Estimating the Flow of Information in Deep Neural Networks

Ziv Goldfeld

MIT

March 2019

Deep Learning - What's Under the Hood?

- Unprecedented practical success in hosts of tasks

Deep Learning - What's Under the Hood?

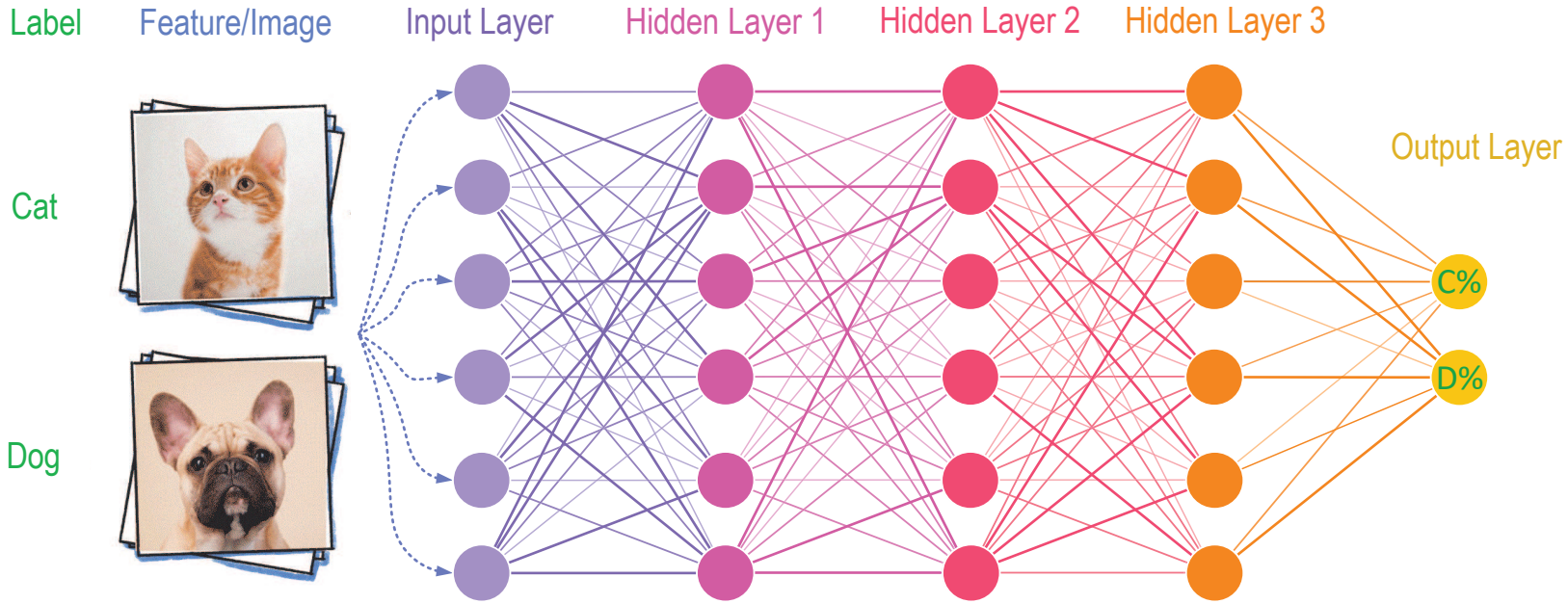
- Unprecedented practical success in hosts of tasks

Deep Learning - What's Under the Hood?

- Unprecedented practical success in hosts of tasks
- **Lacking Theory:** Macroscopic understanding of Deep Learning

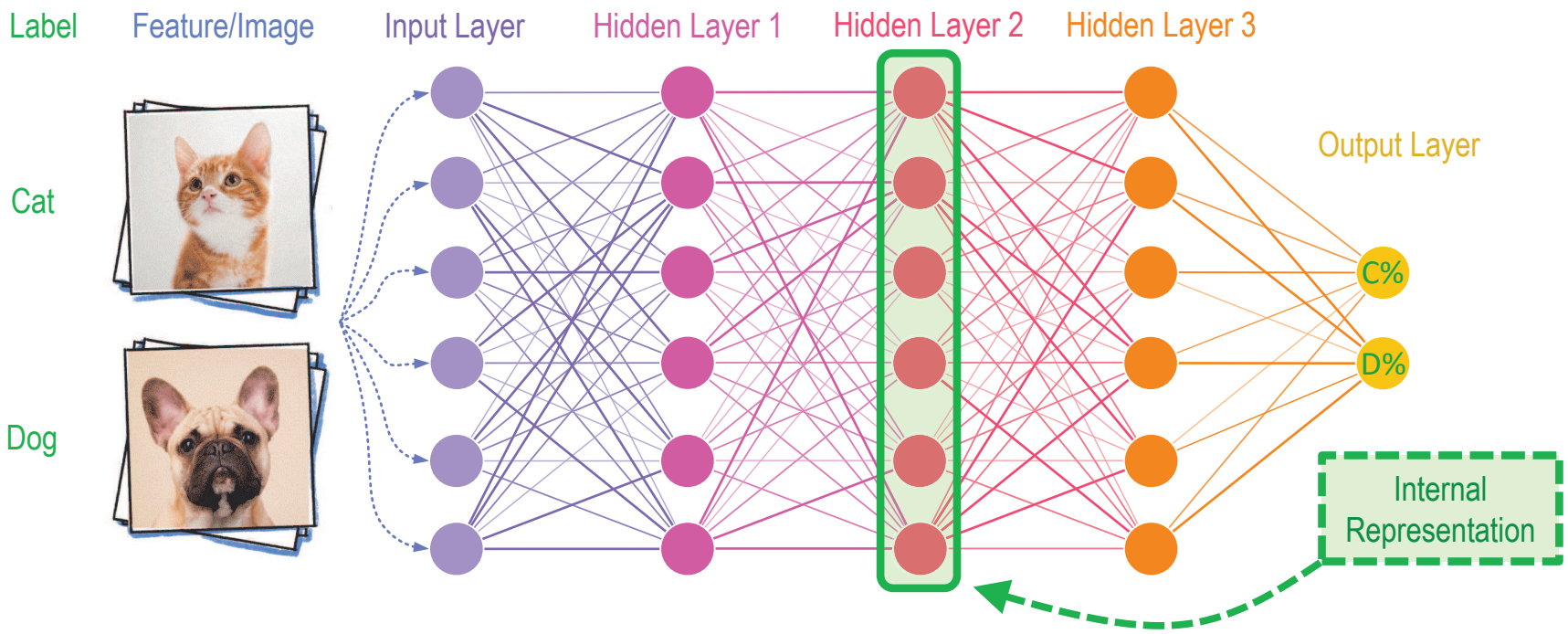
Deep Learning - What's Under the Hood?

- Unprecedented practical success in hosts of tasks
- **Lacking Theory:** Macroscopic understanding of Deep Learning



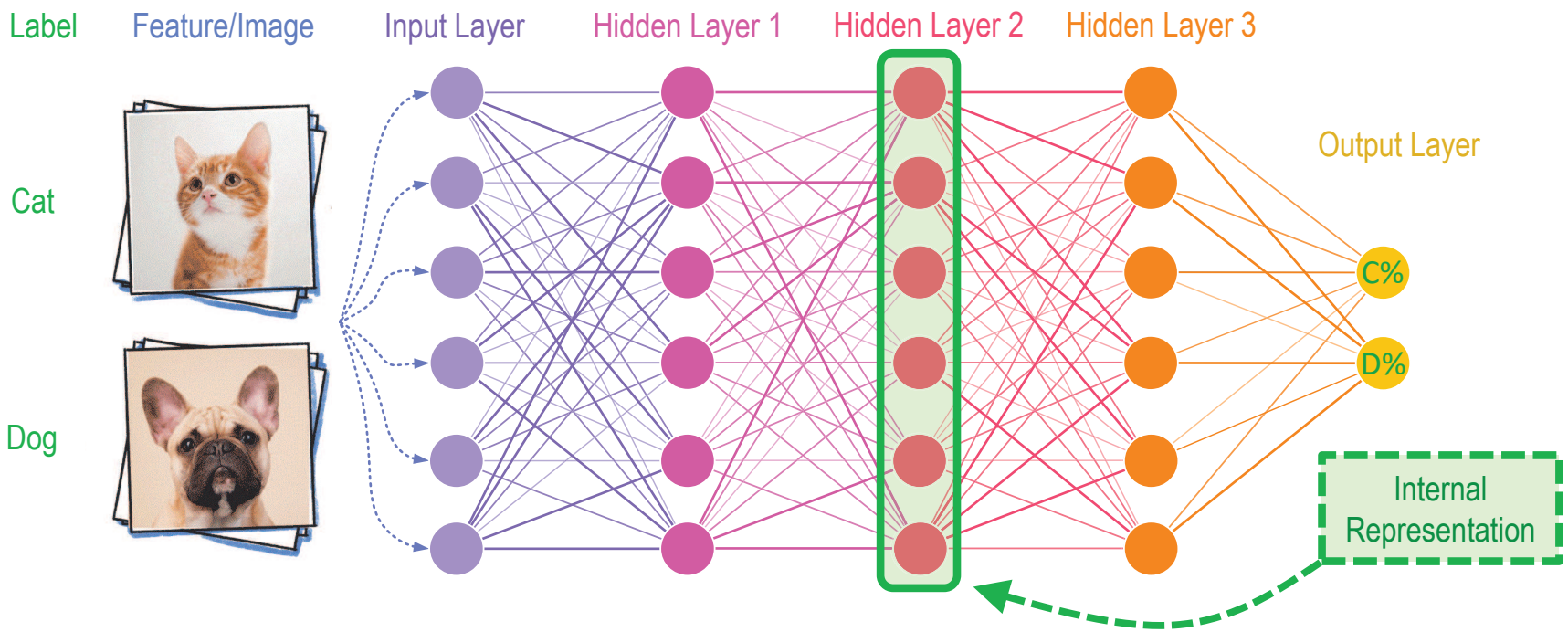
Deep Learning - What's Under the Hood?

- Unprecedented practical success in hosts of tasks
- **Lacking Theory:** Macroscopic understanding of Deep Learning



Deep Learning - What's Under the Hood?

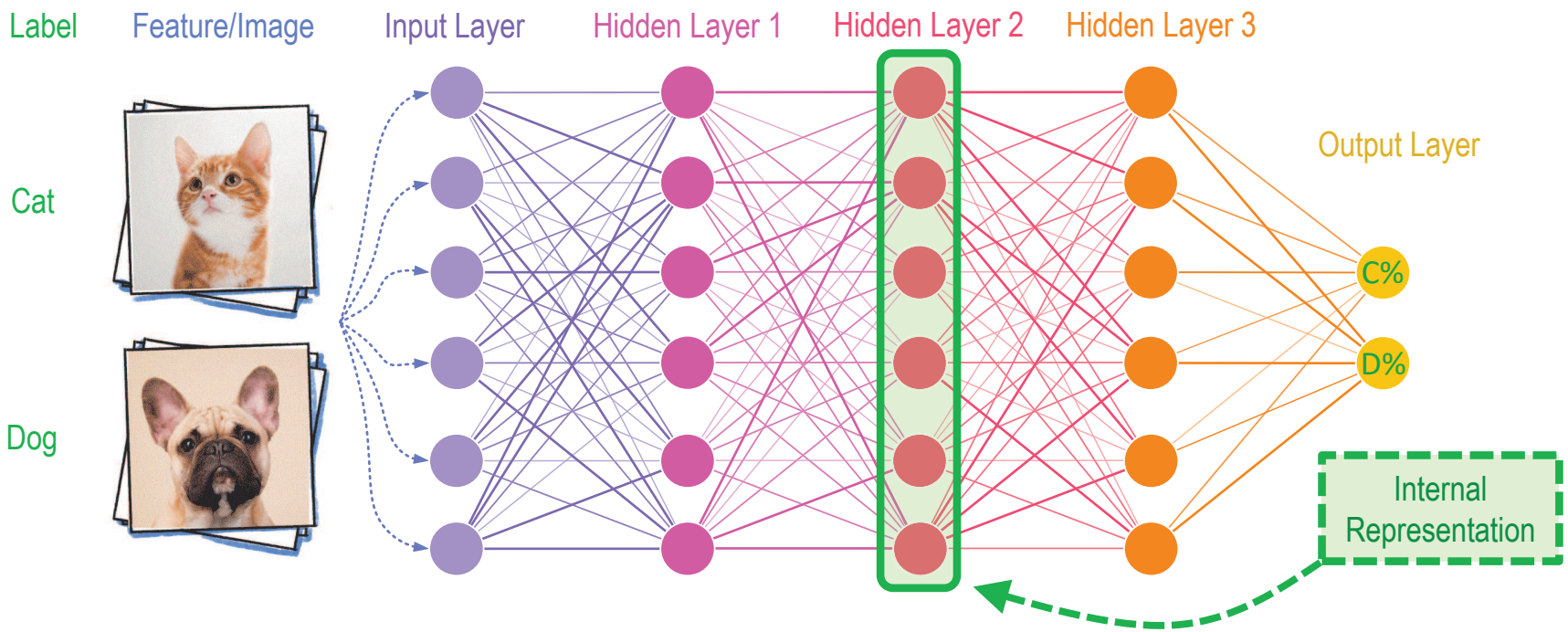
- Unprecedented practical success in hosts of tasks
- **Lacking Theory:** Macroscopic understanding of Deep Learning



? What drives the evolution of internal representations?

Deep Learning - What's Under the Hood?

- Unprecedented practical success in hosts of tasks
- **Lacking Theory:** Macroscopic understanding of Deep Learning

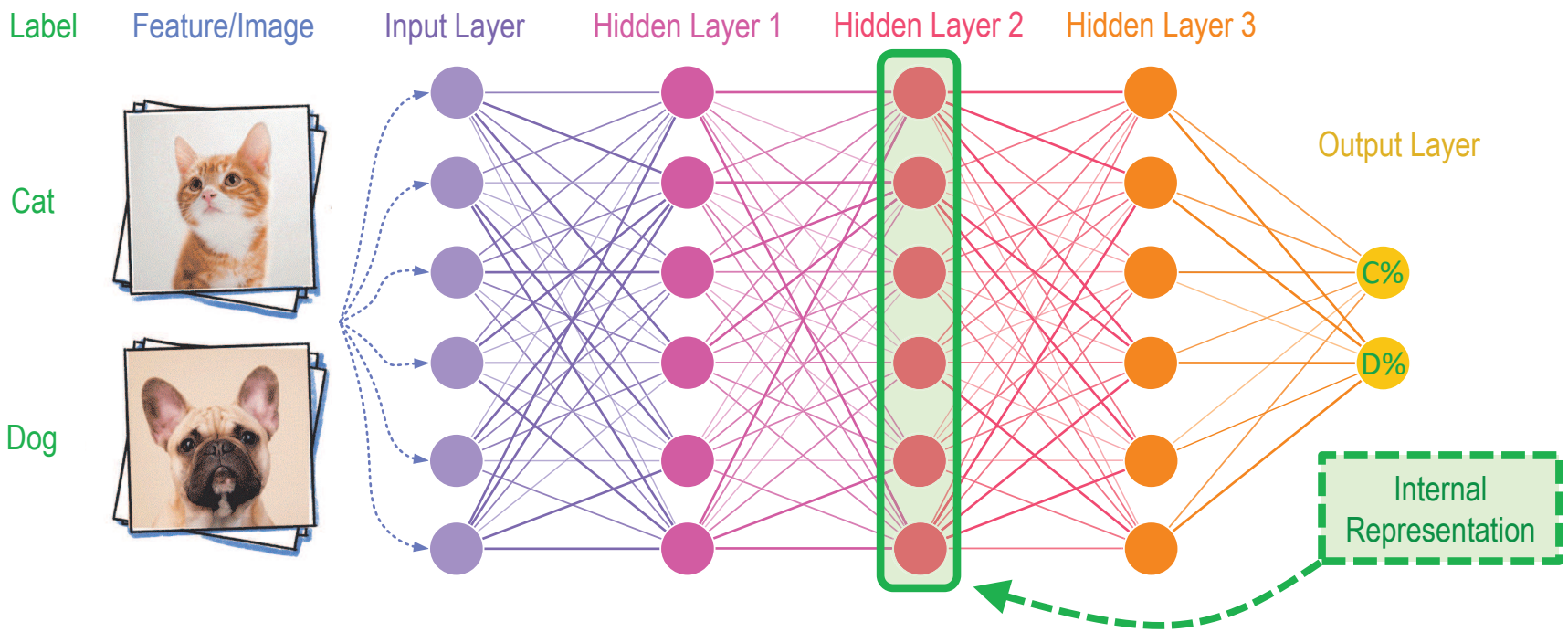


? What drives the evolution of internal representations?

? What are properties of learned representations?

Deep Learning - What's Under the Hood?

- Unprecedented practical success in hosts of tasks
- **Lacking Theory:** Macroscopic understanding of Deep Learning



- ? What drives the evolution of internal representations?
- ? What are properties of learned representations?
- ? How fully trained networks process information?

Past Attempts to Understand Effectiveness of DL:

Deep Learning - An Information-Theoretic Lens

Past Attempts to Understand Effectiveness of DL:

- Optimization dynamics in parameter space

  [Saxe-McClelland-Ganguli'14, Chen-Wang-Zhao-Papailiopoulos-Koutris'18, Li-Liang'18]

Deep Learning - An Information-Theoretic Lens

Past Attempts to Understand Effectiveness of DL:

- Optimization dynamics in parameter space

  [Saxe-McClelland-Ganguli'14, Chen-Wang-Zhao-Papailiopoulos-Koutris'18, Li-Liang'18]

- Classes of efficiently representable functions

  [Hajnal-et al'93, Delalleau-Bengio'11, Eldan-Shamir'15, Telgarsky'16, Poggio-et al'17]

Deep Learning - An Information-Theoretic Lens

Past Attempts to Understand Effectiveness of DL:

- Optimization dynamics in parameter space

  [Saxe-McClelland-Ganguli'14, Chen-Wang-Zhao-Papailiopoulos-Koutris'18, Li-Liang'18]

- Classes of efficiently representable functions

  [Hajnal-et al'93, Delalleau-Bengio'11, Eldan-Shamir'15, Telgarsky'16, Poggio-et al'17]

- Information Bottleneck Theory

  [Tishby-Zaslavsky'15, Shwartz-Tishby'17, Saxe *et al.*'18, Gabrié *et al.*'18]

Deep Learning - An Information-Theoretic Lens

Past Attempts to Understand Effectiveness of DL:

- Optimization dynamics in parameter space

  [Saxe-McClelland-Ganguli'14, Chen-Wang-Zhao-Papailiopoulos-Koutris'18, Li-Liang'18]

- Classes of efficiently representable functions

  [Hajnal-et al'93, Delalleau-Bengio'11, Eldan-Shamir'15, Telgarsky'16, Poggio-et al'17]

- Information Bottleneck Theory

  [Tishby-Zaslavsky'15, Shwartz-Tishby'17, Saxe *et al.*'18, Gabrié *et al.*'18]

  - Study DL dynamics via an information-theoretic perspective

Deep Learning - An Information-Theoretic Lens

Past Attempts to Understand Effectiveness of DL:

- Optimization dynamics in parameter space

  [Saxe-McClelland-Ganguli'14, Chen-Wang-Zhao-Papailiopoulos-Koutris'18, Li-Liang'18]

- Classes of efficiently representable functions

  [Hajnal-et al'93, Delalleau-Bengio'11, Eldan-Shamir'15, Telgarsky'16, Poggio-et al'17]

- Information Bottleneck Theory

  [Tishby-Zaslavsky'15, Shwartz-Tishby'17, Saxe *et al.*'18, Gabrié *et al.*'18]

  - Study DL dynamics via an information-theoretic perspective
  - Intriguing empirical results

Deep Learning - An Information-Theoretic Lens

Past Attempts to Understand Effectiveness of DL:

- Optimization dynamics in parameter space

  [Saxe-McClelland-Ganguli'14, Chen-Wang-Zhao-Papailiopoulos-Koutris'18, Li-Liang'18]

- Classes of efficiently representable functions

  [Hajnal-et al'93, Delalleau-Bengio'11, Eldan-Shamir'15, Telgarsky'16, Poggio-et al'17]

- Information Bottleneck Theory

  [Tishby-Zaslavsky'15, Shwartz-Tishby'17, Saxe *et al.*'18, Gabrié *et al.*'18]

  - Study DL dynamics via an information-theoretic perspective
  - Intriguing empirical results
  - Theory mostly relies on heuristic arguments

Deep Learning - An Information-Theoretic Lens

Past Attempts to Understand Effectiveness of DL:

- Optimization dynamics in parameter space

  [Saxe-McClelland-Ganguli'14, Chen-Wang-Zhao-Papailiopoulos-Koutris'18, Li-Liang'18]

- Classes of efficiently representable functions

  [Hajnal-et al'93, Delalleau-Bengio'11, Eldan-Shamir'15, Telgarsky'16, Poggio-et al'17]

- Information Bottleneck Theory

  [Tishby-Zaslavsky'15, Shwartz-Tishby'17, Saxe *et al.*'18, Gabrié *et al.*'18]

  - Study DL dynamics via an information-theoretic perspective
  - Intriguing empirical results
  - Theory mostly relies on heuristic arguments

★ **Goal:** IB theory mathematical analysis

Deep Learning - An Information-Theoretic Lens

Past Attempts to Understand Effectiveness of DL:

- Optimization dynamics in parameter space

  [Saxe-McClelland-Ganguli'14, Chen-Wang-Zhao-Papailiopoulos-Koutris'18, Li-Liang'18]

- Classes of efficiently representable functions

  [Hajnal-et al'93, Delalleau-Bengio'11, Eldan-Shamir'15, Telgarsky'16, Poggio-et al'17]

- Information Bottleneck Theory

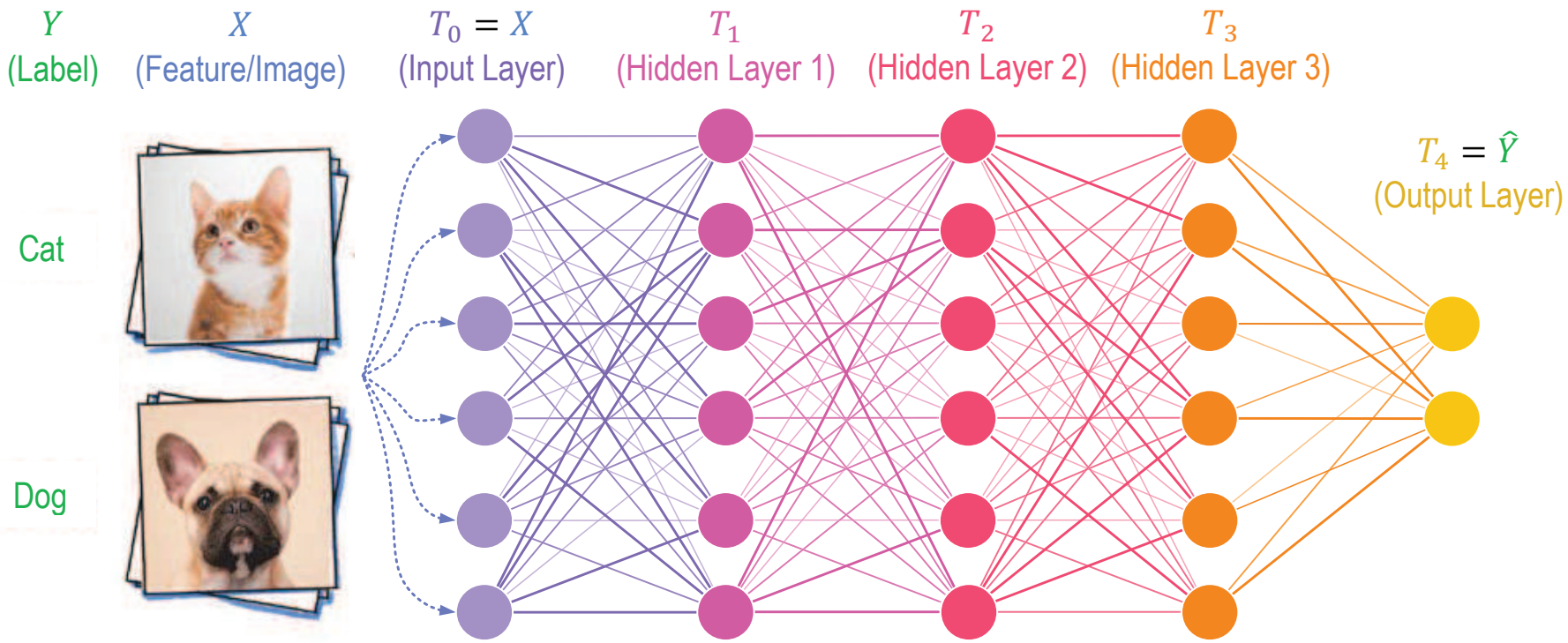
  [Tishby-Zaslavsky'15, Shwartz-Tishby'17, Saxe *et al.*'18, Gabrié *et al.*'18]

  - Study DL dynamics via an information-theoretic perspective
  - Intriguing empirical results
  - Theory mostly relies on heuristic arguments

★ **Goal:** IB theory mathematical analysis $\Longrightarrow$ better DNN designs

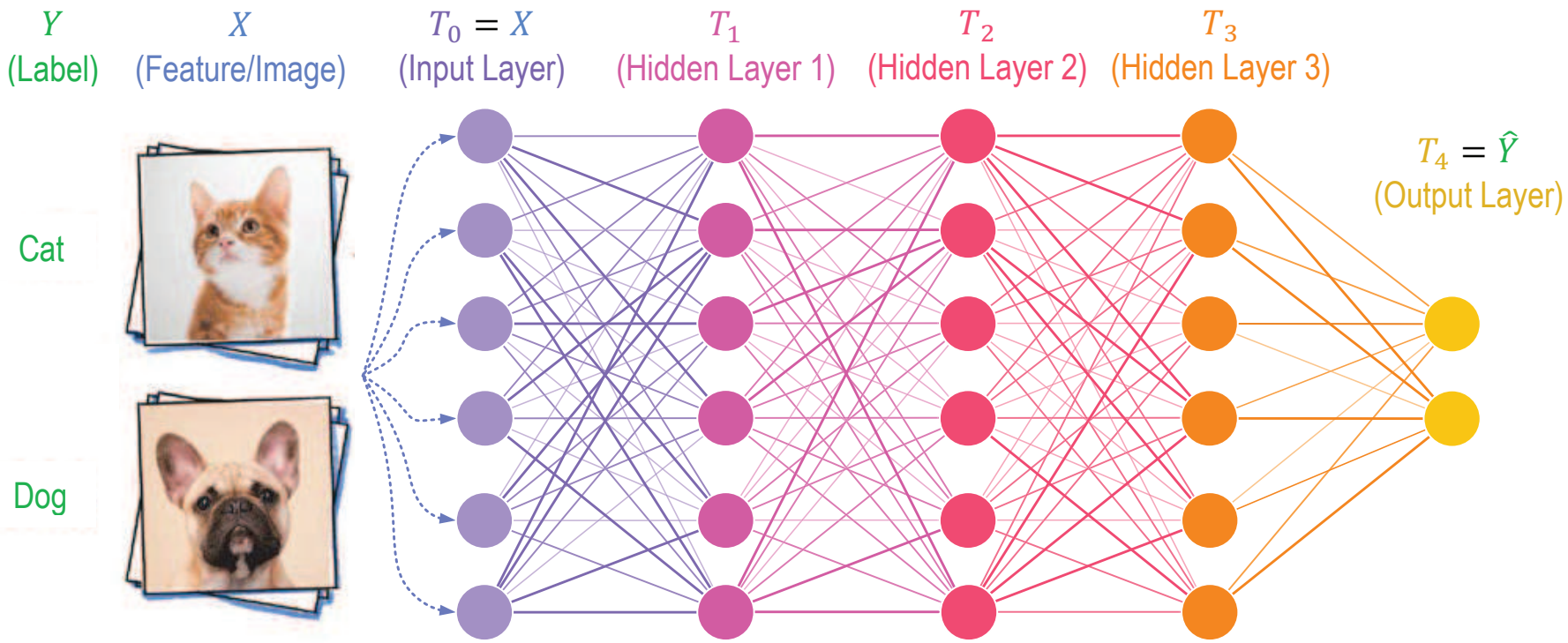
Setup and Preliminaries

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



Setup and Preliminaries

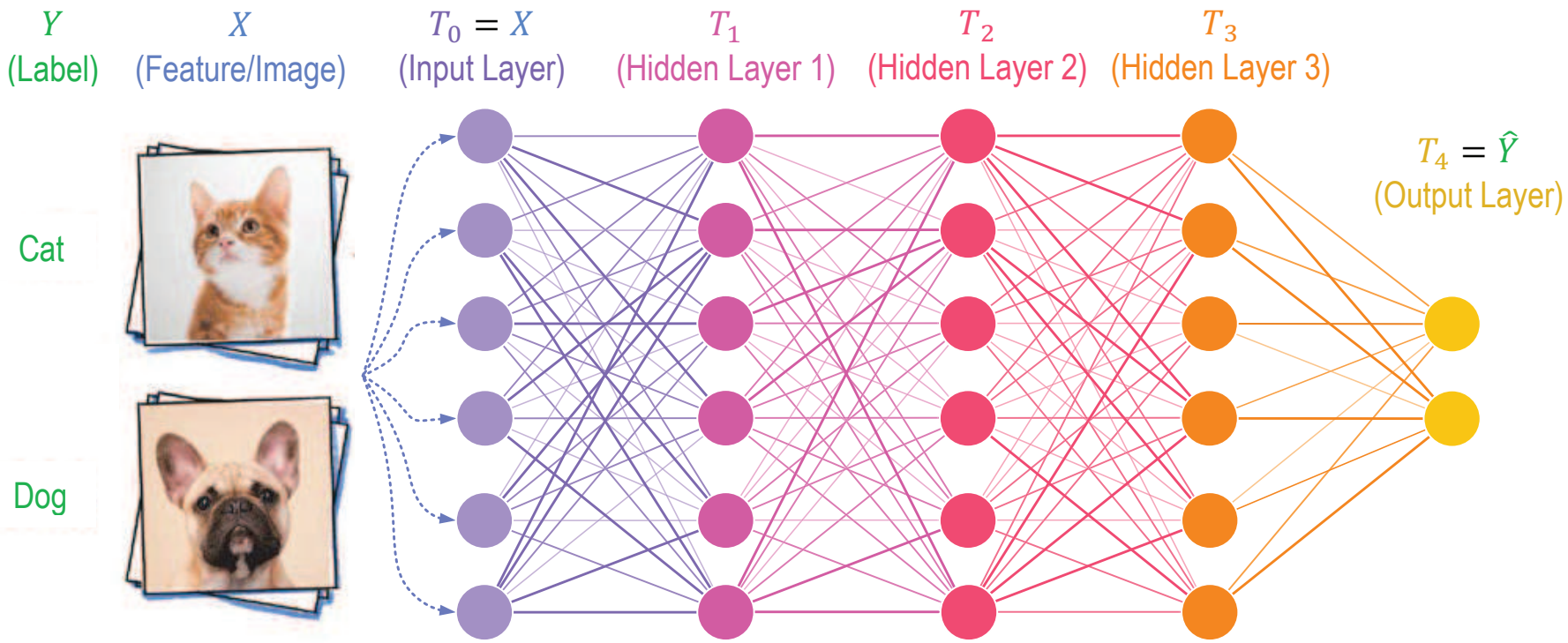
(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$



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

Setup and Preliminaries

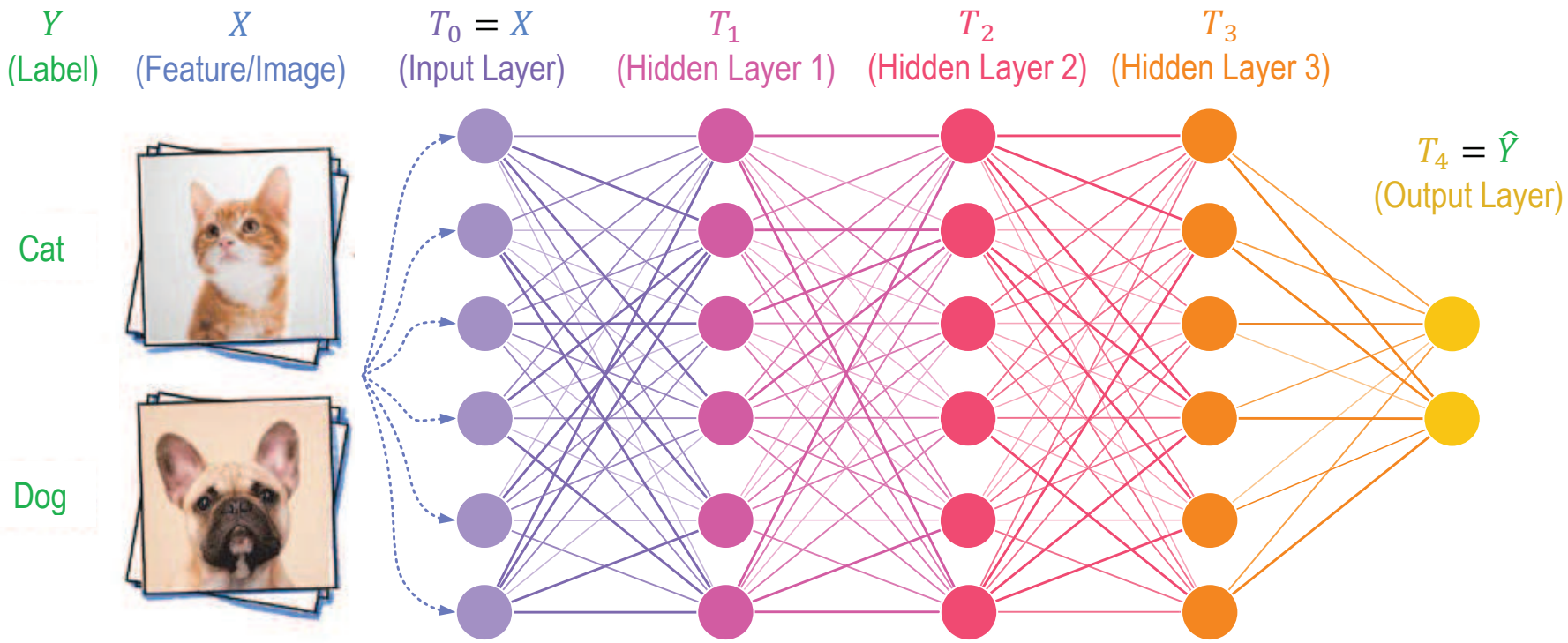
(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$



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

Setup and Preliminaries

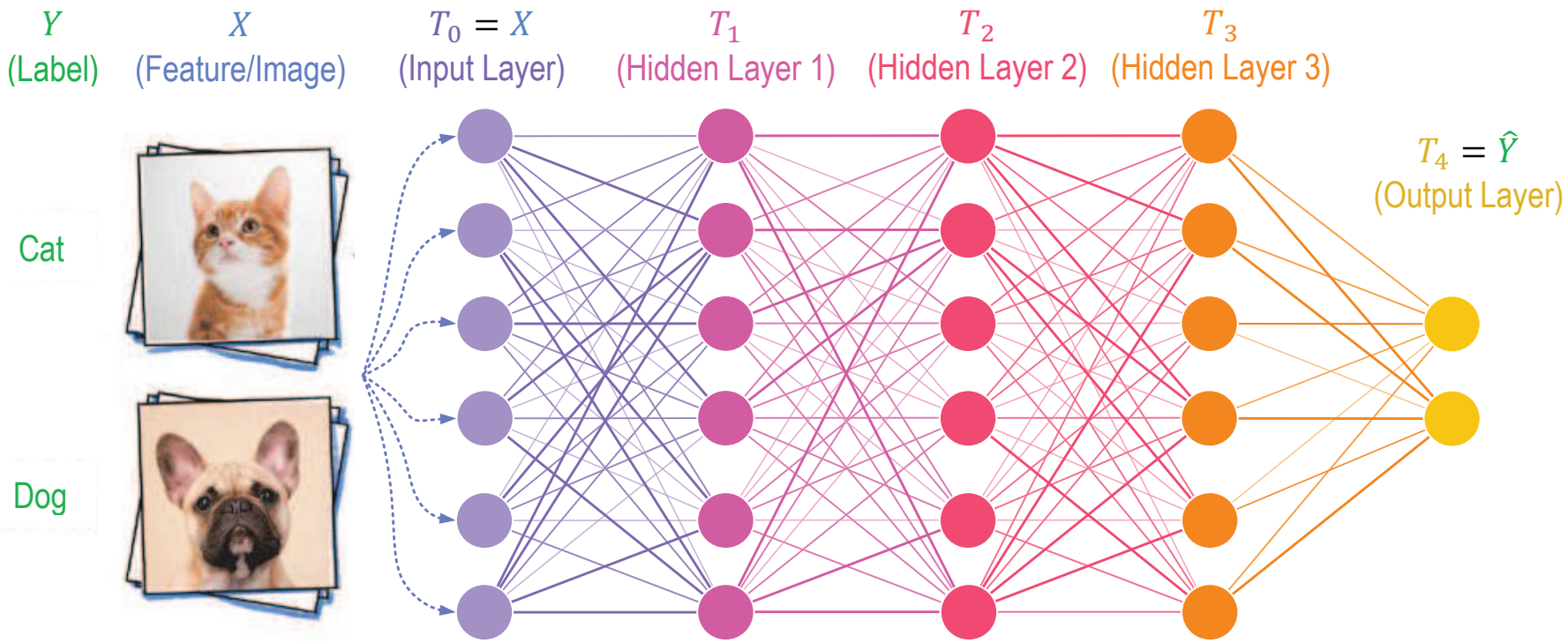
(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$



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

Setup and Preliminaries

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

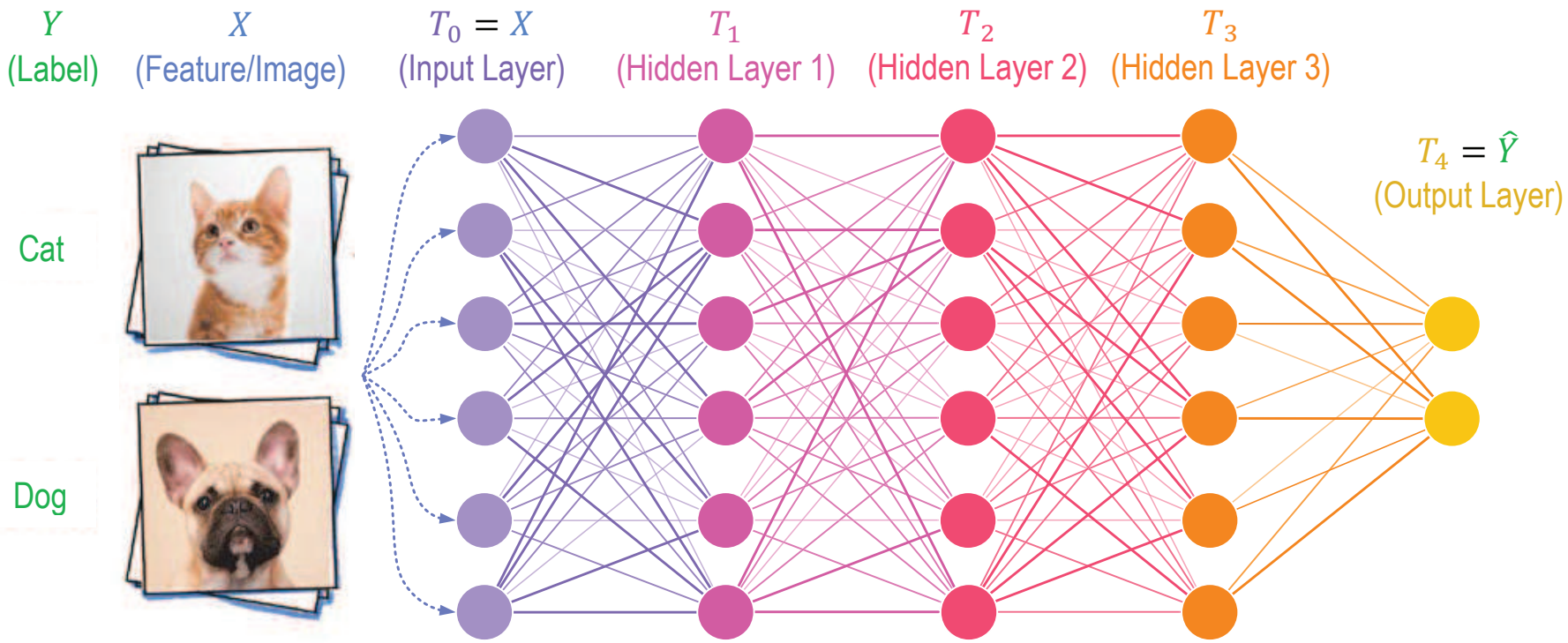


- **Joint Distribution:** $P_{X,Y} \implies P_{X,Y} \cdot P_{T_1,\ldots,T_L|X}$
- **Information Plane:** Evolution of $\big(I(X;T_\ell), I(Y;T_\ell)\big)$ during training

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

Setup and Preliminaries

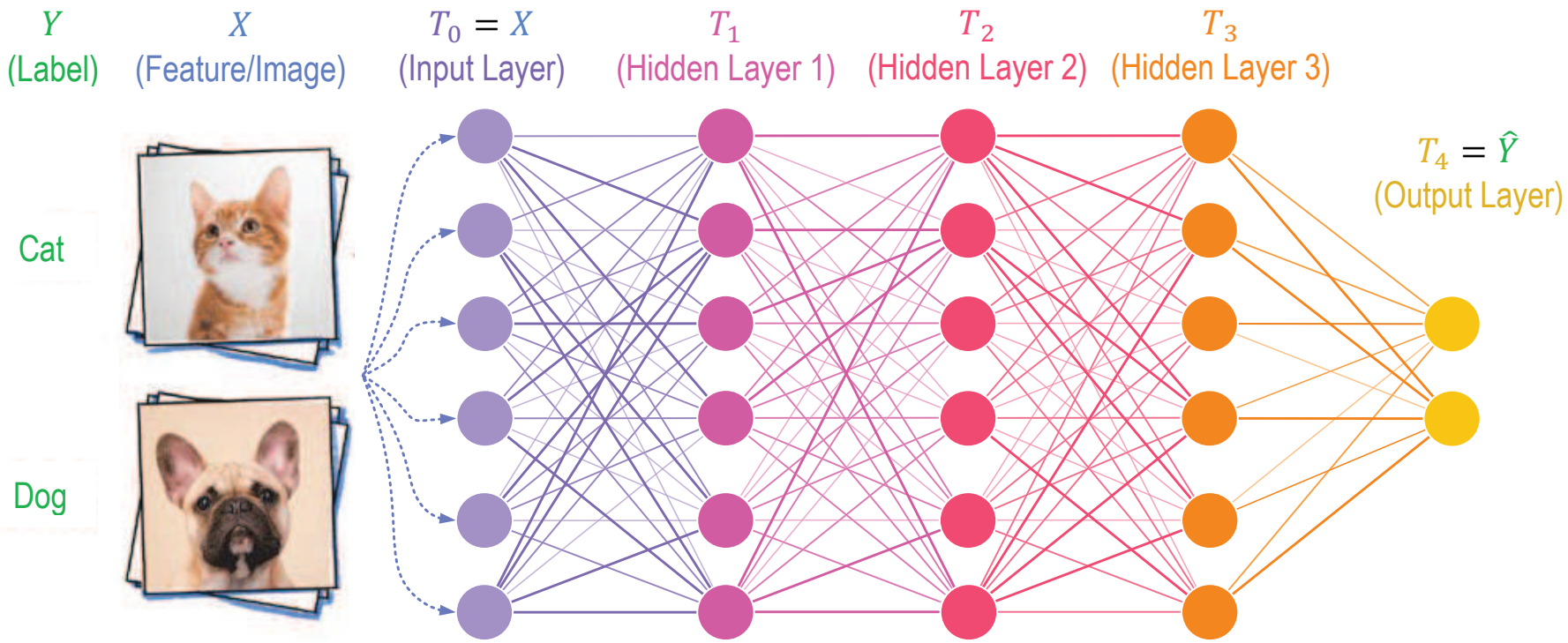
(Deterministic) Feedforward DNN: Each layer $T_\ell = f_\ell(T_{\ell-1})$



IB Theory Claim: Training comprises 2 phases

Setup and Preliminaries

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

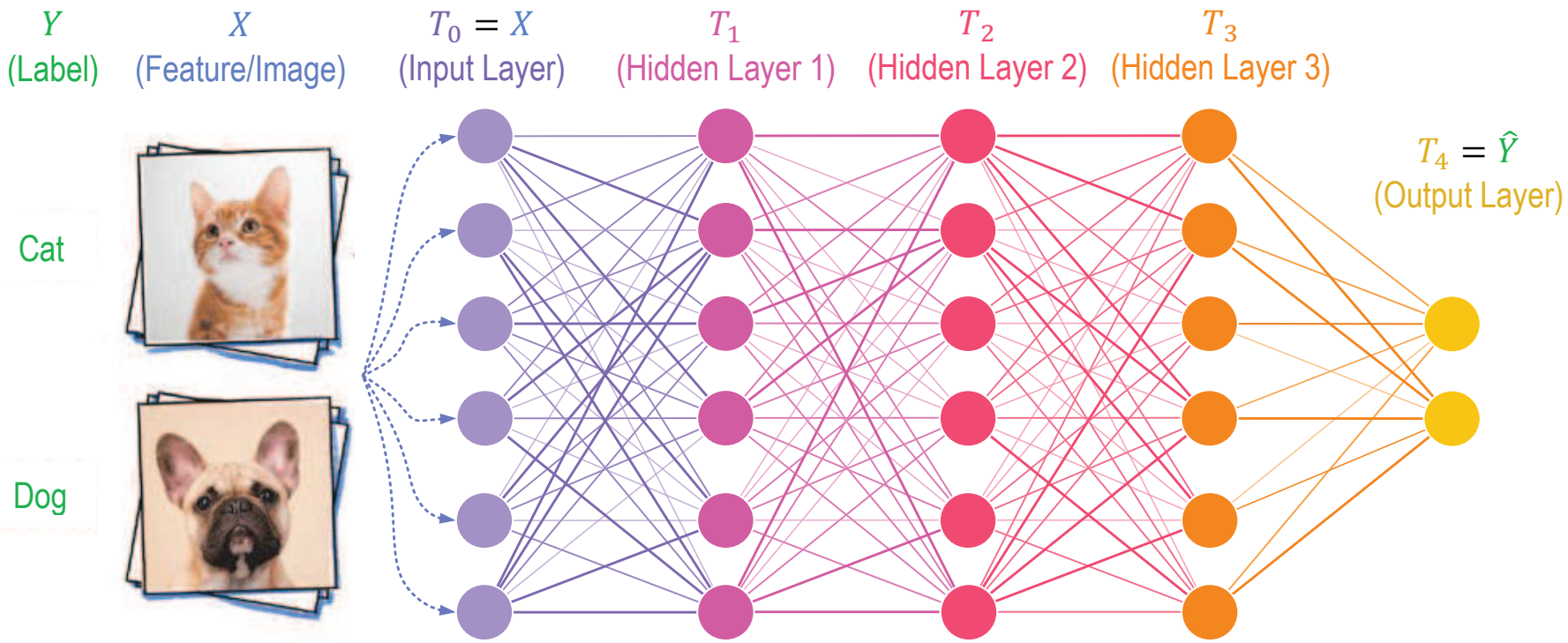


IB Theory Claim: Training comprises 2 phases

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

Setup and Preliminaries

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

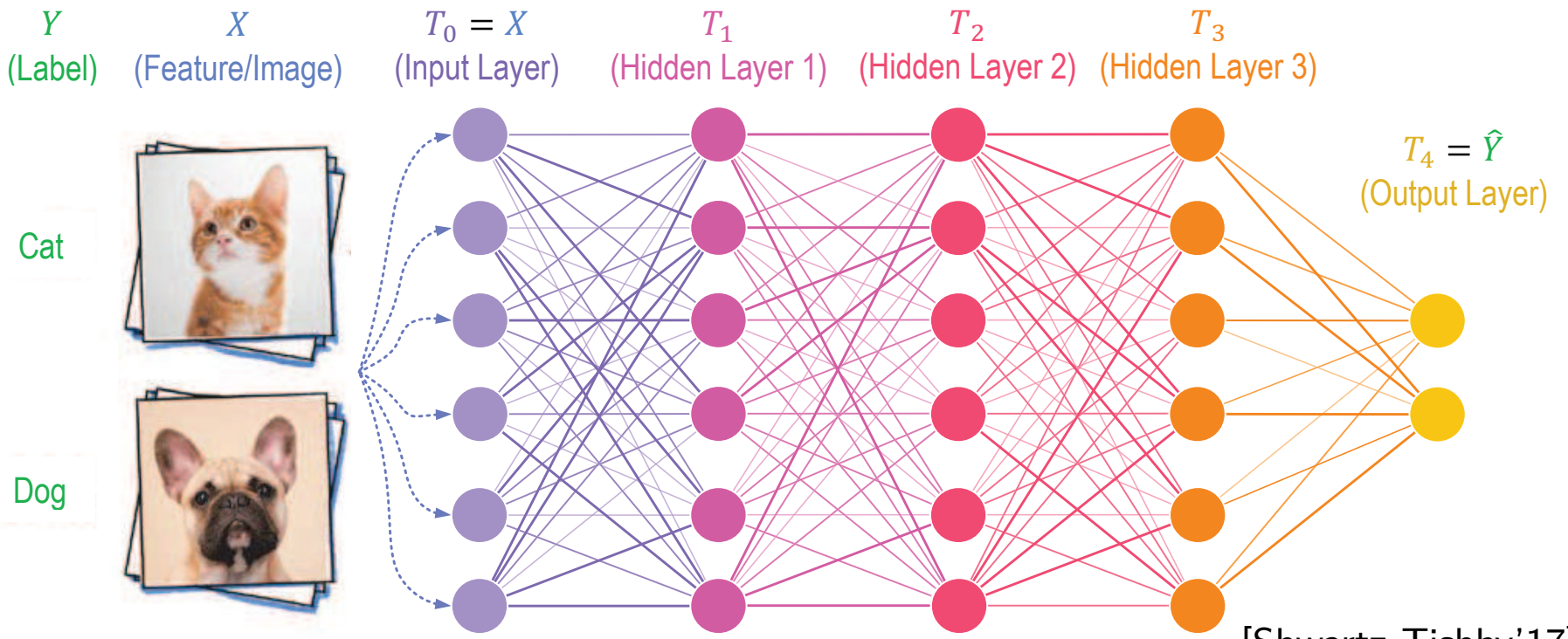


IB Theory Claim: Training comprises 2 phases

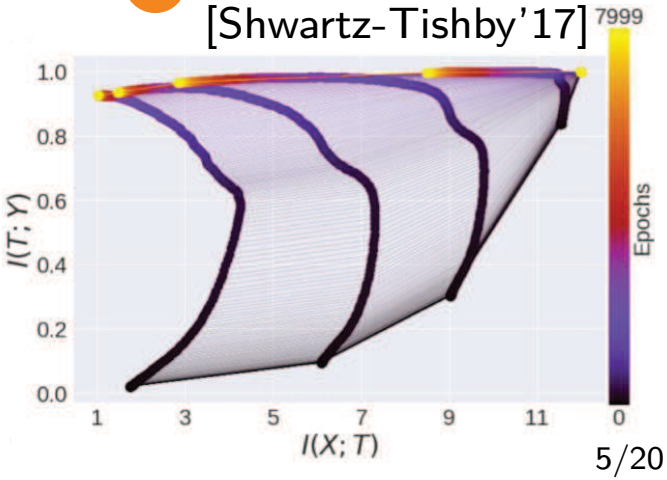
1. **Fitting:** $I(Y;T_\ell)$ & $I(X;T_\ell)$ rise (short)
2. **Compression:** $I(X;T_\ell)$ slowly drops (long)

Setup and Preliminaries

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



[Shwartz-Tishby'17]



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)

Vacuous Mutual Information & Misestimation

Proposition (Informal)

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

Vacuous Mutual Information & Misestimation

Proposition (Informal)

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

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

Vacuous Mutual Information & Misestimation

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)

Vacuous Mutual Information & Misestimation

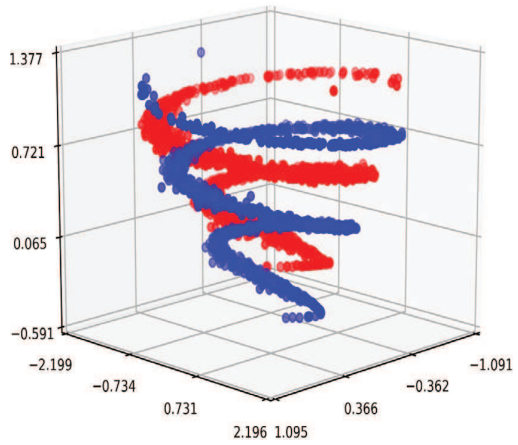
Proposition (Informal)

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)

Feature Space (X)



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

$|\mathcal{X}| = 3000$

Vacuous Mutual Information & Misestimation

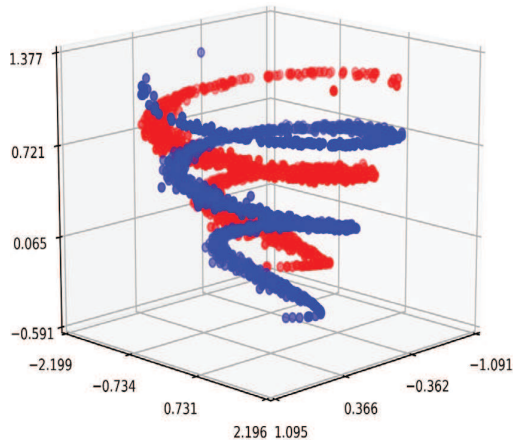
Proposition (Informal)

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

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

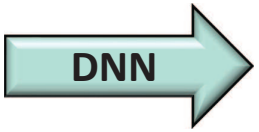
- $I(X;T_\ell)$ a.s. **infinite** (continuous X) or **constant** $H(X)$ (discrete X)

Feature Space (X)

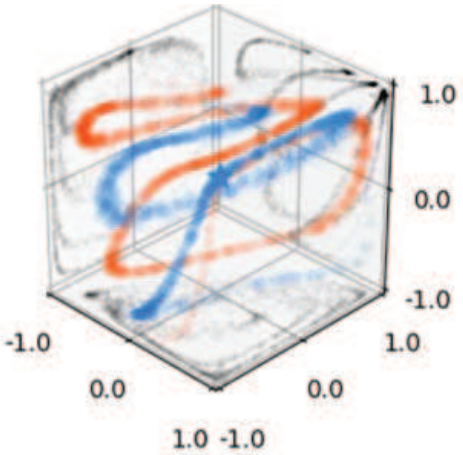


$X \sim \text{Unif}(\mathcal{X})$

$|\mathcal{X}| = 3000$



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



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

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

Vacuous Mutual Information & Misestimation

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)

Vacuous Mutual Information & Misestimation

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

Vacuous Mutual Information & Misestimation

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

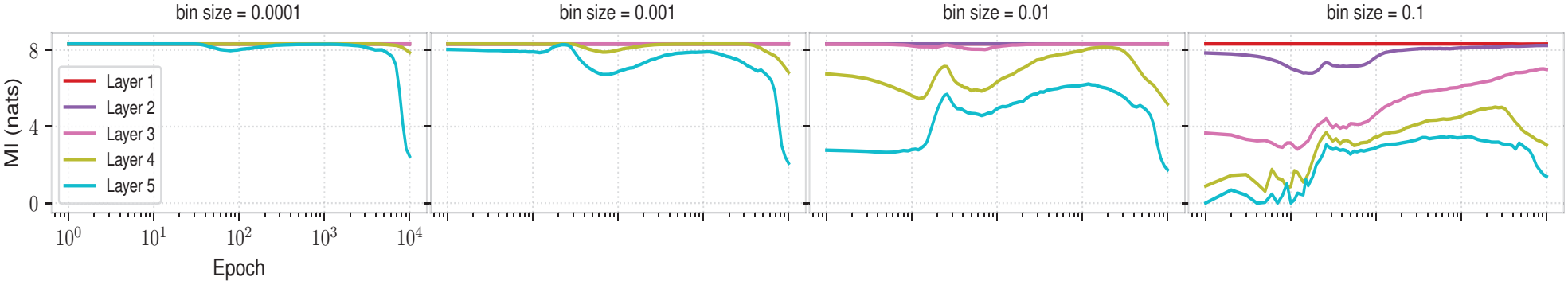
Vacuous Mutual Information & Misestimation

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



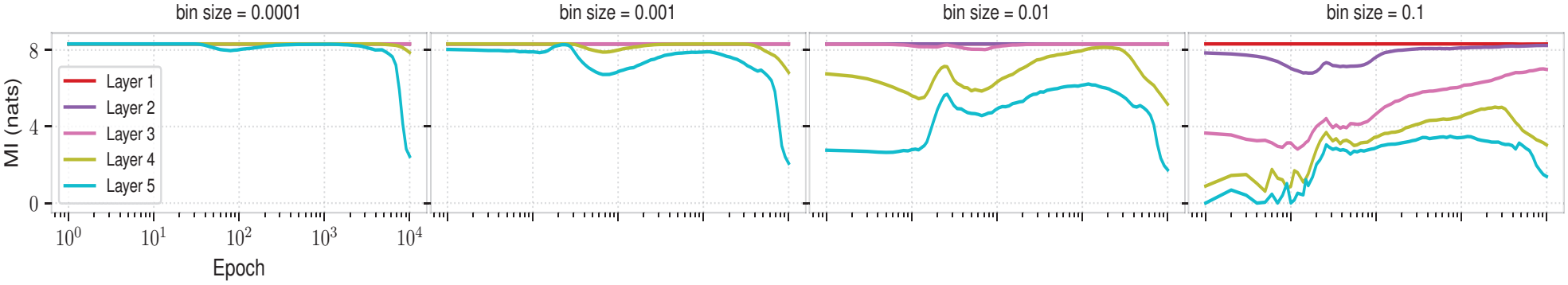
Vacuous Mutual Information & Misestimation

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 $I\big(X;\text{Bin}(T_\ell)\big) \neq I(X;T_\ell)$
  2. $I\big(X;\text{Bin}(T_\ell)\big)$ 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

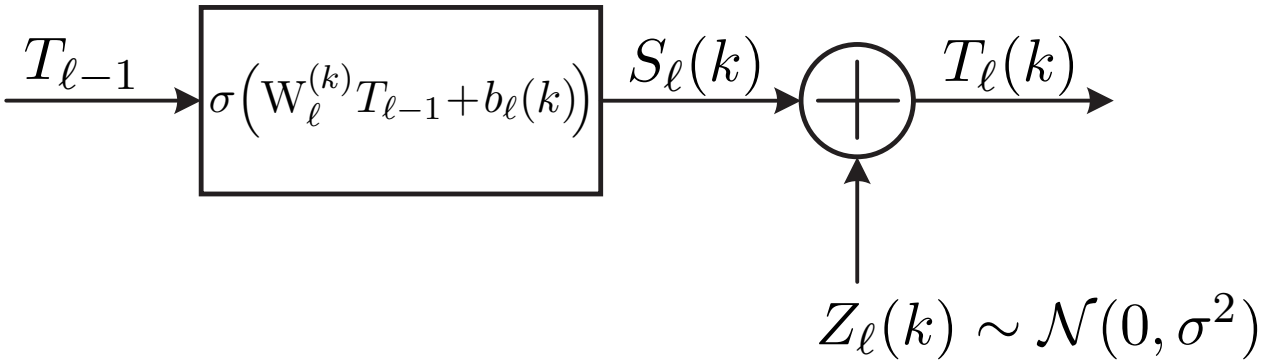
[Goldfeld-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy'18]

Auxiliary Framework - Noisy Deep Neural Networks

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

[Goldfeld-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy'18]

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

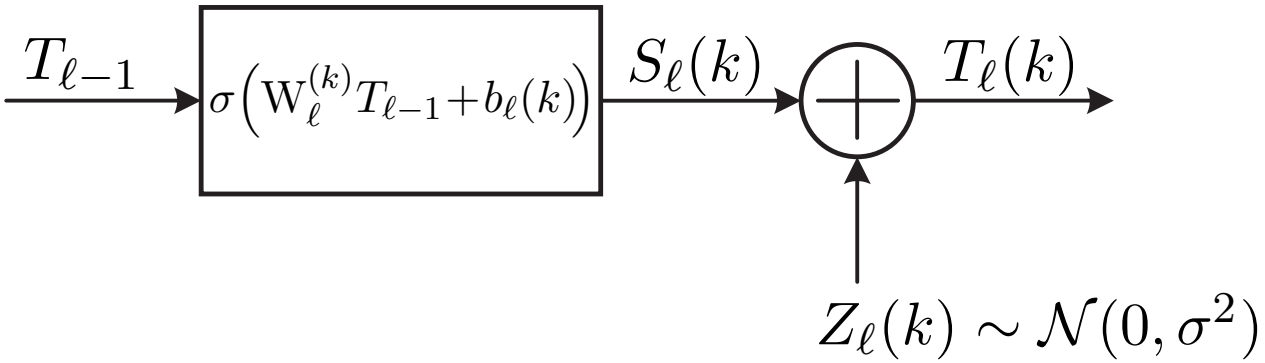


Auxiliary Framework - Noisy Deep Neural Networks

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

[Goldfeld-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy'18]

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



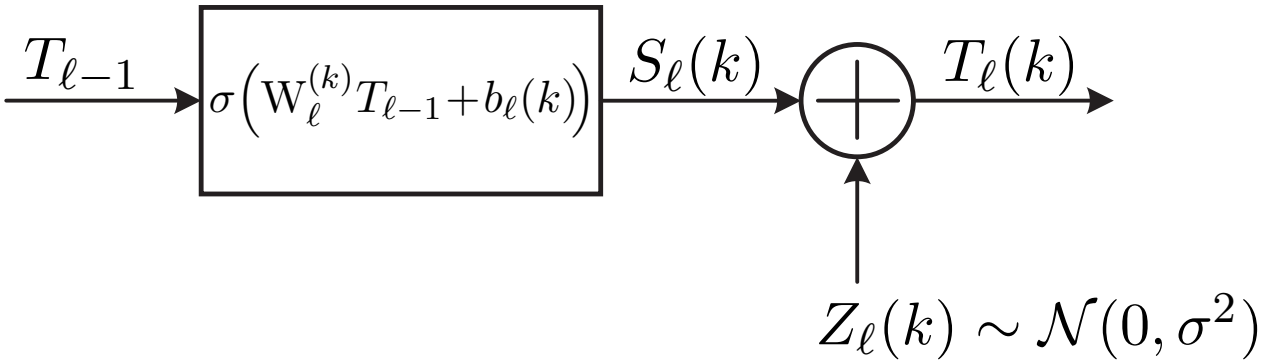
$\implies X \mapsto T_\ell$ is a **parametrized channel** (by DNN's parameters)

Auxiliary Framework - Noisy Deep Neural Networks

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

[Goldfeld-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy'18]

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



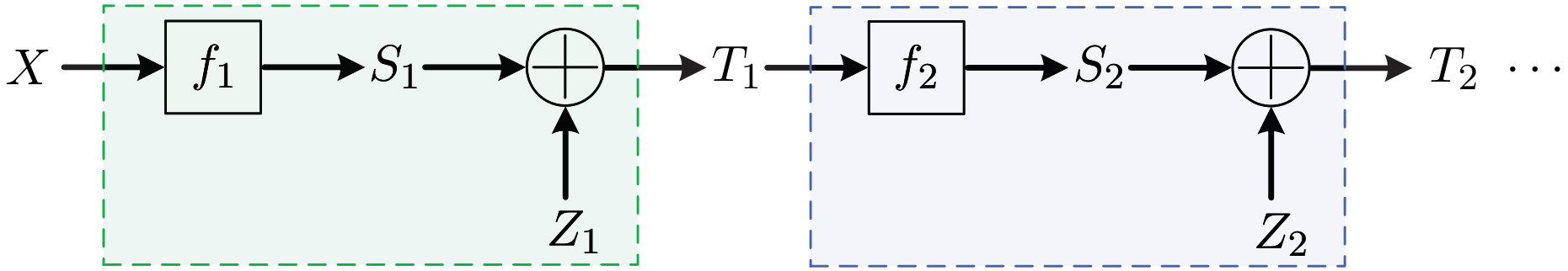
$\Longrightarrow$ $X \mapsto T_\ell$ is a **parametrized channel** (by DNN's parameters)

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

Estimating $I(X;T_\ell)$ in Noisy DNNs

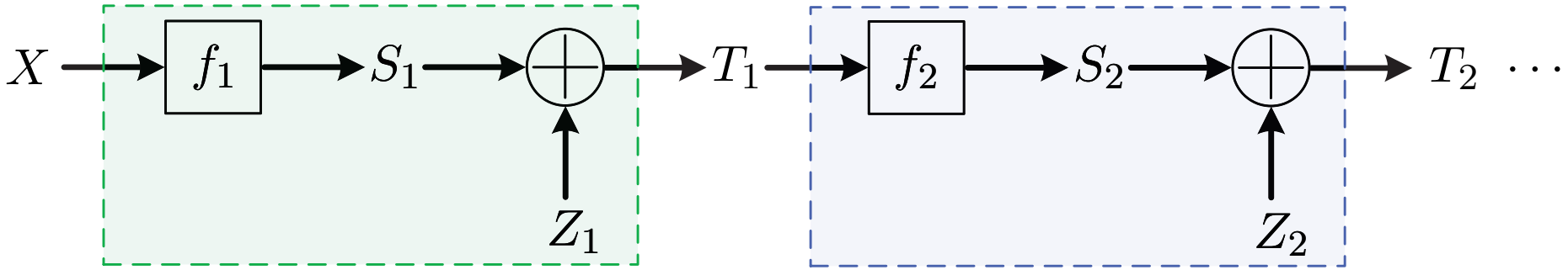
Mutual Information Estimation in Noisy DNNs

Noisy DNN:



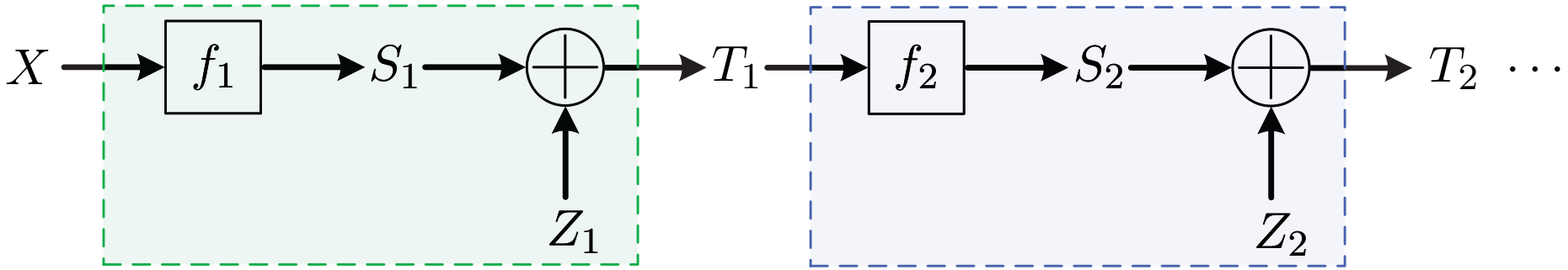
Mutual Information Estimation in Noisy DNNs

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



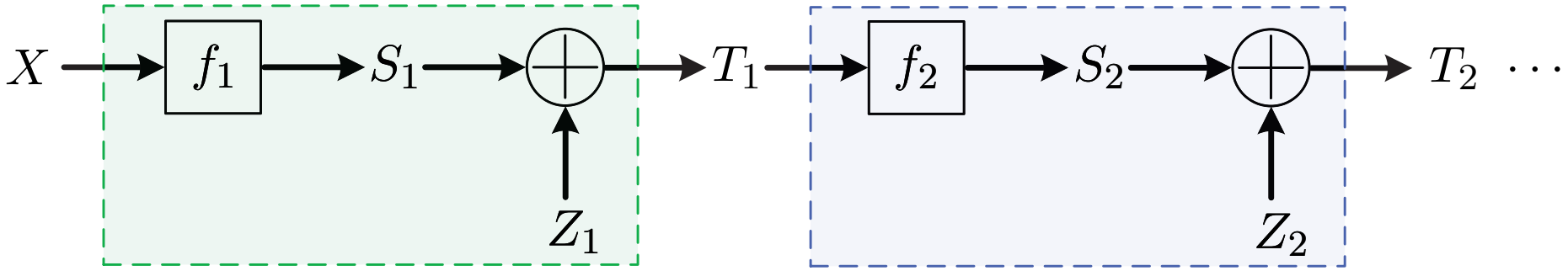
Mutual Information Estimation in Noisy DNNs

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



Mutual Information Estimation in Noisy DNNs

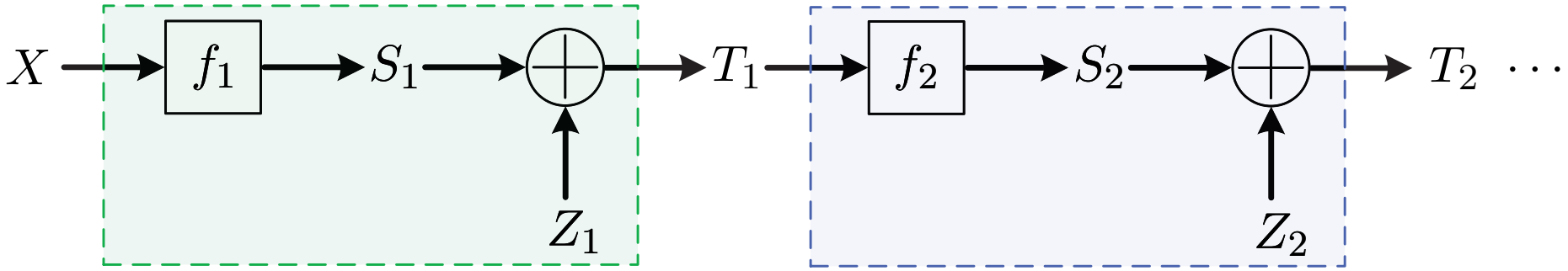
Noisy DNN: $S_\ell \triangleq f_\ell(T_{\ell-1}) \quad \Longrightarrow \quad T_\ell = S_\ell + Z_\ell, \quad Z_\ell \sim \mathcal{N}(0, \sigma^2 \mathrm{I}_d)$



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

Mutual Information Estimation in Noisy DNNs

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

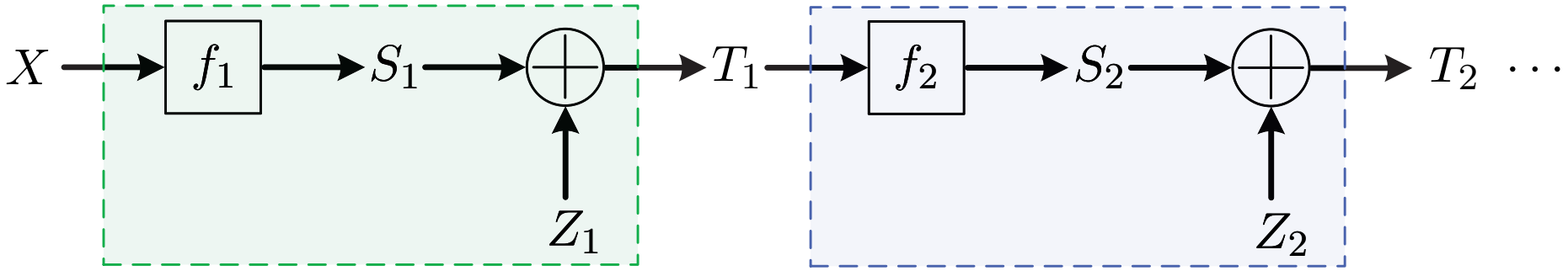


- **Assume:** $X \sim \mathrm{Unif}(\mathcal{X})$, where $\mathcal{X} \triangleq \{x_i\}_{i=1}^m$ is empirical dataset

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

Mutual Information Estimation in Noisy DNNs

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



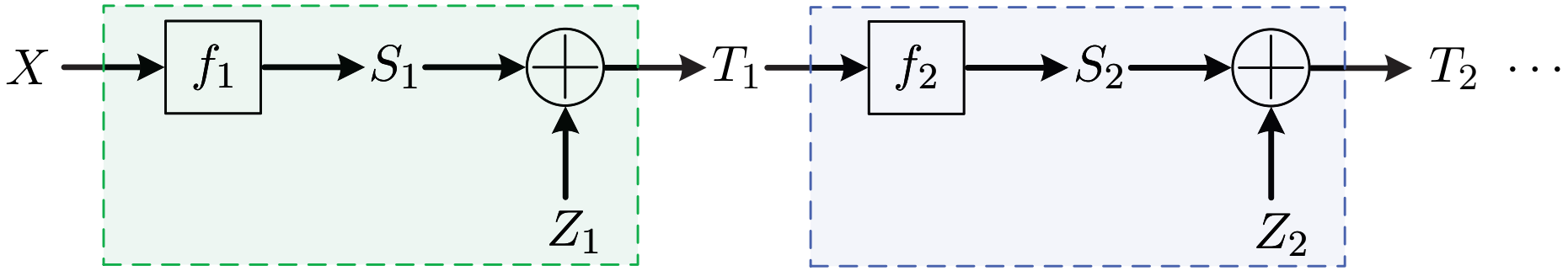
- **Assume:** $X \sim \mathrm{Unif}(\mathcal{X})$, where $\mathcal{X} \triangleq \{x_i\}_{i=1}^m$ is empirical dataset

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

- **Structure:** $S_\ell \perp Z_\ell \quad \Longrightarrow \quad T_\ell = S_\ell + Z_\ell \sim P * \mathcal{N}_\sigma$

Mutual Information Estimation in Noisy DNNs

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



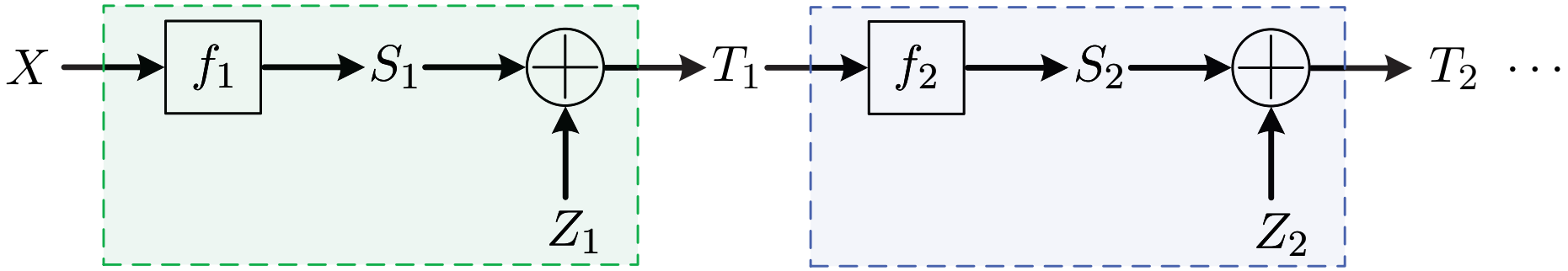
- **Assume:** $X \sim \mathrm{Unif}(\mathcal{X})$, where $\mathcal{X} \triangleq \{x_i\}_{i=1}^m$ is empirical dataset

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

- **Structure:** $S_\ell \perp Z_\ell \quad \Longrightarrow \quad T_\ell = \boldsymbol{S_\ell} + Z_\ell \sim \boldsymbol{P} * \mathcal{N}_\sigma$

Mutual Information Estimation in Noisy DNNs

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



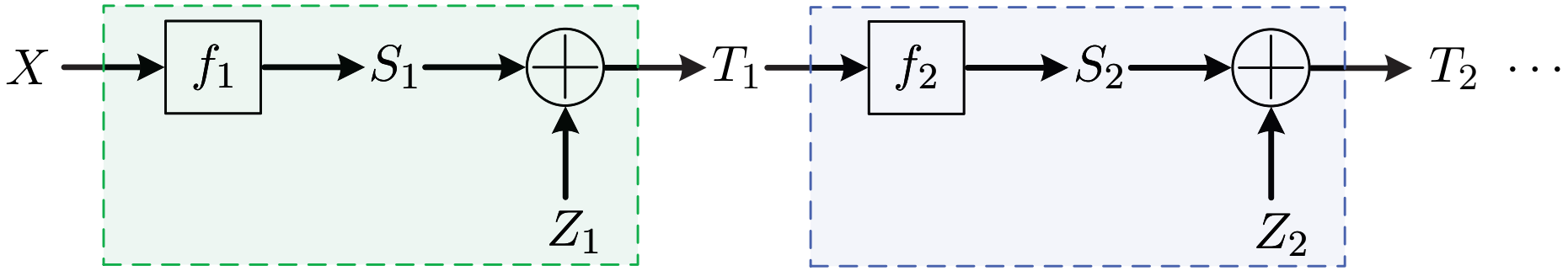
- **Assume:** $X \sim \mathrm{Unif}(\mathcal{X})$, where $\mathcal{X} \triangleq \{x_i\}_{i=1}^m$ is empirical dataset

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

- **Structure:** $S_\ell \perp Z_\ell \quad \Longrightarrow \quad T_\ell = S_\ell + \boldsymbol{Z_\ell} \sim P * \boldsymbol{\mathcal{N}_\sigma}$

Mutual Information Estimation in Noisy DNNs

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



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

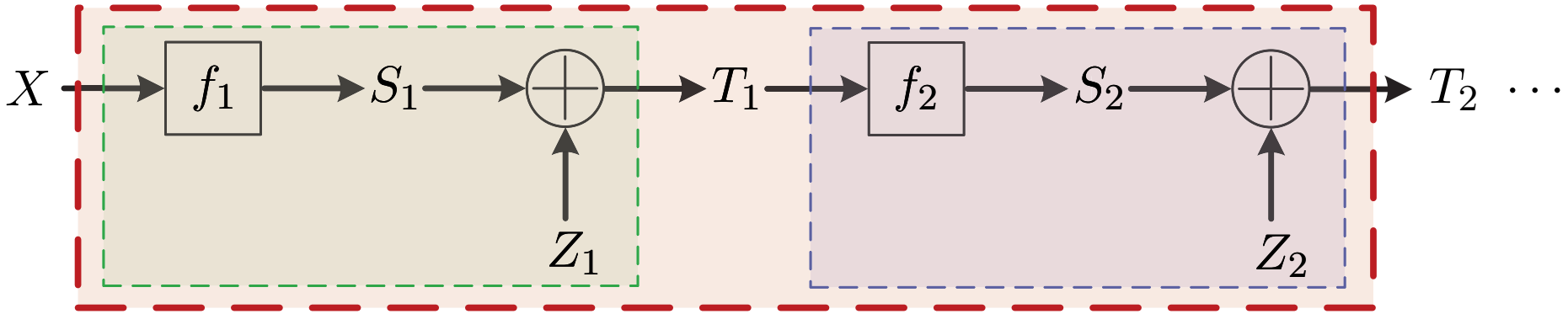
$\Longrightarrow$ **Mutual Information:** $I(X; T_\ell) = h(T_\ell) - \frac{1}{m}\sum_{i=1}^m h(T_\ell | X = x_i)$

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



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

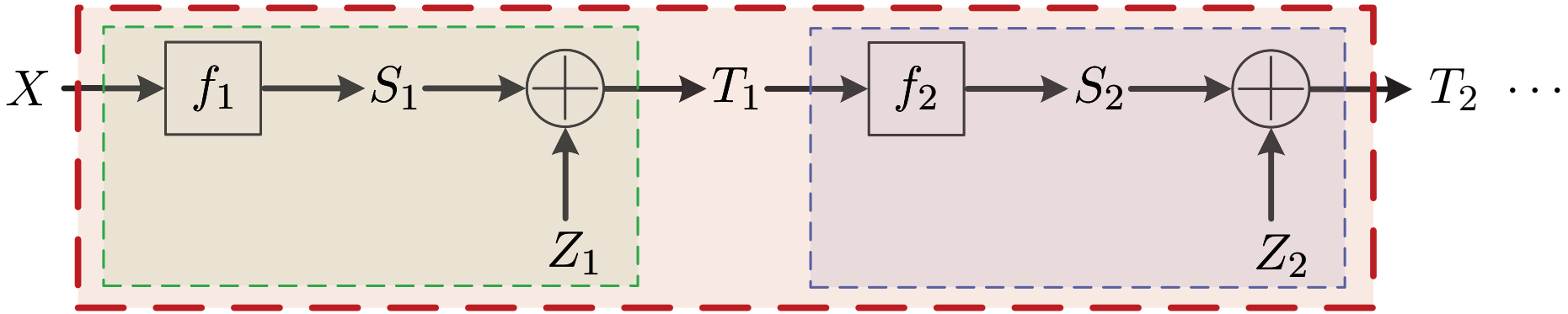
$\Longrightarrow$ **Mutual Information:** $I(X; T_\ell) = h(T_\ell) - \frac{1}{m}\sum_{i=1}^m h(T_\ell | X = x_i)$

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



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

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

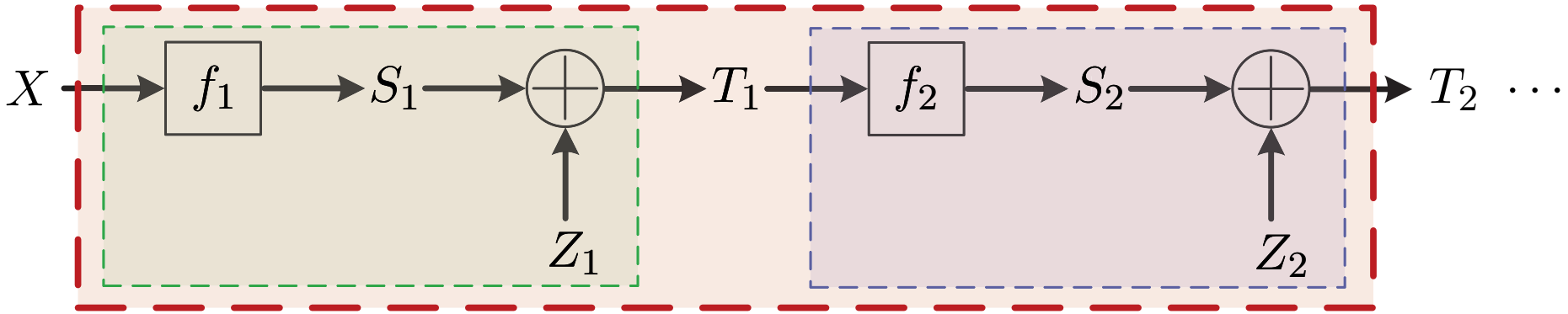
- **Structure:** $S_\ell \perp Z_\ell \quad \Longrightarrow \quad T_\ell = S_\ell + Z_\ell \sim P * \mathcal{N}_\sigma$

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

⊛ **Extremely complicated** $P \quad \Longrightarrow$ Treat as unknown

Mutual Information Estimation in Noisy DNNs

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



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

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

- **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** $P \quad \Longrightarrow$ 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* <u>*unknown*</u> $P \in \mathcal{F}_d$ *(nonparametric class) and* <u>*knowledge*</u> *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* <u>*unknown*</u> $P \in \mathcal{F}_d$ *(nonparametric class) and* <u>*knowledge*</u> *of* $\mathcal{N}_\sigma$ *(noise distribution).*

<u>**Nonparametric Class:**</u> 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* <u>*unknown*</u> $P \in \mathcal{F}_d$ *(nonparametric class) and* <u>*knowledge*</u> *of* $\mathcal{N}_\sigma$ *(noise distribution).*

<u>**Nonparametric Class:**</u> Specified by DNN architecture ($d = T_\ell$ 'width')

<u>**Goal:**</u> 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* <u>*unknown*</u> $P \in \mathcal{F}_d$ *(nonparametric class) and* <u>*knowledge*</u> *of* $\mathcal{N}_\sigma$ *(noise distribution).*

<u>**Nonparametric Class:**</u> Specified by DNN architecture ($d = T_\ell$ 'width')

<u>**Goal:**</u> Simple & parallelizable for efficient implementation

<u>**Estimator:**</u> $\hat{h}(S^n, \sigma) \triangleq h(\hat{P}_{S^n} * \mathcal{N}_\sigma)$, 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 (Goldfeld-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 (Goldfeld-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 (Goldfeld-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 (Goldfeld-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

$$\boxed{C_{\sigma,d,K} = \frac{4}{\sigma^2}\sqrt{32d^2K^4 + d(d+2)\left(\frac{\sigma}{\sqrt{2}} + K\right)^4}\left(\left(\frac{1}{\sqrt{2}} + \frac{K}{\sigma}\right)e^{\frac{3}{8}}\right)^{\frac{d}{2}}}$$

Structured Estimator - Convergence Rate

Theorem (Goldfeld-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 $O\left(n^{-\frac{1}{2}}\right)$

Structured Estimator - Convergence Rate

Theorem (Goldfeld-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 (Goldfeld-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

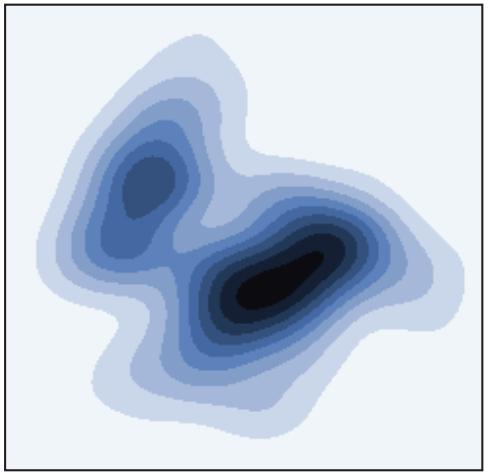
Empirical 1-Wasserstein Distance:

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:

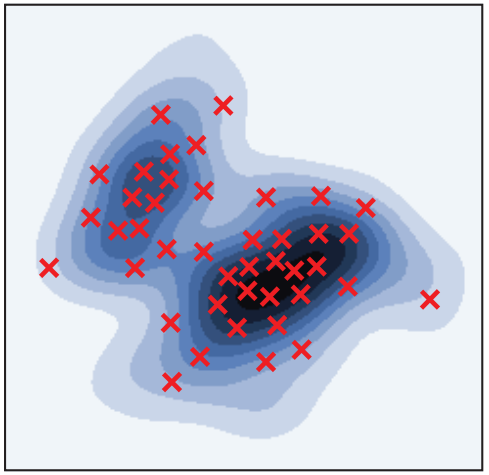
- Distribution P on $\mathbb{R}^d$

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:

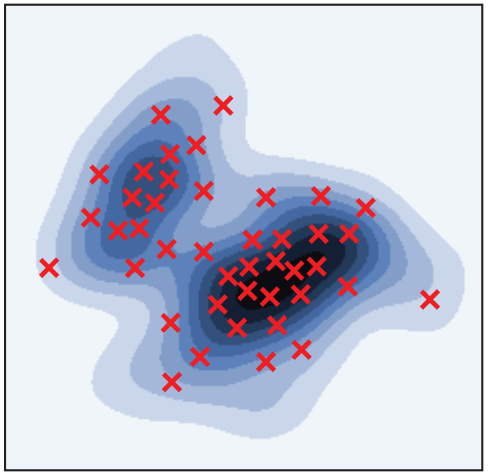
- Distribution P on $\mathbb{R}^d$ $\implies$ i.i.d. Samples $(S_i)_{i=1}^n$

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:

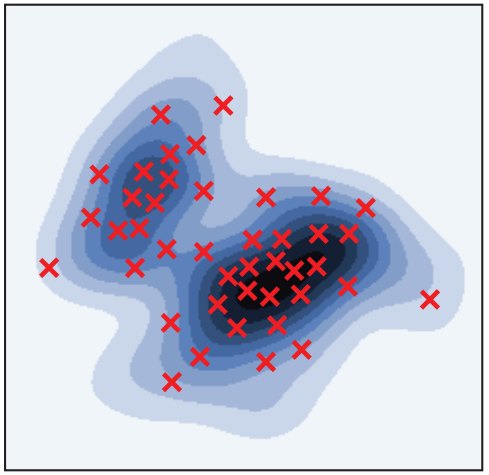
- Distribution P on $\mathbb{R}^d \implies$ i.i.d. Samples $(S_i)_{i=1}^n$
- Empirical distribution $\hat{P}_{S^n} \triangleq \frac{1}{n}\sum_{i=1}^{n} \delta_{S_i}$

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

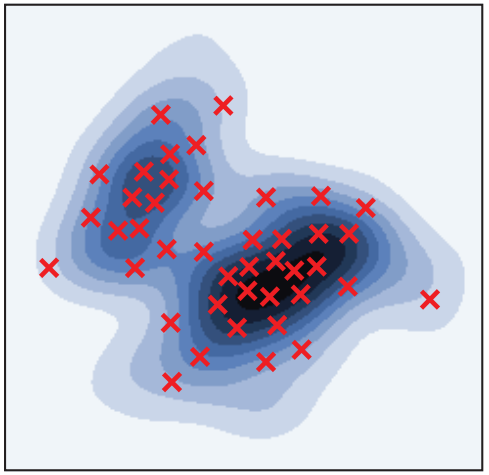
$\implies$ Dependence on (n,d) of $\mathbb{E}W_1(P, \hat{P}_{S^n})$

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

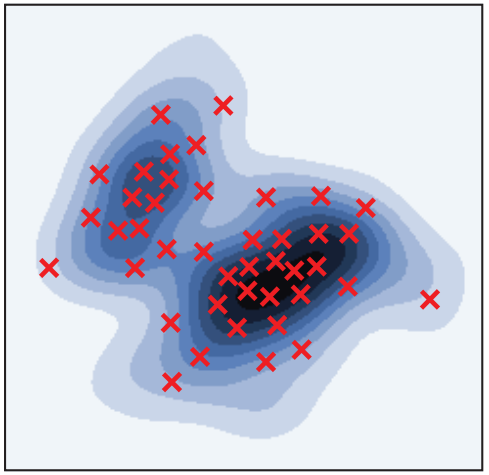
$\implies$ Dependence on (n,d) of $\mathbb{E}W_1(P,\hat{P}_{S^n}) \gtrsim n^{-\frac{1}{d}}$

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

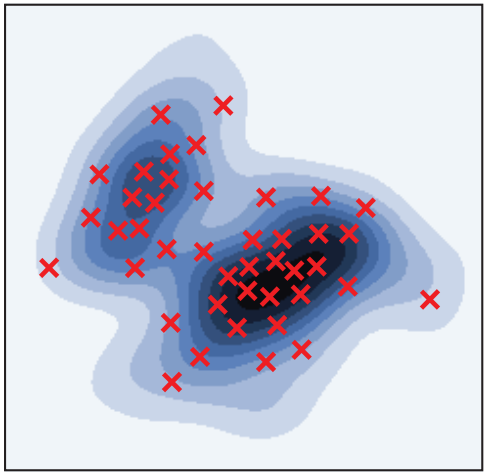
$\implies$ Dependence on (n,d) of $\mathbb{E}W_1(P, \hat{P}_{S^n}) \gtrsim n^{-\frac{1}{d}}$

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 (Goldfeld-Greenewald-Weed-Polyanskiy'19)

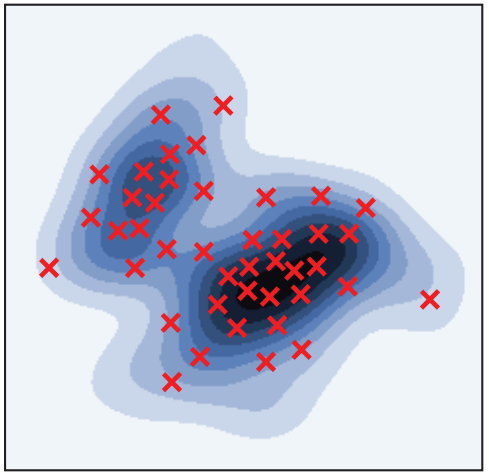
For any d, we have $\mathbb{E}W_1(P * \mathcal{N}_\sigma, \hat{P}_{S^n} * \mathcal{N}_\sigma) \leq O_{\sigma,d}(n^{-\frac{1}{2}})$

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

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

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

Is Exponentiality in Dimension Necessary?

Is Exponentiality in Dimension Necessary?

Theorem (Goldfeld-Greenewald-Polyanskiy'18)

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 (Goldfeld-Greenewald-Polyanskiy'18)

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 (Goldfeld-Greenewald-Polyanskiy'18)

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

Proof (main ideas):

Is Exponentiality in Dimension Necessary?

Theorem (Goldfeld-Greenewald-Polyanskiy'18)

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.

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

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 (Goldfeld-Greenewald-Polyanskiy'18)

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

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]

Back to Noisy DNNs

Back to Noisy DNNs

✓ Simple-to-compute & Parallelizable estimator for $I(X;T_\ell)$

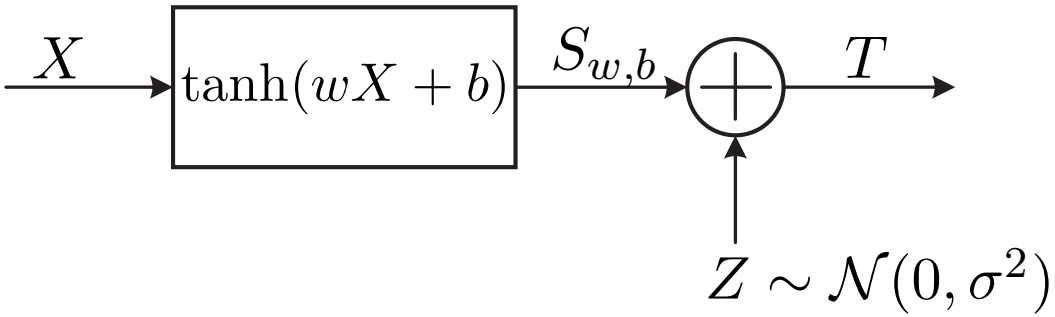
Back to Noisy DNNs

✓ Simple-to-compute & Parallelizable estimator for $I(X; T_\ell)$

✓ Statistically minimax rate optimal

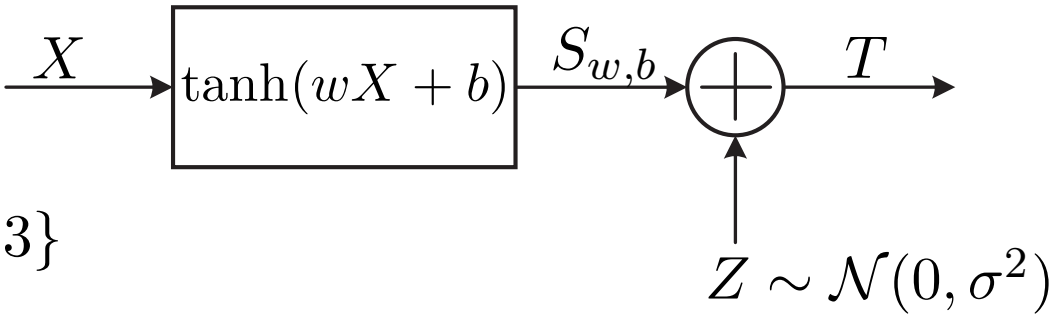
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:



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

Single Neuron Classification:



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

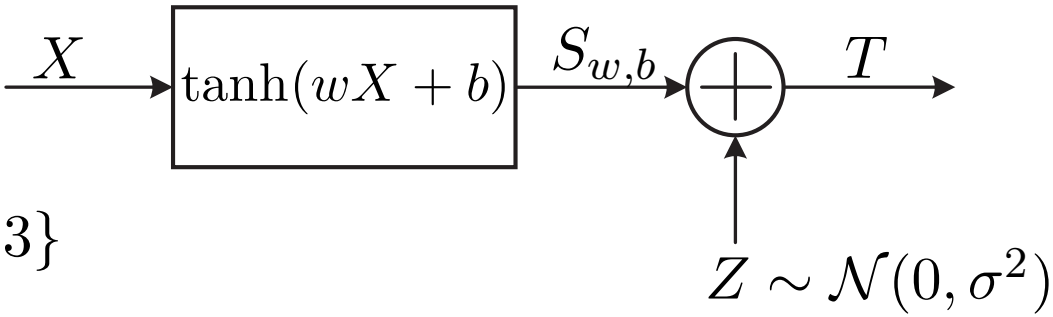
  $\mathcal{X}_{y=-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_{y=1} \triangleq \{3\}$

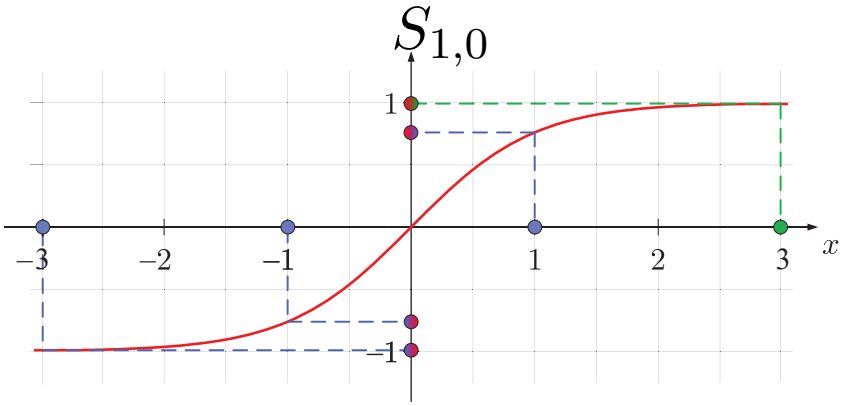
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

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

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



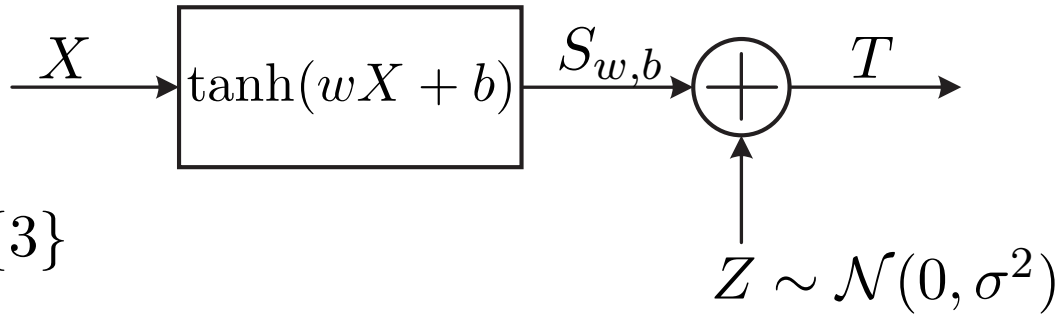


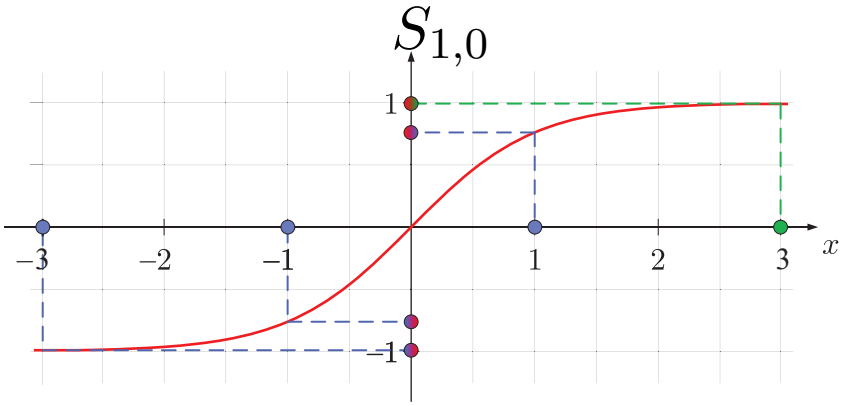
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

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

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





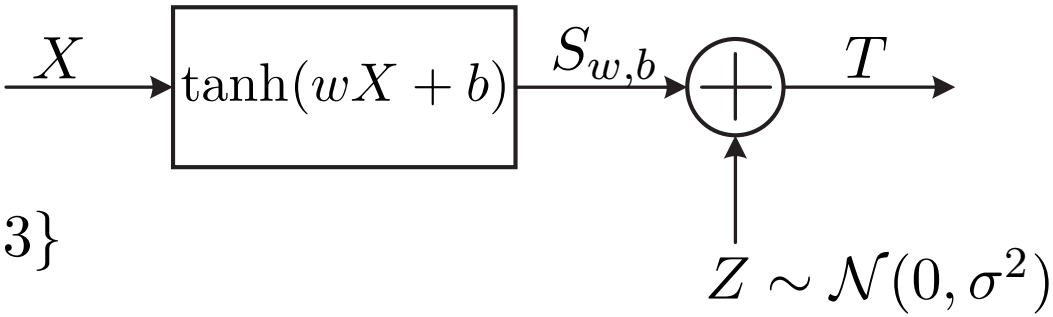
⊛ Center & sharpen transition ($\iff$ increase w and keep $b = -2w$)

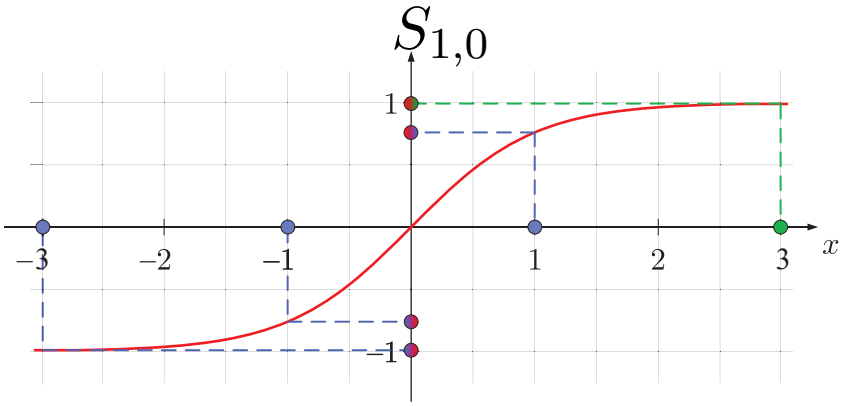
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

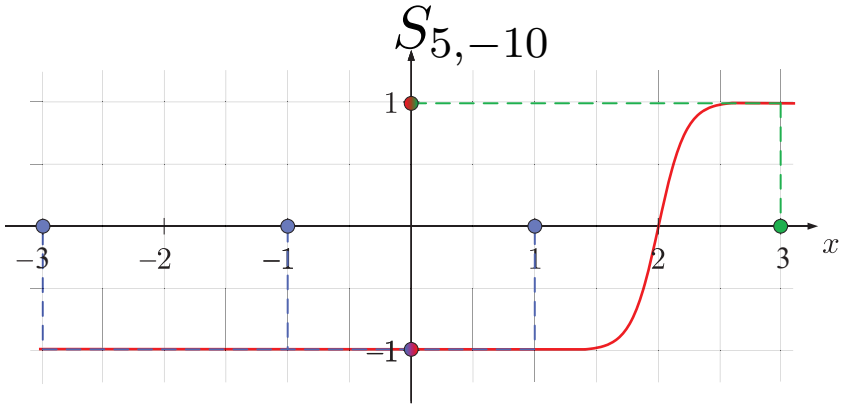
- **Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$

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





$\Longrightarrow$

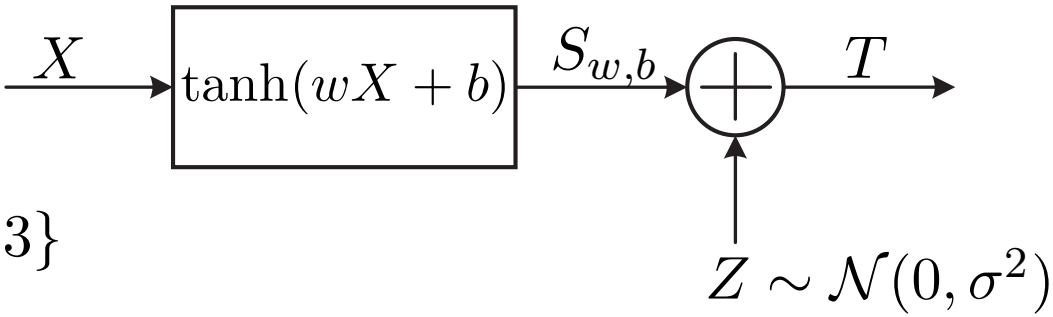


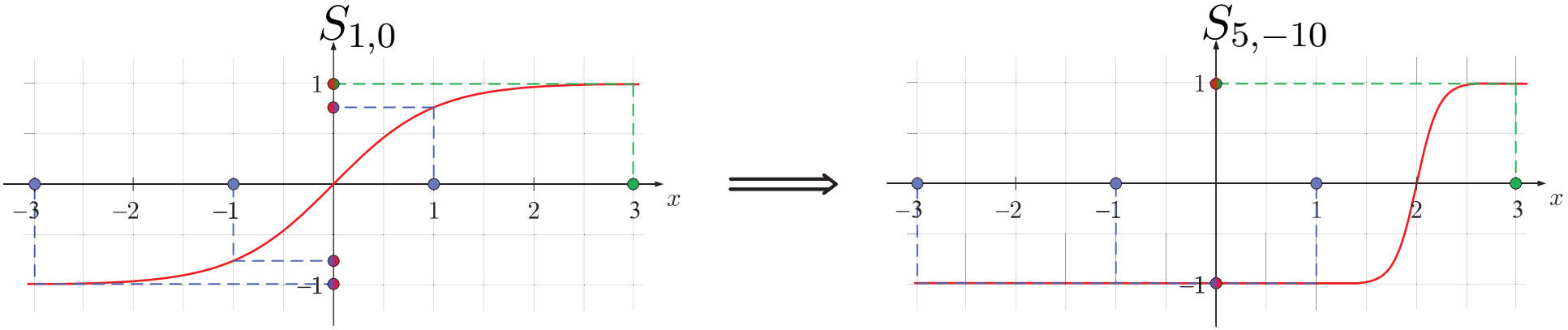
$I(X;T_\ell)$ Dynamics - Illustrative Minimal Example

Single Neuron Classification:

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

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

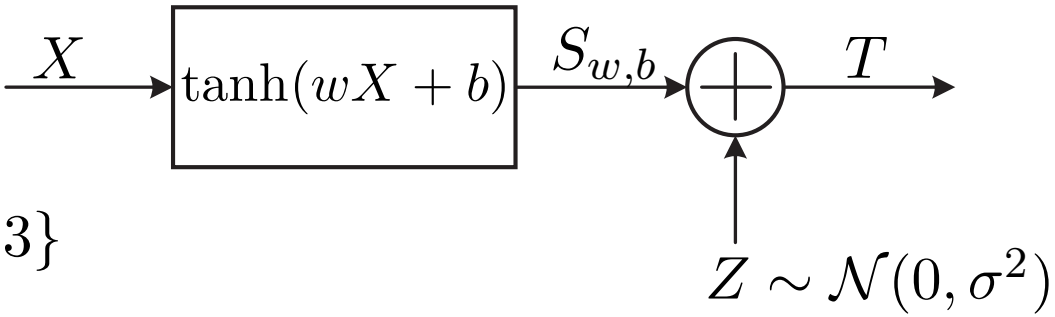




✓ Correct classification performance

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

Single Neuron Classification:



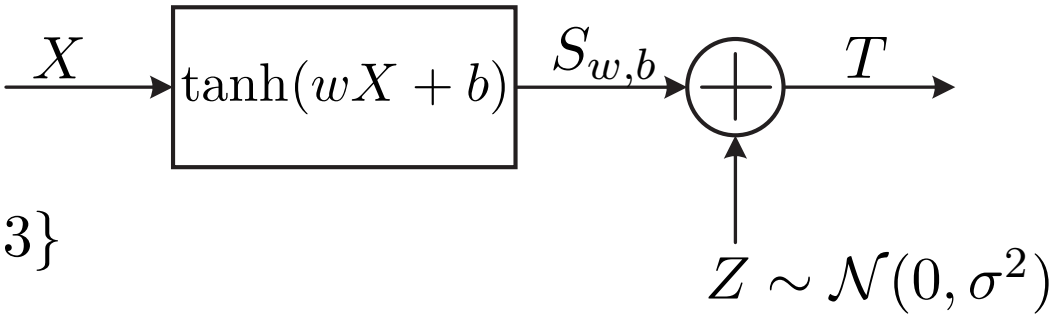
- **Input:** $X \sim \mathsf{Unif}\{\pm 1, \pm 3\}$

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

- **Mutual Information:**

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

Single Neuron Classification:



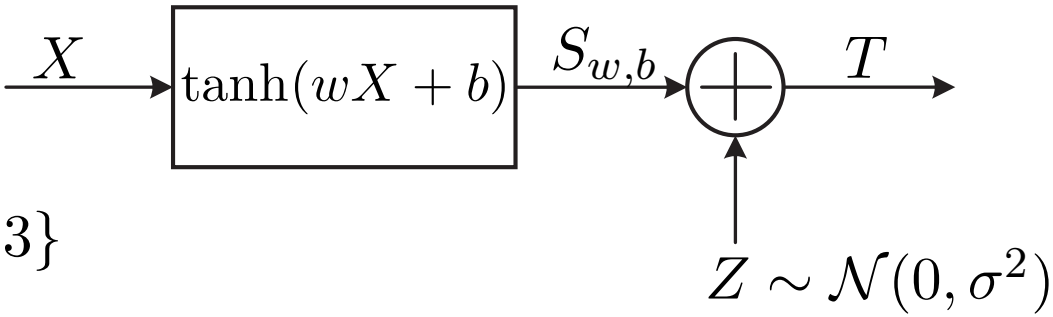
- **Input:** $X \sim \mathsf{Unif}\{\pm 1, \pm 3\}$

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

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

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

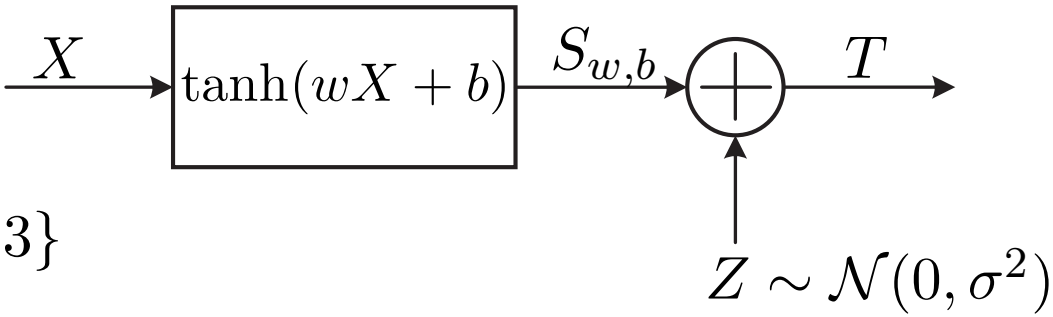
- **Mutual Information:** $I(X;T) = I(S_{w,b}; S_{w,b} + Z)$

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

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

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

Single Neuron Classification:



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

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

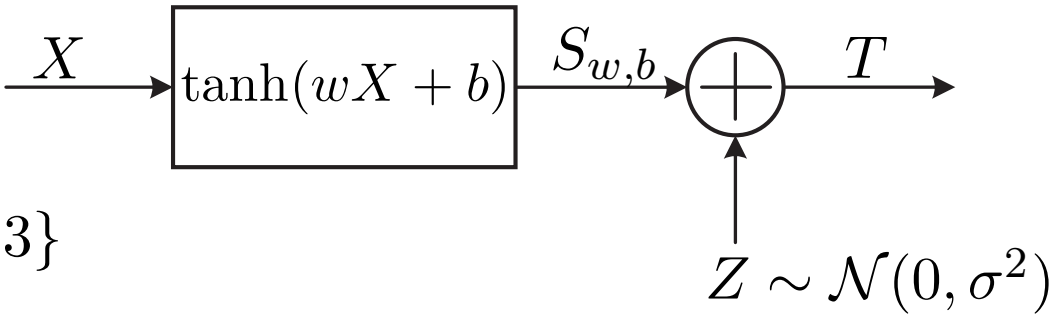
- **Mutual Information:** $I(X;T) = I(S_{w,b}; S_{w,b} + Z)$

$\Longrightarrow$ $I(X;T)$ 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:



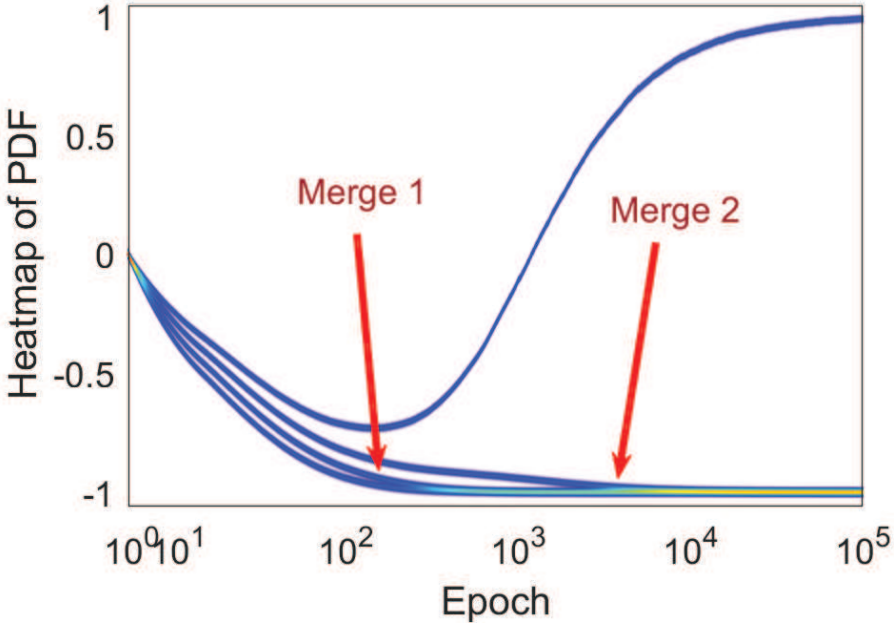
- **Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$

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

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

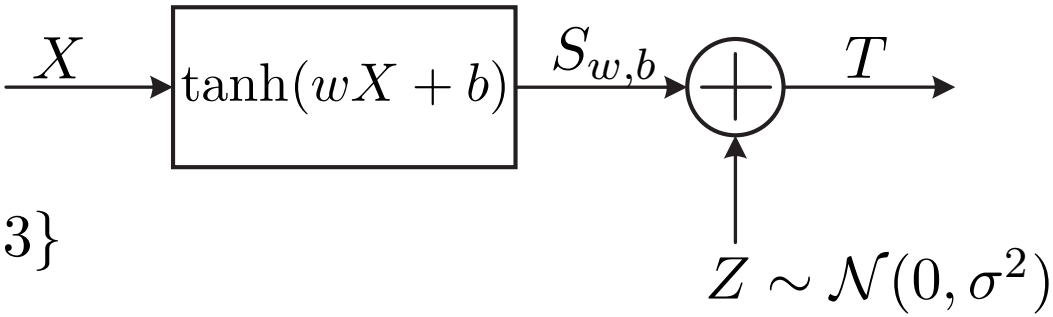
$\Longrightarrow$ $I(X;T)$ is # bits (nats) transmittable over AWGN with symbols

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



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

Single Neuron Classification:



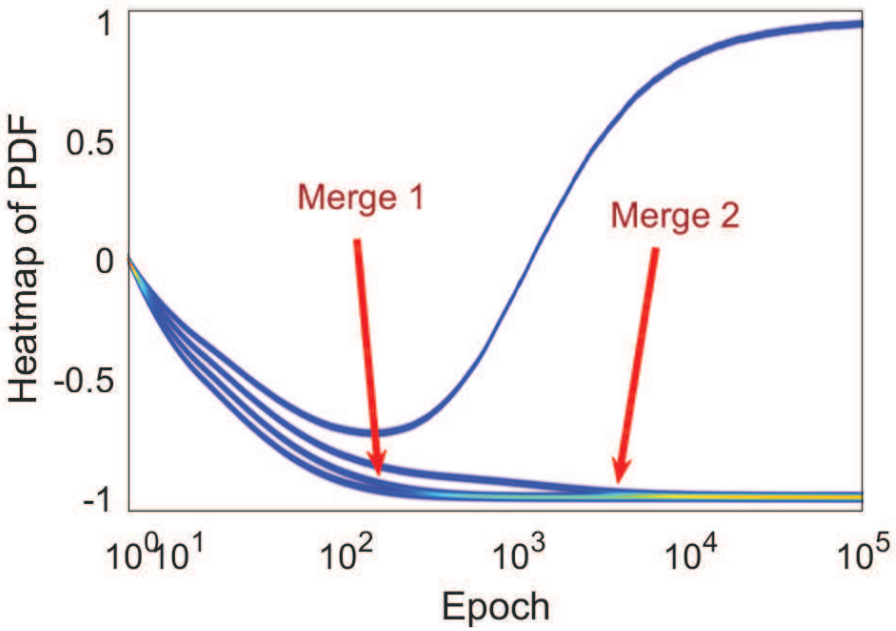
- **Input:** $X \sim \text{Unif}\{\pm 1, \pm 3\}$

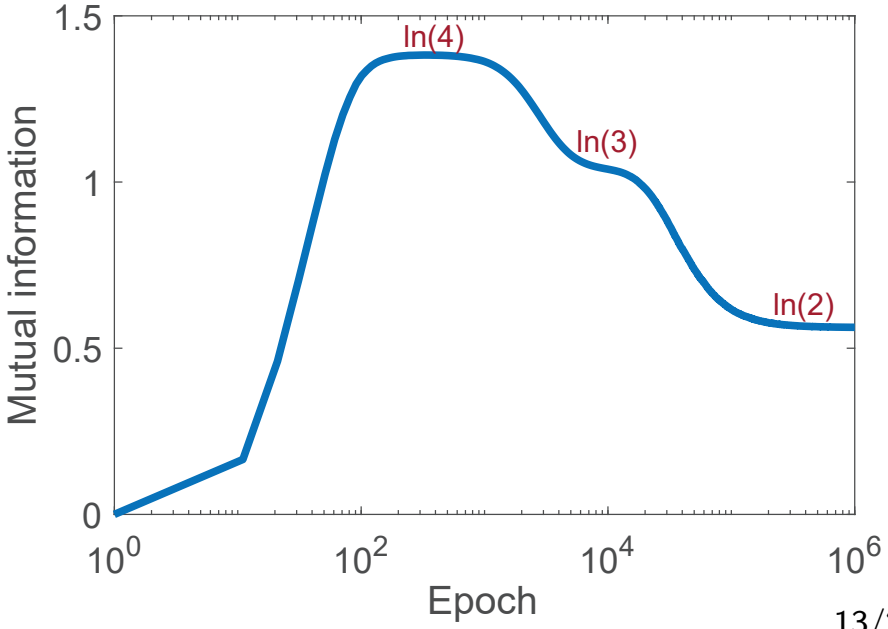
  $\mathcal{X}_{y=-1} \triangleq \{-3,-1,1\}$, $\mathcal{X}_{y=1} \triangleq \{3\}$

- **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 MLP arch.

Clustering of Representations - Larger Networks

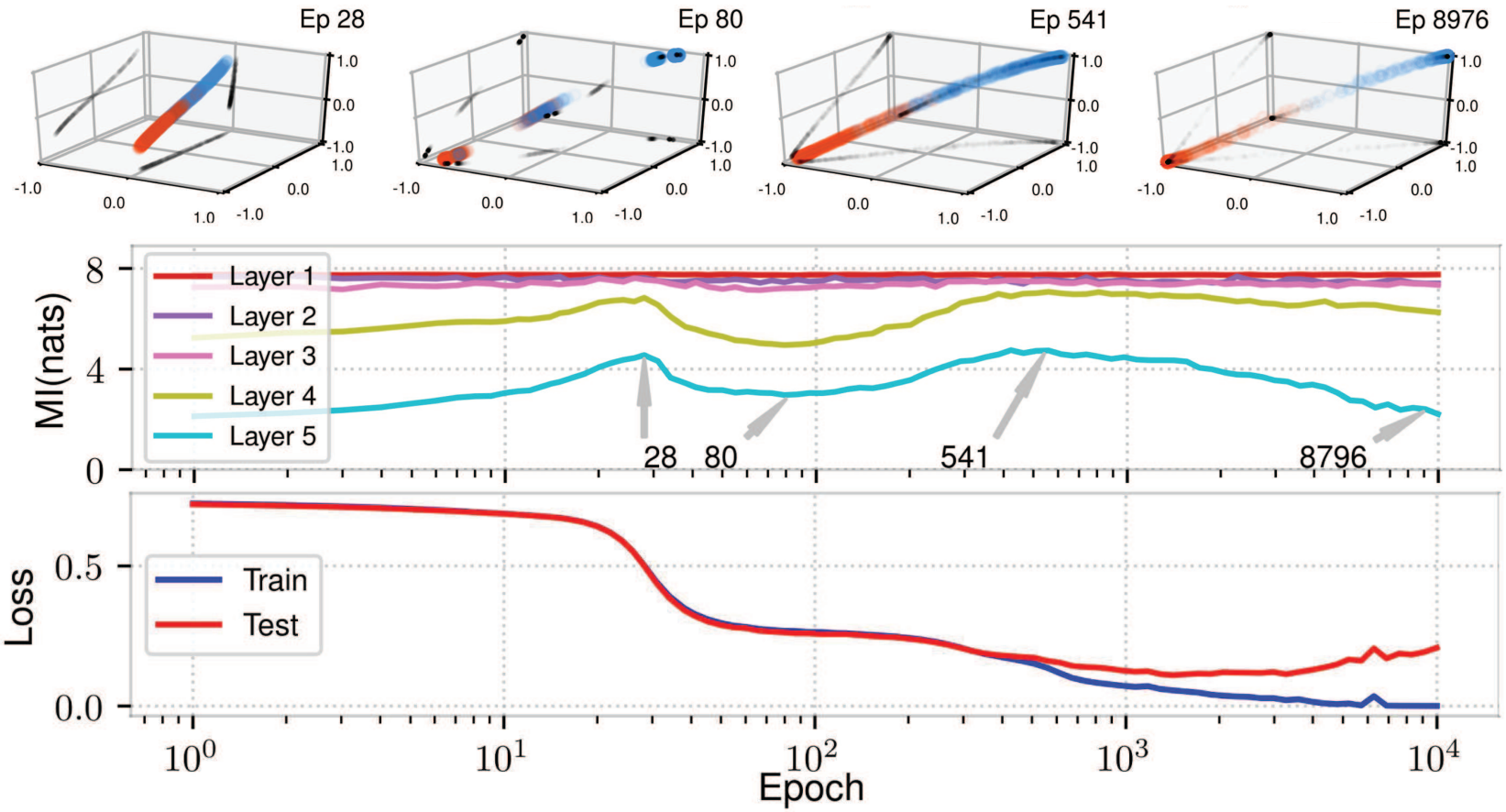
Noisy version of DNN from [Shwartz-Tishby'17]:

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

Clustering of Representations - Larger Networks

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

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



Clustering of Representations - Larger Networks

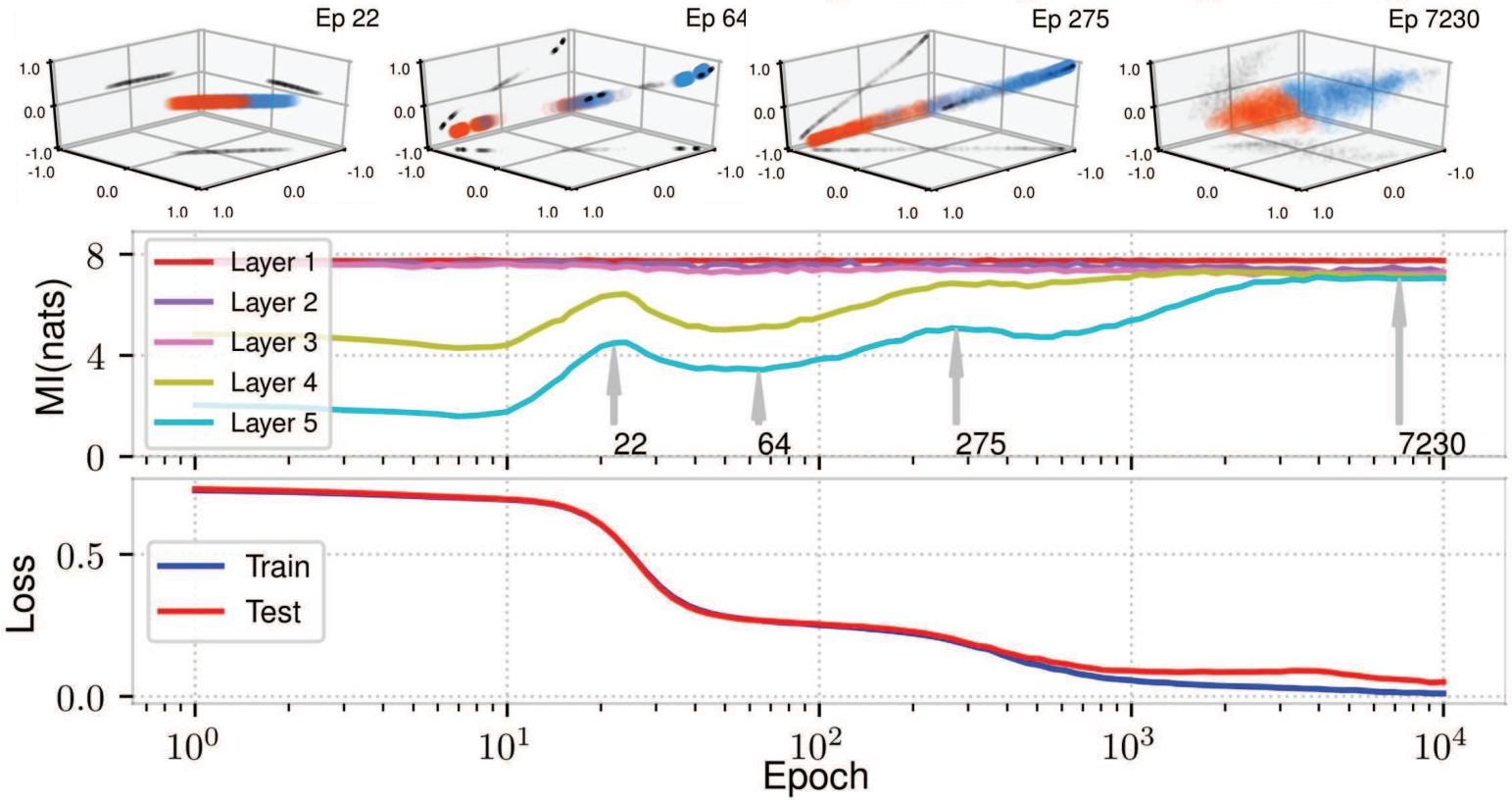
Noisy version of DNN from [Shwartz-Tishby'17]:

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

Clustering of Representations - Larger Networks

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

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



⊛ weight orthonormality regularization

Clustering of Representations - Larger Networks

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

- **Binary Classification:** 12-bit input & 12–**10**–**7**–**5**–**4**–**3**–2 MLP arch.
- **Noise std.:** Set to $\sigma = 0.01$
- Verified in multiple additional experiments [Goldfeld-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy'18]

Clustering of Representations - Larger Networks

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

- **Binary Classification:** 12-bit input & 12–**10**–**7**–**5**–**4**–**3**–2 MLP arch.
- **Noise std.:** Set to $\sigma = 0.01$
- Verified in multiple additional experiments [Goldfeld-Berg-Greenewald-Melnyk-Nguyen-Kingsbury-Polyanskiy'18]

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

Circling Back to Deterministic DNNs

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

Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I(X;\mathrm{Bin}(T_\ell))$

Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I\big(X;\mathsf{Bin}(T_\ell)\big)$

- $T_\ell = \tilde{f}_\ell(X)$

Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I\big(X;\mathsf{Bin}(T_\ell)\big)$

- $T_\ell = \tilde{f}_\ell(X) \quad \implies \quad I\big(X;\mathsf{Bin}(T_\ell)\big) = H\big(\mathsf{Bin}(T_\ell)\big)$

Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I\big(X;\mathsf{Bin}(T_\ell)\big)$

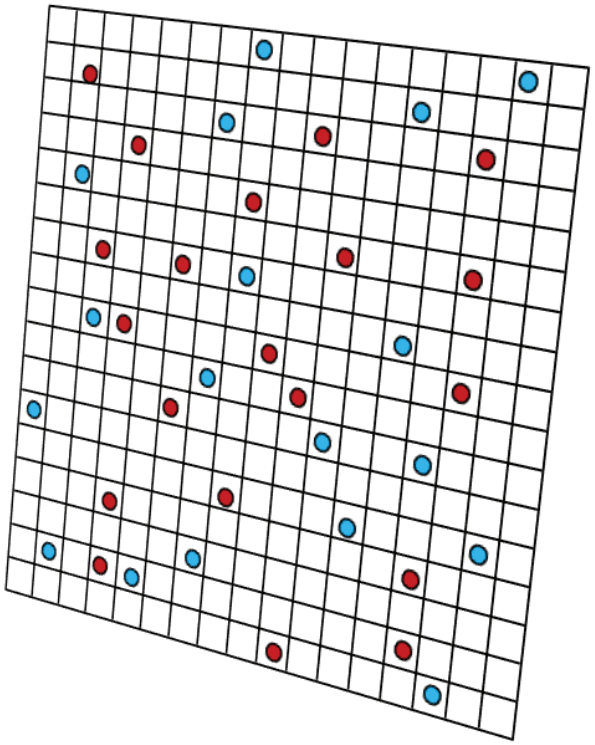
- $T_\ell = \tilde{f}_\ell(X) \quad \implies \quad I\big(X;\mathsf{Bin}(T_\ell)\big) = H\big(\mathsf{Bin}(T_\ell)\big)$
- $H\big(\mathsf{Bin}(T_\ell)\big)$ measures clustering

Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I(X;\text{Bin}(T_\ell))$

- $T_\ell = \tilde{f}_\ell(X) \quad \implies \quad I(X;\text{Bin}(T_\ell)) = H(\text{Bin}(T_\ell))$
- $H(\text{Bin}(T_\ell))$ measures clustering $\uparrow$

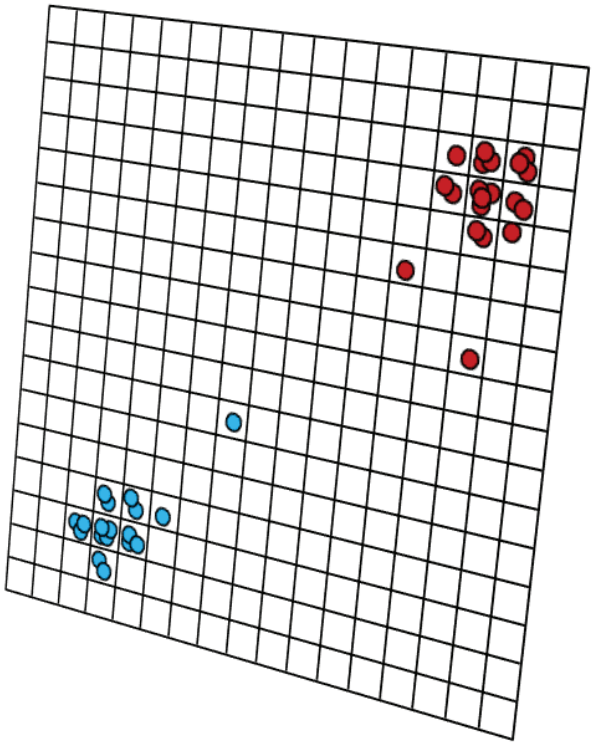


Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I(X;\text{Bin}(T_\ell))$

- $T_\ell = \tilde{f}_\ell(X) \quad \implies \quad I(X;\text{Bin}(T_\ell)) = H(\text{Bin}(T_\ell))$
- $H(\text{Bin}(T_\ell))$ measures clustering $\downarrow$



Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I\big(X;\mathsf{Bin}(T_\ell)\big)$

- $T_\ell = \tilde{f}_\ell(X) \quad \implies \quad I\big(X;\mathsf{Bin}(T_\ell)\big) = H\big(\mathsf{Bin}(T_\ell)\big)$
- $H\big(\mathsf{Bin}(T_\ell)\big)$ measures clustering

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

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

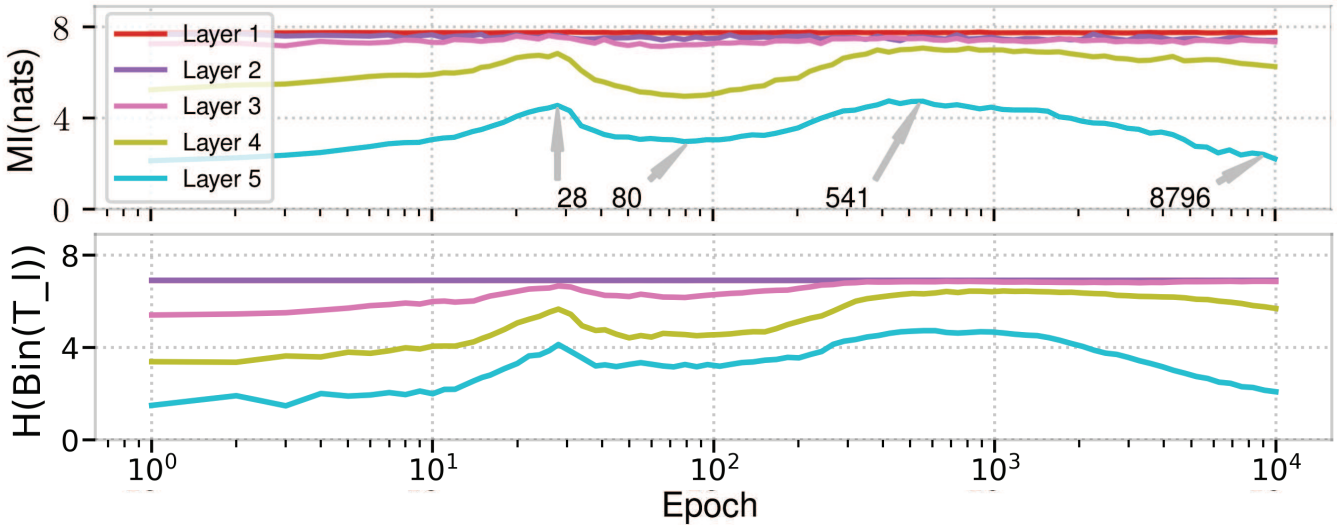
Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I(X;\text{Bin}(T_\ell))$

- $T_\ell = \tilde{f}_\ell(X) \implies I(X;\text{Bin}(T_\ell)) = H(\text{Bin}(T_\ell))$
- $H(\text{Bin}(T_\ell))$ measures clustering

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



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

Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I\big(X;\mathsf{Bin}(T_\ell)\big)$

- $T_\ell = \tilde{f}_\ell(X) \quad \implies \quad I\big(X;\mathsf{Bin}(T_\ell)\big) = H\big(\mathsf{Bin}(T_\ell)\big)$
- $H\big(\mathsf{Bin}(T_\ell)\big)$ measures clustering

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

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

Reexamining Past Measurements: Computed $I\big(X;\mathsf{Bin}(T_\ell)\big)$

- $T_\ell = \tilde{f}_\ell(X) \quad \implies \quad I\big(X;\mathsf{Bin}(T_\ell)\big) = H\big(\mathsf{Bin}(T_\ell)\big)$
- $H\big(\mathsf{Bin}(T_\ell)\big)$ measures clustering

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)!**

Bi-Product Results:

Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I\big(X;\mathsf{Bin}(T_\ell)\big)$

- $T_\ell = \tilde{f}_\ell(X) \quad \implies \quad I\big(X;\mathsf{Bin}(T_\ell)\big) = H\big(\mathsf{Bin}(T_\ell)\big)$
- $H\big(\mathsf{Bin}(T_\ell)\big)$ measures clustering

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)!**

Bi-Product Results:

1. Refute 'compression (tight clustering) improves generalization' claim

Circling Back to Deterministic DNNs

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

Reexamining Past Measurements: Computed $I(X;\text{Bin}(T_\ell))$

- $T_\ell = \tilde{f}_\ell(X) \implies I(X;\text{Bin}(T_\ell)) = H(\text{Bin}(T_\ell))$
- $H(\text{Bin}(T_\ell))$ measures clustering

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)!**

Bi-Product Results:

1. Refute 'compression (tight clustering) improves generalization' claim
2. Computational feasibility of tracking clustering

Towards Broader Impact

Towards Broader Impact

Towards Broader Impact

Towards Broader Impact

Towards Broader Impact

Design

Towards Broader Impact

Design **Optimize**

Towards Broader Impact

Design | **Optimize** | **Understand**

Towards Broader Impact: Design

How to Better Design DNNs?

Towards Broader Impact: Design

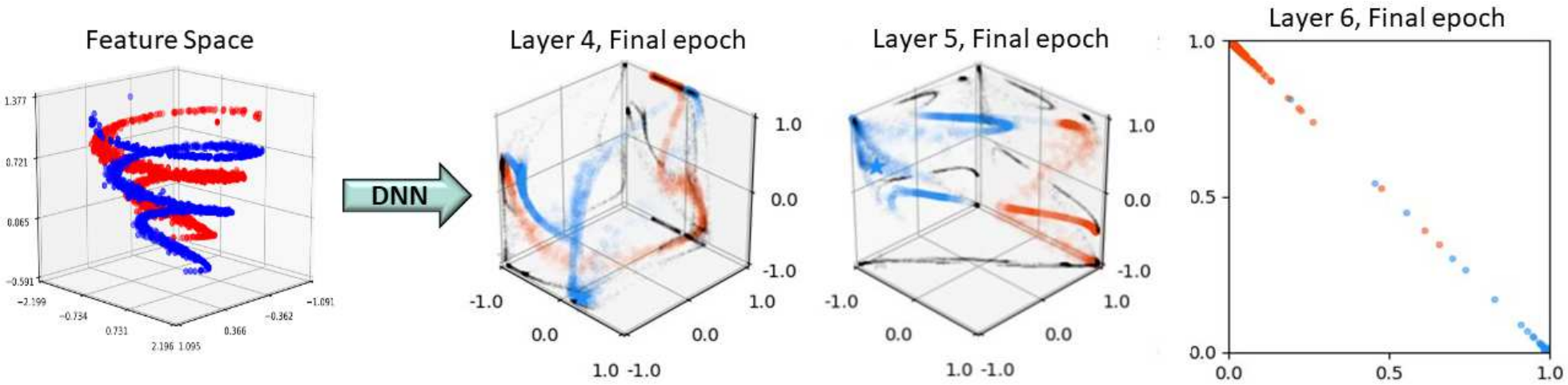
How to Better Design DNNs?

- T_ℓ compression $\implies$ $T_{\ell-1}$ linear separation

Towards Broader Impact: Design

How to Better Design DNNs?

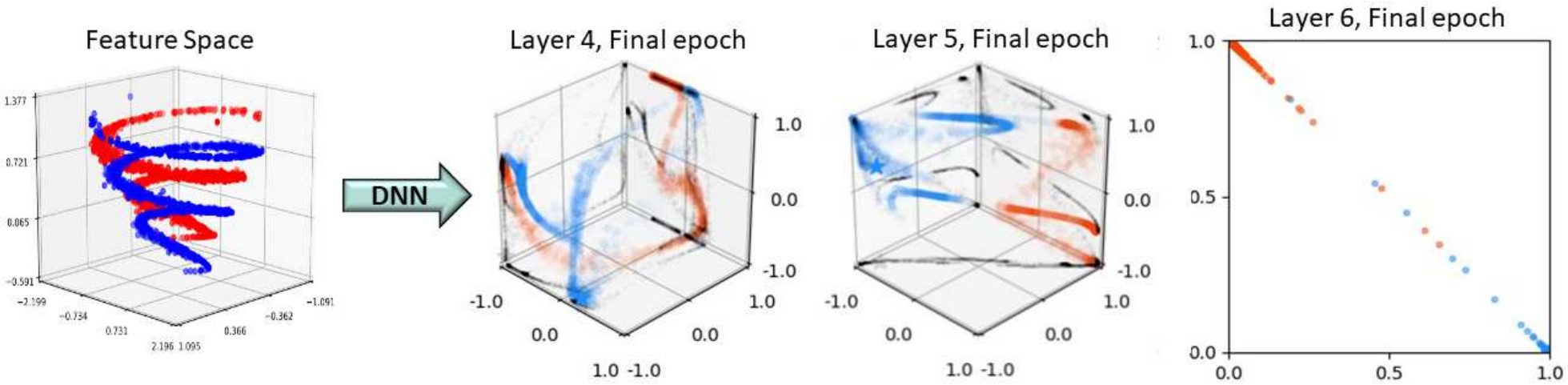
- T_ℓ compression $\implies$ $T_{\ell-1}$ linear separation



Towards Broader Impact: Design

How to Better Design DNNs?

- T_ℓ compression $\implies$ $T_{\ell-1}$ linear separation

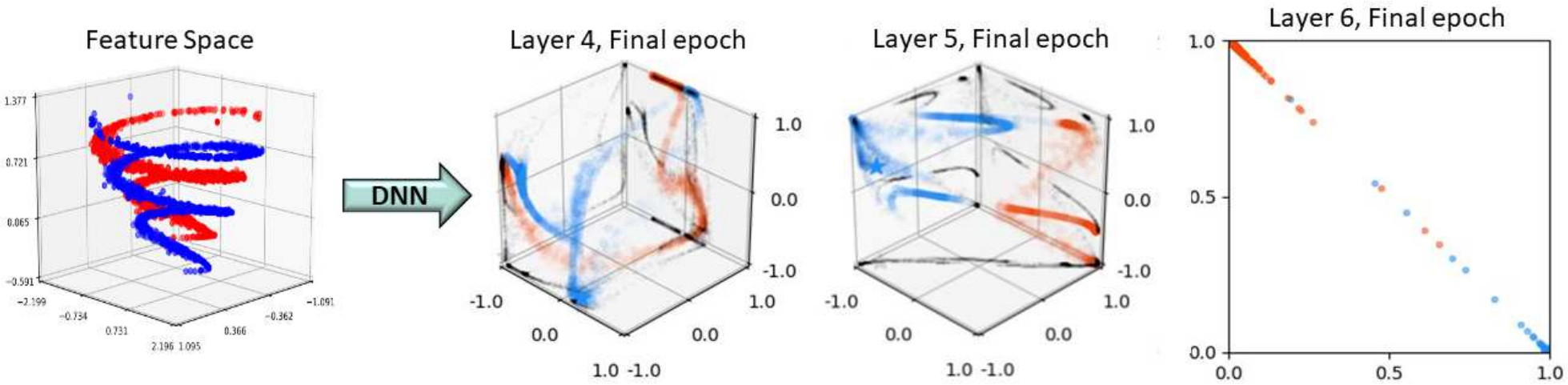


- Combine with efficient high-dimensional clustering measure

Towards Broader Impact: Design

How to Better Design DNNs?

- T_ℓ compression $\implies$ $T_{\ell-1}$ linear separation



- Combine with efficient high-dimensional clustering measure

$\Rightarrow$ **Optimize architecture** by shedding redundant layers

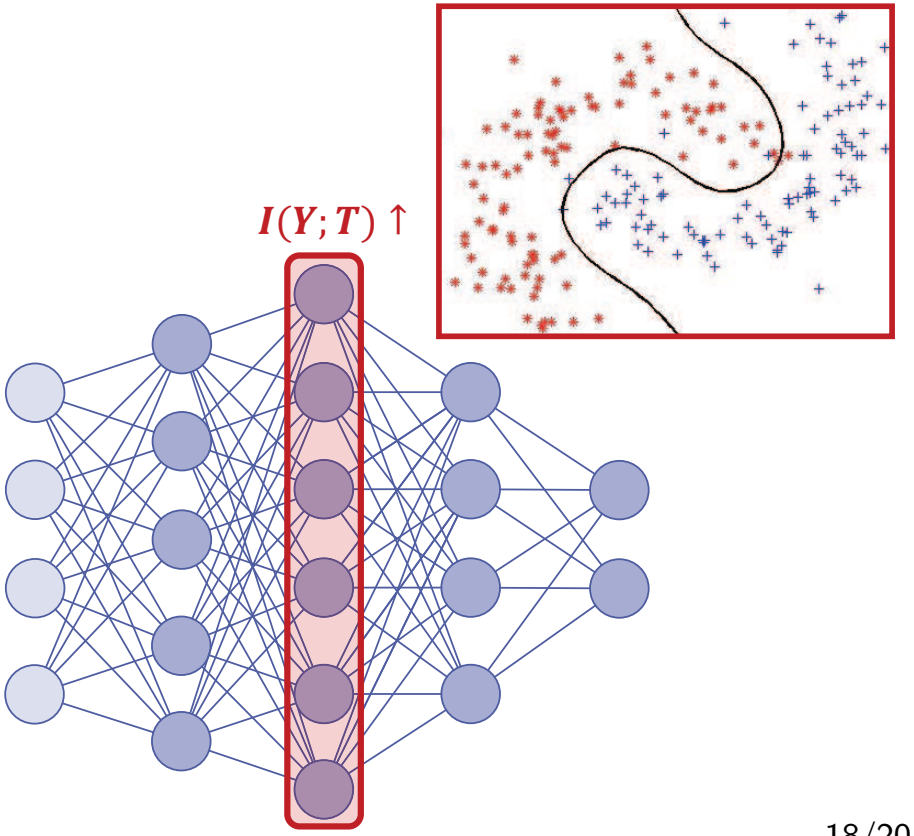
Towards Broader Impact: Optimization

How to Better Train DNNs?

Towards Broader Impact: Optimization

How to Better Train DNNs?

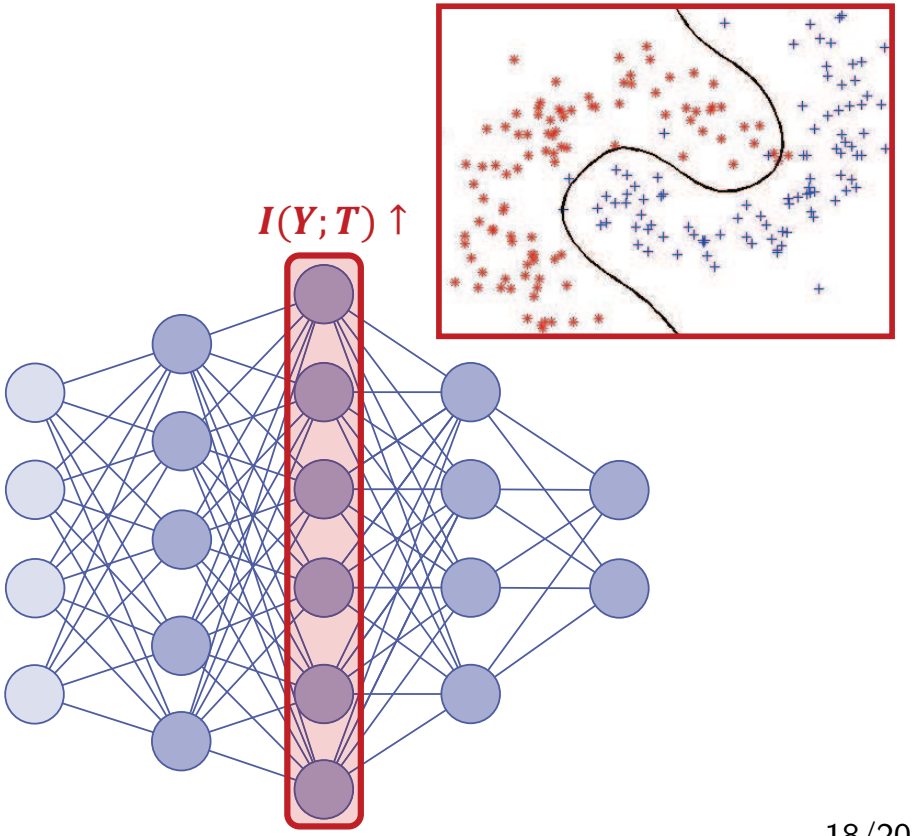
- Regularize intermediate layer to increase $I(Y;T_\ell)$



Towards Broader Impact: Optimization

How to Better Train DNNs?

- Regularize intermediate layer to increase $I(Y;T_\ell)$
- Learn well-separated (nonlinear) representations

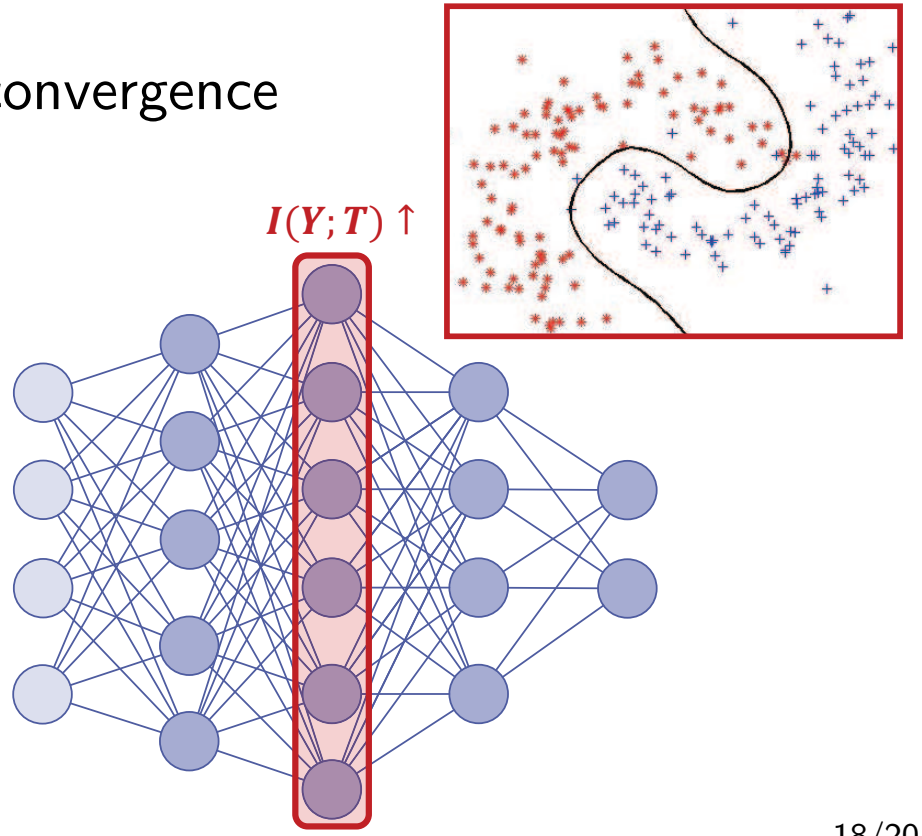


Towards Broader Impact: Optimization

How to Better Train DNNs?

- Regularize intermediate layer to increase $I(Y;T_\ell)$
- Learn well-separated (nonlinear) representations

$\Rightarrow$ **Enhanced algorithms** for faster convergence



Towards Broader Impact: Understanding

How to Better Understand DNNs?

Towards Broader Impact: Understanding

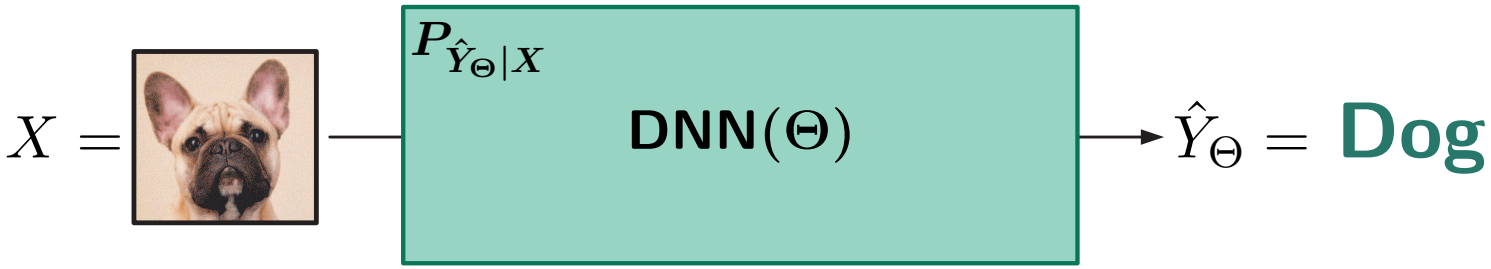
How to Better Understand DNNs?

1. **Channel Synthesis:** Quantify #bits needed for 'emulating' a channel

Towards Broader Impact: Understanding

How to Better Understand DNNs?

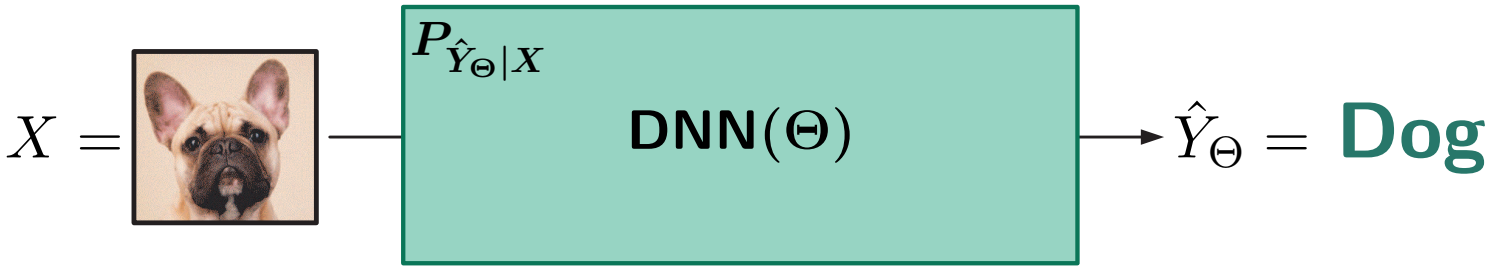
1. **Channel Synthesis:** Quantify #bits needed for 'emulating' a channel



Towards Broader Impact: Understanding

How to Better Understand DNNs?

1. **Channel Synthesis:** Quantify #bits needed for 'emulating' a channel

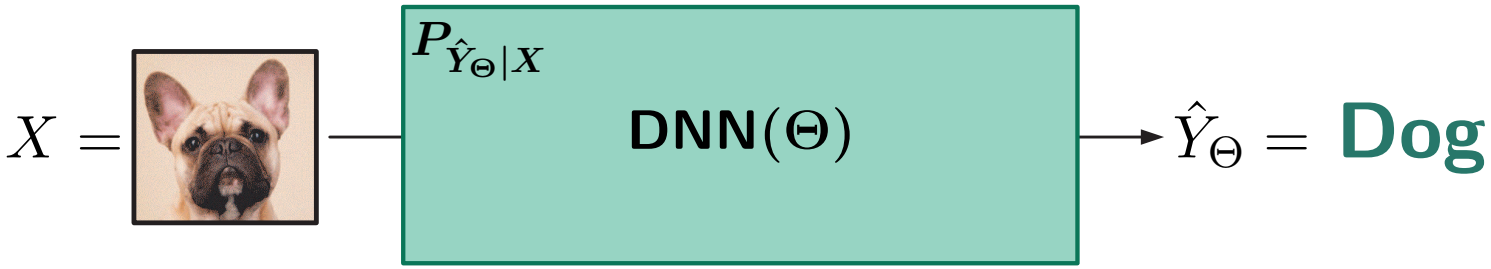


- Compare **DNN synthesis #bits** vs. $\log$ (**#classes**)

Towards Broader Impact: Understanding

How to Better Understand DNNs?

1. **Channel Synthesis:** Quantify #bits needed for 'emulating' a channel



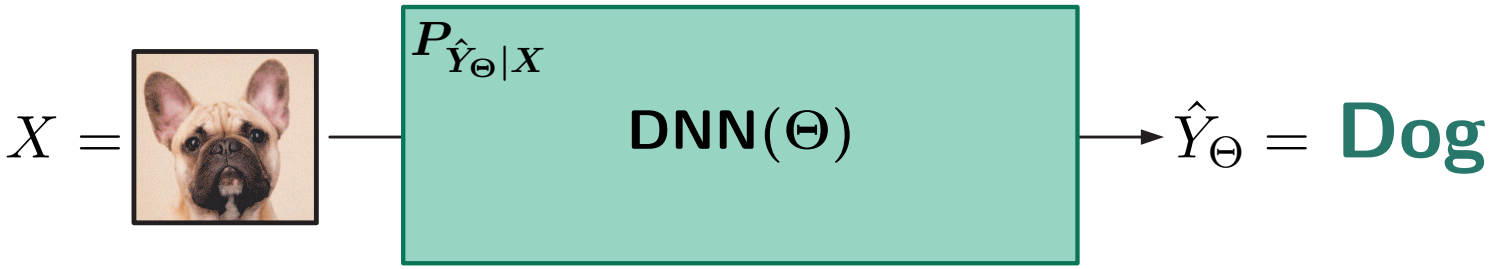
- Compare **DNN synthesis #bits** vs. $\log$ (**#classes**)

$\Rightarrow$ **Scoring systems** for DNNs performing the same task

Towards Broader Impact: Understanding

How to Better Understand DNNs?

1. **Channel Synthesis:** Quantify #bits needed for 'emulating' a channel



- Compare **DNN synthesis #bits** vs. $\log$ (**#classes**)

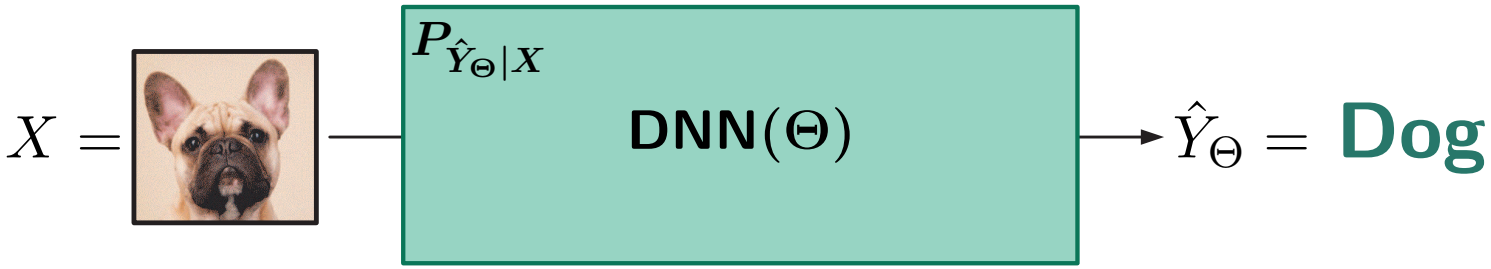
$\Rightarrow$ **Scoring systems** for DNNs performing the same task

2. **DNN Neural Activity:** Which are the 'dog' neurons?

Towards Broader Impact: Understanding

How to Better Understand DNNs?

1. **Channel Synthesis:** Quantify #bits needed for 'emulating' a channel



- Compare **DNN synthesis #bits** vs. $\log$ (**#classes**)

$\Rightarrow$ **Scoring systems** for DNNs performing the same task

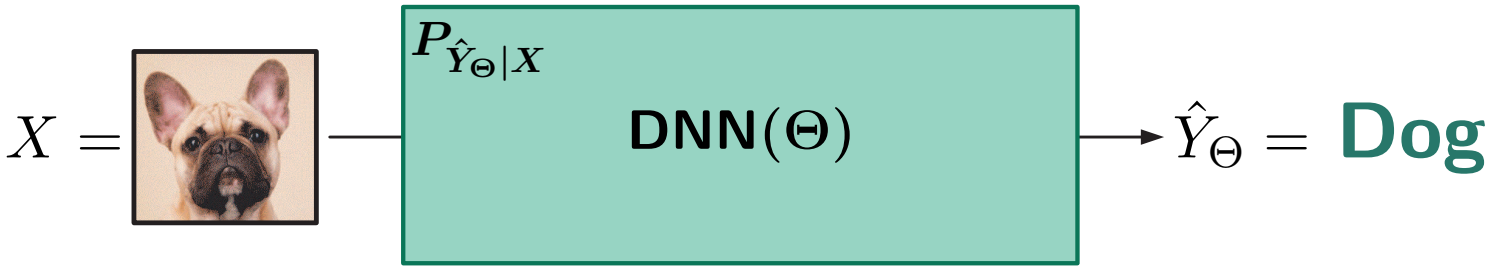
2. **DNN Neural Activity:** Which are the 'dog' neurons?

- MI estimator convergence rate **independent of input dimension**!

Towards Broader Impact: Understanding

How to Better Understand DNNs?

1. **Channel Synthesis:** Quantify #bits needed for 'emulating' a channel



- Compare **DNN synthesis #bits** vs. $\log$ (**#classes**)

$\Rightarrow$ **Scoring systems** for DNNs performing the same task

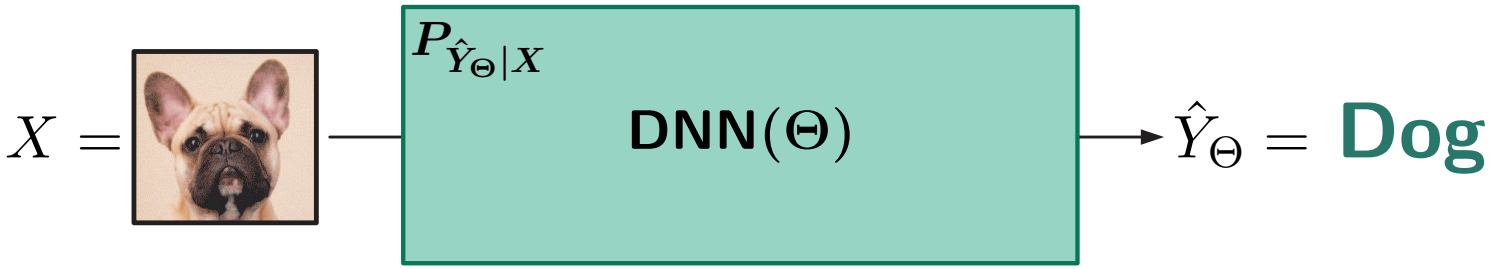
2. **DNN Neural Activity:** Which are the 'dog' neurons?

- MI estimator convergence rate **independent of input dimension**!
- Measure MI between $X_{\text{Dog}}/X_{\text{Cat}}$ and **single** (pairs, triples of) neurons

Towards Broader Impact: Understanding

How to Better Understand DNNs?

1. **Channel Synthesis:** Quantify #bits needed for 'emulating' a channel



- Compare **DNN synthesis #bits** vs. $\log$ (**#classes**)

$\Rightarrow$ **Scoring systems** for DNNs performing the same task

2. **DNN Neural Activity:** Which are the 'dog' neurons?

- MI estimator convergence rate **independent of input dimension**!
- Measure MI between $X_{\text{Dog}}/X_{\text{Cat}}$ and **single** (pairs, triples of) neurons

$\Rightarrow$ **Heatmap** of DNN neural activity

Information Theory in the Age of Information

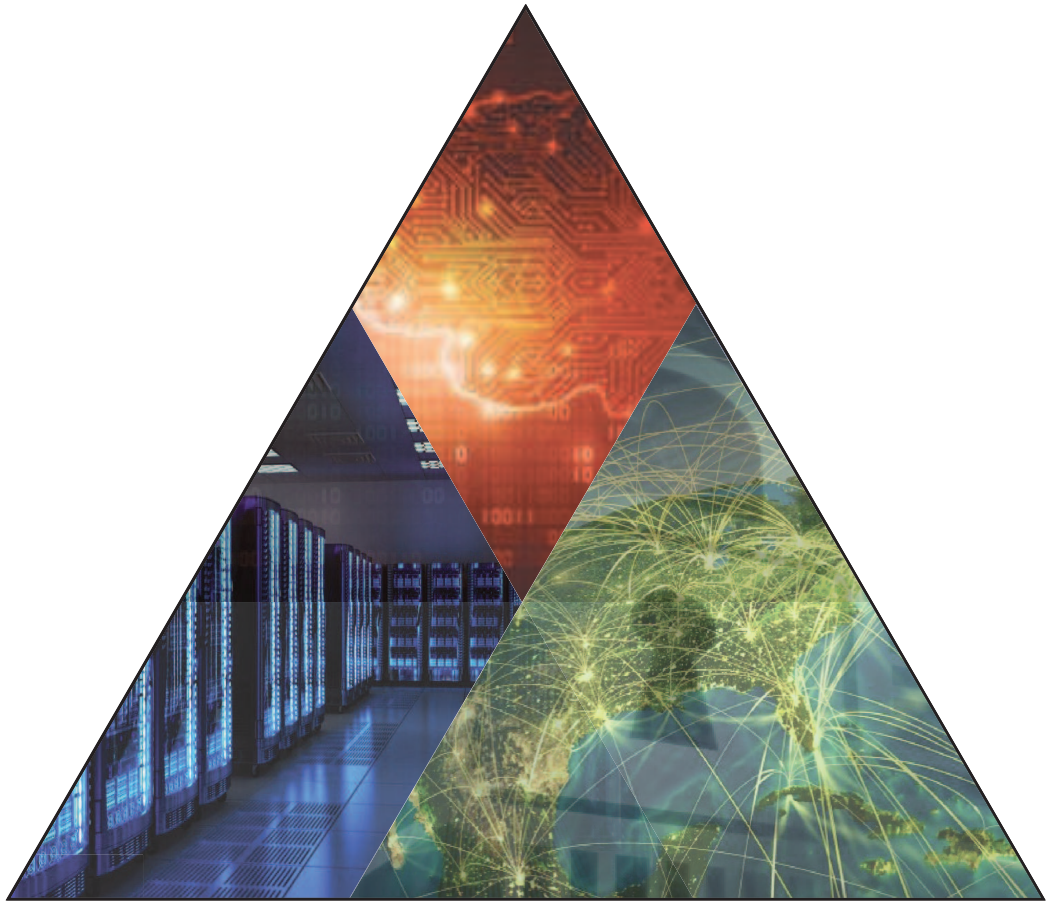
Information Theory in the Age of Information

Goal: Fundamental properties and opt. designs (math. modeling & solutions)

Information Theory in the Age of Information

Goal: Fundamental properties and opt. designs (math. modeling & solutions)

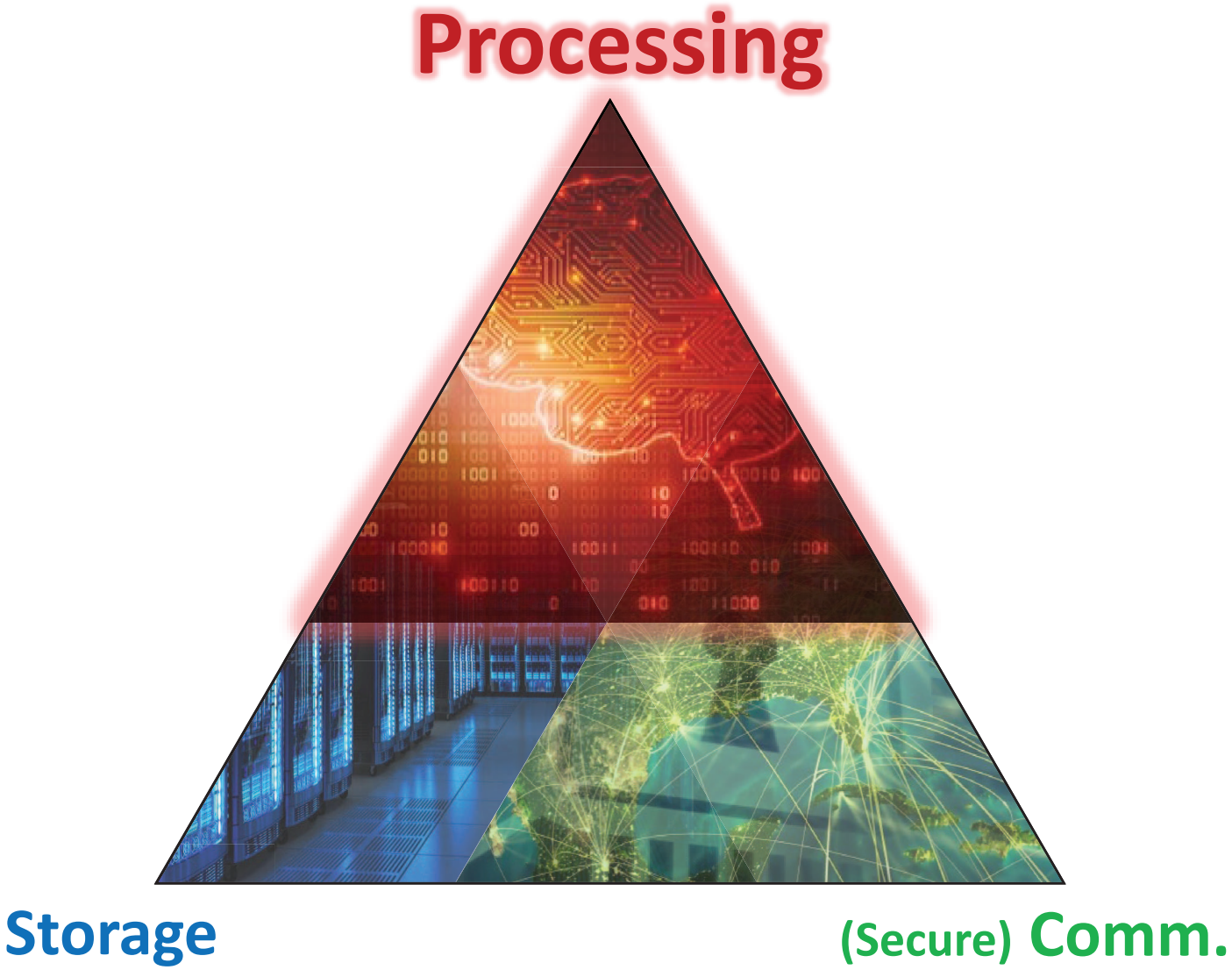
Processing



Storage **(Secure) Comm.**

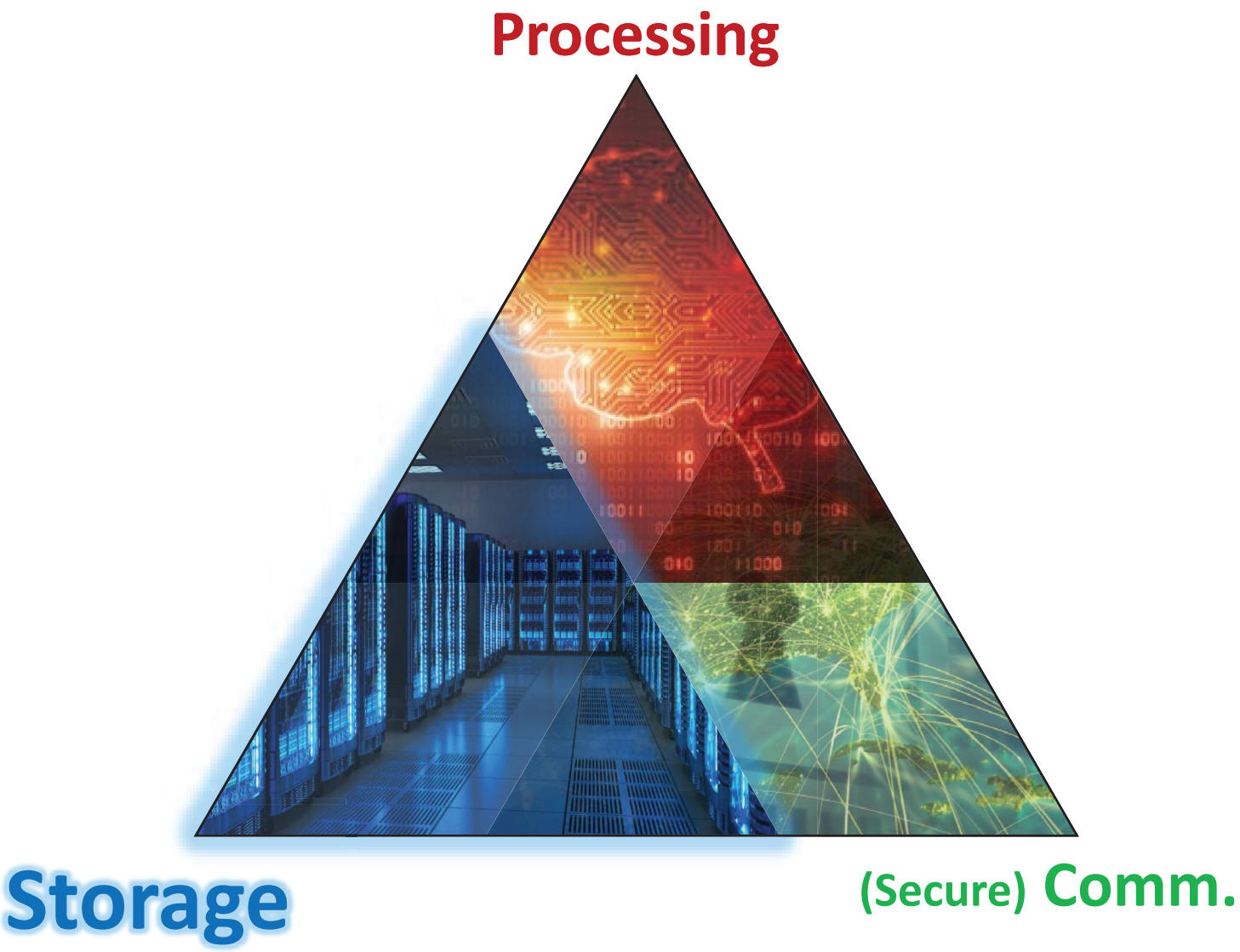
Information Theory in the Age of Information

Goal: Fundamental properties and opt. designs (math. modeling & solutions)



Information Theory in the Age of Information

Goal: Fundamental properties and opt. designs (math. modeling & solutions)



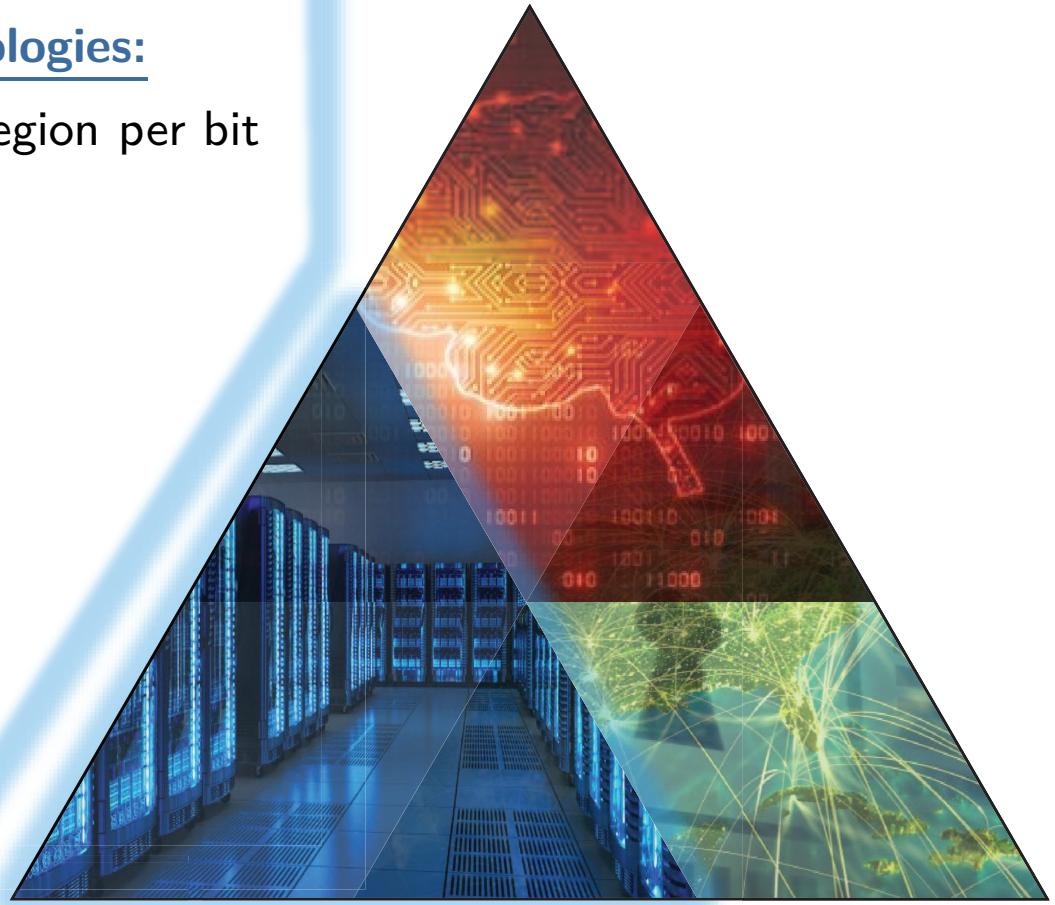
Information Theory in the Age of Information

Goal: Fundamental properties and opt. designs (math. modeling & solutions)

Emerging Technologies:

Shrink magnetic region per bit

Processing



Storage

(Secure) Comm.

Information Theory in the Age of Information

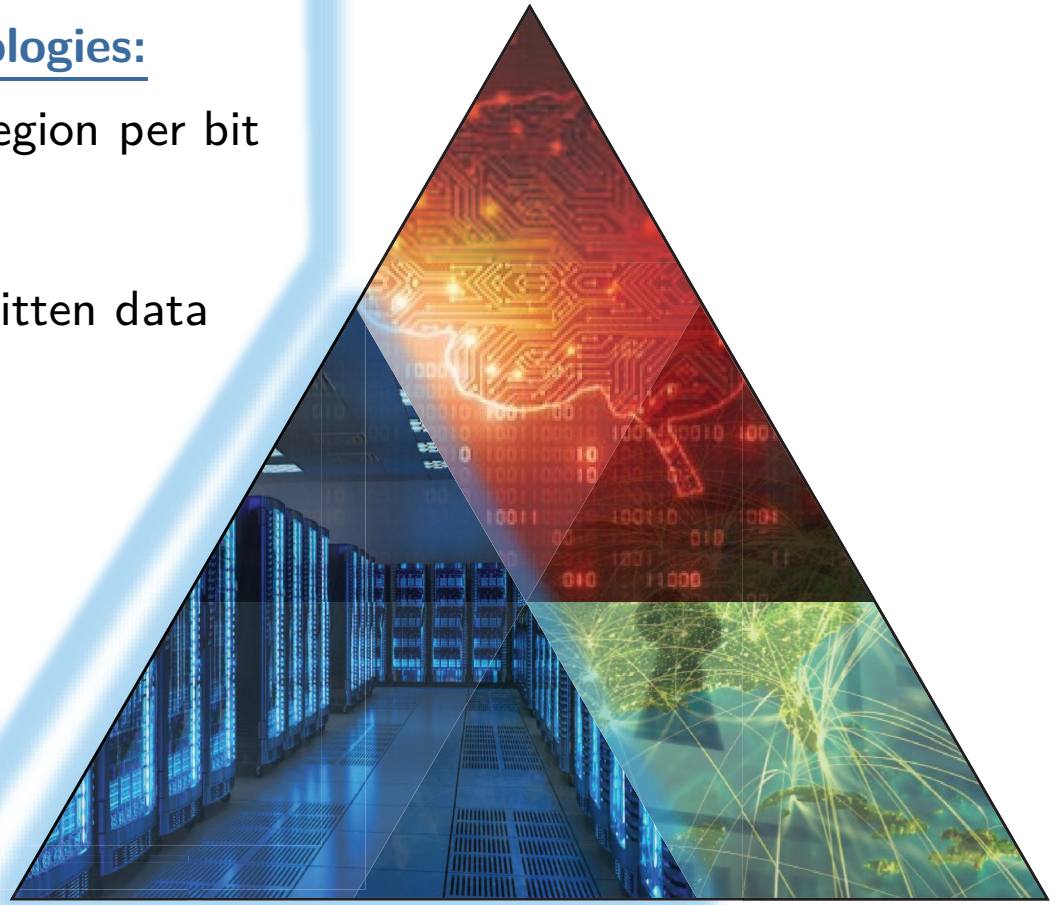
Goal: Fundamental properties and opt. designs (math. modeling & solutions)

Emerging Technologies:

Shrink magnetic region per bit

Challenges:

Stabilization of written data



Information Theory in the Age of Information

Goal: Fundamental properties and opt. designs (math. modeling & solutions)

Emerging Technologies:

Shrink magnetic region per bit

Challenges:

Stabilization of written data

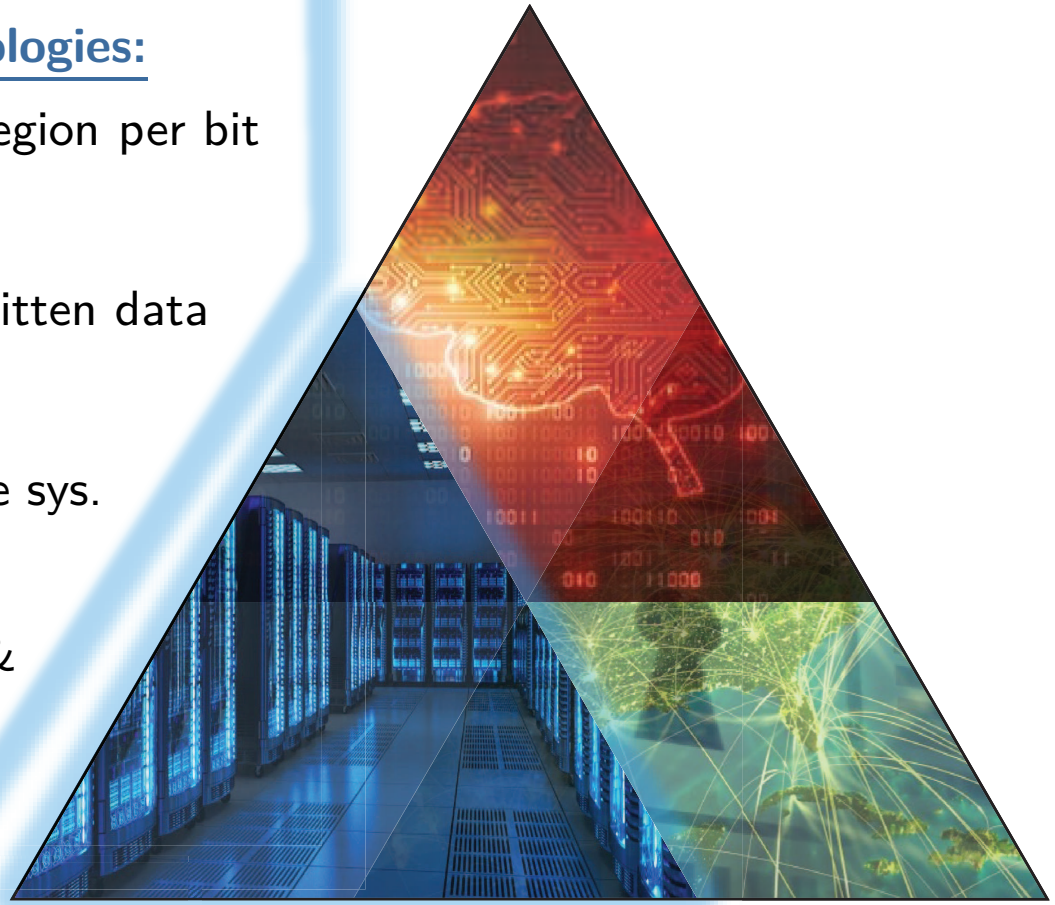
Model & Study:

Interacting particle sys.

$\Downarrow$

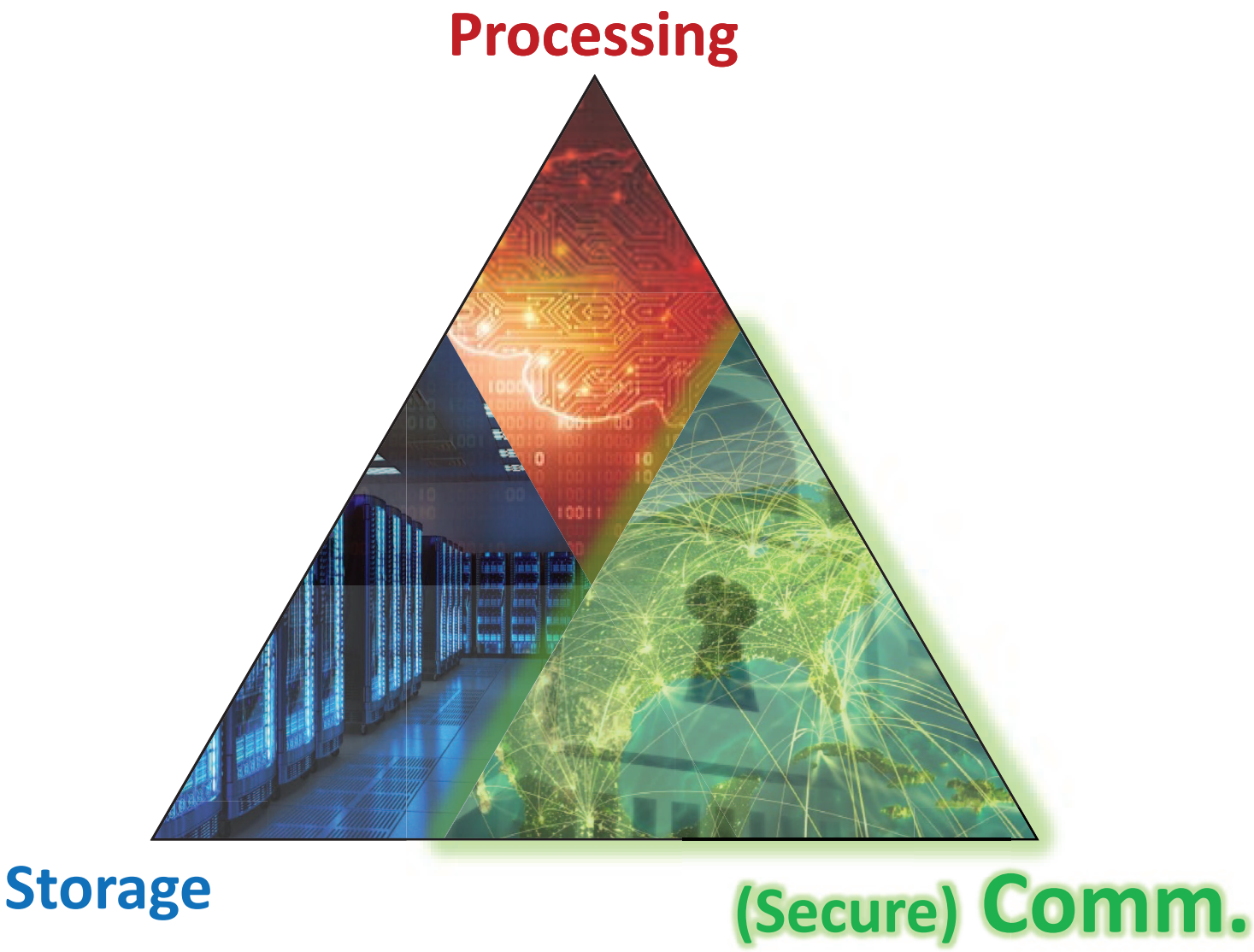
Storage capacity &

HDD designs



Information Theory in the Age of Information

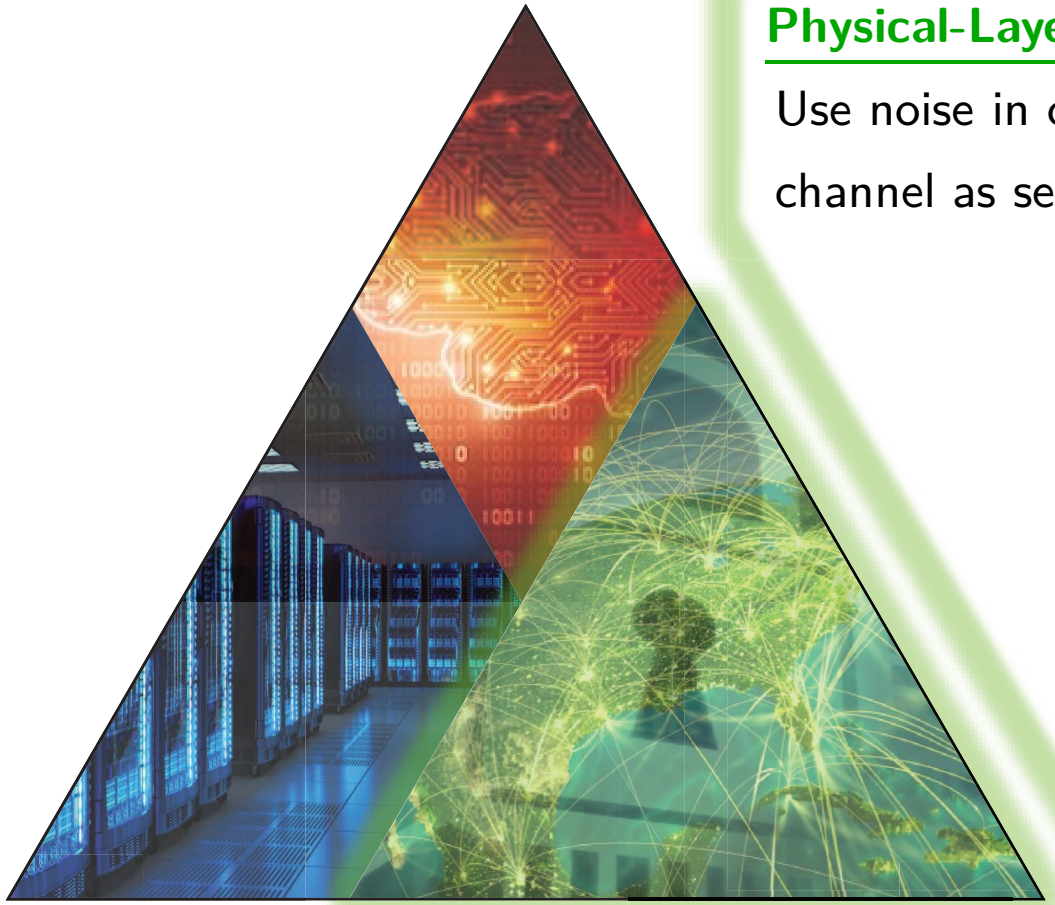
Goal: Fundamental properties and opt. designs (math. modeling & solutions)



Information Theory in the Age of Information

Goal: Fundamental properties and opt. designs (math. modeling & solutions)

Processing



Storage

(Secure) Comm.

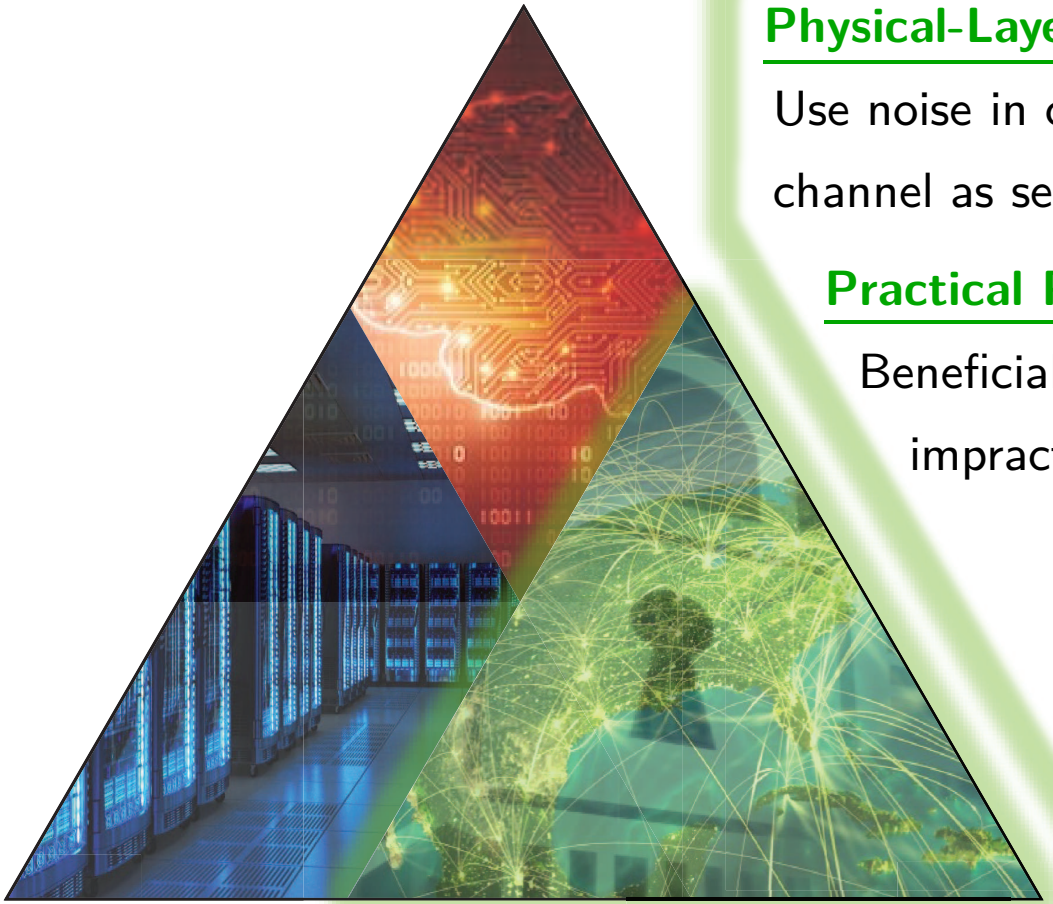
Physical-Layer Security (PLS):

Use noise in communication channel as security resource

Information Theory in the Age of Information

Goal: Fundamental properties and opt. designs (math. modeling & solutions)

Processing



Storage

(Secure) Comm.

Physical-Layer Security (PLS):

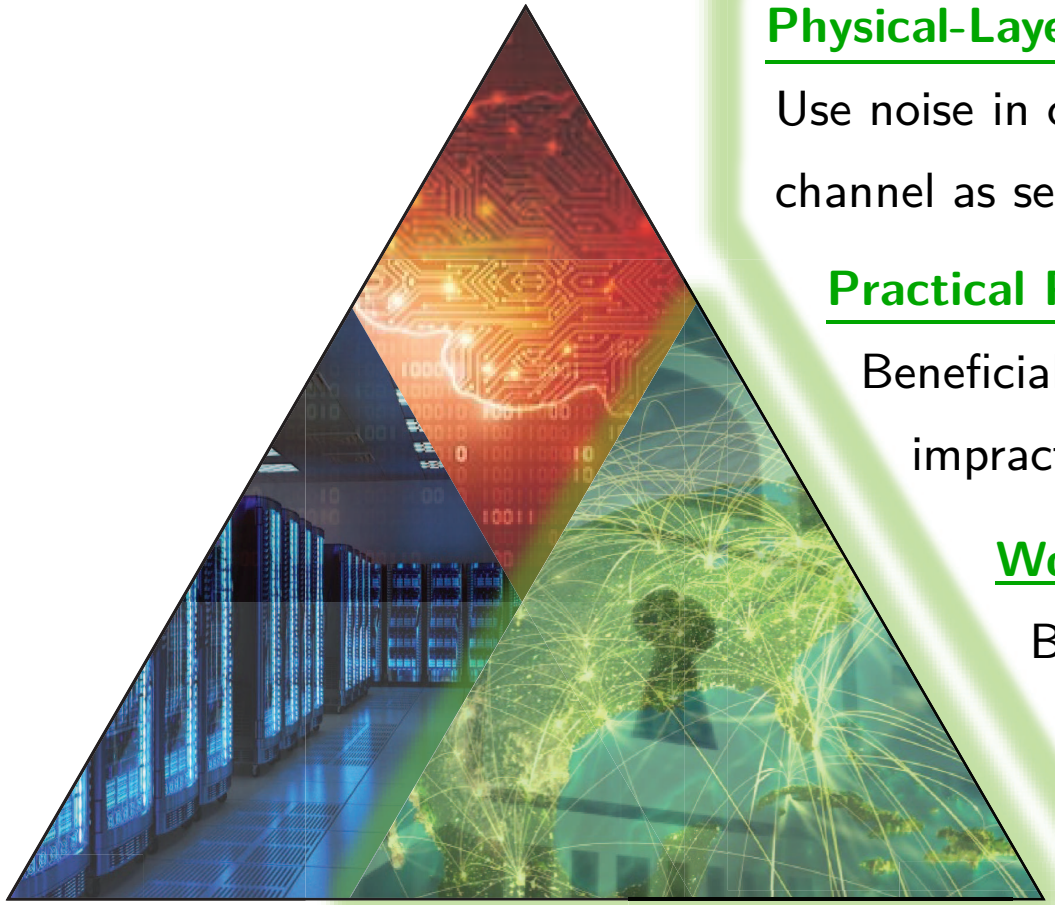
Use noise in communication channel as security resource

Practical Perspective

Beneficial properties but impractical assumptions

Information Theory in the Age of Information

Goal: Fundamental properties and opt. designs (math. modeling & solutions)



Physical-Layer Security (PLS):

Use noise in communication channel as security resource

Practical Perspective

Beneficial properties but impractical assumptions

Work & Vision:

Bridge gaps for interdisciplinary security paradigm