustering:**
  - Is it necessary? Desirable?
  - Design tool for DNN architectures

- **Algorithmic Perspective:**
Future/Ongoing Clustering Inspired Research

- **Track Clustering in High-Dimensions:**
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability [Czumaj-Peng-Sohler’15]

- **Role of Compression/Clustering:**
  - Is it necessary? Desirable?
  - Design tool for DNN architectures

- **Algorithmic Perspective:**
  - Better understanding of internal representation evolution & final state
Future/Ongoing Clustering Inspired Research

- **Track Clustering in High-Dimensions:**
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability [Czumaj-Peng-Sohler’15]

- **Role of Compression/Clustering:**
  - Is it necessary? Desirable?
  - Design tool for DNN architectures

- **Algorithmic Perspective:**
  - Better understanding of internal representation evolution & final state
  - Enhanced DNN training alg. (regularize intermediate layers wrt $I(Y;T)$)
Future/Ongoing Clustering Inspired Research

- **Track Clustering in High-Dimensions:**
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability [Czumaj-Peng-Sohler’15]

- **Role of Compression/Clustering:**
  - Is it necessary? Desirable?
  - Design tool for DNN architectures

- **Algorithmic Perspective:**
  - Better understanding of internal representation evolution & final state
  - Enhanced DNN training alg. (regularize intermediate layers wrt $I(Y; T)$)


**Motivation:** Demand for high-capacity data storage devices
**Motivation:** Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
Information Storage in Interacting Particle Systems

**Motivation:** Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
- **Challenges** in stabilizing stored data (interparticle interactions)
Motivation: Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
- **Challenges** in stabilizing stored data (interparticle interactions)

Approach:
Motivation: Demand for high-capacity data storage devices

- Emerging technologies drastically shrink magnetic region per bit
- Challenges in stabilizing stored data (interparticle interactions)

Approach: View storage medium as Interacting Particle System:
**Motivation:** Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
- **Challenges** in stabilizing stored data (interparticle interactions)

**Approach:** View storage medium as **Interacting Particle System:**

- Hard-drive topology
**Motivation:** Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
- **Challenges** in stabilizing stored data (interparticle interactions)

**Approach:** View storage medium as **Interacting Particle System:**

- Hard-drive topology $\rightarrow$ Graph
Motivation: Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
- **Challenges** in stabilizing stored data (interparticle interactions)

Approach: View storage medium as **Interacting Particle System**:

- Hard-drive topology \(\Rightarrow\) Graph
- Interactions
Information Storage in Interacting Particle Systems

**Motivation:** Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
- **Challenges** in stabilizing stored data (interparticle interactions)

**Approach:** View storage medium as **Interacting Particle System:**

- Hard-drive topology $\rightarrow$ Graph
- Interactions $\rightarrow$ Stochastic dynamics
**Motivation:** Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
- **Challenges** in stabilizing stored data (interparticle interactions)

**Approach:** View storage medium as **Interacting Particle System:**

- Hard-drive topology $\Rightarrow$ Graph
- Interactions $\Rightarrow$ Stochastic dynamics
**Motivation:** Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
- **Challenges** in stabilizing stored data (interparticle interactions)

**Approach:** View storage medium as **Interacting Particle System:**

- Hard-drive topology $\rightarrow$ Graph
- Interactions $\rightarrow$ Stochastic dynamics

**Setup:** **Design** initial configuration and **recover** data after $t$ time
Motivation: Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
- **Challenges** in stabilizing stored data (interparticle interactions)

Approach: View storage medium as **Interacting Particle System:**

- Hard-drive topology $\rightarrow$ Graph
- Interactions $\rightarrow$ Stochastic dynamics

Setup: **Design** initial configuration and **recover** data after $t$ time
**Motivation:** Demand for high-capacity data storage devices

- **Emerging technologies** drastically shrink magnetic region per bit
- **Challenges** in stabilizing stored data (interparticle interactions)

**Approach:** View storage medium as **Interacting Particle System:**

- Hard-drive topology $\rightarrow$ Graph
- Interactions $\rightarrow$ Stochastic dynamics

**Setup:** Design initial configuration and **recover** data after $t$ time

[**G.-Bresler-Polyanskiy’18**] Performance benchmarks & hard-drive designs
**Motivation:** Modern networks are large, decentralized and ad hoc
Motivation: Modern networks are large, decentralized and ad hoc
Motivation: Modern networks are large, decentralized and ad hoc

- Most devices are simple & low-complexity
Motivation: Modern networks are large, decentralized and ad hoc

- Most devices are simple & low-complexity
- Nodes constantly join and leave the network
Cooperative Communication Networks

Motivation: Modern networks are large, decentralized and ad hoc

- Most devices are simple & low-complexity
- Nodes constantly join and leave the network
- Cooperation for long-distance communication (mitigate interference)
Cooperative Communication Networks

Motivation: Modern networks are large, decentralized and ad hoc

- Most devices are simple & low-complexity
- Nodes constantly join and leave the network
- Cooperation for long-distance communication (mitigate interference)

Modus Operandi: Primitive building blocks $\implies$ Entire network
Cooperative Communication Networks

**Motivation:** Modern networks are large, decentralized and ad hoc

- Most devices are **simple** & **low-complexity**
- Nodes constantly **join** and **leave** the network
- **Cooperation** for long-distance communication (mitigate interference)

**Modus Operandi:** Primitive building blocks $\implies$ Entire network

- P2P, feedback, relay, uplink, downlink channels
Cooperative Communication Networks

**Motivation:** Modern networks are large, decentralized and ad hoc

- Most devices are **simple** & **low-complexity**
- Nodes constantly **join** and **leave** the network
- **Cooperation** for long-distance communication (mitigate interference)

**Modus Operandi:** Primitive building blocks $\Rightarrow$ Entire network

- P2P, feedback, relay, uplink, downlink channels
- Public vs. private vs. confidential transmissions
**Motivation:** Modern networks are large, decentralized and ad hoc

- Most devices are **simple** & **low-complexity**
- Nodes constantly **join** and **leave** the network
- **Cooperation** for long-distance communication (mitigate interference)

**Modus Operandi:** Primitive building blocks ➞ Entire network

- P2P, feedback, relay, uplink, downlink channels
- Public vs. private vs. confidential transmissions

**Next Steps:** Model & study large-scale networks
Cooperative Communication Networks

**Motivation:** Modern networks are large, decentralized and ad hoc

- Most devices are **simple & low-complexity**
- Nodes constantly **join** and **leave** the network
- **Cooperation** for long-distance communication (mitigate interference)

**Modus Operandi:** Primitive building blocks $\implies$ Entire network

- P2P, feedback, relay, uplink, downlink channels
- Public vs. private vs. confidential transmissions

**Next Steps:** **Model & study** large-scale networks

- **Stochastic** connectivity patterns (random / small-world / free-scale)
**Motivation:** Modern networks are large, decentralized and ad hoc

- Most devices are **simple** & **low-complexity**
- Nodes constantly **join** and **leave** the network
- **Cooperation** for long-distance communication (mitigate interference)

**Modus Operandi:** Primitive building blocks $\implies$ Entire network

- P2P, feedback, relay, uplink, downlink channels
- Public vs. private vs. confidential transmissions

**Next Steps:** Model & study large-scale networks

- **Stochastic** connectivity patterns (random / small-world / free-scale)
- Local interactions **distilled** from primitive results
Cooperative Communication Networks

**Motivation:** Modern networks are large, decentralized and ad hoc

- Most devices are **simple** & **low-complexity**
- Nodes constantly **join** and **leave** the network
- **Cooperation** for long-distance communication (mitigate interference)

**Modus Operandi:** Primitive building blocks $\implies$ Entire network

- P2P, feedback, relay, uplink, downlink channels
- Public vs. private vs. confidential transmissions

**Next Steps:** Model & study large-scale networks

- **Stochastic** connectivity patterns (random / small-world / free-scale)
- Local interactions **distilled** from primitive results

**Q:** Reliable (& secure) information passing protocols? Fundamental limits?
Mutual Information in Noisy DNNs

Noisy DNN:

\[ X \rightarrow f_1 \rightarrow S_1 \rightarrow T_1 \rightarrow f_2 \rightarrow S_2 \rightarrow T_2 \rightarrow \ldots \]

\[ Z_1 \quad Z_2 \]
Mutual Information in Noisy DNNs

Noisy DNN:

$X \xrightarrow{f_1} S_1 \xrightarrow{T_1} f_2 \xrightarrow{S_2} T_2 \cdots$

$Z_1 \xrightarrow{}$
Mutual Information in Noisy DNNs

Noisy DNN:

\[ X \overset{f_1}{\rightarrow} S_1 \overset{T_1}{\rightarrow} f_2 \overset{S_2}{\rightarrow} T_2 \ldots \]
Mutual Information in Noisy DNNs

Noisy DNN: $S_{\ell} \triangleq f_{\ell}(T_{\ell-1})$

![Diagram of Noisy DNN](attachment:image.png)
Mutual Information in Noisy DNNs

Noisy DNN: $S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I)$
Assume: \( X \sim \text{Unif}(\mathcal{X}), \) where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset
**Mutual Information in Noisy DNNs**

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

$X$ \[ f_1 \] $S_1$ \[ T_1 \] \[ f_2 \] $S_2$ \[ T_2 \] \[ Z_1 \] \[ Z_2 \] \[ \cdots \]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\[ \implies \text{Mutual Information:} \quad I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^{m} h(T_\ell | X = x_i) \]
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

**Assume:** \( X \sim \text{Unif}(\mathcal{X}), \text{ where } \mathcal{X} \triangleq \{x_i\}_{i=1}^m \text{ is empirical dataset} \)

\implies **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \)
Mutual Information in Noisy DNNs

**Noisy DNN:** $S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \ Z_\ell \sim \mathcal{N}(0, \beta^2 I)$

![Diagram of Noisy DNN]

- **Assume:** $X \sim \text{Unif}(\mathcal{X})$, where $\mathcal{X} \triangleq \{x_i\}_{i=1}^m$ is empirical dataset
  
  $\implies$ **Mutual Information:** $I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i)$

- $P_{T_\ell}$ and $P_{T_\ell|X}$ are extremely complicated to compute/evaluate
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \quad \implies \quad T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
X \xrightarrow{f_1} S_1 \xrightarrow{Z_1} T_1 \xrightarrow{f_2} S_2 \xrightarrow{Z_2} T_2 \ldots
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}), \) where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \)

- \( P_{T_\ell} \) and \( P_{T_\ell | X} \) are **extremely** complicated to compute/evaluate

- But both are **easily** sampled via the DNN forward pass
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{array}{c}
X \rightarrow f_1 \rightarrow S_1 \rightarrow T_1 \rightarrow f_2 \rightarrow S_2 \rightarrow T_2 \ldots \\
\uparrow Z_1 \quad \uparrow Z_2
\end{array}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\[ I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \]

- \( P_{T_\ell} \) and \( P_{T_\ell | X} \) are extremely complicated to compute/evaluate
- But both are easily sampled via the DNN forward pass
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \ Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{array}{c}
X \\
\downarrow f_1 \\
S_1 \\
\downarrow T_1 \\
\downarrow f_2 \\
S_2 \\
\downarrow T_2 \\
\end{array}
\]

\( Z_1 \)

\( Z_2 \)

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\( \implies \text{Mutual Information:} \ I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell|X = x_i) \)

- \( P_{T_\ell} \) and \( P_{T_\ell|X} \) are extremely complicated to compute/evaluate

- But both are easily sampled via the DNN forward pass

  - **Sampling** \( P_{T_\ell} \): Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{array}{cccccccccc}
X & \rightarrow & f_1 & \rightarrow & S_1 & \rightarrow & T_1 & \rightarrow & f_2 & \rightarrow & S_2 & \rightarrow & T_2 & \cdots \\
& & & & & & & & & & & \\
& & & & & Z_1 & & & & & & Z_2 \\
\end{array}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\[
\implies \text{Mutual Information:} \quad I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i)
\]

- **\( P_{T_\ell} \) and \( P_{T_\ell|X} \) are extremely complicated to compute/evaluate**

- **But both are easily** sampled via the DNN forward pass

  - **Sampling \( P_{T_\ell} \):** Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{align*}
X &\rightarrow f_1 & \rightarrow S_1 &\rightarrow T_1 &\rightarrow f_2 &\rightarrow S_2 &\rightarrow T_2 &\cdots \\
\uparrow &\quad & \uparrow &\quad & \uparrow &\quad & \uparrow &\quad \\
x_1 & & & & & & t_{\ell,1}
\end{align*}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}), \) where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\[ I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \]

- \( P_{T_\ell} \) and \( P_{T_\ell | X} \) are **extremely** complicated to compute/evaluate

- But both are **easily** sampled via the DNN forward pass

  - **Sampling** \( P_{T_\ell} \): Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \ Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{align*}
X & \xrightarrow{f_1} S_1 & \xrightarrow{+} T_1 & \xrightarrow{f_2} S_2 & \xrightarrow{+} T_2 & \cdots \\
&S_1 & & & & \\
&S_2 & & & & \\
&S_\ell & & & & \\
&Z_1 & & & & \\
&Z_2 & & & & \\
&x_1 & & & & t_{\ell,1} \\
&x_2 & & & & \\
\end{align*}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \)

- \( P_{T_\ell} \) and \( P_{T_\ell | X} \) are extremely complicated to compute/evaluate

- But both are easily sampled via the DNN forward pass

  - **Sampling** \( P_{T_\ell} \): Feed randomly chosen \( x_i \)’s & read \( T_\ell \) values
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \overset{\Delta}{=} f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{align*}
X &\xrightarrow{f_1} S_1 & &\xrightarrow{f_2} S_2 & &\cdots \\
x_1 &\quad &Z_1 &\quad &Z_2 &\quad &t_{\ell,1} &\quad &t_{\ell,2}
\end{align*}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \overset{\Delta}{=} \{x_i\}_{i=1}^m \) is empirical dataset

\[
\implies \text{Mutual Information: } I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell|X = x_i)
\]

- \( P_{T_\ell} \) and \( P_{T_\ell|X} \) are extremely complicated to compute/evaluate
- But both are easily sampled via the DNN forward pass
  - **Sampling** \( P_{T_\ell} \): Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{align*}
X & \xrightarrow{f_1} S_1 & \xrightarrow{+} & T_1 & \xrightarrow{f_2} S_2 & \xrightarrow{+} & T_2 & \cdots \\
x_1 & & & \downarrow Z_1 & & & \downarrow Z_2 \quad & \quad t_{\ell,1} & \quad t_{\ell,2} \\
x_2 & & & \vdots & & & \vdots \\
\vdots & & & & & & & \vdots & \vdots
\end{align*}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}), \) where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset
- \( \implies \) **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \)
- \( P_{T_\ell} \) and \( P_{T_\ell|X} \) are extremely complicated to compute/evaluate
- But both are easily sampled via the DNN forward pass
- **Sampling** \( P_{T_\ell}: \) Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 \mathbf{I}) \)

\[
\begin{array}{c}
X \xrightarrow{f_1} S_1 \xrightarrow{+} T_1 \xrightarrow{f_2} S_2 \xrightarrow{+} T_2 \ldots
\end{array}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\[
\implies \text{Mutual Information:} \quad I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i)
\]

- \( P_{T_\ell} \) and \( P_{T_\ell | X} \) are extremely complicated to compute/evaluate

- But both are easily sampled via the DNN forward pass

  - **Sampling** \( P_{T_\ell} \): Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values

  - **Sampling** \( P_{T_\ell | X = x_i} \): Feed \( x_i \) multiples times & read \( T_\ell \) values
Mutual Information in Noisy DNNs

Noisy DNN: \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{align*}
X & \rightarrow f_1 \rightarrow S_1 \rightarrow T_1 \rightarrow f_2 \rightarrow S_2 \rightarrow T_2 \cdots \\
\mathbf{x}_i & \\
Z_1 & \\
\mathbf{Z}_2 & \\
\end{align*}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}), \) where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\[ I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \]

- \( P_{T_\ell} \) and \( P_{T_\ell | X} \) are extremely complicated to compute/evaluate
- But both are easily sampled via the DNN forward pass
  - **Sampling** \( P_{T_\ell} \): Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values
  - **Sampling** \( P_{T_\ell | X=x_i} \): Feed \( x_i \) multiples times & read \( T_\ell \) values
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \ Z_\ell \sim \mathcal{N}(0, \beta^2I) \)

\[
\begin{array}{c}
X \\ x_i
\end{array} \xrightarrow{f_1} S_1 \xrightarrow{+} T_1 \xrightarrow{f_2} S_2 \xrightarrow{+} T_2 \cdots
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

- \( \implies \) **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^{m} h(T_\ell | X = x_i) \)

- \( P_{T_\ell} \) and \( P_{T_\ell | X} \) are **extremely** complicated to compute/evaluate

- But both are **easily** sampled via the DNN forward pass
  - **Sampling** \( P_{T_\ell} \): Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values
  - **Sampling** \( P_{T_\ell | X=x_i} \): Feed \( x_i \) multiples times & read \( T_\ell \) values
Mutual Information in Noisy DNNs

Noisy DNN: \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{array}{cccccc}
X & \xrightarrow{f_1} & S_1 & + & T_1 & \xrightarrow{f_2} & S_2 & + & T_2 & \cdots \\
& & Z_1 & & & & Z_2 & & & \\
\end{array}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell | X = x_i) \)

- \( P_{T_\ell} \) and \( P_{T_\ell | X} \) are **extremely** complicated to compute/evaluate

- **But both are easily** sampled via the DNN forward pass
  - **Sampling** \( P_{T_\ell} \): Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values
  - **Sampling** \( P_{T_\ell | X=x_i} \): Feed \( x_i \) multiples times & read \( T_\ell \) values
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \overset{\Delta}{=} f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
X \xrightarrow{f_1} S_1 \xrightarrow{+} T_1 \xrightarrow{f_2} S_2 \xrightarrow{+} T_2 \cdots
\]

\( x_i \)

\( Z_1 \)

\( t^{(i)}_{\ell,1} \)

\( Z_2 \)

\( t^{(i)}_{\ell,2} \)

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}), \text{ where } \mathcal{X} \overset{\Delta}{=} \{x_i\}_{i=1}^m \) is empirical dataset

- \( \implies \text{Mutual Information:} \quad I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell|X = x_i) \)

- \( P_{T_\ell} \) and \( P_{T_\ell|X} \) are **extremely** complicated to compute/evaluate

- But both are **easily** sampled via the DNN forward pass
  - **Sampling** \( P_{T_\ell} \): Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values
  - **Sampling** \( P_{T_\ell|X=x_i} \): Feed \( x_i \) multiples times & read \( T_\ell \) values
Mutual Information in Noisy DNNs

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{array}{cccccc}
X & \rightarrow & f_1 & \rightarrow & S_1 & \rightarrow & T_1 & \rightarrow & f_2 & \rightarrow & S_2 & \rightarrow & T_2 & \cdots \\
\vdots & & & & & & & & & & & & & \\
\end{array}
\]

- **Assume:** \( X \sim \text{Unif}(\mathcal{X}) \), where \( \mathcal{X} \triangleq \{x_i\}_{i=1}^m \) is empirical dataset

\[ I(X; T_\ell) = h(T_\ell) - \frac{1}{m} \sum_{i=1}^m h(T_\ell|X = x_i) \]

- \( P_{T_\ell} \) and \( P_{T_\ell|X} \) are extremely complicated to compute/evaluate

But both are **easily** sampled via the DNN forward pass

- **Sampling** \( P_{T_\ell} \): Feed randomly chosen \( x_i \)'s & read \( T_\ell \) values
- **Sampling** \( P_{T_\ell|X=x_i} \): Feed \( x_i \) multiples times & read \( T_\ell \) values
Estimate $I(X; T_\ell)$ from samples via **general-purpose** $h(P)$ est.:
General-Purpose Differential Entropy Estimators

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{align*}
X & \xrightarrow{f_1} S_1 & 
T_1 & \xrightarrow{f_2} S_2 & \cdots \\
& \quad \downarrow Z_1 & & \quad \downarrow Z_2
\end{align*}
\]

\[ \implies \text{Estimate } I(X; T_\ell) \text{ from samples via general-purpose } h(P) \text{ est.}: \]

- Most results assume lower bounded density
General-Purpose Differential Entropy Estimators

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[ X \xrightarrow{f_1} S_1 \xrightarrow{} T_1 \xrightarrow{f_2} S_2 \xrightarrow{} T_2 \cdots \]

\( \implies \) Estimate \( I(X; T_\ell) \) from samples via **general-purpose** \( h(P) \) est.:

- Most results assume lower bounded density \( \implies \text{Inapplicable} \)
**General-Purpose Differential Entropy Estimators**

**Noisy DNN:** $S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I)$

\[ X \xrightarrow{f_1} S_1 \xrightarrow{\oplus} T_1 \xrightarrow{f_2} S_2 \xrightarrow{\oplus} T_2 \cdots \]

$Z_1 \xrightarrow{\oplus} S_1 \xrightarrow{\oplus} T_1 \xrightarrow{\oplus} S_2 \xrightarrow{\oplus} T_2 \cdots$

$\implies \text{Estimate } I(X; T_\ell) \text{ from samples via general-purpose } h(P) \text{ est.}:$

- Most results assume lower bounded density $\implies \text{Inapplicable}$
- 2 Works Drop Assumption:
General-Purpose Differential Entropy Estimators

Noisy DNN: \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{align*}
X \xrightarrow{f_1} S_1 & \quad \bigoplus \quad T_1 \xrightarrow{f_2} S_2 & \quad \bigoplus \quad T_2 & \quad \cdots \\
& \quad Z_1 & \quad Z_2
\end{align*}
\]

\( \implies \) Estimate \( I(X; T_\ell) \) from samples via general-purpose \( h(P) \) est.:

- Most results assume lower bounded density \( \implies \) **Inapplicable**
- **2 Works Drop Assumption:**
  - KDE + Best poly. approximation [Han-Jiao-Weissman-Wu’17]
**General-Purpose Differential Entropy Estimators**

**Noisy DNN:**
\[ S_\ell \overset{\Delta}{=} f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \]

\[ X \xrightarrow{f_1} S_1 \xrightarrow{\oplus} T_1 \xrightarrow{f_2} S_2 \xrightarrow{\oplus} T_2 \cdots \]

\[ Z_1 \quad \text{and} \quad Z_2 \]

\[ \implies \text{Estimate } I(X; T_\ell) \text{ from samples via general-purpose } h(P) \text{ est.:} \]

- Most results assume lower bounded density \( \implies \text{Inapplicable} \)
- **2 Works Drop Assumption:**
  1. KDE + Best poly. approximation [Han-Jiao-Weissman-Wu’17]
  2. Kozachenko-Leonenko (kNN) estimator [Jiao-Gao-Han’17]
General-Purpose Differential Entropy Estimators

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
X \xrightarrow{f_1} S_1 \xrightarrow{} T_1 \xrightarrow{f_2} S_2 \xrightarrow{} T_2 \cdots
\]

\( Z_1 \)

\( Z_2 \)

\( \implies \) Estimate \( I(X; T_\ell) \) from samples via **general-purpose** \( h(P) \) est.:

- Most results assume lower bounded density \( \implies \) **Inapplicable**

- **2 Works Drop Assumption:**
  - 1. KDE + Best poly. approximation [Han-Jiao-Weissman-Wu’17]
  - 2. Kozachenko-Leonenko (kNN) estimator [Jiao-Gao-Han’17]

- **Assume:** \( \text{supp} = [0, 1]^d \) & Periodic BC & \( s \in (0, 2] \)
General-Purpose Differential Entropy Estimators

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{array}{cccccccccccc}
X & \rightarrow & f_1 & \rightarrow & S_1 & \rightarrow & \oplus & \rightarrow & f_2 & \rightarrow & S_2 & \rightarrow & \oplus & \rightarrow & T_2 & \cdots \\
& & \uparrow & & Z_1 & & & & \uparrow & & Z_2 & & & & \\
\end{array}
\]

\( \implies \) Estimate \( I(X; T_\ell) \) from samples via **general-purpose** \( h(P) \) est.:

- Most results assume lower bounded density \( \implies \) Inapplicable
- **2 Works Drop Assumption:**
  1. KDE + Best poly. approximation [Han-Jiao-Weissman-Wu’17]
  2. Kozachenko-Leonenko (kNN) estimator [Jiao-Gao-Han’17]
- **Assume:** \( \text{supp} = [0, 1]^d \) \& Periodic BC \& \( s \in (0, 2) \) \( \implies \) Inapplicable*

* Except sub-Gaussian result from [Han-Jiao-Weissman-Wu’17]
Noisy DNN: $S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2I)$

$X \xrightarrow{f_1} S_1 \xrightarrow{} T_1 \xrightarrow{f_2} S_2 \xrightarrow{} T_2 \cdots$

$\implies$ Estimate $I(X; T_\ell)$ from samples via **general-purpose $h(P)$ est.**:

- Most results assume lower bounded density $\implies$ **Inapplicable**

- **2 Works Drop Assumption:**
  1. KDE + Best poly. approximation [Han-Jiao-Weissman-Wu’17]
  2. Kozachenko-Leonenko (kNN) estimator [Jiao-Gao-Han’17]

- **Assume:** supp $= [0, 1]^d$ \& Periodic BC \& $s \in (0,2) \implies$ **Inapplicable***

- **Rate:** Risk $\leq O\left(n^{-\frac{\alpha s}{\beta s + d}}\right)$, \quad w/ $\alpha, \beta \in \mathbb{N}$, $s$ smoothness, $d$ dimension
Exploit Structure - Ad Hoc Estimation

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

![Diagram showing the flow of data through layers with noise]

**Exploit structure:** We know \( T_\ell = S_\ell + Z_\ell \sim P \ast \varphi \) and:
Exploit Structure - Ad Hoc Estimation

Noisy DNN: $S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I)$

Exploit structure: We know $T_\ell = S_\ell + Z_\ell \sim P * \varphi$ and:

- **Genie1:** Sample $P = P_{S_\ell}$ and $P = P_{S_\ell|X=x_i}$ (sample $T_{\ell-1}$ & apply $f_\ell$)
Exploit Structure - Ad Hoc Estimation

**Noisy DNN:** \( S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I) \)

\[
\begin{align*}
  X & \xrightarrow{f_1} S_1 & \xrightarrow{+} T_1 & \xrightarrow{f_2} S_2 & \xrightarrow{+} T_2 & \cdots
\end{align*}
\]

\[ Z_1 \quad Z_2 \]

**Exploit structure:** We know \( T_\ell = S_\ell + Z_\ell \sim P \ast \varphi \) and:

- **Genie1:** Sample \( P = P_{S_\ell} \) and \( P = P_{S_\ell \mid X=x_i} \) (sample \( T_{\ell-1} \) & apply \( f_\ell \))

- **Genie2:** Know the distribution \( \varphi \) of \( Z_\ell \) (noise injected by design)
Exploit Structure - Ad Hoc Estimation

\textbf{Noisy DNN:} \quad S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 I)

\textbf{Exploit structure:} We know \( T_\ell = S_\ell + Z_\ell \sim P \ast \varphi \) and:

- **Genie1:** Sample \( P = P_{S_\ell} \) and \( P = P_{S_\ell|X=x_i} \) (sample \( T_{\ell-1} \) & apply \( f_\ell \))
- **Genie2:** Know the distribution \( \varphi \) of \( Z_\ell \) (noise injected by design)

\textbf{Differential Entropy Estimation under Gaussian Convolutions}

Estimate \( h(P \ast \varphi) \) based on \( n \) i.i.d. samples from \( P \in \mathcal{F}_d \) (nonparametric class) and knowledge of \( \varphi \) (PDF of \( \mathcal{N}(0, \beta^2 I_d) \)).
Exploit Structure - Ad Hoc Estimation

**Noisy DNN:** \[ S_\ell \triangleq f_\ell(T_{\ell-1}) \implies T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim N(0, \beta^2 I) \]

**Exploit structure:** We know \( T_\ell = S_\ell + Z_\ell \sim P * \varphi \) and:

- **Genie1:** Sample \( P = P_{S_\ell} \) and \( P = P_{S_\ell|X=x_i} \) (sample \( T_{\ell-1} \) & apply \( f_\ell \))
- **Genie2:** Know the distribution \( \varphi \) of \( Z_\ell \) (noise injected by design)

---

**Differential Entropy Estimation under Gaussian Convolutions**

Estimate \( h(P * \varphi) \) based on \( n \) i.i.d. samples from \( P \in \mathcal{F}_d \) (nonparametric class) and knowledge of \( \varphi \) (PDF of \( N(0, \beta^2 I_d) \)).

**Nonparametric Class:** Depends on DNN architecture (nonlinearities)
Abs. Error Minimax Risk: $S^n$ are $n$ i.i.d. samples from $P$, define

$$\mathcal{R}_d^*(n, \beta) \triangleq \inf_{\hat{h}} \sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}(S^n, \beta) \right|$$
Abs. Error Minimax Risk: \( S^n \) are \( n \) i.i.d. samples from \( P \), define

\[
\mathcal{R}_d^*(n, \beta) \triangleq \inf_{\hat{h}} \sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P * \varphi) - \hat{h}(S^n, \beta) \right|
\]

Curse of Dimensionality: Sample complexity exponential in \( d \)
Abs. Error Minimax Risk: $S^n$ are $n$ i.i.d. samples from $P$, define

$$R_d^*(n, \beta) \triangleq \inf_{\hat{h}} \sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P * \varphi) - \hat{h}(S^n, \beta) \right|$$

Curse of Dimensionality: Sample complexity exponential in $d$

'Sample Propagation' Estimator: Empirical distribution $\hat{P}_n = \frac{1}{n} \sum_{i=1}^{n} \delta_{S_i}$

$$\hat{h}_{SP}(S^n, \beta) \triangleq h(\hat{P}_n * \varphi)$$
The Sample Propagation Estimator

Abs. Error Minimax Risk: \( S^n \) are \( n \) i.i.d. samples from \( P \), define

\[
\mathcal{R}_d^*(n, \beta) \triangleq \inf_{\hat{h}} \sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}(S^n, \beta) \right|
\]

Curse of Dimensionality: Sample complexity exponential in \( d \)

‘Sample Propagation’ Estimator: Empirical distribution \( \hat{P}_n = \frac{1}{n} \sum_{i=1}^{n} \delta_{S_i} \)

\[
\hat{h}_{SP}(S^n, \beta) \triangleq h(\hat{P}_n \ast \varphi)
\]

Comments:
Abs. Error Minimax Risk: $S^n$ are $n$ i.i.d. samples from $P$, define

$$R^*_d(n, \beta) \triangleq \inf_{\hat{h}} \sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P \diamond \varphi) - \hat{h}(S^n, \beta) \right|$$

Curse of Dimensionality: Sample complexity exponential in $d$

'Sample Propagation' Estimator: Empirical distribution $\hat{P}_n = \frac{1}{n} \sum_{i=1}^{n} \delta_{S_i}$

$$\hat{h}_{SP}(S^n, \beta) \triangleq h(\hat{P}_n \diamond \varphi)$$

Comments:

- Plug-in: $\hat{h}_{SP}$ is just plug-in est. for the functional $T_{\varphi}(P) \triangleq h(P \diamond \varphi)$
The Sample Propagation Estimator

**Abs. Error Minimax Risk:** $S^n$ are \( n \) i.i.d. samples from \( P \), define

\[
\mathcal{R}_d^*(n, \beta) \triangleq \inf_{\hat{h}} \sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}(S^n, \beta) \right|
\]

**Curse of Dimensionality:** Sample complexity exponential in \( d \)

**‘Sample Propagation’ Estimator:** Empirical distribution \( \hat{P}_n = \frac{1}{n} \sum_{i=1}^{n} \delta_{S_i} \)

\[
\hat{h}_{SP}(S^n, \beta) \triangleq h(\hat{P}_n \ast \varphi)
\]

**Comments:**

- **Plug-in:** \( \hat{h}_{SP} \) is just plug-in est. for the functional \( T_{\varphi}(P) \triangleq h(P \ast \varphi) \)
- **Mixture:** \( \hat{h}_{SP} \) is the diff. entropy of a known Gaussian mixture
The Sample Propagation Estimator

Abs. Error Minimax Risk: \( S^n \) are \( n \) i.i.d. samples from \( P \), define

\[
R_d^*(n, \beta) \triangleq \inf_{\hat{h}} \sup_{P \in F_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}(S^n, \beta) \right|
\]

※ Curse of Dimensionality: Sample complexity exponential in \( d \)

‘Sample Propagation’ Estimator: Empirical distribution \( \hat{P}_n = \frac{1}{n} \sum_{i=1}^{n} \delta_{S_i} \)

\[
\hat{h}_{SP}(S^n, \beta) \triangleq h(\hat{P}_n \ast \varphi)
\]

Comments:

- **Plug-in:** \( \hat{h}_{SP} \) is just plug-in est. for the functional \( T_\varphi(P) \triangleq h(P \ast \varphi) \)
- **Mixture:** \( \hat{h}_{SP} \) is the diff. entropy of a **known** Gaussian mixture
- **Computing:** Can be efficiently computed via MC integration
The Sample Propagation Estimator - Convergence

**Theorem (ZG-Greenewald-Polyanskiy ’18)**

For $\mathcal{F}_d \triangleq \{ P | \text{supp}(P) \subseteq [-1, 1]^d \}$ and any $\beta > 0$ and $d \geq 1$, we have

$$\sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}_{SP}(S^n, \beta) \right| \leq O_\beta \left( \frac{(\log n)^{d/4}}{\sqrt{n}} \right).$$
The Sample Propagation Estimator - Convergence

**Theorem (ZG-Greeneawald-Polyanskiy ’18)**

For $\mathcal{F}_d \triangleq \{ P | \text{supp}(P) \subseteq [-1, 1]^d \}$ and any $\beta > 0$ and $d \geq 1$, we have

$$\sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}_{SP}(S^n, \beta) \right| \leq O_\beta \left( \frac{(\log n)^{d/4}}{\sqrt{n}} \right).$$

where $c_{\beta,d} \triangleq \frac{d}{2} \log(2\pi \beta^2) + \frac{d}{\beta^2}$. 

$$
\leq \frac{1}{2(4\pi \beta^2)^d/4} \log \left( \frac{n \left( 2 + 2\beta \sqrt{(2 + \epsilon) \log n} \right)^d}{\left( \pi \beta^2 \right)^{d/2}} \right) \left( 2 + 2\beta \sqrt{(2 + \epsilon) \log n} \right)^{d/2} \frac{1}{\sqrt{n}} \\
+ \left( c_{\beta,d}^2 + \frac{2c_{\beta,d}d(1 + \beta^2)}{\beta^2} + \frac{8d(d + 2\beta^4 + d\beta^4)}{\beta^4} \right) \frac{2}{n}
$$
The Sample Propagation Estimator - Convergence

**Theorem (ZG-Greenewald-Polyanskiy ’18)**

For $\mathcal{F}_d \triangleq \{ P | \text{supp}(P) \subseteq [-1, 1]^d \}$ and any $\beta > 0$ and $d \geq 1$, we have

$$\sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}_{SP}(S^n, \beta) \right| \leq O_\beta \left( \frac{(\log n)^{d/4}}{\sqrt{n}} \right).$$

**Pf. Technique:**
The Sample Propagation Estimator - Convergence

**Theorem (ZG-Greenewald-Polyanskiy ’18)**

For $\mathcal{F}_d \triangleq \{ P | \text{supp}(P) \subseteq [-1, 1]^d \}$ and any $\beta > 0$ and $d \geq 1$, we have

$$
\sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P * \varphi) - \hat{h}_{SP}(S^n, \beta) \right| \leq O(\beta) \left( \frac{(\log n)^{d/4}}{\sqrt{n}} \right).
$$

**Pf. Technique:** Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$
The Sample Propagation Estimator - Convergence

**Theorem (ZG-Greenewald-Polyanskiy ’18)**

For $\mathcal{F}_d \triangleq \{ P \mid \text{supp}(P) \subseteq [-1, 1]^d \}$ and any $\beta > 0$ and $d \geq 1$, we have

$$\sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}_{SP}(S^n, \beta) \right| \leq O_{\beta} \left( \frac{(\log n)^{d/4}}{\sqrt{n}} \right).$$

**Pf. Technique:** Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

- **Inside $\mathcal{R}$:** Modulus of cont. & Convex analysis & Functional opt.
**The Sample Propagation Estimator - Convergence**

**Theorem (ZG-Greenewald-Polyanskiy ’18)**

For \( \mathcal{F}_d \triangleq \{ P | \text{supp}(P) \subseteq [-1, 1]^d \} \) and any \( \beta > 0 \) and \( d \geq 1 \), we have

\[
\sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}_{SP}(S^n, \beta) \right| \leq O_\beta \left( \frac{(\log n)^{d/4}}{\sqrt{n}} \right).
\]

**Pf. Technique:** Split analysis to \( \mathcal{R} \triangleq [-1, 1]^d + B(0, \sqrt{c \log n}) \) and \( \mathcal{R}^c \)

- **Inside \( \mathcal{R} \):** Modulus of cont. & Convex analysis & Functional opt.
- **Outside \( \mathcal{R} \):** Chi-squared distribution tail bounds
The Sample Propagation Estimator - Convergence

**Theorem (ZG-Greenewald-Polyanskiy ’18)**

For $\mathcal{F}_d \triangleq \{ P \,|\, \text{supp}(P) \subseteq [-1, 1]^d \}$ and any $\beta > 0$ and $d \geq 1$, we have

$$\sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P * \varphi) - \hat{h}_{SP}(S^n, \beta) \right| \leq O_\beta \left( \frac{(\log n)^{d/4}}{\sqrt{n}} \right).$$

**Pf. Technique:** Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

- **Inside $\mathcal{R}$:** Modulus of cont. & Convex analysis & Functional opt.
- **Outside $\mathcal{R}$:** Chi-squared distribution tail bounds

**Comments:**
The Sample Propagation Estimator - Convergence

**Theorem (ZG-Greenewald-Polyanskiy ’18)**

For \( F_d \triangleq \{ P \mid \text{supp}(P) \subseteq [-1, 1]^d \} \) and any \( \beta > 0 \) and \( d \geq 1 \), we have

\[
\sup_{P \in F_d} \mathbb{E}_{S^n} \left| h(P * \varphi) - \hat{h}_{SP}(S^n, \beta) \right| \leq O_{\beta} \left( \frac{(\log n)^{d/4}}{\sqrt{n}} \right).
\]

**Pf. Technique:** Split analysis to \( R \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n}) \) and \( R^c \)

- **Inside** \( R \): Modulus of cont. & Convex analysis & Functional opt.
- **Outside** \( R \): Chi-squared distribution tail bounds

**Comments:**

- **Faster rate** than \( O \left( n^{-\frac{\alpha s}{\beta s + d}} \right) \) for kNN/KDE est. via ‘noisy’ samples
The Sample Propagation Estimator - Convergence

**Theorem (ZG-Greenewald-Polyanskiy '18)**

For $F_d \triangleq \{ P \mid \text{supp}(P) \subseteq [-1, 1]^d \}$ and any $\beta > 0$ and $d \geq 1$, we have

$$\sup_{P \in F_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}_{SP}(S^n, \beta) \right| \leq O_\beta \left( \frac{(\log n)^{d/4}}{\sqrt{n}} \right).$$

**Pf. Technique:** Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + B(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

- **Inside $\mathcal{R}$:** Modulus of cont. & Convex analysis & Functional opt.
- **Outside $\mathcal{R}$:** Chi-squared distribution tail bounds

**Comments:**

- **Faster rate** than $O \left( n^{-\frac{\alpha_s}{\beta s+d}} \right)$ for kNN/KDE est. via ‘noisy’ samples
- **Explicit expression** enables **concrete error bounds** in simulations
The Sample Propagation Estimator - Convergence

Theorem (ZG-Greenewald-Polyanskiy '18)

For $\mathcal{F}_d \triangleq \{P | \text{supp}(P) \subseteq [-1, 1]^d \}$ and any $\beta > 0$ and $d \geq 1$, we have

$$\sup_{P \in \mathcal{F}_d} \mathbb{E}_{S^n} \left| h(P \ast \varphi) - \hat{h}_{\text{SP}}(S^n, \beta) \right| \leq O_\beta \left( \frac{(\log n)^{d/4}}{\sqrt{n}} \right).$$

Pf. Technique: Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

- **Inside $\mathcal{R}$**: Modulus of cont. & Convex analysis & Functional opt.
- **Outside $\mathcal{R}$**: Chi-squared distribution tail bounds

Comments:

- **Faster rate** than $O \left( n^{-\frac{\alpha s}{\beta s+d}} \right)$ for kNN/KDE est. via ‘noisy’ samples
- **Explicit expression** enables **concrete error bounds** in simulations
- **Extension**: $P$ with sub-Gaussian marginals (ReLU + Weight regular.)
Strategy: Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$
**Strategy:** Split analysis to \( \mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n}) \) and \( \mathcal{R}^c \)

- **Restricted Entropy:** \( h_{\mathcal{R}}(p) \triangleq \mathbb{E}[-\log p(X) \mathbb{1}_{\{X \in \mathcal{R}\}}] \)
**Strategy:** Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

**Restricted Entropy:**

$$h_{\mathcal{R}}(p) \triangleq \mathbb{E}[-\log p(X) \mathbb{1}_{\{X \in \mathcal{R}\}}]$$

$$\sup \mathbb{E} |h(P \ast \varphi) - h(\hat{P}_n \ast \varphi)| \leq \sup \mathbb{E} |h_{\mathcal{R}}(P \ast \varphi) - h_{\mathcal{R}}(\hat{P}_n \ast \varphi)| + 2 \sup |h_{\mathcal{R}^c}(P \ast \varphi)|$$
The Sample Propagation Estimator - Proof Ideas

**Strategy:** Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

- **Restricted Entropy:** $h_\mathcal{R}(p) \triangleq \mathbb{E}[-\log p(X)1_{\{X \in \mathcal{R}\}}]$

$$\sup \mathbb{E}|h(P \ast \varphi) - h(\hat{P}_n \ast \varphi)| \leq \sup \mathbb{E}|h_\mathcal{R}(P \ast \varphi) - h_\mathcal{R}(\hat{P}_n \ast \varphi)| + 2 \sup \sup \mathbb{E}|h_\mathcal{R}^c(P \ast \varphi)|$$

- **Inside $\mathcal{R}$:** $-t \log t$ modulus of cont. for $x \mapsto x \log x$ & Jensen’s ineq.
The Sample Propagation Estimator - Proof Ideas

**Strategy:** Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

- **Restricted Entropy:**
  
  $$
  h_{\mathcal{R}}(p) \triangleq \mathbb{E}[-\log p(X) 1_{\{X \in \mathcal{R}\}}]
  $$

  $$
  \sup \mathbb{E}|h(P \ast \varphi) - h(\hat{P}_n \ast \varphi)| \leq \sup \mathbb{E}|h_{\mathcal{R}}(P \ast \varphi) - h_{\mathcal{R}}(\hat{P}_n \ast \varphi)| + 2 \sup |h_{\mathcal{R}^c}(P \ast \varphi)|
  $$

- **Inside $\mathcal{R}$:**
  
  $$
  -t \log t \text{ modulus of cont. for } x \mapsto x \log x \text{ & Jensen's ineq.}
  $$

  $$
  \implies \text{ Focus on analyzing } \mathbb{E}|(P \ast \varphi)(x) - (\hat{P}_n \ast \varphi)(x)|
  $$
The Sample Propagation Estimator - Proof Ideas

**Strategy:** Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

- **Restricted Entropy:** $h_{\mathcal{R}}(p) \triangleq \mathbb{E}[-\log p(X)1_{\{X \in \mathcal{R}\}}]$

$$\sup \mathbb{E}|h(P*\varphi) - h(\hat{P}_n*\varphi)| \leq \sup \mathbb{E}|h_{\mathcal{R}}(P*\varphi) - h_{\mathcal{R}}(\hat{P}_n*\varphi)| + 2 \sup |h_{\mathcal{R}^c}(P*\varphi)|$$

- **Inside $\mathcal{R}$:** $-t \log t$ modulus of cont. for $x \mapsto x \log x$ & Jensen’s ineq.

  $\implies$ Focus on analyzing $\mathbb{E}|(P*\varphi)(x) - (\hat{P}_n*\varphi)(x)|$

  $\implies$ Bias & variance analysis
**Strategy:** Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

- **Restricted Entropy:** $h_\mathcal{R}(p) \triangleq \mathbb{E}[-\log p(X) 1_{\{X \in \mathcal{R}\}}]$

$$\sup \mathbb{E}|h(P \ast \varphi) - h(\hat{P}_n \ast \varphi)| \leq \sup \mathbb{E}|h_\mathcal{R}(P \ast \varphi) - h_\mathcal{R}(\hat{P}_n \ast \varphi)| + 2 \sup |h_{\mathcal{R}^c}(P \ast \varphi)|$$

- **Inside $\mathcal{R}$:** $-t \log t$ modulus of cont. for $x \mapsto x \log x$ & Jensen’s ineq.

$\implies$ Focus on analyzing $\mathbb{E}|(P \ast \varphi)(x) - (\hat{P}_n \ast \varphi)(x)|$

- Bias & variance analysis

$\implies$ $\mathbb{E}|(P \ast \varphi)(x) - (\hat{P}_n \ast \varphi)(x)| \leq c_1 \sqrt{\frac{(P \ast \tilde{\varphi})(x)}{n}}, \quad \tilde{\varphi} = \mathcal{N}(0, \frac{\beta^2}{2} I)$
The Sample Propagation Estimator - Proof Ideas

**Strategy:** Split analysis to $\mathcal{R} \triangleq [−1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

- **Restricted Entropy:**
  \[ h_\mathcal{R}(p) \triangleq \mathbb{E}[−\log p(X) 1_{\{X \in \mathcal{R}\}}] \]
  \[ \sup \mathbb{E}|h(P \ast \varphi) − h(\hat{P}_n \ast \varphi)| \leq \sup \mathbb{E}|h_\mathcal{R}(P \ast \varphi) − h_\mathcal{R}(\hat{P}_n \ast \varphi)| + 2 \sup |h_{\mathcal{R}^c}(P \ast \varphi)| \]

- **Inside $\mathcal{R}$:**
  - $−t \log t$ modulus of cont. for $x \mapsto x \log x$ & Jensen’s ineq.
  - $\Rightarrow$ Focus on analyzing $\mathbb{E}|(P \ast \varphi)(x) − (\hat{P}_n \ast \varphi)(x)|$
  - $\Rightarrow$ Bias & variance analysis
  - $\Rightarrow$ $\mathbb{E}|(P \ast \varphi)(x) − (\hat{P}_n \ast \varphi)(x)| \leq c_1 \sqrt{\frac{(P \ast \tilde{\varphi})(x)}{n}}, \quad \tilde{\varphi} = \mathcal{N}\left(0, \frac{\beta^2}{2}I\right)$
  - $\Rightarrow$ Plug back in & Convex analysis
The Sample Propagation Estimator - Proof Ideas

**Strategy:** Split analysis to \(\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})\) and \(\mathcal{R}^c\)

- **Restricted Entropy:**
  \[ h_{\mathcal{R}}(p) \triangleq \mathbb{E}[-\log p(X)1_{\{X \in \mathcal{R}\}}] \]

  \[
  \sup \mathbb{E}|h(P \ast \varphi) - h(\hat{P}_n \ast \varphi)| \leq \sup \mathbb{E}|h_{\mathcal{R}}(P \ast \varphi) - h_{\mathcal{R}}(\hat{P}_n \ast \varphi)| + 2 \sup |h_{\mathcal{R}}(P \ast \varphi)|
  \]

- **Inside \(\mathcal{R}\):** \(\downarrow \) \(-t \log t\) modulus of cont. for \(x \mapsto x \log x\) & Jensen’s ineq.

  \[\implies\] Focus on analyzing \(\mathbb{E}|(P \ast \varphi)(x) - (\hat{P}_n \ast \varphi)(x)|\)

  \(\uparrow\) Bias & variance analysis

  \[\implies\] \(\mathbb{E}|(P \ast \varphi)(x) - (\hat{P}_n \ast \varphi)(x)| \leq c_1 \sqrt{\frac{\mathbb{P} \ast \varphi(x)}{n}}, \ \tilde{\varphi} = \mathcal{N}(0, \frac{\beta^2}{2} I)\)

  \(\uparrow\) Plug back in & Convex analysis

  \[\implies\] \(\sup \mathbb{E}|h_{\mathcal{R}}(P \ast \varphi) - h_{\mathcal{R}}(\hat{P}_n \ast \varphi)| \leq c_2 \log \left(\frac{n \lambda(\mathcal{R})}{c_3}\right) \sqrt{\frac{\lambda(\mathcal{R})}{n}}\)
The Sample Propagation Estimator - Proof Ideas

**Strategy:** Split analysis to $\mathcal{R} \triangleq [-1, 1]^d + \mathcal{B}(0, \sqrt{c \log n})$ and $\mathcal{R}^c$

- **Restricted Entropy:**
  
  $h_{\mathcal{R}}(p) \triangleq \mathbb{E}[-\log p(X)1_{\{X \in \mathcal{R}\}}]$ 

  $\sup \mathbb{E}|h(P * \varphi) - h(\hat{P}_n * \varphi)| \leq \sup \mathbb{E}|h_{\mathcal{R}}(P * \varphi) - h_{\mathcal{R}}(\hat{P}_n * \varphi)| + 2 \sup |h_{\mathcal{R}^c}(P * \varphi)|$

- **Inside $\mathcal{R}$:**
  
  $-t \log t$ modulus of cont. for $x \mapsto x \log x$ & Jensen’s ineq.

  $\Rightarrow$ Focus on analyzing $\mathbb{E}|(P * \varphi)(x) - (\hat{P}_n * \varphi)(x)|$

  $\Rightarrow$ Bias & variance analysis

  $\Rightarrow$ $\mathbb{E}|(P * \varphi)(x) - (\hat{P}_n * \varphi)(x)| \leq c_1 \sqrt{\frac{(P * \tilde{\varphi})(x)}{n}}, \quad \tilde{\varphi} = \mathcal{N}(0, \frac{\beta^2}{2} \mathbf{I})$

  $\Rightarrow$ Plug back in & Convex analysis

  $\Rightarrow$ $\sup \mathbb{E}|h_{\mathcal{R}}(P * \varphi) - h_{\mathcal{R}}(\hat{P}_n * \varphi)| \leq c_2 \log \left(\frac{n \lambda(\mathcal{R})}{c_3}\right) \sqrt{\frac{\lambda(\mathcal{R})}{n}}$

- **Outside $\mathcal{R}$:** $O \left(\frac{1}{n}\right)$ decay via Chi-squared distribution tail bounds
Binning vs True Mutual Information

Comparing to Previously Shown MI Plots:
Binning vs True Mutual Information

Comparing to Previously Shown MI Plots:
Binning vs True Mutual Information

Comparing to Previously Shown MI Plots:

⇒ Past works were not showing MI but clustering (via binned-MI)!