Estimating the Information Flow in Deep Neural Networks

Ziv Goldfeld

MIT

IT Forum, Information Systems Laboratory, Stanford University

November 9th, 2018

Collaborators: E. van den Berg, K. Greenewald, I. Melnyk, N. Nguyen, B. Kingsbury and Y. Polyanskiy

MIT-IBM Watson AI Lab

How do Deep Neural Networks Learn?

- Unprecedented practical success in hosts of tasks

How do Deep Neural Networks Learn?

- Unprecedented practical success in hosts of tasks
- Lacking theory:

How do Deep Neural Networks Learn?

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

How do Deep Neural Networks Learn?

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

How do Deep Neural Networks Learn?

- 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?

How do Deep Neural Networks Learn?

- 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?

$$\vdots$$

How do Deep Neural Networks Learn?

- 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

How do Deep Neural Networks Learn?

- 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
  - Shallow networks [Ge-Lee-Ma'17, Mei-Montanari-Nguyen'18]

How do Deep Neural Networks Learn?

- 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
  - Shallow networks [Ge-Lee-Ma'17, Mei-Montanari-Nguyen'18]
  - Opt. in parameter space [Saxe'14, Choromanska'15, Wei'18]

How do Deep Neural Networks Learn?

- 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?

$\vdots$

- Some past attempts to understand effectiveness of deep learning
  - Shallow networks [Ge-Lee-Ma'17, Mei-Montanari-Nguyen'18]
  - Opt. in parameter space [Saxe'14, Choromanska'15, Wei'18]
  - Classes of efficiently representable functions [Montufar'14, Poggio'17]

How do Deep Neural Networks Learn?

- 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
  - Shallow networks [Ge-Lee-Ma'17, Mei-Montanari-Nguyen'18]
  - Opt. in parameter space [Saxe'14, Choromanska'15, Wei'18]
  - Classes of efficiently representable functions [Montufar'14, Poggio'17]
  - Information theory [Tishby'17, Saxe'18, Gabrié'18]

How do Deep Neural Networks Learn?

- 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
  - Shallow networks [Ge-Lee-Ma'17, Mei-Montanari-Nguyen'18]
  - Opt. in parameter space [Saxe'14, Choromanska'15, Wei'18]
  - Classes of efficiently representable functions [Montufar'14, Poggio'17]
  - **Information theory** [Tishby'17, Saxe'18, Gabrié'18]

How do Deep Neural Networks Learn?

- 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?

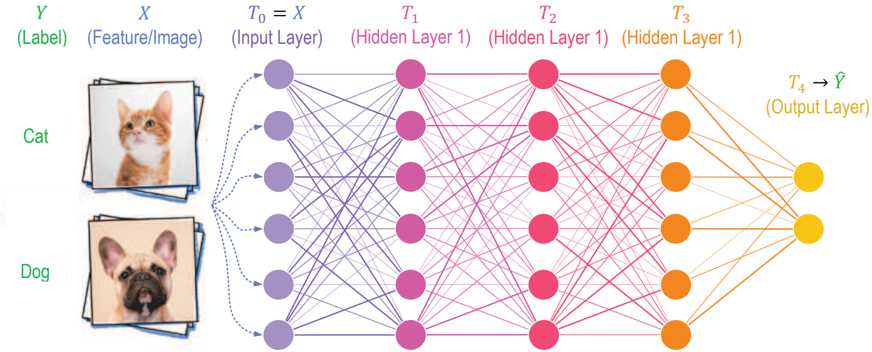
  ⋮

- Some past attempts to understand effectiveness of deep learning
  - Shallow networks [Ge-Lee-Ma'17, Mei-Montanari-Nguyen'18]
  - Opt. in parameter space [Saxe'14, Choromanska'15, Wei'18]
  - Classes of efficiently representable functions [Montufar'14, Poggio'17]
  - **Information theory** [Tishby'17, Saxe'18, Gabrié'18]

★ **Goal:** Explain 'compression' in Information Bottleneck framework

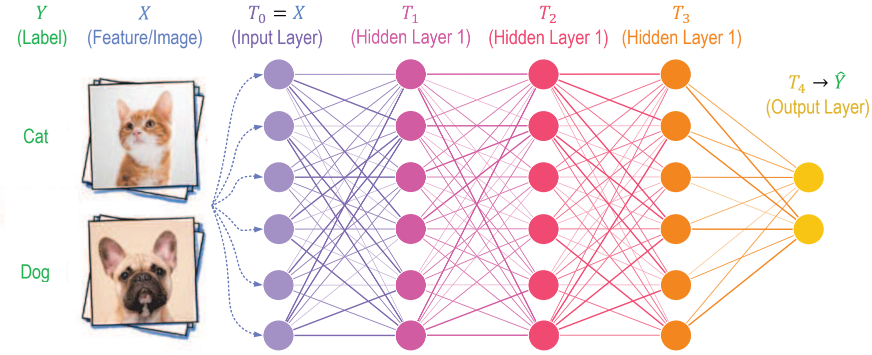
Setup and Preliminaries

Feedforward DNN for Classification:



Setup and Preliminaries

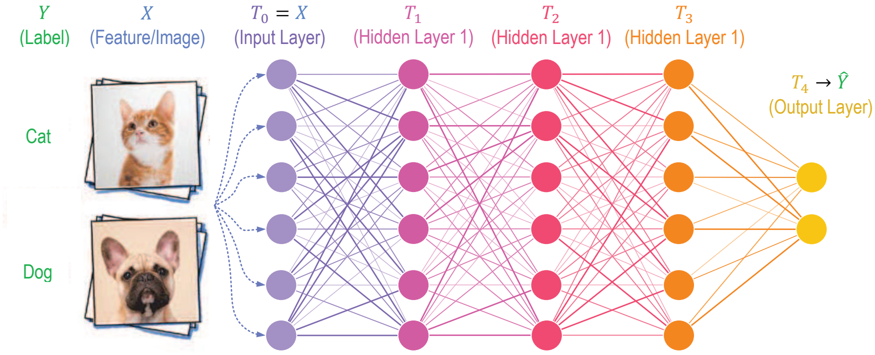
Feedforward DNN for Classification:



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

Setup and Preliminaries

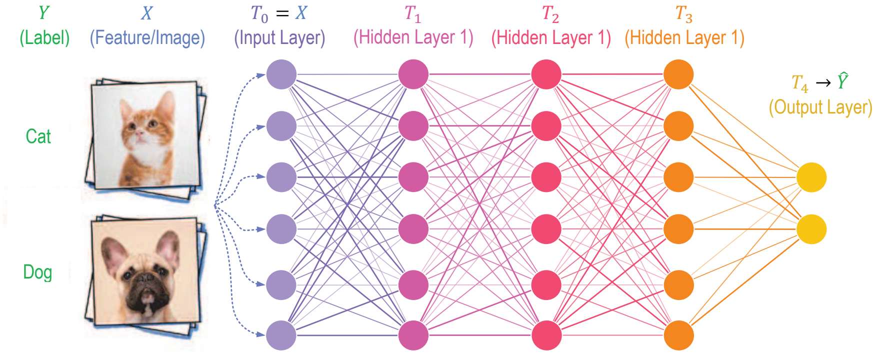
Feedforward DNN for Classification:



- **Deterministic DNN:** $T_\ell = f_\ell(T_{\ell-1})$ (**MLP:** $T_\ell = \sigma(\mathrm{W}_\ell T_{\ell-1} + b_\ell)$)
- **Joint Distribution:** $P_{X,Y}$

Setup and Preliminaries

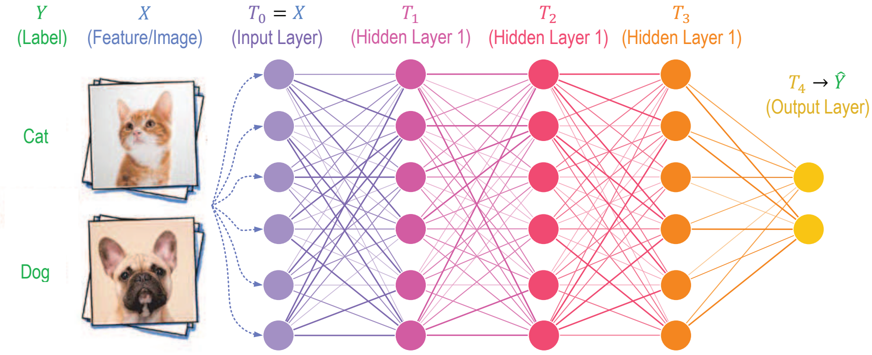
Feedforward DNN for Classification:



- **Deterministic DNN:** $T_\ell = f_\ell(T_{\ell-1})$ **(MLP:** $T_\ell = \sigma(\mathrm{W}_\ell T_{\ell-1} + b_\ell)$**)**
- **Joint Distribution:** $P_{X,Y} \implies P_{X,Y} \cdot P_{T_1,\ldots,T_L|X}$

Setup and Preliminaries

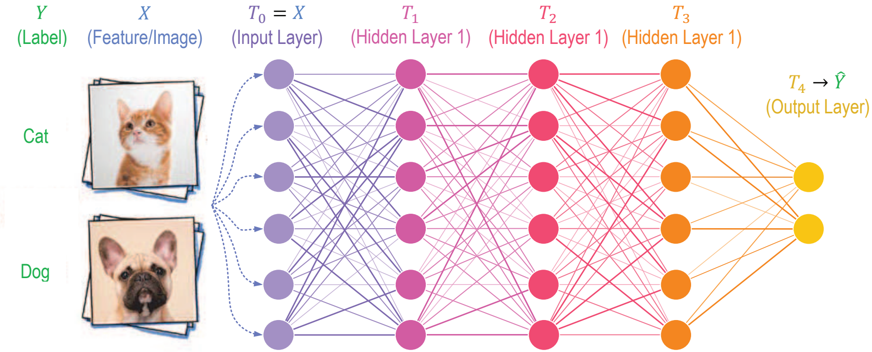
Feedforward DNN for Classification:



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

Setup and Preliminaries

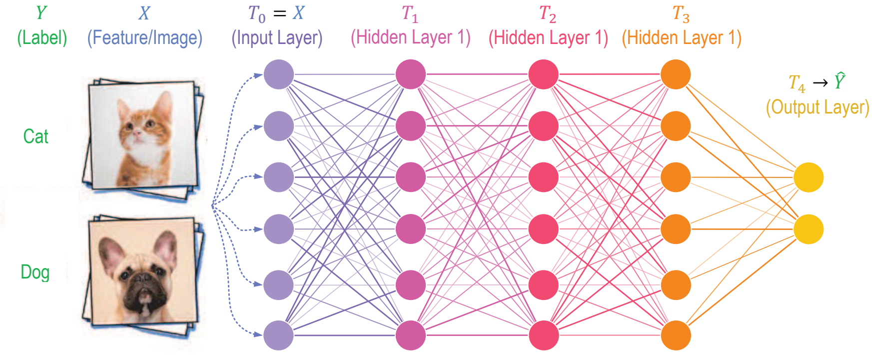
Feedforward DNN for Classification:



IB Theory Claim: Training comprises 2 phases

Setup and Preliminaries

Feedforward DNN for Classification:

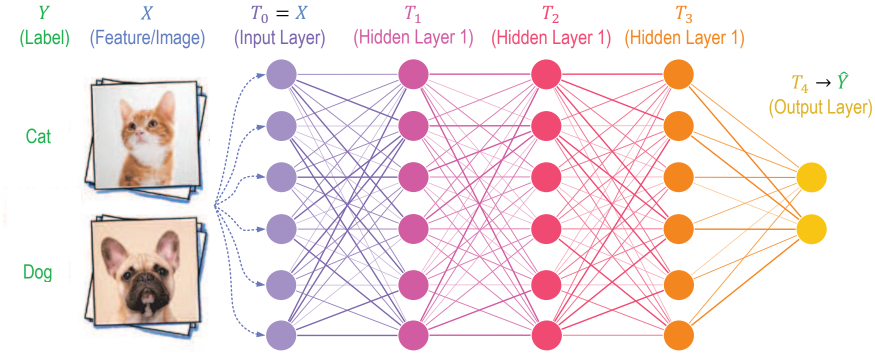


IB Theory Claim: Training comprises 2 phases

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

Setup and Preliminaries

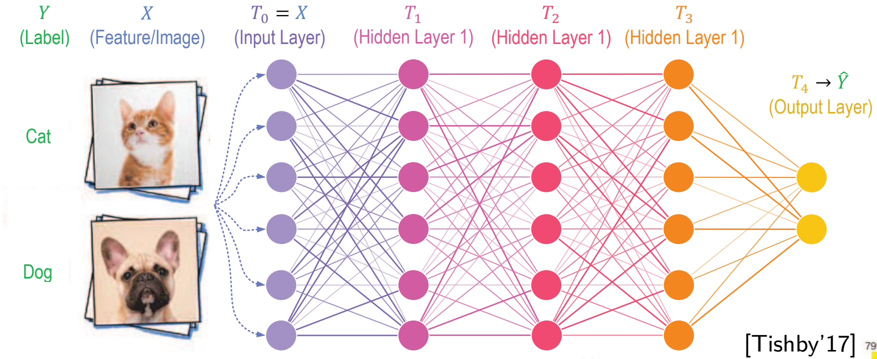
Feedforward DNN for Classification:

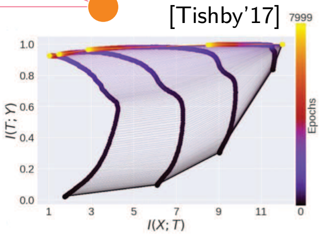


IB Theory Claim: Training comprises 2 phases

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

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)

Meaningless Mutual Information

Observation

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

Meaningless 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*

Meaningless 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?

Meaningless 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?

- **Continuous X:**

Meaningless 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?

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

Meaningless Mutual Information

Observation

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

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

Why?

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

Meaningless Mutual Information

Observation

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

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

Why?

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

Meaningless 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:** $$I(X;T_\ell) = h(T_\ell) - \underbrace{h(\tilde{f}_\ell(X)|X)}_{=-\infty}$$

Meaningless 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?

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

Meaningless Mutual Information

Observation

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

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

Why?

- **Continuous $\boldsymbol{X}$:** $\quad I(X;T_\ell) = h(T_\ell) - h\big(\tilde{f}_\ell(X)\big|X\big) = \infty$
- **Discrete $\boldsymbol{X}$:**

Meaningless 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 $\boldsymbol{X}$:** $I(X;T_\ell) = h(T_\ell) - h\big(\tilde{f}_\ell(X)\big|X\big) = \infty$
- **Discrete $\boldsymbol{X}$:** The map $X \mapsto T_\ell$ is injective*

$\star$ For almost all weight matrices and bias vectors

Meaningless Mutual Information

Observation

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

$$\Longrightarrow \; I(X;T_\ell) \text{ is independent of the DNN parameters}$$

Why?

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

$\star$ For almost all weight matrices and bias vectors

Meaningless Mutual Information

Observation

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

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

Why?

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

Meaningless Mutual Information

Observation

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

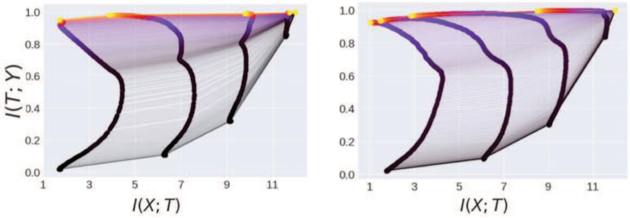
$\Longrightarrow$ $I(X;T_\ell)$ *is* ***independent of the DNN parameters***

Why?

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

Past Works:

[Schwartz-Ziv&Tishby'17, Saxe *et al. '18]*



What is going on here?

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

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 $\quad I\big(X;\text{Bin}(T_\ell)\big)$

What is going on here?

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

$$\Longrightarrow \text{ Plotted values are } \quad I(X;\text{Bin}(T_\ell)) \overset{??}{\approx} I(X;T_\ell)$$

What is going on here?

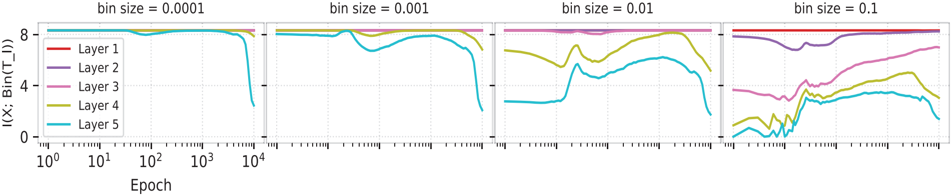
- Plots via binning-based estimator of $I(X;T_\ell)$, for $X \sim \text{Unif}(\text{dataset})$

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

What is going on here?

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

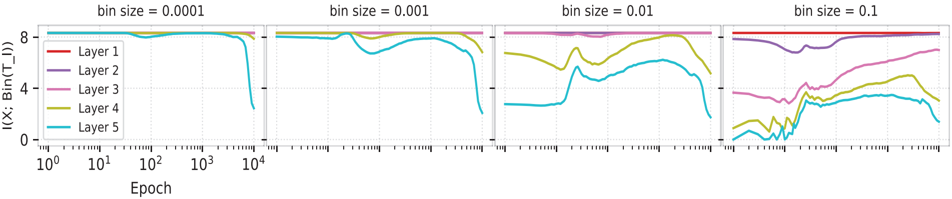
$$\implies \text{Plotted values are} \quad I(X;\mathsf{Bin}(T_\ell)) \overset{??}{\approx} I(X;T_\ell) \qquad \textbf{No!}$$



What is going on here?

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

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

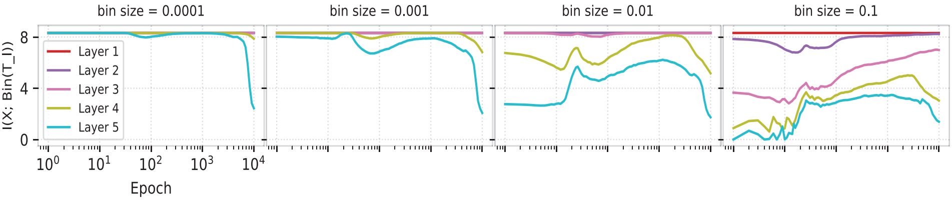


- Smaller bins $\Longrightarrow$ Closer to truth: $\quad I(X;T_\ell) = \ln(2^{12}) \approx 8.31$

What is going on here?

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

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

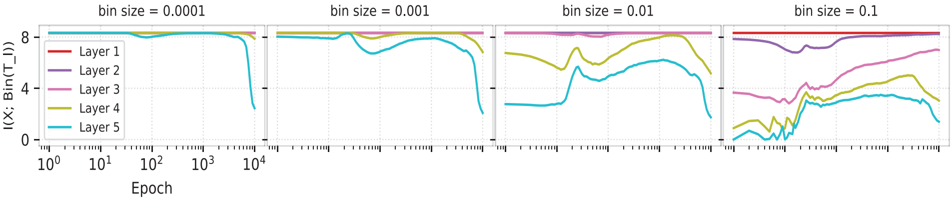


- Smaller bins $\implies$ Closer to truth: $I(X;T_\ell) = \ln(2^{12}) \approx 8.31$
- Binning introduces "noise" into estimator (not present in the DNN)

What is going on here?

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

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

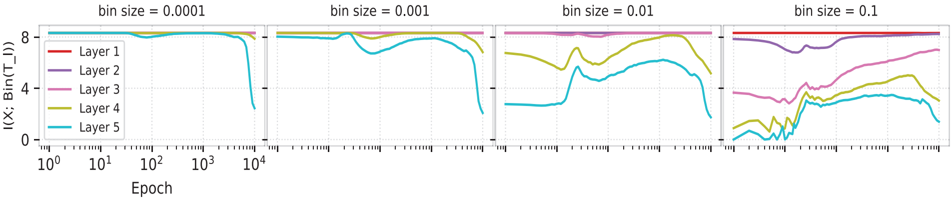


- Smaller bins $\implies$ Closer to truth: $\quad I(X;T_\ell) = \ln(2^{12}) \approx 8.31$
- Binning introduces "noise" into estimator (not present in the DNN)
- Plots showing estimation errors

What is going on here?

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

$$\Longrightarrow \text{ Plotted values are } \quad I(X;\mathsf{Bin}(T_\ell)) \overset{??}{\approx} I(X;T_\ell) \qquad \textbf{No!}$$



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

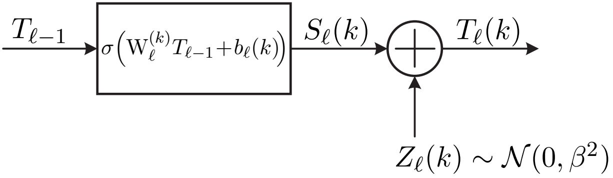
Auxiliary Framework - Noisy Deep Neural Networks

Modification: Inject (small) Gaussian noise to neurons' output

Auxiliary Framework - Noisy Deep Neural Networks

Modification: Inject (small) Gaussian noise to neurons' output

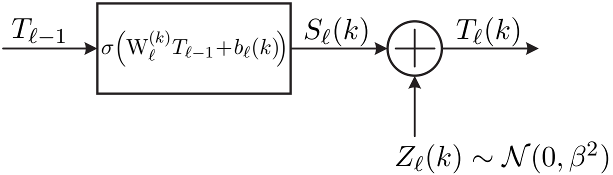
- **Formally:** $T_\ell = f_\ell(T_{\ell-1}) + Z_\ell$, where $Z_\ell \sim \mathcal{N}(0, \beta^2 \mathrm{I})$



Auxiliary Framework - Noisy Deep Neural Networks

Modification: Inject (small) Gaussian noise to neurons' output

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

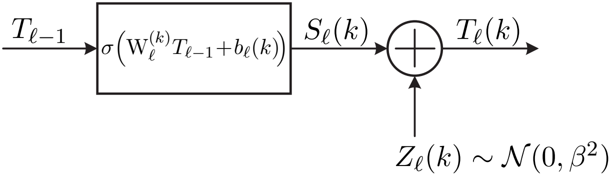


$\implies X \mapsto T_\ell$ is a **parametrized channel** that depends on DNN param.!

Auxiliary Framework - Noisy Deep Neural Networks

Modification: Inject (small) Gaussian noise to neurons' output

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



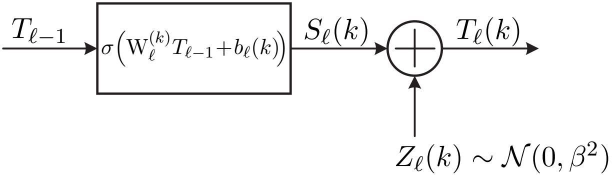
$\Longrightarrow$ $X \mapsto T_\ell$ is a **parametrized channel** that depends on DNN param.!

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

Auxiliary Framework - Noisy Deep Neural Networks

Modification: Inject (small) Gaussian noise to neurons' output

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



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

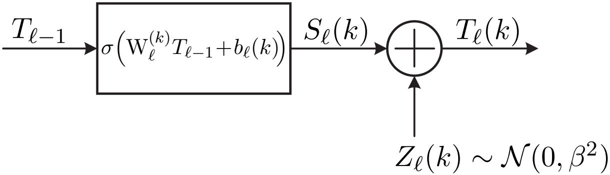
$\Longrightarrow$ $I(X; T_\ell)$ is a **function** of weights and biases!

- **Operational Perspective:**

Auxiliary Framework - Noisy Deep Neural Networks

Modification: Inject (small) Gaussian noise to neurons' output

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



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

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

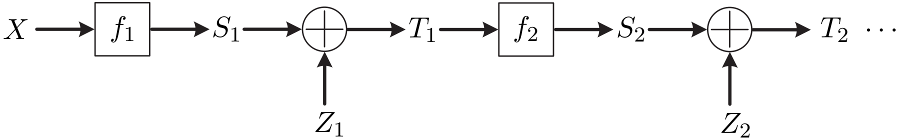
- **Operational Perspective:**

  Performance & learned representations similar to det. DNNs ($\beta \approx 10^{-1}$)

Mutual Information in Noisy DNNs

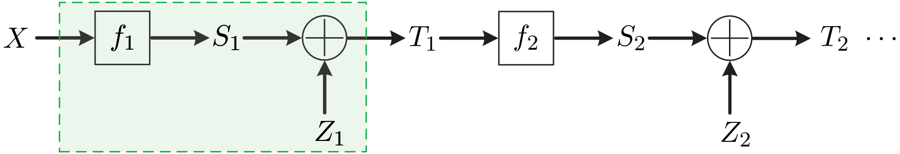
Mutual Information in Noisy DNNs

Noisy DNN:



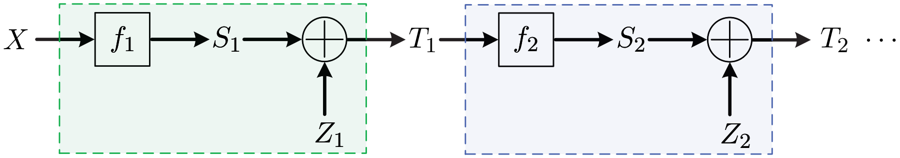
Mutual Information in Noisy DNNs

Noisy DNN:



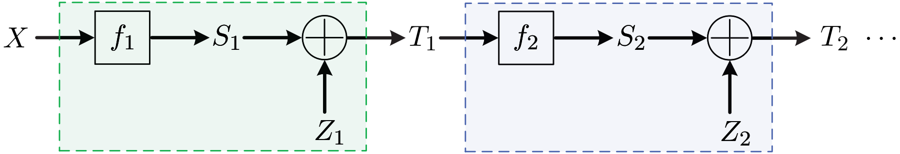
Mutual Information in Noisy DNNs

Noisy DNN:



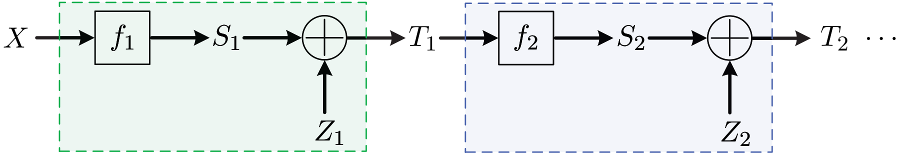
Mutual Information in Noisy DNNs

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



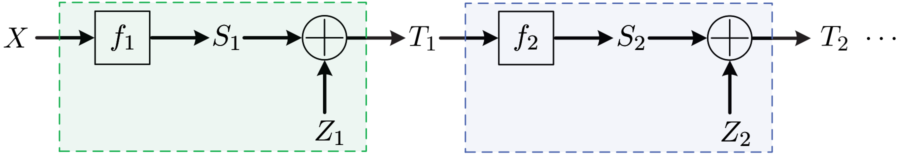
Mutual Information in Noisy DNNs

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



Mutual Information in Noisy DNNs

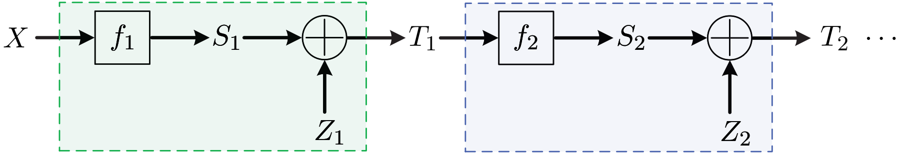
<u>**Noisy DNN:**</u> $S_\ell \triangleq f_\ell(T_{\ell-1}) \quad \Longrightarrow \quad T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 \mathrm{I})$



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

Mutual Information in Noisy DNNs

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

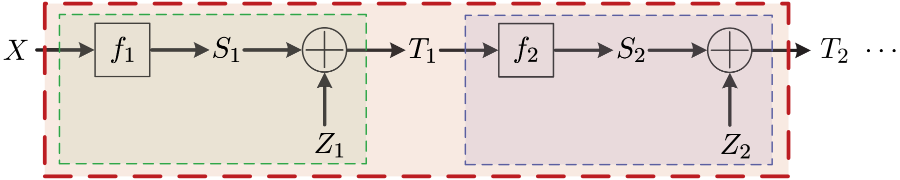


- **Assume:** $X \sim \mathsf{Unif}(\mathcal{X})$, where $\mathcal{X} \triangleq \{x_i\}_{i=1}^m$ is empirical dataset

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

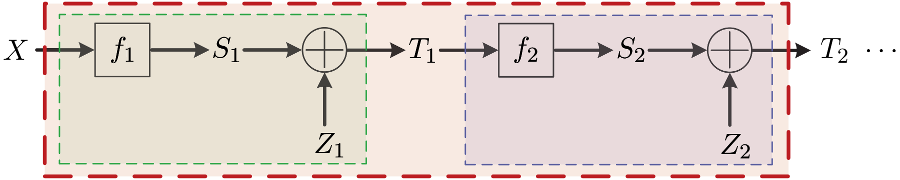


- **Assume:** $X \sim \mathsf{Unif}(\mathcal{X})$, where $\mathcal{X} \triangleq \{x_i\}_{i=1}^m$ is empirical dataset

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

Mutual Information in Noisy DNNs

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



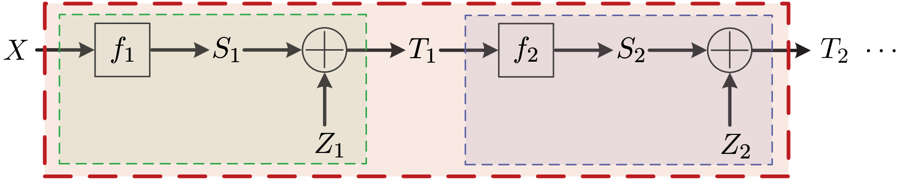
- **Assume:** $X \sim \mathsf{Unif}(\mathcal{X})$, where $\mathcal{X} \triangleq \{x_i\}_{i=1}^m$ is empirical dataset

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

⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

Mutual Information in Noisy DNNs

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



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

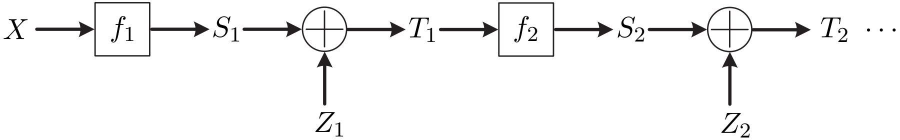
$\Longrightarrow$ **Mutual Information:** $I(X;T_\ell) = h(T_\ell) - \frac{1}{m}\sum_{i=1}^m h(T_\ell|X = x_i)$

⊛ P_{T_ℓ} 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: $S_\ell \triangleq f_\ell(T_{\ell-1}) \quad \Longrightarrow \quad T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 \mathrm{I})$



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

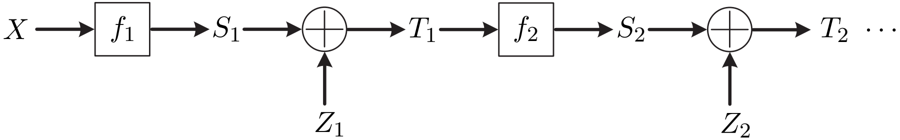
$\Longrightarrow$ **Mutual Information:** $I(X;T_\ell) = h(T_\ell) - \frac{1}{m}\sum_{i=1}^m h(T_\ell|X = x_i)$

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



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

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

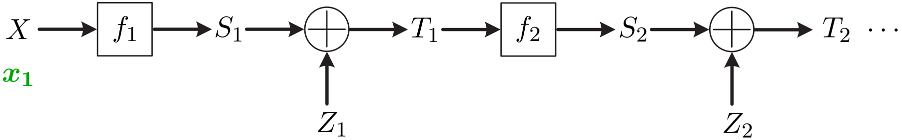
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

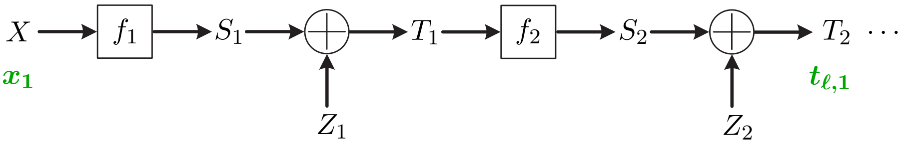
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

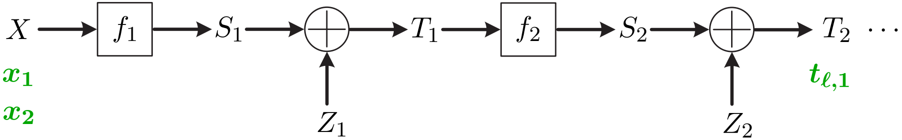
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

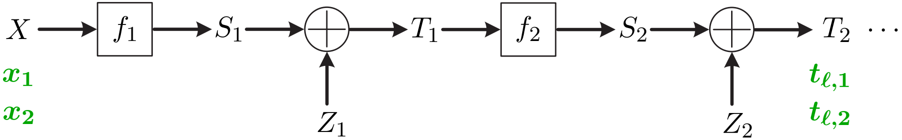
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

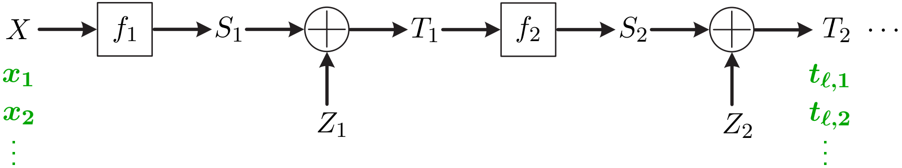
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

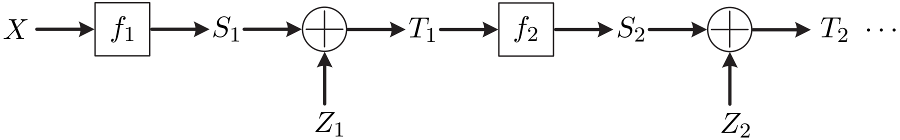
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

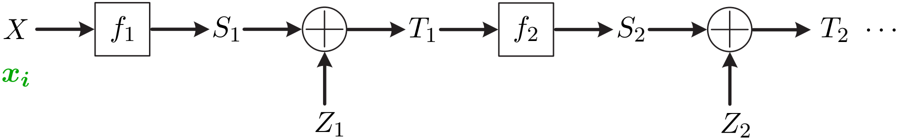
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values
- **Sampling $P_{T_\ell|X=x_i}$:** Feed x_i multiples times & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

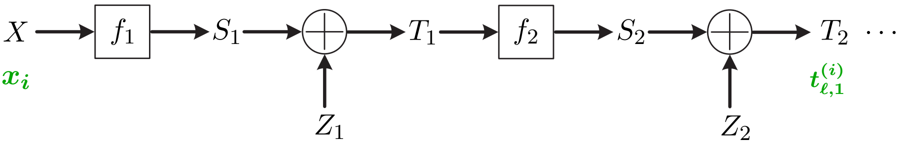
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values
- **Sampling $P_{T_\ell|X=x_i}$:** Feed x_i multiples times & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

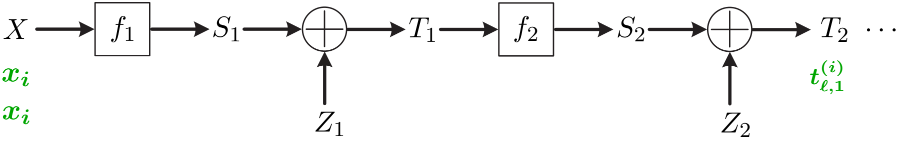
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values
- **Sampling $P_{T_\ell|X=x_i}$:** Feed x_i multiples times & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

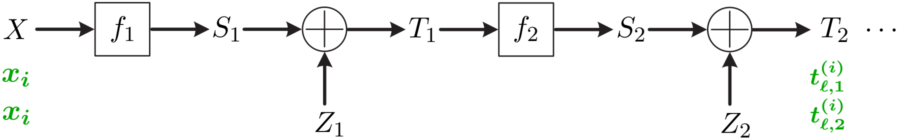
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values
- **Sampling $P_{T_\ell|X=x_i}$:** Feed x_i multiples times & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

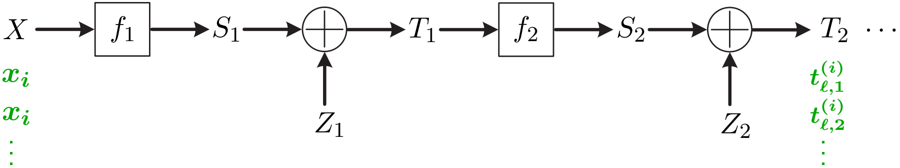
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values
- **Sampling $P_{T_\ell|X=x_i}$:** Feed x_i multiples times & read T_ℓ values

Mutual Information in Noisy DNNs

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



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

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

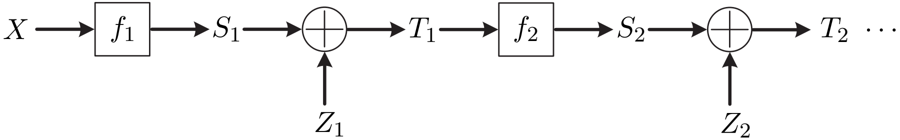
⊛ P_{T_ℓ} and $P_{T_\ell|X}$ are **extremely** complicated to compute/evaluate

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

- **Sampling P_{T_ℓ}:** Feed randomly chosen x_i's & read T_ℓ values
- **Sampling $P_{T_\ell|X=x_i}$:** Feed x_i multiples times & read T_ℓ values

General-Purpose Differential Entropy Estimators

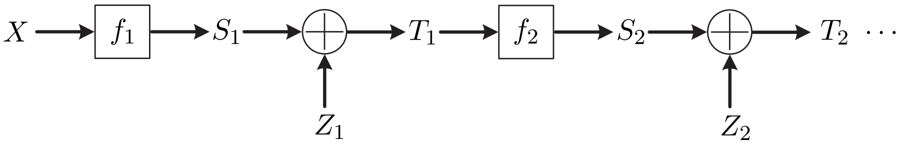
Noisy DNN: $S_\ell \triangleq f_\ell(T_{\ell-1}) \quad \Longrightarrow \quad T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 \mathrm{I})$



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

General-Purpose Differential Entropy Estimators

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

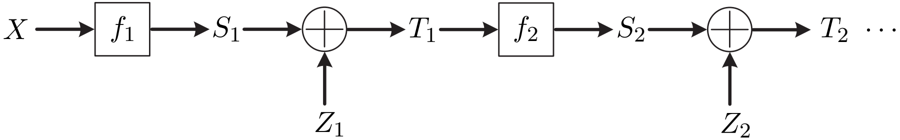


$\Longrightarrow$ Estimate $I(X;T_\ell)$ from samples via **general-purpose $h(P)$ est.:**

- Most results assume lower bounded density

General-Purpose Differential Entropy Estimators

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

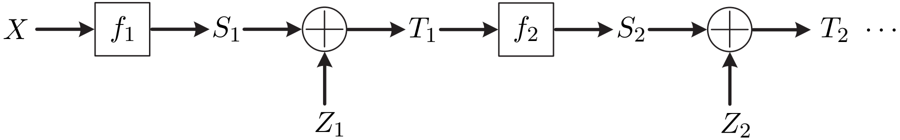


$\Longrightarrow$ Estimate $I(X;T_\ell)$ from samples via **general-purpose $h(P)$ est.:**

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

General-Purpose Differential Entropy Estimators

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

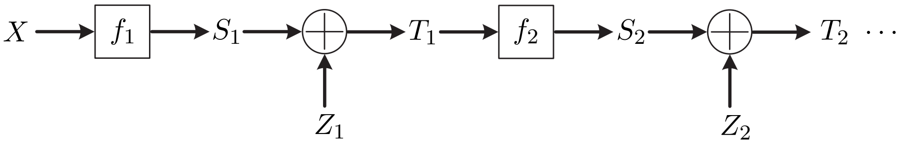


$\Longrightarrow$ Estimate $I(X;T_\ell)$ from samples via **general-purpose $h(P)$ est.:**

- Most results assume lower bounded density $\Longrightarrow$ **Inapplicable**
- **2 Works Drop Assumption:**

General-Purpose Differential Entropy Estimators

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

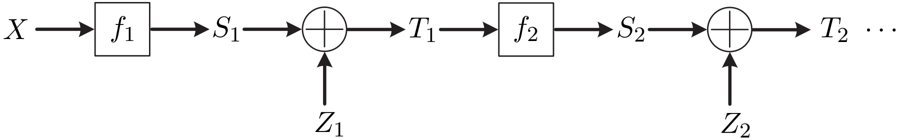


$\implies$ Estimate $I(X;T_\ell)$ from samples via **general-purpose $h(P)$ est.:**

- Most results assume lower bounded density $\implies$ **Inapplicable**
- **2 Works Drop Assumption:**
  1. KDE + Best poly. approximation [Han-Jiao-Weissman-Wu'17]

General-Purpose Differential Entropy Estimators

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

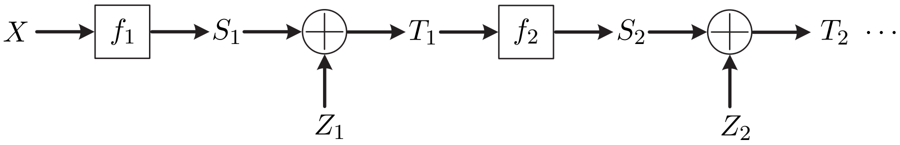


$\implies$ Estimate $I(X;T_\ell)$ from samples via **general-purpose $h(P)$ est.:**

- 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]

General-Purpose Differential Entropy Estimators

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

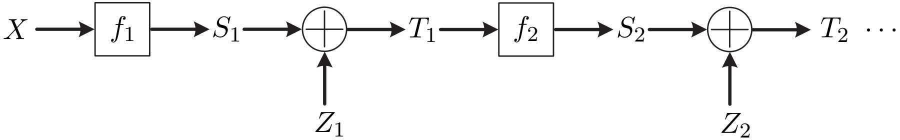


$\implies$ Estimate $I(X;T_\ell)$ from samples via **general-purpose $h(P)$ est.:**

- 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 & $s \in (0,2]$

General-Purpose Differential Entropy Estimators

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



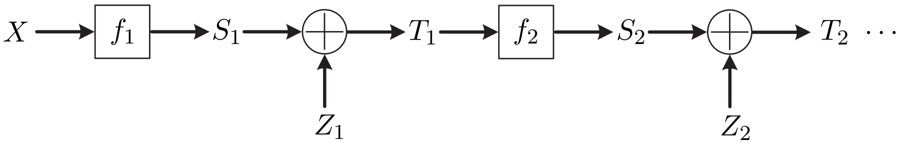
$\implies$ Estimate $I(X;T_\ell)$ from samples via **general-purpose $h(P)$ est.:**

- 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***

⋆ Except sub-Gaussian result from [Han-Jiao-Weissman-Wu'17]

General-Purpose Differential Entropy Estimators

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

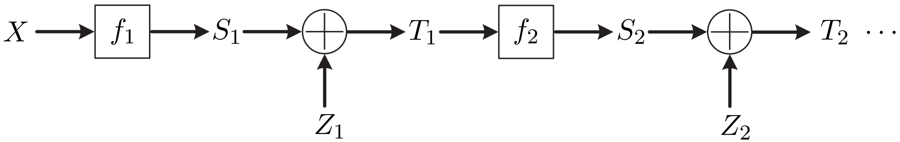


$\Longrightarrow$ Estimate $I(X;T_\ell)$ from samples via **general-purpose $h(P)$ est.:**

- Most results assume lower bounded density $\Longrightarrow$ **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 & $s \in (0,2] \Longrightarrow$ **Inapplicable***
- **Rate:** Risk $\leq O\left(n^{-\frac{\alpha s}{\beta s + d}}\right)$, w/ $\alpha, \beta \in \mathbb{N}$, s smoothness, d dimension

Exploit Structure - Ad Hoc Estimation

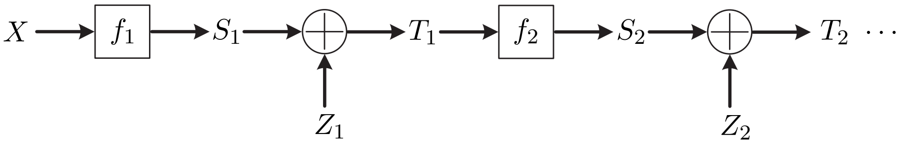
Noisy DNN: $S_\ell \triangleq f_\ell(T_{\ell-1}) \quad \Longrightarrow \quad T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \beta^2 \mathrm{I})$



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

Exploit Structure - Ad Hoc Estimation

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

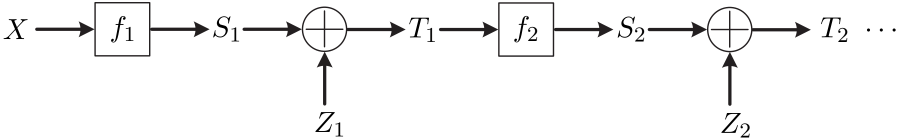


⊛ **Exploit structure:** We know $T_\ell = S_\ell + Z_\ell \sim P * \varphi$ and:

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

Exploit Structure - Ad Hoc Estimation

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

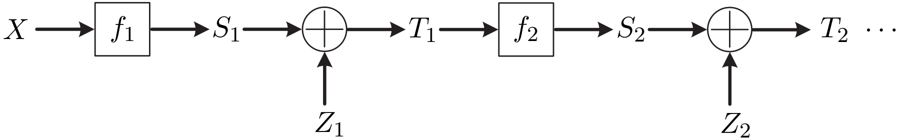


Exploit structure: We know $T_\ell = S_\ell + Z_\ell \sim P * \varphi$ and:

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

Exploit Structure - Ad Hoc Estimation

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



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

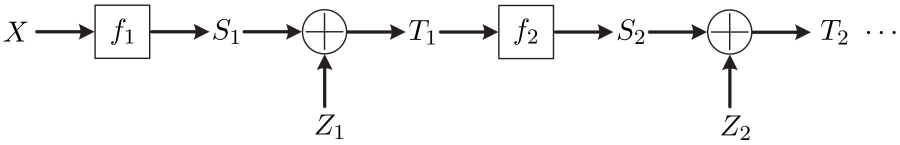
- **Genie1:** Sample $P = P_{S_\ell}$ and $P = P_{S_\ell | X = x_i}$ (sample $T_{\ell-1}$ & apply f_ℓ)
- **Genie2:** Know the distribution φ of Z_ℓ (noise injected by design)

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 φ (PDF of $\mathcal{N}(0, \beta^2 \mathrm{I}_d)$).

Exploit Structure - Ad Hoc Estimation

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



⊛ **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_ℓ)
- **Genie2:** Know the distribution φ of Z_ℓ (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 φ (PDF of $\mathcal{N}(0, \beta^2 \mathrm{I}_d)$).

Nonparametric Class: Depends on DNN architecture (nonlinearities)

The Sample Propagation Estimator

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

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

The Sample Propagation Estimator

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

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

⊛ **Curse of Dimensionality:** Sample complexity exponential in d

The Sample Propagation Estimator

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

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

⊛ **Curse of Dimensionality:** Sample complexity exponential in d

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

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

The Sample Propagation Estimator

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

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

⊛ **Curse of Dimensionality:** Sample complexity exponential in d

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

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

Comments:

The Sample Propagation Estimator

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

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

⊛ **Curse of Dimensionality:** Sample complexity exponential in d

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

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

Comments:

- **Plug-in:** $\hat{h}_{\mathsf{SP}}$ is just plug-in est. for the functional $\mathsf{T}_\varphi(P) \triangleq h(P*\varphi)$

The Sample Propagation Estimator

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

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

⊛ **Curse of Dimensionality:** Sample complexity exponential in d

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

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

Comments:

- **Plug-in:** $\hat{h}_{\mathsf{SP}}$ is just plug-in est. for the functional $\mathsf{T}_\varphi(P) \triangleq h(P*\varphi)$
- **Mixture:** $\hat{h}_{\mathsf{SP}}$ 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

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

⊛ **Curse of Dimensionality:** Sample complexity exponential in d

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

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

Comments:

- **Plug-in:** $\hat{h}_{\mathsf{SP}}$ is just plug-in est. for the functional $\mathsf{T}_\varphi(P) \triangleq h(P*\varphi)$
- **Mixture:** $\hat{h}_{\mathsf{SP}}$ 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 $\mathcal{F}_d \triangleq \left\{P \middle| \mathsf{supp}(P) \subseteq [-1,1]^d\right\}$ *and any* $\beta > 0$ *and* $d \geq 1$*, we have*

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

The Sample Propagation Estimator - Convergence

Theorem (ZG-Greenewald-Polyanskiy '18)

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

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

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

$$\leq \frac{1}{2(4\pi\beta^2)^{\frac{d}{4}}}\log\left(\frac{n\left(2+2\beta\sqrt{(2+\epsilon)\log n}\right)^d}{(\pi\beta^2)^{\frac{d}{2}}}\right)\left(2+2\beta\sqrt{(2+\epsilon)\log n}\right)^{\frac{d}{2}}\frac{1}{\sqrt{n}}$$

$$+\left(c_{\beta,d}^2 + \frac{2c_{\beta,d}d(1+\beta^2)}{\beta^2} + \frac{8d(d+2\beta^4+d\beta^4)}{\beta^4}\right)\frac{2}{n}$$

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

The Sample Propagation Estimator - Convergence

Theorem (ZG-Greenewald-Polyanskiy '18)

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

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

Pf. Technique:

The Sample Propagation Estimator - Convergence

Theorem (ZG-Greenewald-Polyanskiy '18)

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

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

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

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

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

- **Inside R:** Modulus of cont. & Convex analysis & Functional opt.

The Sample Propagation Estimator - Convergence

Theorem (ZG-Greenewald-Polyanskiy '18)

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

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

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

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

The Sample Propagation Estimator - Convergence

Theorem (ZG-Greenewald-Polyanskiy '18)

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

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

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

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

Comments:

The Sample Propagation Estimator - Convergence

Theorem (ZG-Greenewald-Polyanskiy '18)

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

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

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

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

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

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

- **Inside 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
- **Explicit expression** enables **concrete error bounds** in simulations

The Sample Propagation Estimator - Convergence

Theorem (ZG-Greenewald-Polyanskiy '18)

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

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

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

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

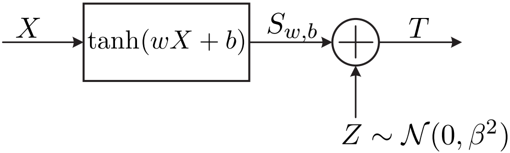
Comments:

- **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.)

Back to Noisy DNNs

$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

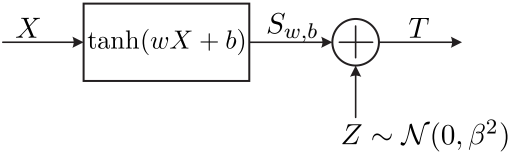
Single Neuron Classification:



$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

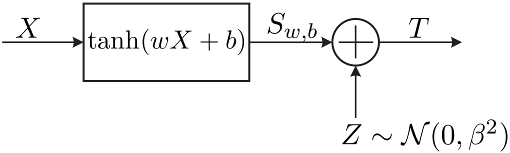
- **Input:** $X \sim \mathsf{Unif}\big(\mathcal{X}_{-1} \cup \mathcal{X}_1\big)$
  $\mathcal{X}_{-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_1 \triangleq \{3\}$

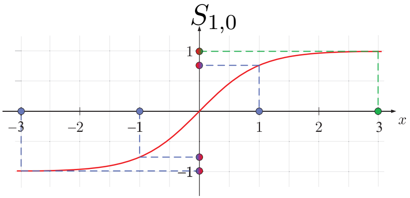


$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

- **Input:** $X \sim \mathsf{Unif}\left(\mathcal{X}_{-1} \cup \mathcal{X}_1\right)$
  $\mathcal{X}_{-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_1 \triangleq \{3\}$



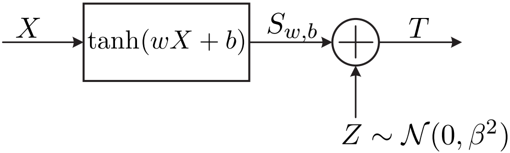


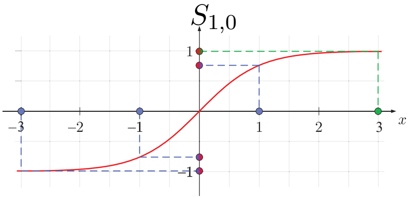
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

- **Input:** $X \sim \mathsf{Unif}(\mathcal{X}_{-1} \cup \mathcal{X}_1)$

  $\mathcal{X}_{-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_1 \triangleq \{3\}$





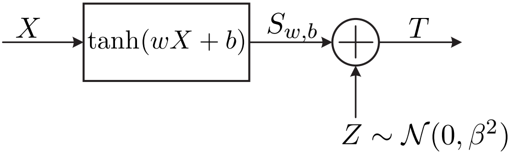
⊛ Center & sharpen transition ($\Longleftrightarrow$ increase w and keep $b = -2w$)

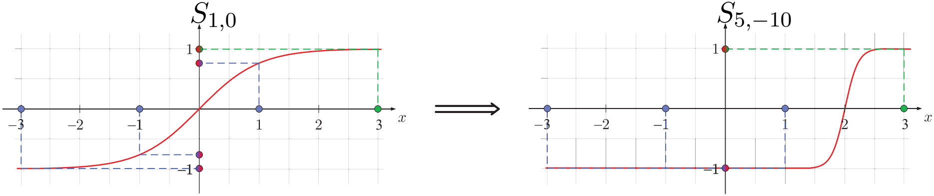
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

- **Input:** $X \sim \mathsf{Unif}\big(\mathcal{X}_{-1} \cup \mathcal{X}_1\big)$

  $\mathcal{X}_{-1} \triangleq \{-3, -1, 1\}$, $\mathcal{X}_1 \triangleq \{3\}$



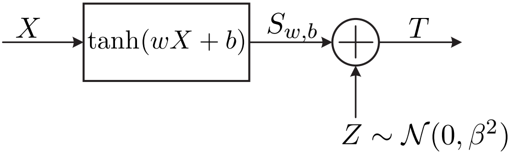


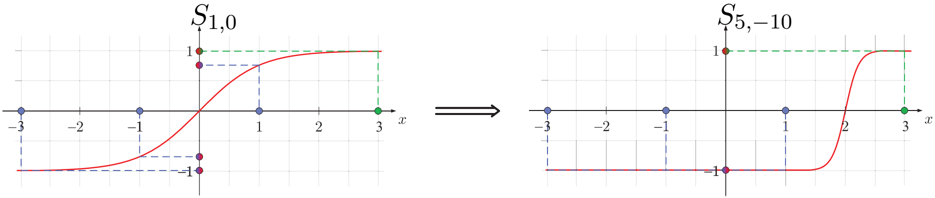
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

- **Input:** $X \sim \text{Unif}\left(\mathcal{X}_{-1} \cup \mathcal{X}_1\right)$

  $\mathcal{X}_{-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_1 \triangleq \{3\}$



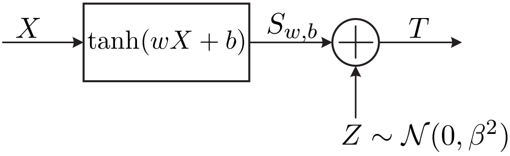


✓ Correct classification performance

$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

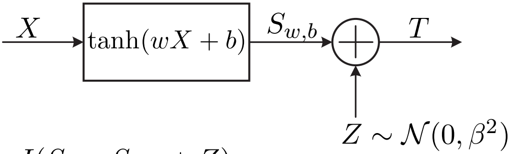
Single Neuron Classification:

- **Input:** $X \sim \mathsf{Unif}\left(\mathcal{X}_{-1} \cup \mathcal{X}_1\right)$
  $\mathcal{X}_{-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_1 \triangleq \{3\}$
- **Mutual Information:**



$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:



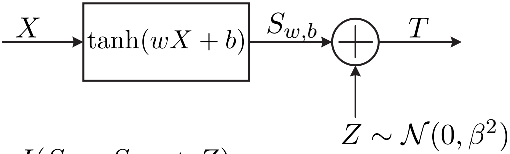
- **Input:** $X \sim \mathsf{Unif}\big(\mathcal{X}_{-1} \cup \mathcal{X}_1\big)$

  $\mathcal{X}_{-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_1 \triangleq \{3\}$

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

$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

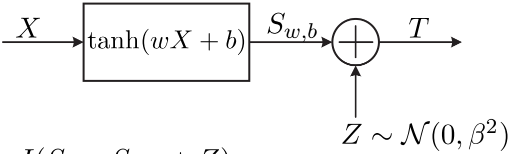


- **Input:** $X \sim \mathsf{Unif}(\mathcal{X}_{-1} \cup \mathcal{X}_1)$
  $\mathcal{X}_{-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_1 \triangleq \{3\}$
- **Mutual Information:** $I(X;T) = I(S_{w,b}; S_{w,b}+Z)$

$\Longrightarrow$ $I(X;T)$ is # bits (nats) transmittable over AWGN w. symbols

$\mathcal{S}_{w,b} \triangleq \{\tanh(-3w+b), \tanh(-w+b), \tanh(w+b), \tanh(3w+b)\}$

$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example



Single Neuron Classification:

- **Input:** $X \sim \text{Unif}(\mathcal{X}_{-1} \cup \mathcal{X}_1)$

  $\mathcal{X}_{-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_1 \triangleq \{3\}$

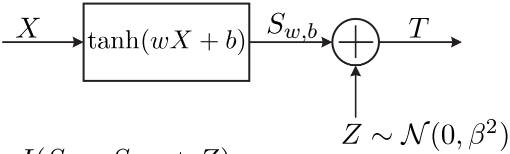
- **Mutual Information:** $I(X;T) = I(S_{w,b}; S_{w,b}+Z)$

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

$$\mathcal{S}_{w,b} \triangleq \{\tanh(-3w+b), \tanh(-w+b), \tanh(w+b), \tanh(3w+b)\} \longrightarrow \{\pm 1\}$$

$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:



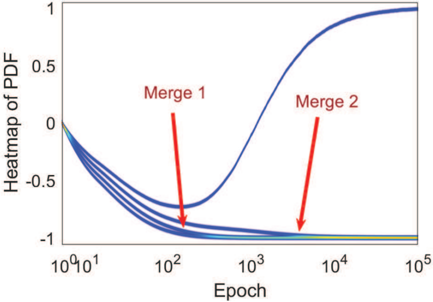
- **Input:** $X \sim \mathsf{Unif}(\mathcal{X}_{-1} \cup \mathcal{X}_1)$

  $\mathcal{X}_{-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_1 \triangleq \{3\}$

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

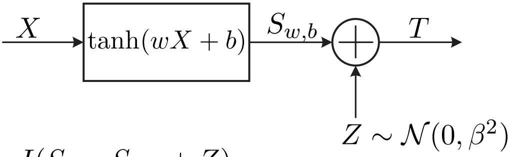
$\Longrightarrow$ $I(X;T)$ is # bits (nats) transmittable over AWGN w. symbols

$$\mathcal{S}_{w,b} \triangleq \{\tanh(-3w+b), \tanh(-w+b), \tanh(w+b), \tanh(3w+b)\} \longrightarrow \{\pm 1\}$$



$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:



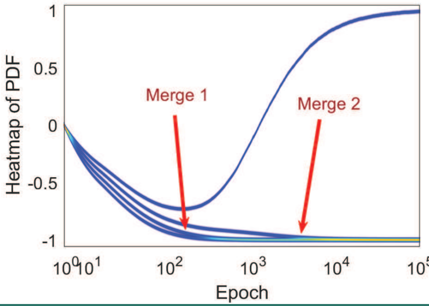
- **Input:** $X \sim \text{Unif}(\mathcal{X}_{-1} \cup \mathcal{X}_1)$

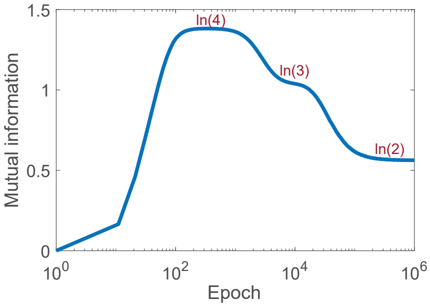
  $\mathcal{X}_{-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_1 \triangleq \{3\}$

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

$\Longrightarrow$ $I(X;T)$ is # bits (nats) transmittable over AWGN w. symbols

$$\mathcal{S}_{w,b} \triangleq \{\tanh(-3w+b), \tanh(-w+b), \tanh(w+b), \tanh(3w+b)\} \longrightarrow \{\pm 1\}$$





Clustering of Representations - Larger Networks

Noisy version of DNN from [Schwartz-Ziv&Tishby'17]:

Clustering of Representations - Larger Networks

Noisy version of DNN from [Schwartz-Ziv&Tishby'17]:

- **Binary Classification:** 12-bit input & 12–**10–7–5–4–3**–2 MLP arch.

Clustering of Representations - Larger Networks

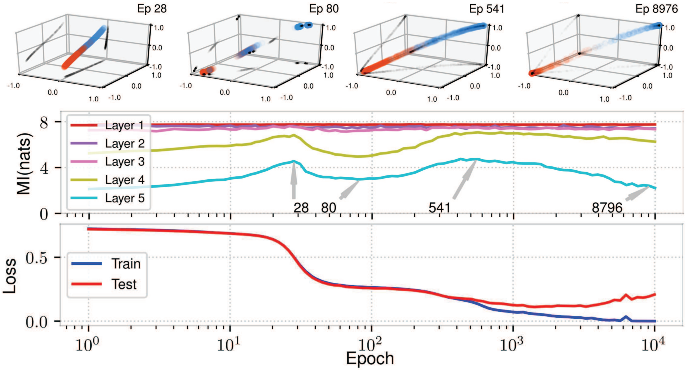
Noisy version of DNN from [Schwartz-Ziv&Tishby'17]:

- **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 [Schwartz-Ziv&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

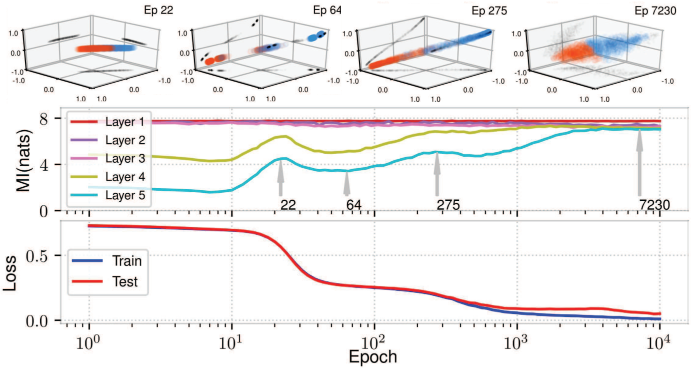
Noisy version of DNN from [Schwartz-Ziv&Tishby'17]:

- **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 [Schwartz-Ziv&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

Clustering of Representations - Larger Networks

Noisy version of DNN from [Schwartz-Ziv&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 [Schwartz-Ziv&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

$\Longrightarrow$ 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

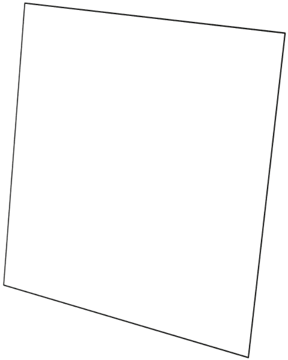
- $I(X;T_\ell)$ is constant $\implies$ Doesn't measure clustering
- Alternative measures for clustering (det. and noisy DNNs):
  - Scatter plots (up to 3D layers)

Circling back to Deterministic DNNs

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

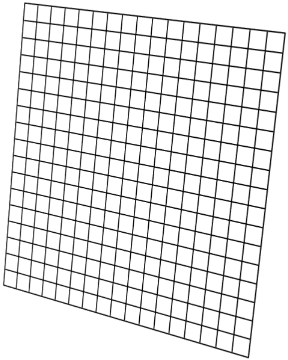
Circling back to Deterministic DNNs

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



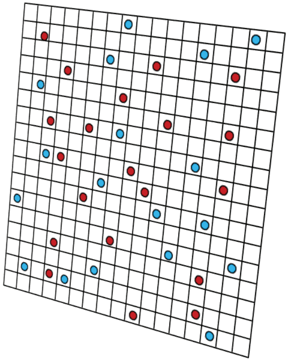
Circling back to Deterministic DNNs

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



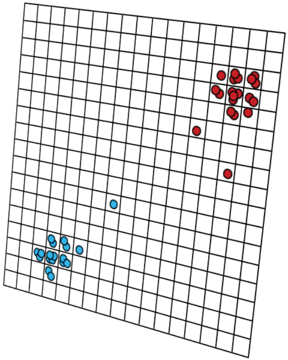
Circling back to Deterministic DNNs

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



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 $H\big(\mathrm{Bin}(T_\ell)\big) \downarrow$



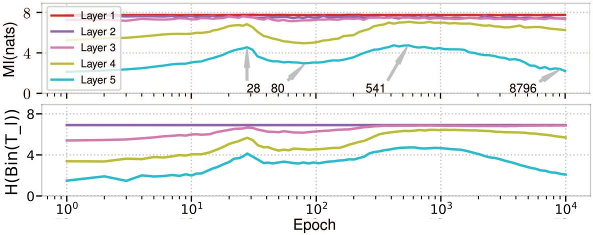
Circling back to Deterministic DNNs

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

  ✗ Incapable of accurately estimating MI values

  ✓ 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 $H\big(\mathsf{Bin}(T_\ell)\big)$
- **Noisy DNNs:** $I(X;T_\ell)$ and $H\big(\mathsf{Bin}(T_\ell)\big)$ highly correlated!*
- **Det. DNNs:** $H\big(\mathsf{Bin}(T_\ell)\big) = I\big(X;\mathsf{Bin}(T_\ell)\big)$ 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

Summary

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

Summary

- **Reexamined Information Bottleneck Compression:**
  - $I(X;T)$ fluctuations in det. DNNs are theoretically impossible
  - Yet, past works presented $I(X;T)$ 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 $I(X;T)$ dynamics during training
- **Noisy DNN Framework:** Studying IT quantities over DNNs
  - SP estimator for accurate MI estimation over this framework

Summary

- **Reexamined Information Bottleneck Compression:**
  - $I(X;T)$ fluctuations in det. DNNs are theoretically impossible
  - Yet, past works presented $I(X;T)$ dynamics during training
- **Noisy DNN Framework:** Studying IT quantities over DNNs
  - SP 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 $I(X;T)$ dynamics during training
- **Noisy DNN Framework:** Studying IT quantities over DNNs
  - SP 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 $I(X;T)$ dynamics during training
- **Noisy DNN Framework:** Studying IT quantities over DNNs
  - SP 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

$\Longrightarrow$ **Clustering** is the common phenomenon of interest!

Future Research

- **Curse of Dimensionality:** Track clustering in high-dimensions?

Future Research

- **Curse of Dimensionality:** Track clustering in high-dimensions?
  - Lower-dimensional embedding

Future Research

- **Curse of Dimensionality:** Track clustering in high-dimensions?
  - Lower-dimensional embedding
  - Summarizing statistics

Future Research

- **Curse of Dimensionality:** Track clustering in high-dimensions?
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability measures [Czumaj-Peng-Sohler'15]

Future Research

- **Curse of Dimensionality:** Track clustering in high-dimensions?
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability measures [Czumaj-Peng-Sohler'15]
- **The Role of Compression:**

Future Research

- **Curse of Dimensionality:** Track clustering in high-dimensions?
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability measures [Czumaj-Peng-Sohler'15]
- **The Role of Compression:**
  - Is compression necessary? Desirable?

Future Research

- **Curse of Dimensionality:** Track clustering in high-dimensions?
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability measures [Czumaj-Peng-Sohler'15]
- **The Role of Compression:**
  - Is compression necessary? Desirable?
  - Design tool for DNN architectures

Future Research

- **Curse of Dimensionality:** Track clustering in high-dimensions?
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability measures [Czumaj-Peng-Sohler'15]
- **The Role of Compression:**
  - Is compression necessary? Desirable?
  - Design tool for DNN architectures
- **Algorithmic Perspective:**

Future Research

- **Curse of Dimensionality:** Track clustering in high-dimensions?
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability measures [Czumaj-Peng-Sohler'15]
- **The Role of Compression:**
  - Is compression necessary? Desirable?
  - Design tool for DNN architectures
- **Algorithmic Perspective:**
  - Better understanding of internal representation evolution & final state

Future Research

- **Curse of Dimensionality:** Track clustering in high-dimensions?
  - Lower-dimensional embedding
  - Summarizing statistics
  - Graph clusterability measures [Czumaj-Peng-Sohler'15]
- **The Role of Compression:**
  - Is compression necessary? Desirable?
  - Design tool for DNN architectures
- **Algorithmic Perspective:**
  - Better understanding of internal representation evolution & final state
  - Enhanced DNN training algorithms

The Sample Propagation Estimator - Proof Ideas

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

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

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

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

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

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

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

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

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

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

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

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

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

    $\implies$ Focus on analyzing $\mathbb{E}\left|(P*\varphi)(x) - (\hat{P}_n*\varphi)(x)\right|$
  - Bias & variance analysis

The Sample Propagation Estimator - Proof Ideas

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

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

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

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

    $\Longrightarrow$ Focus on analyzing $\mathbb{E}\left|(P*\varphi)(x) - (\hat{P}_n*\varphi)(x)\right|$
  - Bias & variance analysis

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

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

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

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

    $\Longrightarrow$ Focus on analyzing $\mathbb{E}\left|(P*\varphi)(x) - (\hat{P}_n*\varphi)(x)\right|$
  - Bias & variance analysis

    $\Longrightarrow$ $\mathbb{E}\left|(P*\varphi)(x) - (\hat{P}_n*\varphi)(x)\right| \leq c_1\sqrt{\frac{(P*\tilde{\varphi})(x)}{n}}, \;\; \tilde{\varphi} = \mathcal{N}\left(0, \frac{\beta^2}{2}\mathrm{I}\right)$
  - 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:** $h_{\mathcal{R}}(p) \triangleq \mathbb{E}\left[-\log p(X) \mathbb{1}_{\{X \in \mathcal{R}\}}\right]$

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

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

    $\Longrightarrow$ Focus on analyzing $\mathbb{E}\left|(P*\varphi)(x) - (\hat{P}_n*\varphi)(x)\right|$
  - Bias & variance analysis

    $$\Longrightarrow \mathbb{E}\left|(P*\varphi)(x) - (\hat{P}_n*\varphi)(x)\right| \leq c_1\sqrt{\frac{(P*\tilde{\varphi})(x)}{n}}, \quad \tilde{\varphi} = \mathcal{N}\left(0, \frac{\beta^2}{2}\mathrm{I}\right)$$
  - Plug back in & Convex analysis

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

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

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

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

    $\Longrightarrow$ Focus on analyzing $\mathbb{E}\left|(P*\varphi)(x) - (\hat{P}_n*\varphi)(x)\right|$
  - Bias & variance analysis

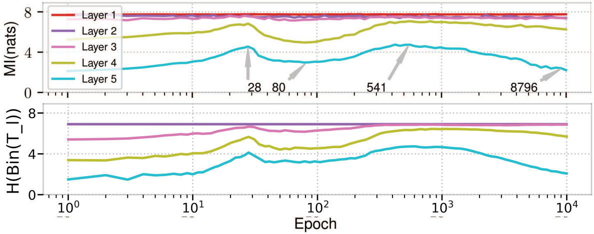
    $$\Longrightarrow \mathbb{E}\left|(P*\varphi)(x) - (\hat{P}_n*\varphi)(x)\right| \leq c_1\sqrt{\frac{(P*\tilde{\varphi})(x)}{n}}, \quad \tilde{\varphi} = \mathcal{N}\left(0, \frac{\beta^2}{2}\mathrm{I}\right)$$
  - Plug back in & Convex analysis

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

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

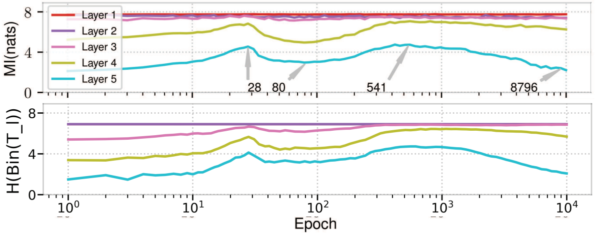
Binning vs True Mutual Information

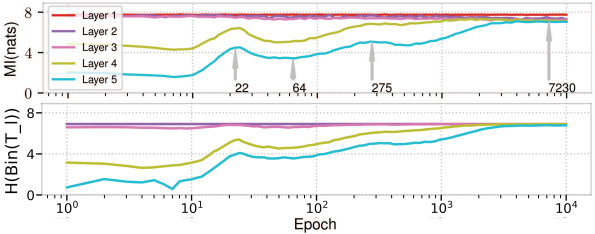
Comparing to Previously Shown MI Plots:



Binning vs True Mutual Information

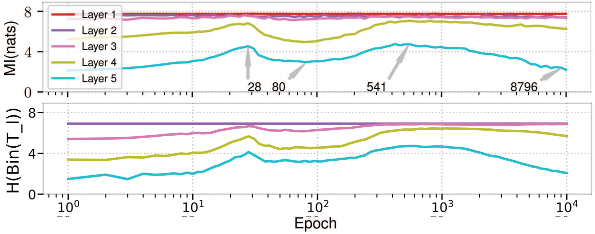
Comparing to Previously Shown MI Plots:

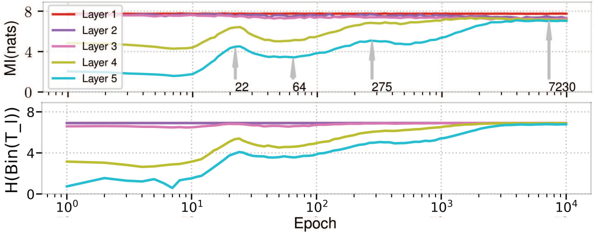




Binning vs True Mutual Information

Comparing to Previously Shown MI Plots:





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

References

[1] Z. Goldfeld, E. van den Berg, K. Greenewald, I. Melnyk, N. Nguyen, B. Kingsbury and Y. Polyanskiy, “Estimating information flow in DNNs,” Submitted to the *International Conference on Learning Representations (ICLR-2019)*, New Orleans, Louisiana, US, May 2019.
Arxiv (extended): `https://arxiv.org/abs/1810.05728`

[2] Z. Goldfeld, K. Greenewald and Y. Polyanskiy, “Estimating differential entropy under Gaussian convolutions,” Submitted to the *IEEE Transactions on Information Theory*, October 2018.
Arxiv: `https://arxiv.org/abs/1810.11589`