Estimating Information Flow in Deep Neural Networks

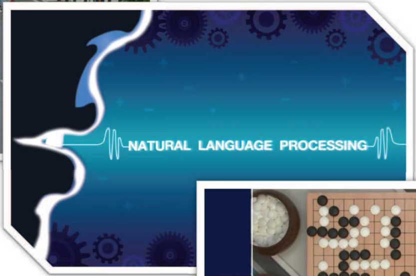
Ziv Goldfeld, Ewout van den Berg, Kristjan Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury and Yury Polyanskiy

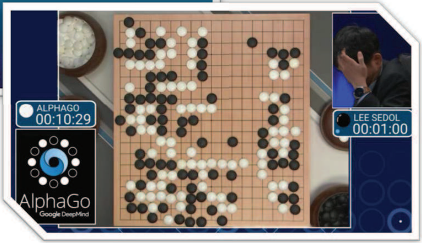
MIT, IBM Research, MIT-IBM Watson AI Lab

International Conference on Machine Learning

June 12th, 2019

Deep Learning - What's Under the Hood?





Deep Learning - What's Under the Hood?

- **Lacking Theory:** Macroscopic understanding of Deep Learning

Deep Learning - What's Under the Hood?

- **Lacking Theory:** Macroscopic understanding of Deep Learning
  - ? What drives the evolution of internal representations?

Deep Learning - What's Under the Hood?

- **Lacking Theory:** Macroscopic understanding of Deep Learning
  - What drives the evolution of internal representations?
  - What are properties of learned representations?

Deep Learning - What's Under the Hood?

- **Lacking Theory:** Macroscopic understanding of Deep Learning
  - What drives the evolution of internal representations?
  - What are properties of learned representations?
  - How do fully trained networks process information?

Deep Learning - What's Under the Hood?

- **Lacking Theory:** Macroscopic understanding of Deep Learning
  - ? What drives the evolution of internal representations?
  - ? What are properties of learned representations?
  - ? How do fully trained networks process information?

- **Attempts to Understand Effectiveness of DL:**
  - Structure of loss landscape
    [Saxe *et al.*'14, Choromanska *et al.*'15, Kawaguchi'16, Keskar *et al.*'17]
  - Wavelets and sparse coding
    [Bruna-Mallat'13, Giryes *et al.*'16, Papyan *et al.*'16]
  - Adversarial examples
    [Szegedy *et al.*'14, Nguyen *et al.*'17, Liu *et al.*'16, Cisse *et al.*'16]
  - Information Bottleneck Theory
    [Tishby-Zaslavsky'15, Shwartz-Tishby'17, Saxe *et al.*'18, Gabrié *et al.*'18]

Deep Learning - What's Under the Hood?

- **Lacking Theory:** Macroscopic understanding of Deep Learning
  - ? What drives the evolution of internal representations?
  - ? What are properties of learned representations?
  - ? How do fully trained networks process information?

- **Attempts to Understand Effectiveness of DL:**
  - Structure of loss landscape
    [Saxe *et al.*'14, Choromanska *et al.*'15, Kawaguchi'16, Keskar *et al.*'17]
  - Wavelets and sparse coding
    [Bruna-Mallat'13, Giryes *et al.*'16, Papyan *et al.*'16]
  - Adversarial examples
    [Szegedy *et al.*'14, Nguyen *et al.*'17, Liu *et al.*'16, Cisse *et al.*'16]
  - **Information Bottleneck Theory**
    [Tishby-Zaslavsky'15, Shwartz-Tishby'17, Saxe *et al.*'18, Gabrié *et al.*'18]

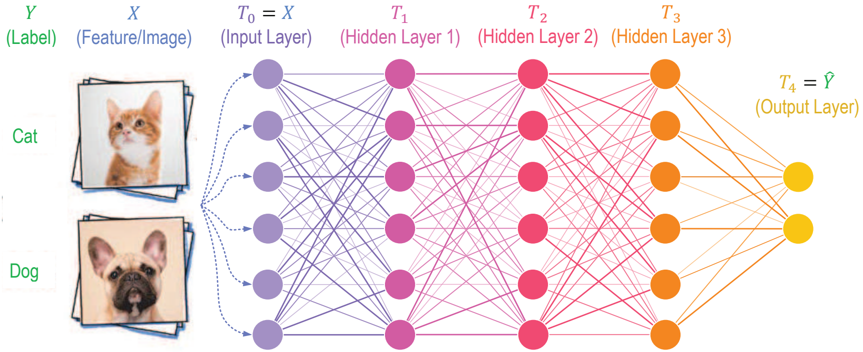
Deep Learning - What's Under the Hood?

- **Lacking Theory:** Macroscopic understanding of Deep Learning
  - ? What drives the evolution of internal representations?
  - ? What are properties of learned representations?
  - ? How do fully trained networks process information?

- **Attempts to Understand Effectiveness of DL:**
  - Structure of loss landscape
    [Saxe *et al.*'14, Choromanska *et al.*'15, Kawaguchi'16, Keskar *et al.*'17]
  - Wavelets and sparse coding
    [Bruna-Mallat'13, Giryes *et al.*'16, Papyan *et al.*'16]
  - Adversarial examples
    [Szegedy *et al.*'14, Nguyen *et al.*'17, Liu *et al.*'16, Cisse *et al.*'16]
  - **Information Bottleneck Theory**
    [Tishby-Zaslavsky'15, Shwartz-Tishby'17, Saxe *et al.*'18, Gabrié *et al.*'18]

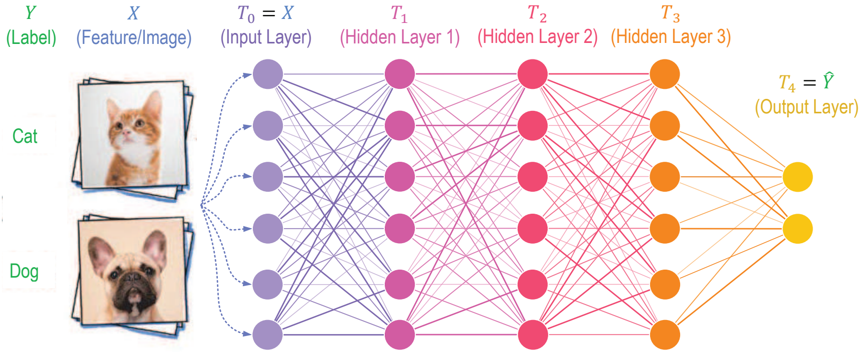
★ **Goal:** Mathematically analyze IB theory & test 'Compression'

(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$



Setup and Preliminaries

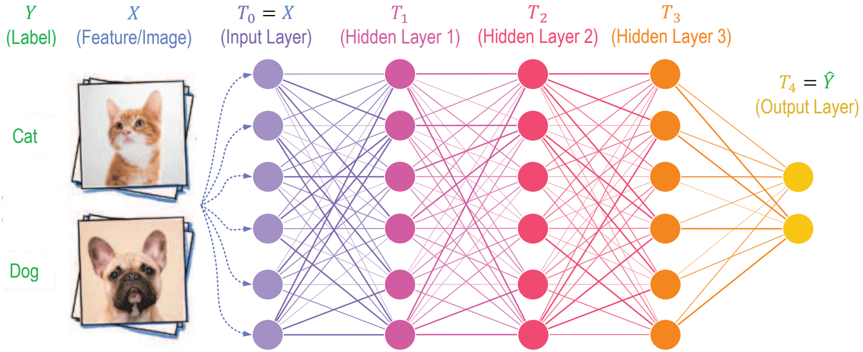
(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$



- **Joint Distribution:** $P_{X,Y}$

Setup and Preliminaries

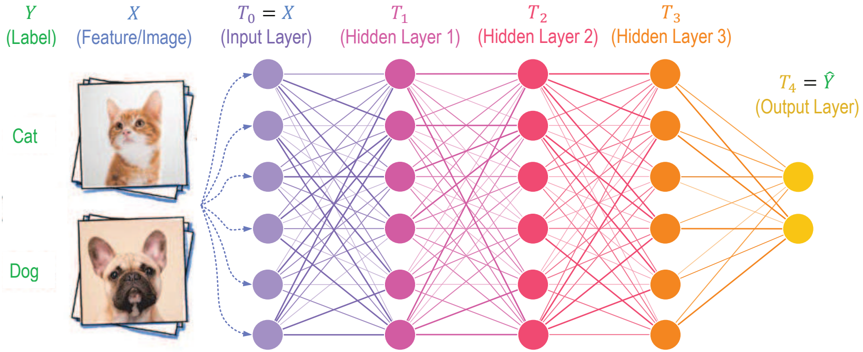
(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$



- **Joint Distribution:** $P_{X,Y} \implies P_{X,Y} \cdot P_{T_1,\ldots,T_L|X}$

Setup and Preliminaries

(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$

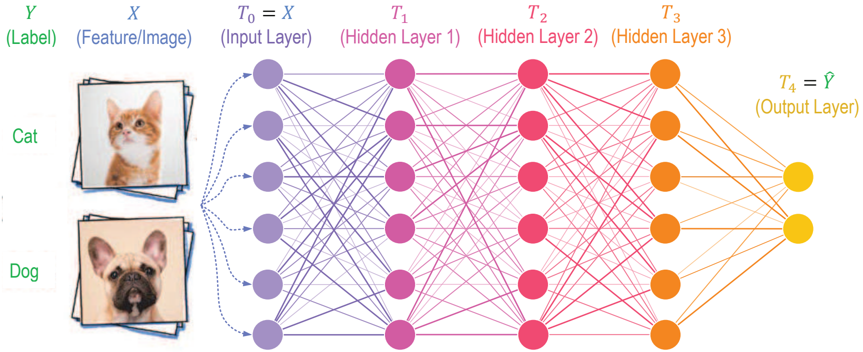


- **Joint Distribution:** $P_{X,Y} \quad \Longrightarrow \quad P_{X,Y} \cdot P_{T_1,\ldots,T_L|X}$
- **Information Plane:** Evolution of $\big(I(X;T_\ell), I(Y;T_\ell)\big)$ during training

$$\left[I(A;B) = \mathsf{D}_{\mathsf{KL}}(P_{A,B} \| P_A \otimes P_B) \overset{\text{Discrete}}{=} \sum_{a,b} P_{A,B}(a,b) \log \frac{P_{A,B}(a,b)}{P_A(a) P_B(b)} \right]$$

Setup and Preliminaries

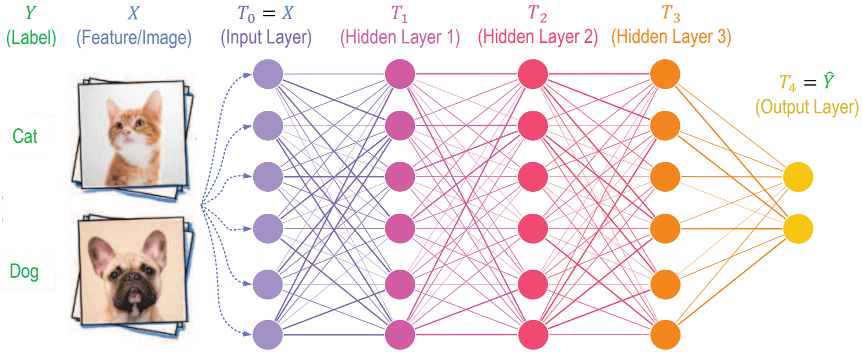
(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$



IB Theory Claim: Training comprises 2 phases

Setup and Preliminaries

(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$

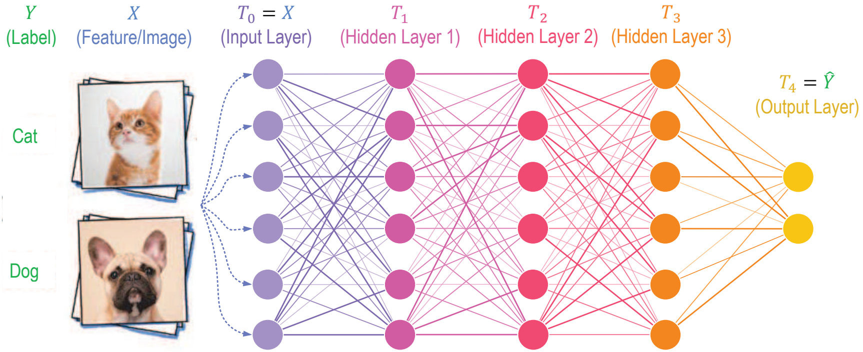


IB Theory Claim: Training comprises 2 phases

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

Setup and Preliminaries

(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$

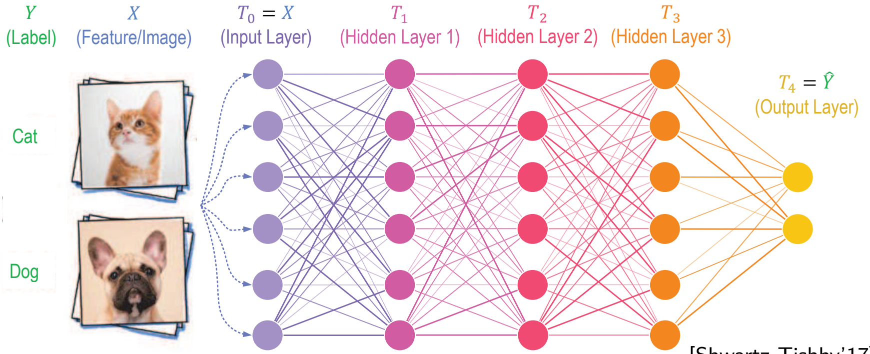


IB Theory Claim: Training comprises 2 phases

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

Setup and Preliminaries

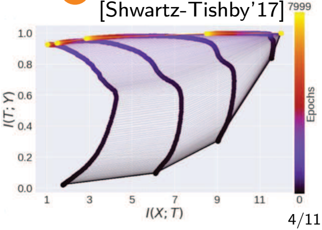
(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$



IB Theory Claim: Training comprises 2 phases

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

[Shwartz-Tishby'17]



Vacuous Mutual Information & Mis-Estimation

Proposition (Informal)

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

Vacuous Mutual Information & Mis-Estimation

Proposition (Informal)

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

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

Vacuous Mutual Information & Mis-Estimation

Proposition (Informal)

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

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

- $I(X;T_\ell)$ a.s. **infinite** (continuous X) or **constant** $H(X)$ (discrete X)

Vacuous Mutual Information & Mis-Estimation

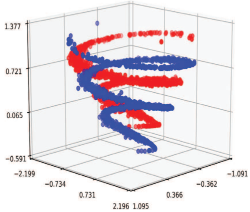
Proposition (Informal)

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

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

- $I(X;T_\ell)$ a.s. **infinite** (continuous X) or **constant** $H(X)$ (discrete X)

Feature Space (X)



$X \sim \text{Unif}(\mathcal{X})$

$|\mathcal{X}| = 3000$

Vacuous Mutual Information & Mis-Estimation

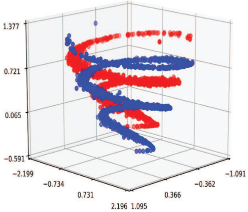
Proposition (Informal)

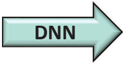
Det. DNNs with strictly monotone nonlinearities (e.g., tanh or sigmoid)

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

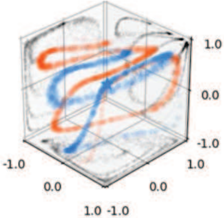
- $I(X;T_\ell)$ a.s. **infinite** (continuous X) or **constant** $H(X)$ (discrete X)

Feature Space (X)





Internal Rep. Space ($T_\ell = \tilde{f}_\ell(X)$)



$X \sim \text{Unif}(\mathcal{X})$

$|\mathcal{X}| = 3000$

$T_\ell \sim \text{Unif}(\mathcal{T}_\ell)$

$|\mathcal{T}_\ell| = |\mathcal{X}| = 3000$

Vacuous Mutual Information & Mis-Estimation

Proposition (Informal)

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

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

- $I(X;T_\ell)$ a.s. **infinite** (continuous X) or **constant** $H(X)$ (discrete X)
- **Past Works:** Use binning-based proxy of $I(X;T_\ell)$ (aka quantization)

Vacuous Mutual Information & Mis-Estimation

Proposition (Informal)

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

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

- $I(X;T_\ell)$ a.s. **infinite** (continuous X) or **constant** $H(X)$ (discrete X)
- **Past Works:** Use binning-based proxy of $I(X;T_\ell)$ (aka quantization)
  1. For non-negligible bin size $I\big(X;\mathrm{Bin}(T_\ell)\big) \neq I(X;T_\ell)$

Vacuous Mutual Information & Mis-Estimation

Proposition (Informal)

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

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

- $I(X;T_\ell)$ a.s. **infinite** (continuous X) or **constant** $H(X)$ (discrete X)
- **Past Works:** Use binning-based proxy of $I(X;T_\ell)$ (aka quantization)
  1. For non-negligible bin size $I\big(X;\mathrm{Bin}(T_\ell)\big) \neq I(X;T_\ell)$
  2. $I\big(X;\mathrm{Bin}(T_\ell)\big)$ highly sensitive to user-defined bin size:

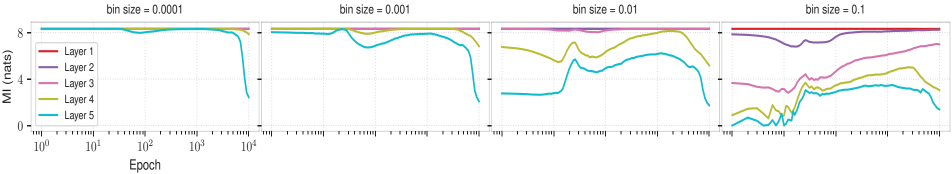
Vacuous Mutual Information & Mis-Estimation

Proposition (Informal)

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

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

- $I(X;T_\ell)$ a.s. **infinite** (continuous X) or **constant** $H(X)$ (discrete X)
- **Past Works:** Use binning-based proxy of $I(X;T_\ell)$ (aka quantization)
  1. For non-negligible bin size $I\big(X;\mathrm{Bin}(T_\ell)\big) \neq I(X;T_\ell)$
  2. $I\big(X;\mathrm{Bin}(T_\ell)\big)$ highly sensitive to user-defined bin size:



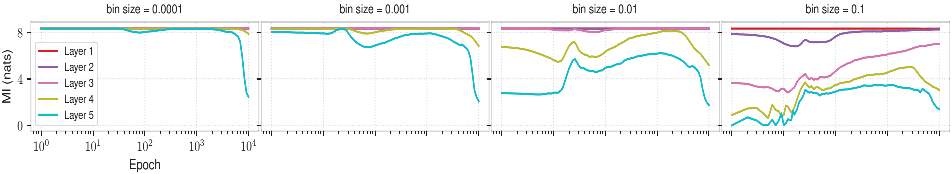
Vacuous Mutual Information & Mis-Estimation

Proposition (Informal)

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

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

- $I(X;T_\ell)$ a.s. **infinite** (continuous X) or **constant** $H(X)$ (discrete X)
- **Past Works:** Use binning-based proxy of $I(X;T_\ell)$ (aka quantization)
    1. For non-negligible bin size $I\big(X;\mathrm{Bin}(T_\ell)\big) \neq I(X;T_\ell)$
    2. $I\big(X;\mathrm{Bin}(T_\ell)\big)$ highly sensitive to user-defined bin size:



⊛ **Real Problem:** Mismatch between $I(X;T_\ell)$ measurement and model

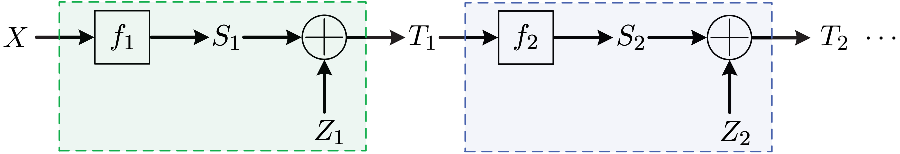
Auxiliary Framework - Noisy Deep Neural Networks

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

Auxiliary Framework - Noisy Deep Neural Networks

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

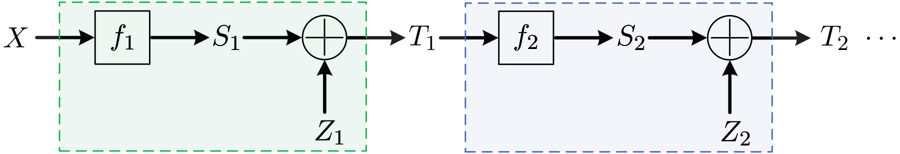
- **Formally:** $T_\ell = S_\ell + Z_\ell$, where $S_\ell \triangleq f_\ell(T_{\ell-1})$ and $Z_\ell \sim \mathcal{N}(0, \sigma^2 \mathrm{I}_d)$



Auxiliary Framework - Noisy Deep Neural Networks

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

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

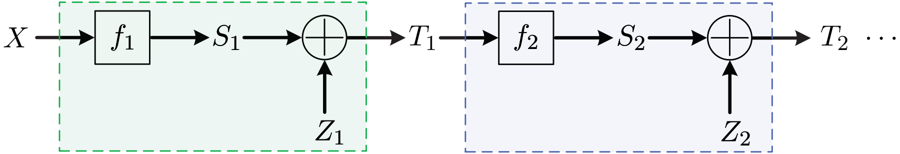


$\Longrightarrow$ $X \mapsto T_\ell$ is a **parametrized channel** (by DNN's parameters)

Auxiliary Framework - Noisy Deep Neural Networks

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

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



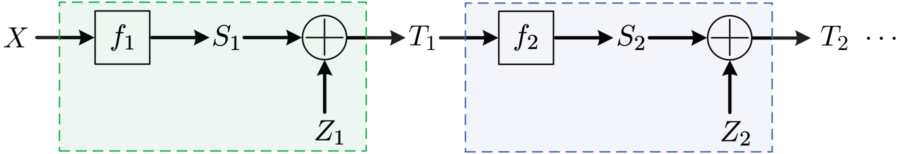
$\Longrightarrow$ $X \mapsto T_\ell$ is a **parametrized channel** (by DNN's parameters)

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

Auxiliary Framework - Noisy Deep Neural Networks

<u>**Modification:**</u> Inject (small) Gaussian noise to neurons' output

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



$\Longrightarrow$ $X \mapsto T_\ell$ is a **parametrized channel** (by DNN's parameters)

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

⊛ **Challenge:** How to accurately track $I(X;T_\ell)$?

High-Dim. & Nonparametric Functional Estimation

High-Dim. & Nonparametric Functional Estimation

Distill $I(X;T_\ell)$ Estimation into Noisy Differential Entropy Estimation:

Estimate $h(P * \mathcal{N}_\sigma)$ from n i.i.d. samples $S^n \triangleq (S_i)_{i=1}^n$ of $P \in \mathcal{F}_d$ (non-parametric class) and knowledge of $\mathcal{N}_\sigma$ (Gaussian measure $\mathcal{N}(0, \sigma^2 \mathrm{I}_d)$).

High-Dim. & Nonparametric Functional Estimation

Distill $I(X;T_\ell)$ Estimation into Noisy Differential Entropy Estimation:

Estimate $h(P * \mathcal{N}_\sigma)$ from n i.i.d. samples $S^n \triangleq (S_i)_{i=1}^n$ of $P \in \mathcal{F}_d$ (non-parametric class) and knowledge of $\mathcal{N}_\sigma$ (Gaussian measure $\mathcal{N}(0, \sigma^2 \mathrm{I}_d)$).

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

Sample complexity of any accurate estimator (additive gap η) is $\Omega\left(\frac{2^d}{\eta d}\right)$

High-Dim. & Nonparametric Functional Estimation

Distill $I(X;T_\ell)$ Estimation into Noisy Differential Entropy Estimation:

Estimate $h(P * \mathcal{N}_\sigma)$ from n i.i.d. samples $S^n \triangleq (S_i)_{i=1}^n$ of $P \in \mathcal{F}_d$ (nonparametric class) and knowledge of $\mathcal{N}_\sigma$ (Gaussian measure $\mathcal{N}(0, \sigma^2 \mathrm{I}_d)$).

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

Sample complexity of any accurate estimator (additive gap η) is $\Omega\left(\frac{2^d}{\eta d}\right)$

Structured Estimator*:** $\hat{h}(S^n, \sigma) \triangleq h(\hat{P}_n * \mathcal{N}_\sigma)$, where $\hat{P}_n = \frac{1}{n}\sum_{i=1}^{n} \delta_{S_i}$

$\star$ Efficient and parallelizable

High-Dim. & Nonparametric Functional Estimation

Distill $I(X;T_\ell)$ Estimation into Noisy Differential Entropy Estimation:

Estimate $h(P*\mathcal{N}_\sigma)$ from n i.i.d. samples $S^n \triangleq (S_i)_{i=1}^n$ of $P \in \mathcal{F}_d$ (non-parametric class) and knowledge of $\mathcal{N}_\sigma$ (Gaussian measure $\mathcal{N}(0,\sigma^2\mathrm{I}_d)$).

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

Sample complexity of any accurate estimator (additive gap η) is $\Omega\left(\frac{2^d}{\eta d}\right)$

Structured Estimator⋆: $\hat{h}(S^n,\sigma) \triangleq h(\hat{P}_n * \mathcal{N}_\sigma)$, where $\hat{P}_n = \frac{1}{n}\sum_{i=1}^n \delta_{S_i}$

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

For $\mathcal{F}_{d,K}^{(\mathsf{SG})} \triangleq \{P \,|\, P$ *is* K*-subgaussian in* $\mathbb{R}^d\}$, $d \geq 1$ *and* $\sigma > 0$, *we have*

$$\sup_{P\in\mathcal{F}_{d,K}^{(\mathsf{SG})}} \mathbb{E}_{S^n}\left|h(P*\mathcal{N}_\sigma) - \hat{h}(S^n,\sigma)\right| \leq c_{\sigma,K}^d \cdot n^{-\frac{1}{2}}$$

High-Dim. & Nonparametric Functional Estimation

Distill $I(X;T_\ell)$ Estimation into Noisy Differential Entropy Estimation:

Estimate $h(P * \mathcal{N}_\sigma)$ from n i.i.d. samples $S^n \triangleq (S_i)_{i=1}^n$ of $P \in \mathcal{F}_d$ (nonparametric class) and knowledge of $\mathcal{N}_\sigma$ (Gaussian measure $\mathcal{N}(0, \sigma^2 \mathrm{I}_d)$).

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

Sample complexity of any accurate estimator (additive gap η) is $\Omega\left(\frac{2^d}{\eta d}\right)$

Structured Estimator***:** $\hat{h}(S^n, \sigma) \triangleq h(\hat{P}_n * \mathcal{N}_\sigma)$, where $\hat{P}_n = \frac{1}{n}\sum_{i=1}^n \delta_{S_i}$

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

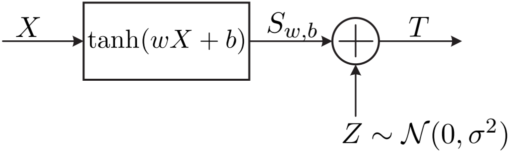
For $\mathcal{F}_{d,K}^{(\mathsf{SG})} \triangleq \{P \,|\, P$ *is* K*-subgaussian in* $\mathbb{R}^d\}$, $d \geq 1$ *and* $\sigma > 0$, *we have*

$$\sup_{P \in \mathcal{F}_{d,K}^{(\mathsf{SG})}} \mathbb{E}_{S^n} \left| h(P * \mathcal{N}_\sigma) - \hat{h}(S^n, \sigma) \right| \leq c_{\sigma,K}^d \cdot n^{-\frac{1}{2}}$$

Optimality: $\hat{h}(S^n, \sigma)$ attains sharp dependence on both n and d!

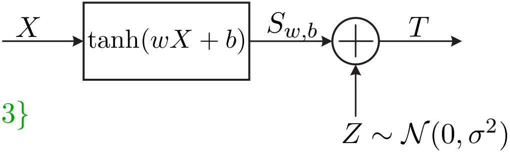
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:



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

Single Neuron Classification:



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

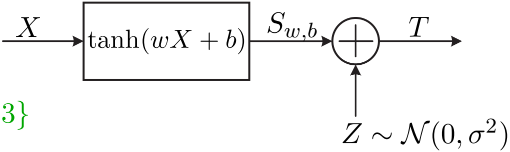
  $\mathcal{X}_{y=-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_{y=1} \triangleq \{3\}$

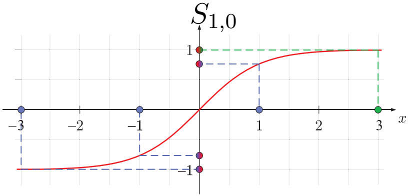
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

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

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



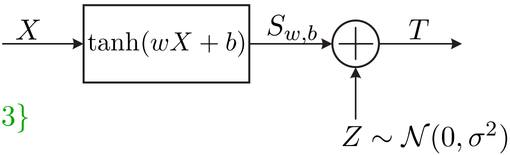


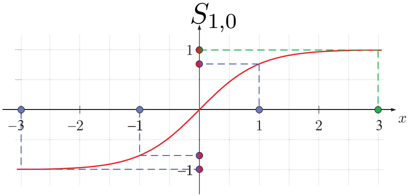
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

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

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



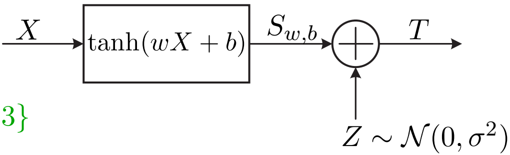


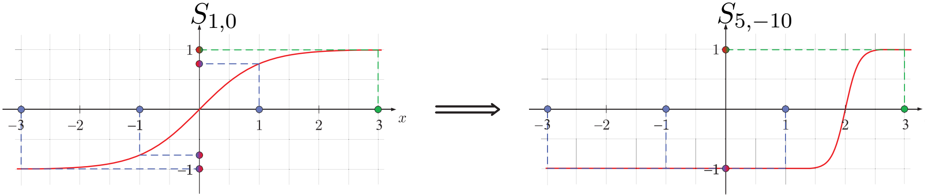
⊛ Center & sharpen transition ($\iff$ increase w and keep $b=-2w$)

Single Neuron Classification:

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

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



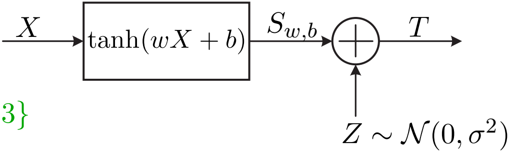


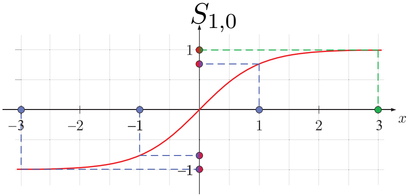
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

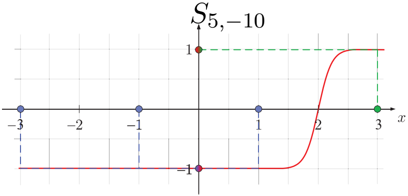
- **Input:** $X \sim \mathsf{Unif}\{\pm 1, \pm 3\}$

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





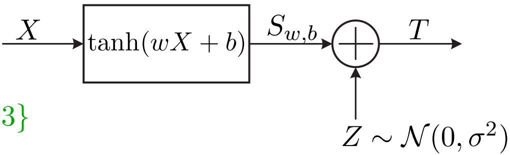
$\Longrightarrow$



✓ Correct classification performance

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

Single Neuron Classification:



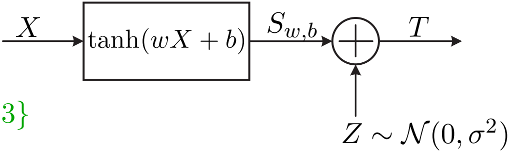
- **Input:** $X \sim \mathsf{Unif}\{\pm 1, \pm 3\}$

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

- **Mutual Information:**

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

Single Neuron Classification:



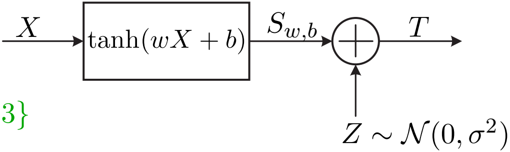
- **Input:** $X \sim \mathsf{Unif}\{\pm 1, \pm 3\}$

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

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

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

Single Neuron Classification:



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

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

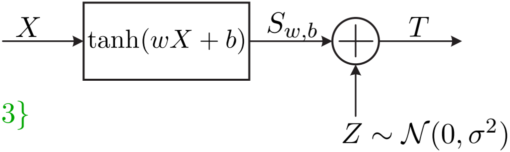
- **Mutual Information:** $I(X;T) = I(S_{w,b}; S_{w,b} + Z)$

$\Longrightarrow$ $I(X;T)$ is # bits (nats) transmittable over AWGN with 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:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$

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

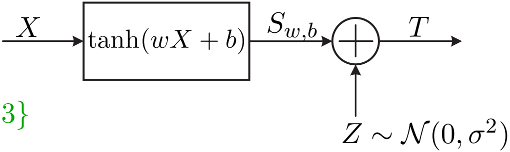
- **Mutual Information:** $I(X;T) = I(S_{w,b}; S_{w,b}+Z)$

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



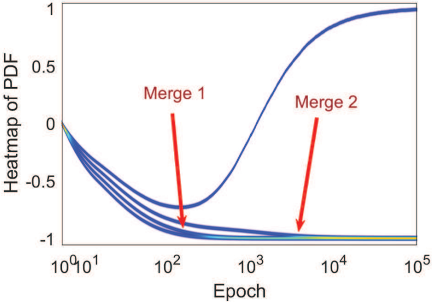
- **Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$

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

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

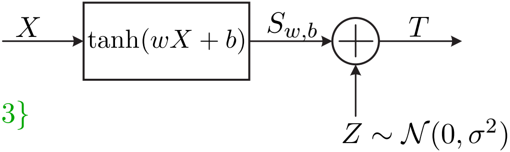
$\Longrightarrow$ $I(X;T)$ is # bits (nats) transmittable over AWGN with symbols

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



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

Single Neuron Classification:



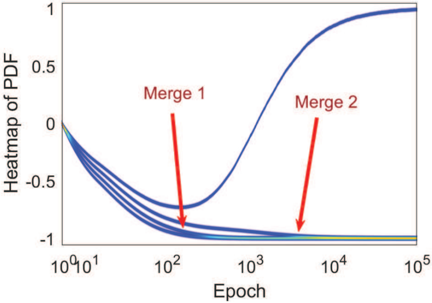
- **Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$

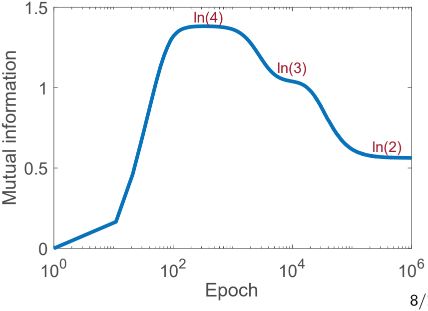
  $\mathcal{X}_{y=-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_{y=1} \triangleq \{3\}$

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

$\Longrightarrow$ $I(X;T)$ is # bits (nats) transmittable over AWGN with 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 [Shwartz-Tishby'17]:

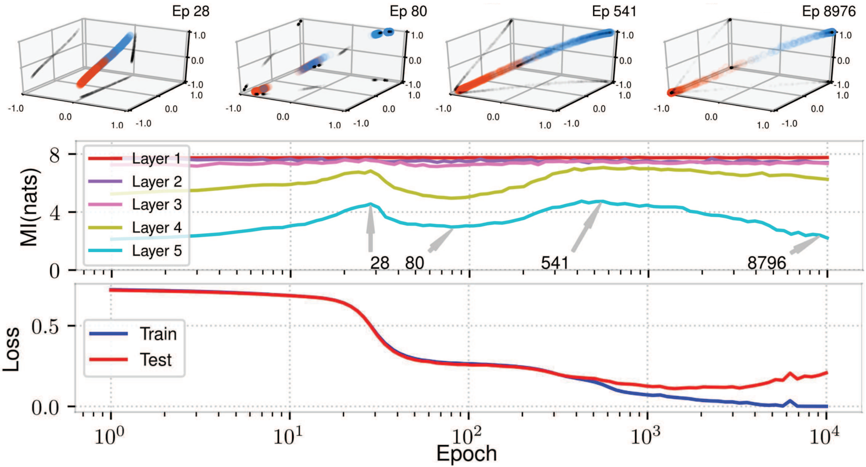
Clustering of Representations - Larger Networks

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

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

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

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



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 tanh MLP
- 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 tanh MLP
- Verified in multiple additional experiments

$\Longrightarrow$ Compression of $I(X;T_\ell)$ driven by clustering of representations

Circling Back to Deterministic DNNs

$\boxed{I(X;T_\ell) \text{ is constant/infinite} \implies \text{Doesn't measure clustering}}$

Circling Back to Deterministic DNNs

$$\boxed{I(X;T_\ell) \text{ is constant/infinite } \implies \text{ Doesn't measure clustering}}$$

Reexamine Measurements: Computed $I\big(X;\mathsf{Bin}(T_\ell)\big) = H\big(\mathsf{Bin}(T_\ell)\big)$

Circling Back to Deterministic DNNs

$\boxed{I(X;T_\ell) \text{ is constant/infinite} \implies \text{Doesn't measure clustering}}$

Reexamine Measurements: Computed $I\big(X;\mathsf{Bin}(T_\ell)\big) = H\big(\mathsf{Bin}(T_\ell)\big)$

- $H\big(\mathsf{Bin}(T_\ell)\big)$ measures clustering (maximized by uniform distribution)

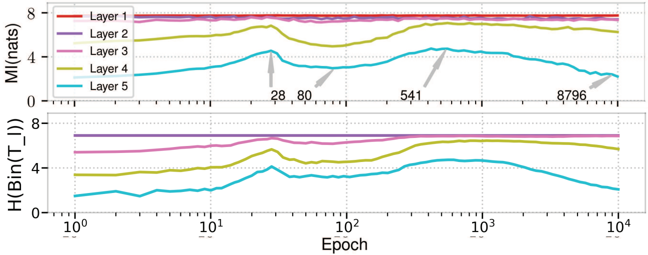
Circling Back to Deterministic DNNs

$\boxed{I(X;T_\ell) \text{ is constant/infinite} \implies \text{Doesn't measure clustering}}$

Reexamine Measurements: Computed $I(X;\mathrm{Bin}(T_\ell)) = H(\mathrm{Bin}(T_\ell))$

- $H(\mathrm{Bin}(T_\ell))$ measures clustering (maximized by uniform distribution)

Test: $I(X;T_\ell)$ and $H(\mathrm{Bin}(T_\ell))$ highly correlated in noisy DNNs*



$\star$ When bin size chosen $\propto$ noise std.

Circling Back to Deterministic DNNs

$\boxed{I(X;T_\ell) \text{ is constant/infinite} \implies \text{Doesn't measure clustering}}$

Reexamine Measurements: Computed $I\big(X;\mathsf{Bin}(T_\ell)\big) = H\big(\mathsf{Bin}(T_\ell)\big)$

- $H\big(\mathsf{Bin}(T_\ell)\big)$ measures clustering (maximized by uniform distribution)

Test: $I(X;T_\ell)$ and $H\big(\mathsf{Bin}(T_\ell)\big)$ highly correlated in noisy DNNs*

$\implies$ **Past works not measuring MI but clustering (via binned-MI)!**

Circling Back to Deterministic DNNs

$\boxed{I(X;T_\ell) \text{ is constant/infinite} \implies \text{Doesn't measure clustering}}$

Reexamine Measurements: Computed $I(X;\text{Bin}(T_\ell)) = H(\text{Bin}(T_\ell))$

- $H(\text{Bin}(T_\ell))$ measures clustering (maximized by uniform distribution)

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

$\implies$ **Past works not measuring MI but clustering (via binned-MI)!**

By-Product Result:

Circling Back to Deterministic DNNs

$\boxed{I(X;T_\ell) \text{ is constant/infinite} \implies \text{Doesn't measure clustering}}$

Reexamine Measurements: Computed $I(X;\mathsf{Bin}(T_\ell)) = H(\mathsf{Bin}(T_\ell))$

- $H(\mathsf{Bin}(T_\ell))$ measures clustering (maximized by uniform distribution)

Test: $I(X;T_\ell)$ and $H(\mathsf{Bin}(T_\ell))$ highly correlated in noisy DNNs*

$\implies$ **Past works not measuring MI but clustering (via binned-MI)!**

By-Product Result:

- Refute 'compression (tight clustering) improves generalization' claim

[Come see us at poster #96 for details]

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 (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
  - Optimal estimator (in n and d) for accurate MI estimation

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
  - Optimal estimator (in n and d) for accurate MI estimation
  - Clustering of 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
  - Optimal estimator (in n and d) for accurate MI estimation
  - Clustering of 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
  - Optimal estimator (in n and d) for accurate MI estimation
  - Clustering of learned representations is the source of compression
- **Clarify Past Observations of Compression:** in fact show clustering
  - Compression/clustering and generalization and not necessarily related

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
  - Optimal estimator (in n and d) for accurate MI estimation
  - Clustering of learned representations is the source of compression
- **Clarify Past Observations of Compression:** in fact show clustering
  - Compression/clustering and generalization and not necessarily related

Thank you!

Clustering of Representations - Larger Networks

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

Clustering of Representations - Larger Networks

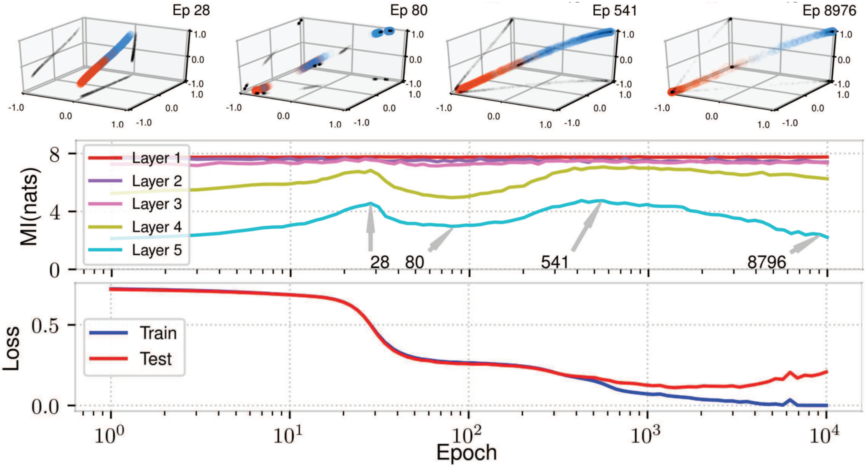
Noisy version of DNN from [Shwartz-Tishby'17]:

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

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 tanh MLP



Clustering of Representations - Larger Networks

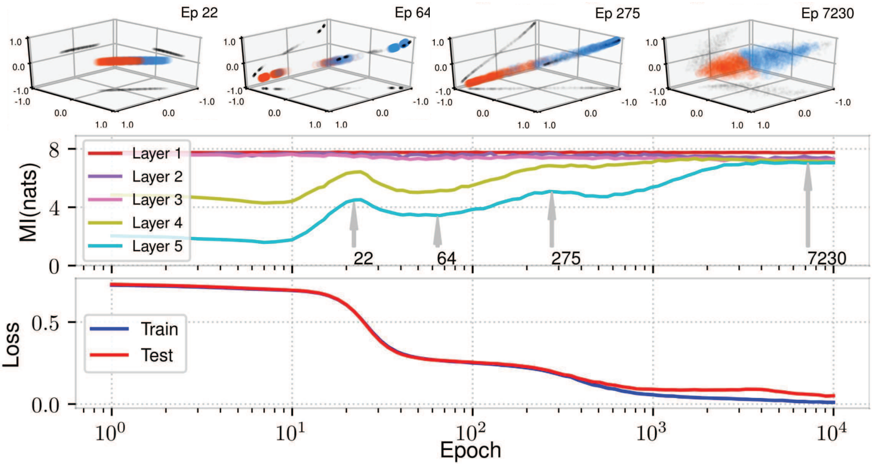
Noisy version of DNN from [Shwartz-Tishby'17]:

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

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 tanh MLP



⊛ weight orthonormality regularization [Cisse *et al.*'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 tanh MLP
- Verified in multiple additional experiments

Clustering of Representations - Larger Networks

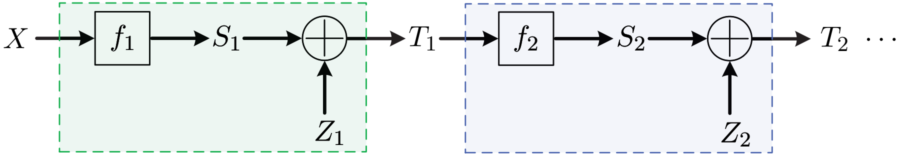
Noisy version of DNN from [Shwartz-Tishby'17]:

- **Binary Classification:** 12-bit input & 12–**10–7–5–4–3**–2 tanh MLP
- Verified in multiple additional experiments

$\Longrightarrow$ Compression of $I(X;T_\ell)$ driven by clustering of representations

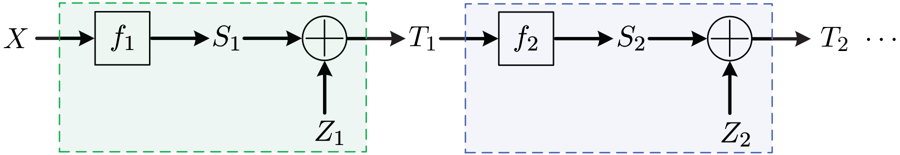
Mutual Information Estimation in Noisy DNNs

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



Mutual Information Estimation in Noisy DNNs

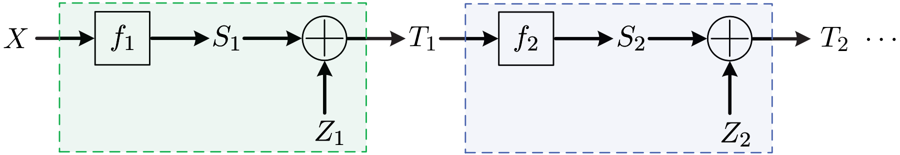
Noisy DNN: $T_\ell = S_\ell + Z_\ell$, where $S_\ell \triangleq f_\ell(T_{\ell-1})$ and $Z_\ell \sim \mathcal{N}(0, \sigma^2 \mathrm{I}_d)$



- **Mutual Information:** $I(X;T_\ell) = h(T_\ell) - \int \mathrm{d}P_X(x) h(T_\ell | X = x)$

Mutual Information Estimation in Noisy DNNs

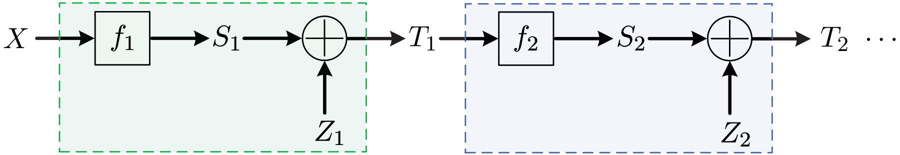
Noisy DNN: $T_\ell = S_\ell + Z_\ell$, where $S_\ell \triangleq f_\ell(T_{\ell-1})$ and $Z_\ell \sim \mathcal{N}(0, \sigma^2 \mathrm{I}_d)$



- **Mutual Information:** $I(X; T_\ell) = h(T_\ell) - \int \mathrm{d}P_X(x) h(T_\ell | X = x)$
- **Structure:** $S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P * \mathcal{N}_\sigma$

Mutual Information Estimation in Noisy DNNs

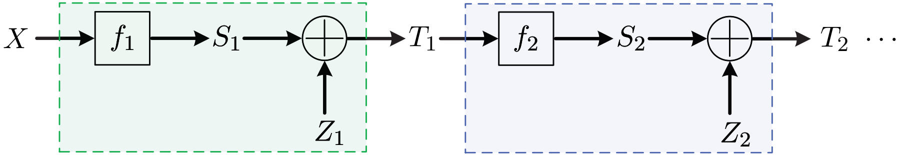
Noisy DNN: $T_\ell = S_\ell + Z_\ell$, where $S_\ell \triangleq f_\ell(T_{\ell-1})$ and $Z_\ell \sim \mathcal{N}(0, \sigma^2 \mathrm{I}_d)$



- **Mutual Information:** $I(X; T_\ell) = h(T_\ell) - \int \mathrm{d}P_X(x) h(T_\ell | X = x)$
- **Structure:** $S_\ell \perp Z_\ell \implies T_\ell = \boldsymbol{S_\ell} + Z_\ell \sim \boldsymbol{P} * \mathcal{N}_\sigma$

Mutual Information Estimation in Noisy DNNs

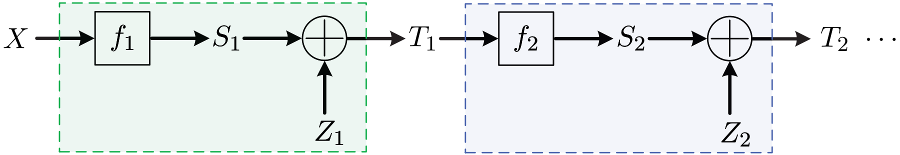
Noisy DNN: $T_\ell = S_\ell + Z_\ell$, where $S_\ell \triangleq f_\ell(T_{\ell-1})$ and $Z_\ell \sim \mathcal{N}(0, \sigma^2 \mathrm{I}_d)$



- **Mutual Information:** $I(X;T_\ell) = h(T_\ell) - \int \mathrm{d}P_X(x) h(T_\ell | X = x)$
- **Structure:** $S_\ell \perp Z_\ell \implies T_\ell = S_\ell + \boldsymbol{Z_\ell} \sim P * \boldsymbol{\mathcal{N}_\sigma}$

Mutual Information Estimation in Noisy DNNs

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

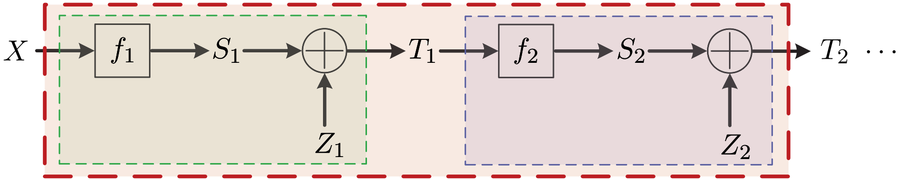


- **Mutual Information:** $I(X; T_\ell) = h(T_\ell) - \int \mathrm{d}P_X(x) h(T_\ell | X = x)$
- **Structure:** $S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P * \mathcal{N}_\sigma$

⊛ **Know** the distribution $\mathcal{N}_\sigma$ of Z_ℓ (noise injected by design)

Mutual Information Estimation in Noisy DNNs

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

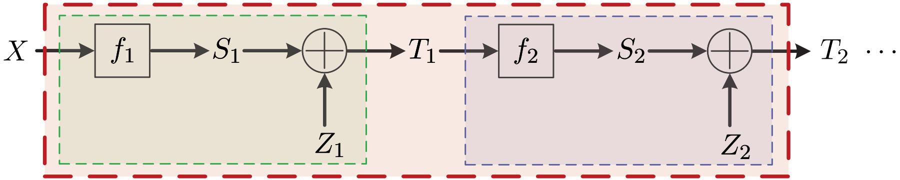


- **Mutual Information:** $I(X;T_\ell) = h(T_\ell) - \int \mathrm{d}P_X(x) h(T_\ell | X = x)$
- **Structure:** $S_\ell \perp Z_\ell \quad \Longrightarrow \quad T_\ell = S_\ell + Z_\ell \sim P * \mathcal{N}_\sigma$

⊛ **Know** the distribution $\mathcal{N}_\sigma$ of Z_ℓ (noise injected by design)

Mutual Information Estimation in Noisy DNNs

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



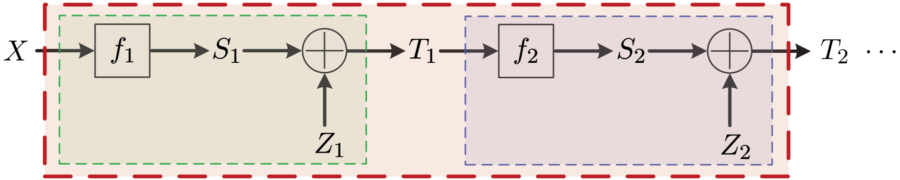
- **Mutual Information:** $I(X;T_\ell) = h(T_\ell) - \int \mathrm{d}P_X(x) h(T_\ell|X=x)$
- **Structure:** $S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P * \mathcal{N}_\sigma$

⊛ **Know** the distribution $\mathcal{N}_\sigma$ of Z_ℓ (noise injected by design)

⊛ **Extremely complicated** $P \implies$ Treat as unknown

Mutual Information Estimation in Noisy DNNs

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



- **Mutual Information:** $I(X;T_\ell) = h(T_\ell) - \int \mathrm{d}P_X(x) h(T_\ell | X = x)$
- **Structure:** $S_\ell \perp Z_\ell \quad \Longrightarrow \quad T_\ell = S_\ell + Z_\ell \sim P * \mathcal{N}_\sigma$

⊛ **Know** the distribution $\mathcal{N}_\sigma$ of Z_ℓ (noise injected by design)

⊛ **Extremely complicated** $P \implies$ Treat as unknown

⊛ **Easily** get i.i.d. samples from P via DNN forward pass

Structured Estimator (with Implementation in Mind)

Differential Entropy Estimation under Gaussian Convolutions

Estimate $h(P * \mathcal{N}_\sigma)$ *via* n *i.i.d. samples* $S^n \triangleq (S_i)_{i=1}^n$ *from unknown* $P \in \mathcal{F}_d$ *(nonparametric class) and knowledge of* $\mathcal{N}_\sigma$ *(noise distribution).*

Structured Estimator (with Implementation in Mind)

Differential Entropy Estimation under Gaussian Convolutions

Estimate $h(P * \mathcal{N}_\sigma)$ *via* n *i.i.d. samples* $S^n \triangleq (S_i)_{i=1}^n$ *from unknown* $P \in \mathcal{F}_d$ *(nonparametric class) and knowledge of* $\mathcal{N}_\sigma$ *(noise distribution).*

Nonparametric Class: Specified by DNN architecture ($d = T_\ell$ 'width')

Structured Estimator (with Implementation in Mind)

Differential Entropy Estimation under Gaussian Convolutions

Estimate $h(P * \mathcal{N}_\sigma)$ via n i.i.d. samples $S^n \triangleq (S_i)_{i=1}^n$ from unknown $P \in \mathcal{F}_d$ (nonparametric class) and knowledge of $\mathcal{N}_\sigma$ (noise distribution).

Nonparametric Class: Specified by DNN architecture ($d = T_\ell$ 'width')

Goal: Simple & parallelizable for efficient implementation

Structured Estimator (with Implementation in Mind)

Differential Entropy Estimation under Gaussian Convolutions

Estimate $h(P * \mathcal{N}_\sigma)$ via n i.i.d. samples $S^n \triangleq (S_i)_{i=1}^n$ from unknown $P \in \mathcal{F}_d$ (nonparametric class) and knowledge of $\mathcal{N}_\sigma$ (noise distribution).

Nonparametric Class: Specified by DNN architecture ($d = T_\ell$ 'width')

Goal: Simple & parallelizable for efficient implementation

Estimator: $\hat{h}(S^n, \sigma) \triangleq h(\hat{P}_{S^n} * \mathcal{N}_\sigma)$, where $\hat{P}_{S^n} \triangleq \frac{1}{n}\sum_{i=1}^{n} \delta_{S_i}$

Structured Estimator (with Implementation in Mind)

Differential Entropy Estimation under Gaussian Convolutions

Estimate $h(P * \mathcal{N}_\sigma)$ via n i.i.d. samples $S^n \triangleq (S_i)_{i=1}^n$ from unknown $P \in \mathcal{F}_d$ (nonparametric class) and knowledge of $\mathcal{N}_\sigma$ (noise distribution).

Nonparametric Class: Specified by DNN architecture ($d = T_\ell$ 'width')

Goal: Simple & parallelizable for efficient implementation

Estimator: $\hat{h}(S^n, \sigma) \triangleq h(\hat{P}_{S^n} * \mathcal{N}_\sigma)$, where $\hat{P}_{S^n} \triangleq \frac{1}{n}\sum_{i=1}^n \delta_{S_i}$

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

Structured Estimator - Convergence Rate

Theorem (ZG-Greenewald-Weed-Polyanskiy'19)

For any $\sigma > 0,\ d \geq 1$*, we have*

$$\sup_{P \in \mathcal{F}_{d,K}^{(\mathsf{SG})}} \mathbb{E}\left| h(P * \mathcal{N}_\sigma) - h(\hat{P}_{S^n} * \mathcal{N}_\sigma) \right| \leq C_{\sigma,d,K} \cdot n^{-\frac{1}{2}}$$

where $C_{\sigma,d,K} = O_{\sigma,K}(c^d)$ *for a constant* c*.*

Structured Estimator - Convergence Rate

Theorem (ZG-Greenewald-Weed-Polyanskiy'19)

For any $\sigma > 0,\ d \geq 1$, we have

$$\sup_{P \in \mathcal{F}_{d,K}^{(\mathsf{SG})}} \mathbb{E}\left| h(P * \mathcal{N}_\sigma) - h(\hat{P}_{S^n} * \mathcal{N}_\sigma) \right| \leq C_{\sigma,d,K} \cdot n^{-\frac{1}{2}}$$

where $C_{\sigma,d,K} = O_{\sigma,K}(c^d)$ for a constant c.

Comments:

Structured Estimator - Convergence Rate

Theorem (ZG-Greenewald-Weed-Polyanskiy'19)

For any $\sigma > 0,\ d \geq 1$, we have

$$\sup_{P \in \mathcal{F}_{d,K}^{(\mathsf{SG})}} \mathbb{E}\left| h(P * \mathcal{N}_\sigma) - h(\hat{P}_{S^n} * \mathcal{N}_\sigma) \right| \leq C_{\sigma,d,K} \cdot n^{-\frac{1}{2}}$$

where $C_{\sigma,d,K} = O_{\sigma,K}(c^d)$ for a constant c.

Comments:

- **Explicit Expression:** Enables concrete error bounds in simulations

Structured Estimator - Convergence Rate

Theorem (ZG-Greenewald-Weed-Polyanskiy'19)

For any $\sigma > 0,\ d \geq 1$, *we have*

$$\sup_{P \in \mathcal{F}_{d,K}^{(\mathsf{SG})}} \mathbb{E}\left| h(P * \mathcal{N}_\sigma) - h(\hat{P}_{S^n} * \mathcal{N}_\sigma) \right| \leq C_{\sigma,d,K} \cdot n^{-\frac{1}{2}}$$

where $C_{\sigma,d,K} = O_{\sigma,K}(c^d)$ *for a constant* c.

Comments:

- **Explicit Expression:** Enables concrete error bounds in simulations
- **Minimax Rate Optimal:** Attains parametric estimation rate $O\big(n^{-\frac{1}{2}}\big)$

Structured Estimator - Convergence Rate

Theorem (ZG-Greenewald-Weed-Polyanskiy'19)

For any $\sigma > 0,\ d \geq 1$, *we have*

$$\sup_{P \in \mathcal{F}_{d,K}^{(\mathsf{SG})}} \mathbb{E}\left| h(P * \mathcal{N}_\sigma) - h(\hat{P}_{S^n} * \mathcal{N}_\sigma) \right| \leq C_{\sigma,d,K} \cdot n^{-\frac{1}{2}}$$

where $C_{\sigma,d,K} = O_{\sigma,K}(c^d)$ *for a constant* c.

Comments:

- **Explicit Expression:** Enables concrete error bounds in simulations
- **Minimax Rate Optimal:** Attains parametric estimation rate $O\left(n^{-\frac{1}{2}}\right)$

Proof (initial step): Based on [Polyanskiy-Wu'16]

$$\boxed{\left| h(P * \mathcal{N}_\sigma) - h(\hat{P}_{S^n} * \mathcal{N}_\sigma) \right| \lesssim W_1(P * \mathcal{N}_\sigma, \hat{P}_{S^n} * \mathcal{N}_\sigma)}$$

Structured Estimator - Convergence Rate

Theorem (ZG-Greenewald-Weed-Polyanskiy'19)

For any $\sigma > 0,\ d \geq 1$, we have

$$\sup_{P \in \mathcal{F}_{d,K}^{(\mathsf{SG})}} \mathbb{E}\left| h(P * \mathcal{N}_\sigma) - h(\hat{P}_{S^n} * \mathcal{N}_\sigma) \right| \leq C_{\sigma,d,K} \cdot n^{-\frac{1}{2}}$$

where $C_{\sigma,d,K} = O_{\sigma,K}(c^d)$ for a constant c.

Comments:

- **Explicit Expression:** Enables concrete error bounds in simulations
- **Minimax Rate Optimal:** Attains parametric estimation rate $O\left(n^{-\frac{1}{2}}\right)$

Proof (initial step): Based on [Polyanskiy-Wu'16]

$$\boxed{\left| h(P * \mathcal{N}_\sigma) - h(\hat{P}_{S^n} * \mathcal{N}_\sigma) \right| \lesssim W_1(P * \mathcal{N}_\sigma, \hat{P}_{S^n} * \mathcal{N}_\sigma)}$$

$\Longrightarrow$ Analyze empirical 1-Wasserstein distance under Gaussian convolutions

Empirical W_1 & The Magic of Gaussian Convolution

p-Wasserstein Distance: For two distributions P and Q on $\mathbb{R}^d$ and $p \geq 1$

$$W_p(P, Q) \triangleq \inf \left(\mathbb{E}\|X - Y\|^p\right)^{1/p}$$

infimum over all couplings of P and Q

Empirical W_1 & The Magic of Gaussian Convolution

p-Wasserstein Distance: For two distributions P and Q on $\mathbb{R}^d$ and $p \geq 1$

$$W_p(P,Q) \triangleq \inf \left(\mathbb{E}\|X-Y\|^p\right)^{1/p}$$

infimum over all couplings of P and Q

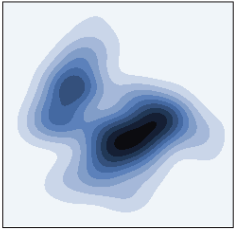
Empirical 1-Wasserstein Distance:

Empirical W_1 & The Magic of Gaussian Convolution

p-Wasserstein Distance: For two distributions P and Q on $\mathbb{R}^d$ and $p \geq 1$

$$W_p(P, Q) \triangleq \inf \left(\mathbb{E}\|X - Y\|^p\right)^{1/p}$$

infimum over all couplings of P and Q



Empirical 1-Wasserstein Distance:

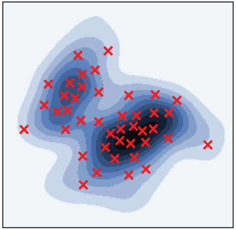
- Distribution P on $\mathbb{R}^d$

Empirical W_1 & The Magic of Gaussian Convolution

p-Wasserstein Distance: For two distributions P and Q on $\mathbb{R}^d$ and $p \geq 1$

$$W_p(P,Q) \triangleq \inf \left(\mathbb{E}\|X - Y\|^p\right)^{1/p}$$

infimum over all couplings of P and Q



Empirical 1-Wasserstein Distance:

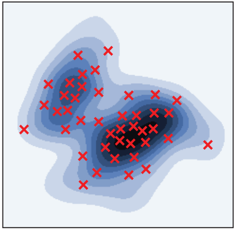
- Distribution P on $\mathbb{R}^d$ $\implies$ i.i.d. Samples $(S_i)_{i=1}^n$

Empirical W_1 & The Magic of Gaussian Convolution

p-Wasserstein Distance: For two distributions P and Q on $\mathbb{R}^d$ and $p \geq 1$

$$W_p(P,Q) \triangleq \inf \left(\mathbb{E}\|X-Y\|^p\right)^{1/p}$$

infimum over all couplings of P and Q



Empirical 1-Wasserstein Distance:

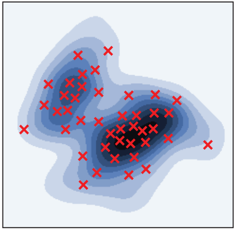
- Distribution P on $\mathbb{R}^d \implies$ i.i.d. Samples $(S_i)_{i=1}^n$
- Empirical distribution $\hat{P}_{S^n} \triangleq \frac{1}{n}\sum_{i=1}^n \delta_{S_i}$

Empirical W_1 & The Magic of Gaussian Convolution

p-Wasserstein Distance: For two distributions P and Q on $\mathbb{R}^d$ and $p \geq 1$

$$W_p(P,Q) \triangleq \inf \left(\mathbb{E}\|X-Y\|^p\right)^{1/p}$$

infimum over all couplings of P and Q



Empirical 1-Wasserstein Distance:

- Distribution P on $\mathbb{R}^d \implies$ i.i.d. Samples $(S_i)_{i=1}^n$
- Empirical distribution $\hat{P}_{S^n} \triangleq \frac{1}{n}\sum_{i=1}^n \delta_{S_i}$

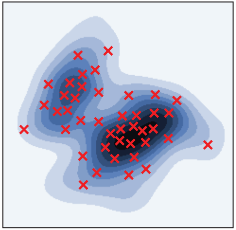
$\implies$ Dependence on (n,d) of $\mathbb{E}W_1(P,\hat{P}_{S^n})$

Empirical W_1 & The Magic of Gaussian Convolution

p-Wasserstein Distance: For two distributions P and Q on $\mathbb{R}^d$ and $p \geq 1$

$$W_p(P,Q) \triangleq \inf \left(\mathbb{E}\|X-Y\|^p\right)^{1/p}$$

infimum over all couplings of P and Q



Empirical 1-Wasserstein Distance:

- Distribution P on $\mathbb{R}^d$ $\implies$ i.i.d. Samples $(S_i)_{i=1}^n$
- Empirical distribution $\hat{P}_{S^n} \triangleq \frac{1}{n}\sum_{i=1}^{n} \delta_{S_i}$

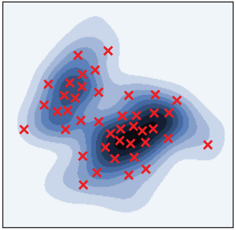
$\implies$ Dependence on (n,d) of $\mathbb{E}W_1(P, \hat{P}_{S^n}) \gtrsim n^{-\frac{1}{d}}$

Empirical W_1 & The Magic of Gaussian Convolution

p-Wasserstein Distance: For two distributions P and Q on $\mathbb{R}^d$ and $p \geq 1$

$$W_p(P,Q) \triangleq \inf \left(\mathbb{E}\|X-Y\|^p\right)^{1/p}$$

infimum over all couplings of P and Q



Empirical 1-Wasserstein Distance:

- Distribution P on $\mathbb{R}^d$ $\implies$ i.i.d. Samples $(S_i)_{i=1}^n$
- Empirical distribution $\hat{P}_{S^n} \triangleq \frac{1}{n}\sum_{i=1}^n \delta_{S_i}$

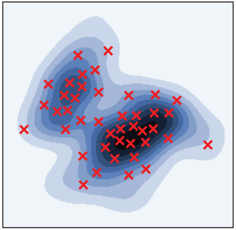
$\implies$ Dependence on (n,d) of $\mathbb{E}W_1(P,\hat{P}_{S^n}) \gtrsim n^{-\frac{1}{d}}$

Empirical W_1 & The Magic of Gaussian Convolution

p-Wasserstein Distance: For two distributions P and Q on $\mathbb{R}^d$ and $p \geq 1$

$$W_p(P,Q) \triangleq \inf \left(\mathbb{E}\|X-Y\|^p\right)^{1/p}$$

infimum over all couplings of P and Q



Empirical 1-Wasserstein Distance:

- Distribution P on $\mathbb{R}^d \implies$ i.i.d. Samples $(S_i)_{i=1}^n$
- Empirical distribution $\hat{P}_{S^n} \triangleq \frac{1}{n}\sum_{i=1}^{n} \delta_{S_i}$

$\implies$ Dependence on (n,d) of $\mathbb{E}W_1(P,\hat{P}_{S^n}) \gtrsim n^{-\frac{1}{d}}$

Theorem (ZG-Greenewald-Weed-Polyanskiy'19)

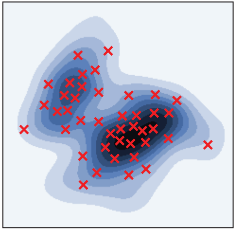
For any d, we have $\mathbb{E}W_1(P*\mathcal{N}_\sigma, \hat{P}_{S^n}*\mathcal{N}_\sigma) \leq O_{\sigma,d}(n^{-\frac{1}{2}})$

Empirical W_1 & The Magic of Gaussian Convolution

p-Wasserstein Distance: For two distributions P and Q on $\mathbb{R}^d$ and $p \geq 1$

$$W_p(P,Q) \triangleq \inf \left(\mathbb{E}\|X-Y\|^p\right)^{1/p}$$

infimum over all couplings of P and Q



Empirical 1-Wasserstein Distance:

- Distribution P on $\mathbb{R}^d \implies$ i.i.d. Samples $(S_i)_{i=1}^n$
- Empirical distribution $\hat{P}_{S^n} \triangleq \frac{1}{n}\sum_{i=1}^{n} \delta_{S_i}$

$\implies$ Dependence on (n,d) of $\mathbb{E}W_1(P,\hat{P}_{S^n}) \gtrsim n^{-\frac{1}{d}}$

Curse of Dimensionality

Theorem (ZG-Greenewald-Weed-Polyanskiy'19)

For any d, we have $\mathbb{E}W_1\left(P * \mathcal{N}_\sigma, \hat{P}_{S^n} * \mathcal{N}_\sigma\right) \leq O_{\sigma,d}\left(n^{-\frac{1}{2}}\right) = O_\sigma\left(c^d n^{-\frac{1}{2}}\right)$

Is Exponentiality in Dimension Necessary?

Is Exponentiality in Dimension Necessary?

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

For any $\sigma > 0$, sufficiently large d and sufficiently small $\eta > 0$, we have $n^\star(\eta, \sigma, \mathcal{F}_d) = \Omega\left(\frac{2^{\gamma(\sigma)d}}{\eta d}\right)$, where $\gamma(\sigma) > 0$ is monotonically decreasing in σ.

Is Exponentiality in Dimension Necessary?

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

For any $\sigma > 0$, sufficiently large d and sufficiently small $\eta > 0$, we have $n^\star(\eta, \sigma, \mathcal{F}_d) = \Omega\left(\frac{2^{\gamma(\sigma)d}}{\eta d}\right)$, where $\gamma(\sigma) > 0$ is monotonically decreasing in σ.

$\implies O\left(\frac{c^d}{\sqrt{n}}\right)$ rate attained by the plugin estimator is sharp in n and d

Is Exponentiality in Dimension Necessary?

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

For any $\sigma > 0$, sufficiently large d and sufficiently small $\eta > 0$, we have $n^\star(\eta, \sigma, \mathcal{F}_d) = \Omega\left(\frac{2^{\gamma(\sigma)d}}{\eta d}\right)$, where $\gamma(\sigma) > 0$ is monotonically decreasing in σ.

$\Longrightarrow$ $O\left(\frac{c^d}{\sqrt{n}}\right)$ rate attained by the plugin estimator is sharp in n and d

Proof (main ideas):

Is Exponentiality in Dimension Necessary?

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

For any $\sigma > 0$, sufficiently large d and sufficiently small $\eta > 0$, we have $n^\star(\eta, \sigma, \mathcal{F}_d) = \Omega\left(\frac{2^{\gamma(\sigma)d}}{\eta d}\right)$, where $\gamma(\sigma) > 0$ is monotonically decreasing in σ.

$\boxed{\Longrightarrow\ O\left(\frac{c^d}{\sqrt{n}}\right)\text{ rate attained by the plugin estimator is sharp in } n \text{ and } d}$

Proof (main ideas):

- Relate $h(P * \mathcal{N}_\sigma)$ to Shannon entropy $H(Q)$

  $\mathrm{supp}(Q) =$ peak-constrained AWGN capacity achieving codebook $\mathcal{C}_d$

Is Exponentiality in Dimension Necessary?

Theorem (ZG-Greenewald-Polyanskiy-Weed'19)

For any $\sigma > 0$, sufficiently large d and sufficiently small $\eta > 0$, we have $n^\star(\eta, \sigma, \mathcal{F}_d) = \Omega\left(\frac{2^{\gamma(\sigma)d}}{\eta d}\right)$, where $\gamma(\sigma) > 0$ is monotonically decreasing in σ.

$\Longrightarrow$ $O\left(\frac{c^d}{\sqrt{n}}\right)$ rate attained by the plugin estimator is sharp in n and d

Proof (main ideas):

- Relate $h(P * \mathcal{N}_\sigma)$ to Shannon entropy $H(Q)$

  $\mathrm{supp}(Q) =$ peak-constrained AWGN capacity achieving codebook $\mathcal{C}_d$
- $H(Q)$ estimation sample complexity $\Omega\left(\frac{|\mathcal{C}_d|}{\eta \log |\mathcal{C}_d|}\right)$ [Valiant-Valiant'10]