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?
Lacking Theory: Macroscopic understanding of Deep Learning
Lacking Theory: Macroscopic understanding of Deep Learning

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

- What drives the evolution of internal representations?
- What are properties of learned representations?
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, Panyan 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, Gabrïé 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 $T_{\ell} = f_{\ell}(T_{\ell-1})$

- $Y$ (Label)
- $X$ (Feature/Image)
- $T_0 = X$ (Input Layer)
- $T_1$ (Hidden Layer 1)
- $T_2$ (Hidden Layer 2)
- $T_3$ (Hidden Layer 3)
- $T_4 = \hat{Y}$ (Output Layer)

Cat

Dog
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:** \( P_{X,Y} \Rightarrow P_{X,Y} \cdot P_{T_1,\ldots,T_L|X} \)
- **Information Plane:** Evolution of \((I(X;T_\ell), I(Y;T_\ell))\) during training

\[
I(A; B) = D_{KL}(P_{A,B}||P_A \otimes P_B) = \text{Discrete} \sum_{a,b} P_{A,B}(a, b) \log \frac{P_{A,B}(a,b)}{P_A(a)P_B(b)}
\]
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 $T_\ell = f_\ell(T_{\ell-1})$

$Y$ (Label) $ightarrow$ $X$ (Feature/Image)

$T_0 = X$ (Input Layer)
$T_1$ (Hidden Layer 1)
$T_2$ (Hidden Layer 2)
$T_3$ (Hidden Layer 3)
$T_4 = \hat{Y}$ (Output Layer)

IB Theory Claim: Training comprises 2 phases

- **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]
Proposition (Informal)

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

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

\[ I(X; T_\ell) \text{ is independent of the DNN parameters} \]
**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 $X$)
Proposition (Informal)

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

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

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

Feature Space ($X$)

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

$$|\mathcal{X}| = 3000$$
Proposition (Informal)

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

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

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

\[ X \sim \text{Unif}(\mathcal{X}) \]

\[ |\mathcal{X}| = 3000 \]

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

\[ |\mathcal{T}_\ell| = |\mathcal{X}| = 3000 \]
**Proposition (Informal)**

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

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

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

**Past Works:** Use binning-based proxy of \( I(X; T_\ell) \) (aka quantization)
Proposition (Informal)

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

\[ I(X; T_\ell) \] 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)
  - For non-negligible bin size \( I(X; \text{Bin}(T_\ell)) \neq I(X; T_\ell) \)
Proposition (Informal)

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

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

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

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

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

- \( I(X; T_\ell) \) 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 \( I(X; \text{Bin}(T_\ell)) \neq I(X; T_\ell) \)
2. \( I(X; \text{Bin}(T_\ell)) \) highly sensitive to user-defined bin size:
**Proposition (Informal)**

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

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

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

- **Past Works:** Use binning-based proxy of \( I(X; T_\ell) \) (aka quantization)
  - For non-negligible bin size \( I(X; \text{Bin}(T_\ell)) \neq I(X; T_\ell) \)
  - \( I(X; \text{Bin}(T_\ell)) \) highly sensitive to user-defined bin size:

- **Real Problem:** Mismatch between \( I(X; T_\ell) \) measurement and model
**Modification:** Inject (small) Gaussian noise to neurons’ output
**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 $Z_\ell \sim \mathcal{N}(0, \sigma^2 I_d)$
**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 $Z_\ell \sim \mathcal{N}(0, \sigma^2 I_d)$

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

$\implies X \mapsto T_\ell$ is a **parametrized channel** (by DNN’s parameters)
**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 \( Z_\ell \sim \mathcal{N}(0, \sigma^2 I_d) \)

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

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

\( I(X; T_\ell) \) is a **function** of parameters!
**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 \( Z_\ell \sim \mathcal{N}(0, \sigma^2 I_d) \)

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

\( \implies X \mapsto T_\ell \) is a **parametrized channel** (by DNN’s parameters)

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

\( \otimes \) **Challenge:** How to accurately track \( I(X; T_\ell) \)?
Distill $I(X; T_\ell)$ Estimation into Noisy Differential Entropy Estimation:

Estimate $h(P \ast \mathcal{N}_\sigma)$ from $n$ i.i.d. samples $S^n \triangleq (S_i)^n_{i=1}$ of $P \in \mathcal{F}_d$ (non-parametric class) and knowledge of $\mathcal{N}_\sigma$ (Gaussian measure $\mathcal{N}(0, \sigma^2 I_d)$).
Distill $I(X;T_\ell)$ Estimation into Noisy Differential Entropy Estimation:

Estimate $h(P \ast \mathcal{N}_\sigma)$ 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 $\mathcal{N}(0, \sigma^2 I_d)$).

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

Sample complexity of any accurate estimator (additive gap $\eta$) 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 $h(P \ast N_\sigma)$ from $n$ i.i.d. samples $S^n \triangleq (S_i)_{i=1}^n$ of $P \in F_d$ (non-parametric class) and knowledge of $N_\sigma$ (Gaussian measure $N(0, \sigma^2 I_d)$).

**Theorem (ZG-Greenewald-Polyanskiy-Weed’19)**

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

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

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

Estimate $h(P \ast \mathcal{N}_\sigma)$ 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 $\mathcal{N}(0, \sigma^2 I_d)$).

**Theorem (ZG-Greenewald-Polyanskiy-Weed’19)**

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

**Structured Estimator**

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

**Theorem (ZG-Greenewald-Polyanskiy-Weed’19)**

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

$$\sup_{P \in \mathcal{F}_{d,K}^{(SG)}} \mathbb{E}_{S^n} \left| h(P \ast \mathcal{N}_\sigma) - \hat{h}(S^n, \sigma) \right| \leq c_{\sigma,K}^d \cdot n^{-\frac{1}{2}}$$
Distill $I(X;T_ℓ)$ Estimation into Noisy Differential Entropy Estimation:

Estimate $h(P*N_σ)$ from $n$ i.i.d. samples $S^n ≜ (S_i)_{i=1}^n$ of $P ∈ F_d$ (non-parametric class) and knowledge of $N_σ$ (Gaussian measure $N(0, σ^2I_d)$).

**Theorem (ZG-Greenewald-Polyanskiy-Weed’19)**

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

**Structured Estimator**: $\hat{h}(S^n, σ) ≜ h(\hat{P}_n*N_σ)$, where $\hat{P}_n = \frac{1}{n} \sum_{i=1}^{n} δ_{S_i}$

**Theorem (ZG-Greenewald-Polyanskiy-Weed’19)**

For $F_d^{(SG)} ≜ \{ P | P \text{ is } K\text{-subgaussian in } \mathbb{R}^d \}$, $d ≥ 1$ and $σ > 0$, we have

$$\sup_{P ∈ F_d^{(SG)}} E_{S^n} \left| h(P*N_σ) - \hat{h}(S^n, σ) \right| ≤ c_{σ,K}^d \cdot n^{-\frac{1}{2}}$$

**Optimality**: $\hat{h}(S^n, σ)$ attains sharp dependence on both $n$ and $d$!
Single Neuron Classification:

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

Single Neuron Classification:

\[ X \xrightarrow{\tanh(wX + b)} S_{w,b} \xrightarrow{\text{sum}} T \]

\[ Z \sim \mathcal{N}(0, \sigma^2) \]
Single Neuron Classification:

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

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

$X_{y=-1} \triangleq \{-3, -1, 1\}$, $X_{y=1} \triangleq \{3\}$

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

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

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

\[ Z \sim N(0, \sigma^2) \]
Single Neuron Classification:

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

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

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

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

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

- $X \xrightarrow{\text{tanh}(wX + b)} S_{w,b} \xrightarrow{T} Z \sim \mathcal{N}(0, \sigma^2)$

- Correct classification performance
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:**
**Single Neuron Classification:**

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

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

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

\[ X_{y=-1} \triangleq \{-3, -1, 1\}, \quad X_{y=1} \triangleq \{3\} \]

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

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

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

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

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

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

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

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

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

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

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

\[ \implies I(X; T) \text{ is } \# \text{ bits (nats) transmittable over AWGN with symbols} \]
\[ S_{w,b} \triangleq \{\tanh(-3w+b), \tanh(-w+b), \tanh(w+b), \tanh(3w+b)\} \rightarrow \{\pm 1\} \]
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
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
- Verified in multiple additional experiments
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

\[ I(X; T_\ell) \] driven by clustering of representations
\[ I(X; T_\ell) \text{ is constant/infinite} \implies \text{Doesn’t measure clustering} \]
I(X; T_\ell) is constant/infinite \implies \text{Doesn’t measure clustering}

**Reexamine Measurements:** Computed \( I(X; \text{Bin}(T_\ell)) = H(\text{Bin}(T_\ell)) \)
Circling Back to Deterministic DNNs

\[ I(X; T\ell) \text{ is constant/infinite} \implies \text{Doesn’t measure clustering} \]

**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)
Circling Back to Deterministic DNNs

\[ I(X; T_\ell) \text{ is constant/infinite} \implies \text{Doesn’t measure clustering} \]

**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*

---

* When bin size chosen \( \propto \) noise std.
Circling Back to Deterministic DNNs

\[ I(X; T_\ell) \text{ is constant/finite } \implies \text{ Doesn’t measure clustering} \]

**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*

\[ \implies \text{ Past works not measuring MI but clustering (via binned-MI)!} \]
Circling Back to Deterministic DNNs

$I(X; T_\ell)$ is constant/infinite $\implies$ 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

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

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*

\[ \implies \text{Past works not measuring MI but clustering (via binned-MI)!} \]

By-Product Result:

- Refute ‘compression (tight clustering) improves generalization’ claim

[Come see us at poster #96 for details]
Reexamined Information Bottleneck Compression:
Reexamined Information Bottleneck Compression:

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

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

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]
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
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

$\rightarrow$ Compression of $I(X; T_\ell)$ driven by clustering of representations
Noisy DNN: \( T_\ell = S_\ell + Z_\ell \), where \( S_\ell \triangleq f_\ell(T_{\ell-1}) \) and \( Z_\ell \sim \mathcal{N}(0, \sigma^2 I_d) \)
**Mutual Information Estimation in Noisy DNNs**

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

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

**Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \int dP_X(x) h(T_\ell | X = x) \)
Noisy DNN: $T_\ell = S_\ell + Z_\ell$, where $S_\ell \triangleq f_\ell(T_{\ell-1})$ and $Z_\ell \sim \mathcal{N}(0, \sigma^2 I_d)$

Mutual Information: $I(X; T_\ell) = h(T_\ell) - \int dP_X(x) h(T_\ell | X = x)$

Structure: $S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \mathcal{N}_\sigma$
Mutual Information Estimation in Noisy DNNs

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

\[
\begin{align*}
X & \rightarrow f_1 \rightarrow S_1 \rightarrow T_1 \rightarrow f_2 \rightarrow S_2 \rightarrow T_2 \ldots \\
\end{align*}
\]

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \int dP_X(x) h(T_\ell | X = x) \)

- **Structure:** \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \mathcal{N}_\sigma \)
Mutual Information Estimation in Noisy DNNs

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

![Diagram of Noisy DNN]

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \int dP_X(x)h(T_\ell | X = x) \)
- **Structure:** \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \mathcal{N}_\sigma \)
Mutual Information Estimation in Noisy DNNs

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

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

• Mutual Information: \( I(X; T_\ell) = h(T_\ell) - \int dP_X(x)h(T_\ell | X = x) \)

• Structure: \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \mathcal{N}_\sigma \)

★ Know the distribution \( \mathcal{N}_\sigma \) of \( Z_\ell \) (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 \( Z_\ell \sim \mathcal{N}(0, \sigma^2 I_d) \)

\[ \begin{array}{c}
  \xymatrix{ X \ar[r]^{f_1} & S_1 \ar[r] & T_1 \ar[r]^{f_2} & S_2 \ar[r] & T_2 \cdots }
  \\
  Z_1 \ar[u]
  \\
  Z_2 \ar[u]
\end{array} \]

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \int dP_X(x)h(T_\ell|X = x) \)
- **Structure:** \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \mathcal{N}_\sigma \)
- **Know** the distribution \( \mathcal{N}_\sigma \) of \( Z_\ell \) (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 \( Z_\ell \sim \mathcal{N}(0, \sigma^2 I_d) \)

![Diagram of Noisy DNN with layers and noises]

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \int dP_X(x)h(T_\ell|X=x) \)
- **Structure:** \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \mathcal{N}_\sigma \)
- **Know** the distribution \( \mathcal{N}_\sigma \) of \( Z_\ell \) (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 \( Z_\ell \sim \mathcal{N}(0, \sigma^2 I_d) \)

\[ h(T_\ell) - \int dP_X(x) h(T_\ell | X = x) \]

- **Mutual Information:** \( I(X; T_\ell) = h(T_\ell) - \int dP_X(x) h(T_\ell | X = x) \)
- **Structure:** \( S_\ell \perp Z_\ell \implies T_\ell = S_\ell + Z_\ell \sim P \ast \mathcal{N}_\sigma \)
- **Know** the distribution \( \mathcal{N}_\sigma \) of \( Z_\ell \) (noise injected by design)
- **Extremely complicated** \( P \implies \) Treat as unknown
- **Easily** get i.i.d. samples from \( P \) via DNN forward pass
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).
Differential Entropy Estimation under Gaussian Convolutions

Estimate $h(P \ast \mathcal{N}_\sigma)$ via $n$ i.i.d. samples $S^n \triangleq (S_i^n)_{i=1}^{n}$ 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 $h(P \ast \mathcal{N}_\sigma)$ via $n$ i.i.d. samples $S^n \triangleq (S_i)^n_{i=1}$ 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)

**Nonparametric Class:** Specified by DNN architecture \((d = T_\ell \text{ ‘width’})\)

**Goal:** Simple & parallelizable for efficient implementation

**Estimator:** 
\[
\hat{h}(S^n, \sigma) \triangleq h(\hat{P}_{S^n} * \mathcal{N}_\sigma), \text{ where } \hat{P}_{S^n} \triangleq \frac{1}{n} \sum_{i=1}^{n} \delta_{S_i}
\]
Structured Estimator (with Implementation in Mind)

### Differential Entropy Estimation under Gaussian Convolutions

Estimate $h(P \ast \mathcal{N}_\sigma)$ 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:**

\[
\hat{h}(S^n, \sigma) \triangleq h(\hat{P}_{S^n} \ast \mathcal{N}_\sigma), \quad \text{where} \quad \hat{P}_{S^n} \triangleq \frac{1}{n} \sum_{i=1}^{n} \delta_{S_i}
\]

- **Plug-in:** $\hat{h}$ is plug-in est. for the functional $T_\sigma(P) \triangleq h(P \ast \mathcal{N}_\sigma)$
For any $\sigma > 0$, $d \geq 1$, we have

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

where $C_{\sigma,d,K} = O_{\sigma,K}(c^d)$ for a constant $c$. 
Theorem (ZG-Greenewald-Weed-Polyanskiy'19)

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

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

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

Comments:
Theorem (ZG-Greenewald-Weed-Polyanskiy’19)

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

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

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

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

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 $O(n^{-\frac{1}{2}})$
Structured Estimator - Convergence Rate

**Theorem (ZG-Greenewald-Weed-Polyanskiy’19)**

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

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

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 $O(n^{-\frac{1}{2}})$

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

$$\left| h(P \ast \mathcal{N}_\sigma) - h(\hat{P}_{Sn} \ast \mathcal{N}_\sigma) \right| \lesssim W_1(P \ast \mathcal{N}_\sigma, \hat{P}_{Sn} \ast \mathcal{N}_\sigma)$$
Theorem (ZG-Greenewald-Weed-Polyanskiy’19)

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

$$\sup_{P \in F_{d, K}^{(SG)}} \mathbb{E} \left| h(P \ast N_\sigma) - h(\hat{P}_{Sn} \ast N_\sigma) \right| \leq C_{\sigma, d, K} \cdot n^{-\frac{1}{2}}$$

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 $O(n^{-\frac{1}{2}})$

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

$$\left| h(P \ast N_\sigma) - h(\hat{P}_{Sn} \ast N_\sigma) \right| \lesssim W_1(P \ast N_\sigma, \hat{P}_{Sn} \ast N_\sigma)$$

$\implies$ Analyze empirical 1-Wasserstein distance under Gaussian convolutions
**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 $Q$
**p-Wasserstein Distance:** For two distributions $P$ and $Q$ on $\mathbb{R}^d$ and $p \geq 1$

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

infimum over all couplings of $P$ and $Q$

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

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

infimum over all couplings of $P$ and $Q$

\textbf{Empirical 1-Wasserstein Distance:}

- Distribution $P$ on $\mathbb{R}^d \quad \Rightarrow \quad$ i.i.d. Samples $(S_i)_{i=1}^n$
$p$-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 $(S_i)^n_{i=1}$
- Empirical distribution $\hat{P}_{Sn} \triangleq \frac{1}{n} \sum_{i=1}^{n} \delta_{S_i}$
**p-Wasserstein Distance:** For two distributions $P$ and $Q$ on $\mathbb{R}^d$ and $p \geq 1$
\[
W_p(P, Q) \triangleq \inf \left( \mathbb{E}\|X - Y\|^p \right)^{1/p}
\]
inimum 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 $\hat{P}_{Sn} \triangleq \frac{1}{n} \sum_{i=1}^n \delta_{S_i}$

$\implies$ Dependence on $(n, d)$ of $\mathbb{E}W_1(P, \hat{P}_{Sn})$
**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 $\hat{P}_{Sn} \triangleq \frac{1}{n} \sum_{i=1}^n \delta_{S_i}$

$\implies$ Dependence on $(n, d)$ of $\mathbb{E} W_1(P, \hat{P}_{Sn}) \gtrsim n^{-1/d}$
**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 $\hat{P}_{Sn} \triangleq \frac{1}{n} \sum_{i=1}^n \delta_{S_i}$

$\implies$ Dependence on $(n, d)$ of $\mathbb{E}W_1(P, \hat{P}_{Sn}) \gtrsim n^{-\frac{1}{d}}$
**p-Wasserstein Distance:** For two distributions $P$ and $Q$ on $\mathbb{R}^d$ and $p \geq 1$

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

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 $\hat{P}_{Sn} \triangleq \frac{1}{n} \sum_{i=1}^n \delta_{S_i}$

\[ \implies \text{Dependence on } (n, d) \text{ of } \mathbb{E}W_1(P, \hat{P}_{Sn}) \gtrsim n^{-\frac{1}{d}} \]

**Theorem (ZG-Greenewald-Weed-Polyanskiy’19)**

For any $d$, we have

\[ \mathbb{E}W_1(P \ast \mathcal{N}_\sigma, \hat{P}_{Sn} \ast \mathcal{N}_\sigma) \leq O_{\sigma,d}(n^{-\frac{1}{2}}) \]
**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 $\mathbb{E}W_1(P \ast \mathcal{N}_\sigma, \hat{P}_{S^n} \ast \mathcal{N}_\sigma) \leq O_{\sigma,d}(n^{-\frac{1}{2}}) = O_\sigma(c^d n^{-\frac{1}{2}})$
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

$$n^*(\eta, \sigma, F_d) = \Omega\left(\frac{2^{\gamma(\sigma)d}}{\eta^d}\right),$$

where $\gamma(\sigma) > 0$ is monotonically decreasing in $\sigma$. 

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

For any $\sigma > 0$, sufficiently large $d$ and sufficiently small $\eta > 0$, we have

$$n^*(\eta, \sigma, \mathcal{F}_d) = \Omega\left(\frac{2^\gamma(\sigma)^d}{\eta^d}\right),$$

where $\gamma(\sigma) > 0$ is monotonically decreasing in $\sigma$.

$$\Rightarrow O\left(\frac{c^d}{\sqrt{n}}\right)$$

rate attained by the plugin estimator is sharp in $n$ and $d$. 

Is Exponentiality in Dimension Necessary?
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 $n^*(\eta, \sigma, F_d) = \Omega\left(\frac{2^{\gamma(\sigma)d}}{\eta^d}\right)$, where $\gamma(\sigma) > 0$ is monotonically decreasing in $\sigma$.

$\Rightarrow O\left(\frac{c^d}{\sqrt{n}}\right)$ 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

$$n^*(\eta, \sigma, \mathcal{F}_d) = \Omega\left(\frac{2^{\gamma(\sigma)d}}{\eta d}\right),$$

where $\gamma(\sigma) > 0$ is monotonically decreasing in $\sigma$.

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

Proof (main ideas):

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

$$\text{supp}(Q) = \text{peak-constrained AWGN capacity achieving codebook } C_d$$
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

$$n^* (\eta, \sigma, F_d) = \Omega \left( \frac{2^{\gamma(\sigma)d}}{\eta d} \right),$$

where $\gamma(\sigma) > 0$ is monotonically decreasing in $\sigma$.

$$\Rightarrow O \left( \frac{c^d}{\sqrt{n}} \right)$$

rate attained by the plugin estimator is sharp in $n$ and $d$

**Proof (main ideas):**

- Relate $h(P \ast \mathcal{N}_\sigma)$ to Shannon entropy $H(Q)$
  $$\text{supp}(Q) = \text{peak-constrained AWGN capacity achieving codebook } C_d$$

- $H(Q)$ estimation sample complexity $\Omega \left( \frac{|C_d|}{\eta \log |C_d|} \right)$ [Valiant-Valiant’10]