

# Mutual Information in Deep Learning

程昊 (louischeng)  
腾讯-优图实验室



2020-04-17

## 1 Mutual Information

- What is mutual information?
- How to estimate mutual information?

## 2 MI in Deep Learning

- Information Bottleneck (MI as visualization tool)
- Deep VIB (MI as regularizer)
- MINE (MI as regularizer)
- Deep Infomax (MI as objective function)
- MoCo (MI as objective function)

## 3 Apply MI in Your Own Research

## 1 Mutual Information

- What is mutual information?
- How to estimate mutual information?

## 2 MI in Deep Learning

- Information Bottleneck (MI as visualization tool)
- Deep VIB (MI as regularizer)
- MINE (MI as regularizer)
- Deep Infomax (MI as objective function)
- MoCo (MI as objective function)

## 3 Apply MI in Your Own Research

# What is mutual information?

First, we define the **information** of a random variable  $X$  as  $H(X)$ :

$$H(X) = - \sum_{p_i} p_i \log p_i$$

$H(X)$  measures the uncertainty of a random variable.

$H(X)$  achieves maximum when  $X$  follows uniform distribution.

# What is mutual information?

The **mutual information (MI)** between two **discrete** random variable  $X$  and  $Y$  is defined as :

$$I(X; Y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} = H(X) - H(X|Y)$$

Similarly, the **MI** between two **continuous** random variable  $X$  and  $Y$  is defined as:

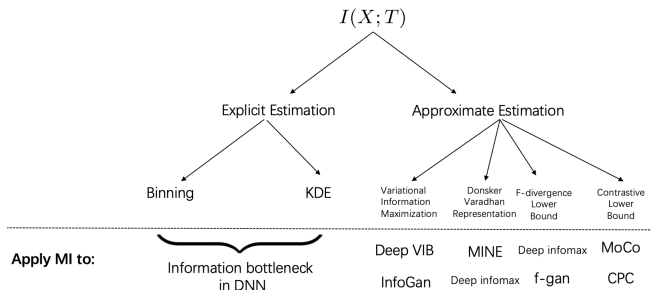
$$I(X; Y) = \iint_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} = H(X) - H(X|Y)$$

MI measures the amount of information obtained about one random variable through observing the other random variable.

# How to estimate mutual information?

There are two ways to estimate mutual information in practice:

Estimating MI (Mutual Information):



Usually, explicit estimation is **non-parametric** method while approximate estimation is **parametric** method.

## 1 Mutual Information

- What is mutual information?
- How to estimate mutual information?

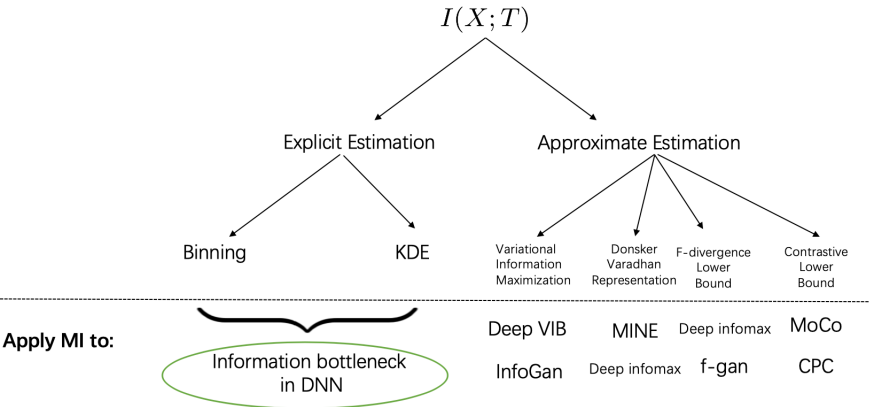
## 2 MI in Deep Learning

- Information Bottleneck (MI as visualization tool)
- Deep VIB (MI as regularizer)
- MINE (MI as regularizer)
- Deep Infomax (MI as objective function)
- MoCo (MI as objective function)

## 3 Apply MI in Your Own Research

# MI in Deep Learning

## Estimating MI (Mutual Information):

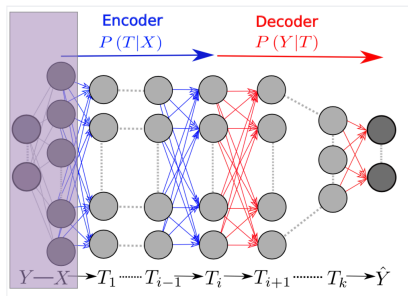




# Information Bottleneck (MI as visualization tool)

Tishby et al. believes that the DNN training is actually optimizing the following objective:

$$\min I(X; T) - \beta I(T; Y)$$



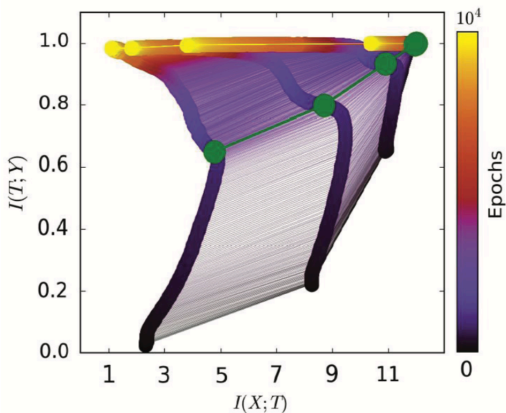
where  $T$  is the feature of each layer,  $X$  is the input and  $Y$  is the label.

Opening the Black Box of Deep Neural Networks via Information, [Arxiv:1703.00810](https://arxiv.org/abs/1703.00810)

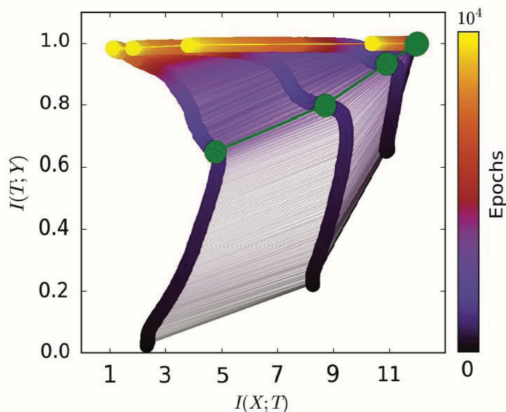
# Information Bottleneck (MI as visualization tool)

Tishby et al. use **binning** to estimate MI. In this case:

$$I(X; T) = \iint_{x,t} p(x, t) \log \frac{p(x, t)}{p(x)p(t)} \approx \sum_{x,t} p(x, t) \log \frac{p(x, t)}{p(x)p(t)}$$



# Information Bottleneck (MI as visualization tool)



- Phase 1:  $I(X; T)$  and  $I(T; Y)$  both increases, which indicates the network memorizes the information about the input.
- Phase 2:  $I(X; T)$  decreases while  $I(T; Y)$  increases, which indicates that the network drops unimportant information to generalize.

# Information Bottleneck (MI as visualization tool)

Andrew Michael Saxe et al. find that two phase phenomenon disappears in RELU case.

They use KDE (Kernel Density Estimation) to estimate mutual information, which is more practical than binning. In KDE (gaussian kernel), each sample forms a gaussian distribution. The overall distribution is the sum of all the gaussian distribution.

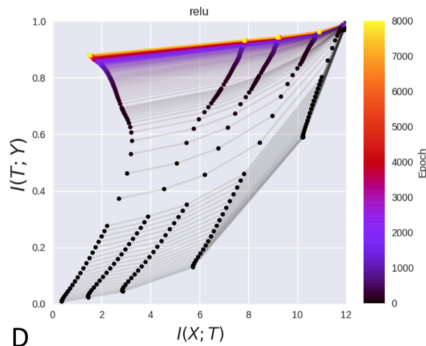
$$p(X) = \frac{1}{N} \sum_{i=1}^N p(X_i)$$

where  $p(X_i)$  is a gaussian distribution centered at  $X_i$ .

On the Information Bottleneck Theory of Deep Learning. ICLR 2018

# Information Bottleneck (MI as visualization tool)

Here the picture shows IB in RELU case (KDE estimation).



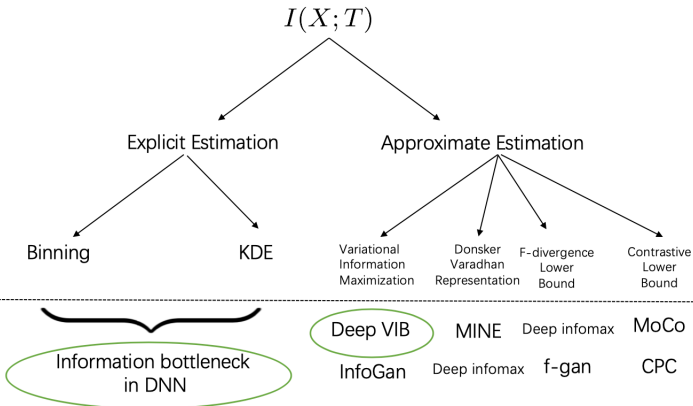
It seems that except for the last layer, all the other layers do not have the second compression phase.

Which is right (Does second phase really exist in RELU case)?

- Tishby shows that with RELU, the network still has the compression phase with more advanced estimator (code is not released).
- Some researchers prove the second phase exists with a specific estimator (EDGE).
- It is still an open problem. However, the ideology of IB is valuable in understanding neural networks and has inspired other great works.

# MI in Deep Learning

Estimating MI (Mutual Information):



# Deep VIB (MI as regularizer)

Suppose  $Y \leftrightarrow X \leftrightarrow Z$  forms a Markov chain. We want to learn an efficient model  $p(z|x)$  from  $IB$  perspective. Then our goal is to minimize:

$$\min I(X; Z) - \beta I(Z; Y)$$

In the above formulation,  $p(z|y)$  in  $I(Z; Y)$  and  $p(z)$  in  $I(X; Z)$  are hard to compute (intratable). Thus we use  $q(z|y)$  and  $r(z)$  to approximate them, where  $q(z|y)$  is parameterized by a neural network and we assume  $r(z)$  follows a standard gaussian distribution. The loss becomes:

$$-\frac{1}{N} \sum_{n=1}^N \int dz p(z|x_n) \log q(y_n|z) - \beta p(z|x_n) \log \frac{p(z|x_n)}{r(z)}$$

which is identical to minimize:

$$-\frac{1}{N} \sum_{n=1}^N \int dz p(z|x_n) \log q(y_n|z) - \beta \text{KL}[p(z|x_n), r(z)]$$

Learning deep representations by mutual information estimation and maximization. [JCLR 2019](#)



# Deep VIB (MI as regularizer)

$$-\frac{1}{N} \sum_{n=1}^N \int dz p(z|x_n) \log q(y_n|z) - \beta \text{KL}[p(z|x_n), r(z)]$$

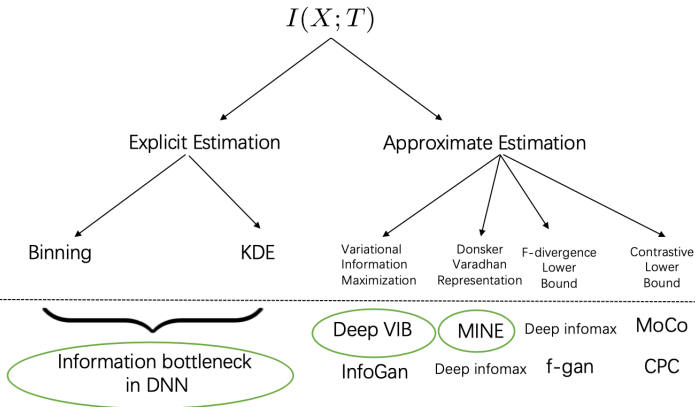
The first term equals to the traditional loss of supervised learning, while the second term wants  $p(z|x)$  follows a standard gaussian distribution. Thus Deep VIB introduces a regularizer to traditional supervised learning.

Result of Deep VIB on MNIST:

	Model	error
	Baseline	1.38%
	Dropout	1.34%
Dropout	(Pereyra et al., 2017)	1.40%
	Confidence Penalty	1.36%
Confidence Penalty	(Pereyra et al., 2017)	1.17%
	Label Smoothing	1.40%
Label Smoothing	(Pereyra et al., 2017)	1.23%
	<b>VIB (<math>\beta = 10^{-3}</math>)</b>	<b>1.13%</b>

# MI in Deep Learning

Estimating MI (Mutual Information):



# MINE: MI as regularizer

## Donsker-Varadhan Representation:

$$\begin{aligned} I(x; z) &= D_{\text{KL}}(p(x, z) \| p(x)p(z)) \\ &= \mathbb{E}_{p(x, z)} \left[ \log \frac{p(x, z)}{p(x)p(z)} \right] \\ &= \mathbb{E}_{p(x, z)} \left[ \log \left( \frac{q(x, z)}{p(x)p(z)} \frac{p(x, z)}{q(x, z)} \right) \right] \\ &= \mathbb{E}_{p(x, z)} \left[ \log \frac{q(x, z)}{p(x)p(z)} \right] + \underbrace{D_{\text{KL}}(p(x, z) \| q(x, z))}_{\geq 0} \\ &\geq \mathbb{E}_{p(x, z)} \left[ \log \frac{q(x, z)}{p(x)p(z)} \right] \end{aligned}$$

$$\text{let } q(x, z) = \frac{1}{K} p(x)p(z) e^{T_\theta(x, z)} = \frac{p(x)p(z) e^{T_\theta(x, z)}}{\int \int p(x')p(z') e^{T_\theta(x', z')} dz' dx'}$$

$$\begin{aligned} \text{Then } D_{\text{KL}}(p(x, z) \| p(x)p(z)) &\geq \sup_{\theta} \mathbb{E}_{p(x, z)} \left[ \log \frac{p(x)p(z) e^{T_\theta(x, z)}}{K p(x)p(z)} \right] \\ &= \sup_{\theta} \mathbb{E}_{p(x, z)} \left[ \log \frac{e^{T_\theta(x, z)}}{\mathbb{E}_{p(x)p(z)} [e^{T_\theta(x, z)}]} \right] \\ &= \sup_{\theta} \mathbb{E}_{p(x, z)} [T_\theta(x, z)] - \log \mathbb{E}_{p(x)p(z)} [e^{T_\theta(x, z)}] \end{aligned}$$

Main idea: Let  $T_\theta$  is parameterized by a neural network.

MINE: Mutual Information Neural Estimation. ICML 2018

# MINE (MI as regularizer)

MINE in GAN:

Vanilla Gan objective is :

$$\min_G \max_D V(D, G) = \mathbb{E}_{P_X}[\log D(X)] + \mathbb{E}_{P_Z}[\log(1 - D(G(Z)))]$$

Write  $Z$  as  $Z = [\epsilon, c]$ , then the generator objective becomes:

$$\max_G \mathbb{E}[\log(D(G([\epsilon, c])))]$$

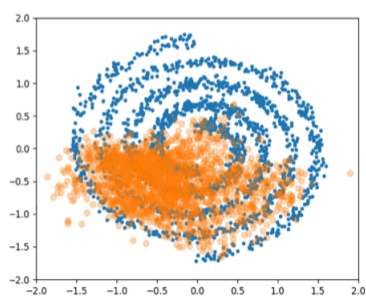
In MINE, the generator objective has a regularization term:

$$\max_G \mathbb{E}[\log(D(G([\epsilon, c])))] + \beta I(G([\epsilon, c]); c)$$

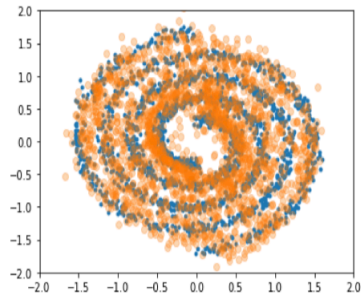
The calculation of  $\beta I(G([\epsilon, c]); c)$  needs  $T_\theta$  parameterized by a neural network.

# MINE (MI as regularizer)

Comparison with GAN and GAN+MINE:



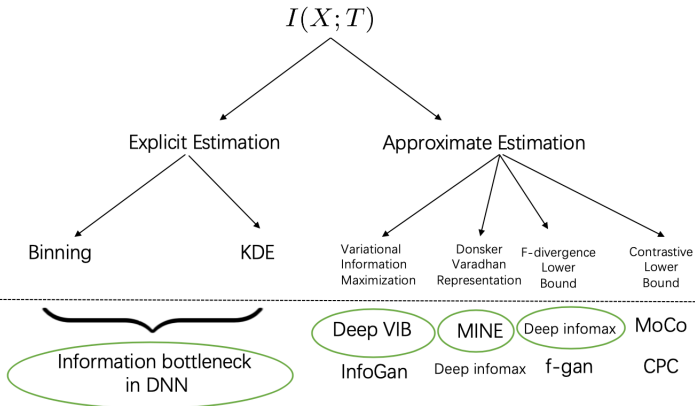
(a) GAN



(b) GAN+MINE

# MI in Deep Learning

Estimating MI (Mutual Information):



# Deep Infomax: MI as objective function

Goal: Learn a good representation by maximize the mutual information between input and output.

Global MI:

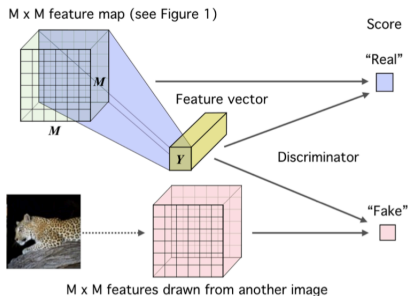


Figure 2: **Deep InfoMax (DIM) with a global  $MI(X; Y)$  objective.** Here, we pass both the high-level feature vector,  $Y$ , and the lower-level  $M \times M$  feature map (see Figure 1) through a discriminator to get the score. Fake samples are drawn by combining the same feature vector with a  $M \times M$  feature map from another image.

# Deep Infomax (MI as objective function)

## Local MI:

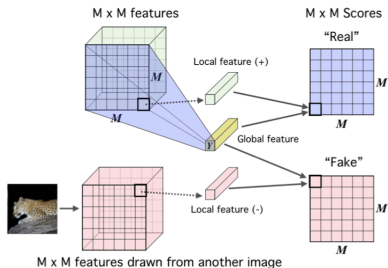


Figure 3: **Maximizing mutual information between local features and global features.** First we encode the image to a feature map that reflects some structural aspect of the data, e.g. spatial locality, and we further summarize this feature map into a global feature vector (see Figure 1). We then concatenate this feature vector with the lower-level feature map *at every location*. A score is produced for each local-global pair through an additional function (see the Appendix A.2 for details).



# Deep Infomax (MI as objective function)

Estimate MI in Deep InfoMax:

The authors propose three MI estimators (DV, JSD, InfoNCE) in the paper and show the estimator based on Jensen-Shannon Divergence (JSD) is the most practical. The estimator is presented as:

$$I_{w,\phi}^{\text{JSD}} = \mathbb{E}_P[-sp(-T_w(x, E_\phi(x)))] - \mathbb{E}_{P \times \tilde{P}}[sp(T_w(x', E_\phi(x)))]$$

where  $x$  is the input.  $E_\phi$  is the encoder to map the input to output.  $T_w$  is a parameterized neural network.  $\mathbb{E}_P$  represents the joint distribution between input and output.  $\mathbb{E}_{P \times \tilde{P}}$  represents the marginal distribution between input and output.  $sp(z) = \log(1 + e^z)$  is the softplus function.

# Deep Infomax (MI as objective function)

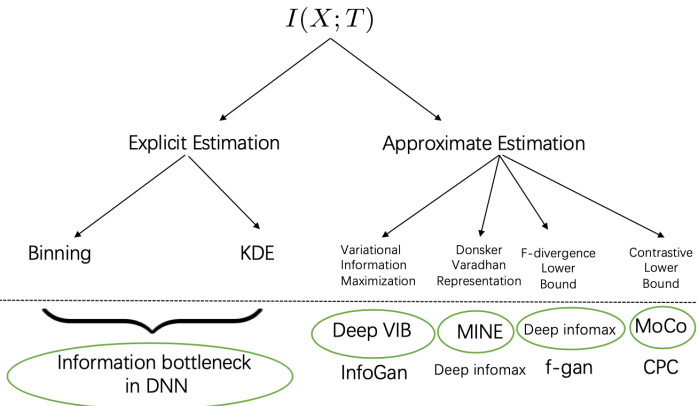
## Comparison with Deep InfoMax and other methods:

Table 1: Classification accuracy (top 1) results on CIFAR10 and CIFAR100. DIM(L) (i.e., with the local-only objective) outperforms all other unsupervised methods presented by a wide margin. In addition, DIM(L) approaches or even surpasses a fully-supervised classifier with similar architecture. DIM with the global-only objective is competitive with some models across tasks, but falls short when compared to generative models and DIM(L) on CIFAR100. Fully-supervised classification results are provided for comparison.

Model	CIFAR10			CIFAR100		
	conv	fc (1024)	Y(64)	conv	fc (1024)	Y(64)
Fully supervised		75.39			42.27	
VAE	60.71	60.54	54.61	37.21	34.05	24.22
AE	62.19	55.78	54.47	31.50	23.89	27.44
$\beta$ -VAE	62.4	57.89	55.43	32.28	26.89	28.96
AAE	59.44	57.19	52.81	36.22	33.38	23.25
BiGAN	62.57	62.74	52.54	37.59	33.34	21.49
NAT	56.19	51.29	31.16	29.18	24.57	9.72
DIM(G)	52.2	52.84	43.17	27.68	24.35	19.98
DIM(L) (DV)	<b>72.66</b>	<b>70.60</b>	<b>64.71</b>	<b>48.52</b>	<b>44.44</b>	<b>39.27</b>
DIM(L) (JSD)	<b>73.25</b>	<b>73.62</b>	<b>66.96</b>	<b>48.13</b>	<b>45.92</b>	<b>39.60</b>
DIM(L) (infoNCE)	<b>75.21</b>	<b>75.57</b>	<b>69.13</b>	<b>49.74</b>	<b>47.72</b>	<b>41.61</b>

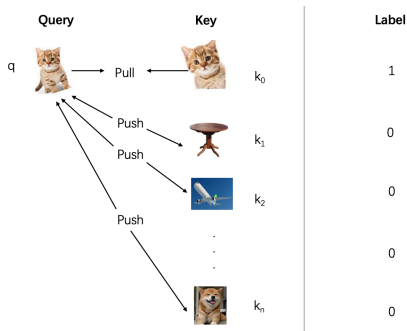
# MI in Deep Learning

Estimating MI (Mutual Information):



# MoCo (MI as objective function)

Goal: Learn a good representation by instance discrimination.



The loss is:

$$L_q = -\log \frac{\exp(q \cdot k_0)}{\sum_{i=0}^n \exp(q \cdot k_i)}$$

# MoCo (MI as objective function)

$$L_q = -\log \frac{\exp(q \cdot k_0)}{\sum_{i=0}^n \exp(q \cdot k_i)}$$

This loss is actually maximizing the mutual information of the two views of the image.

$$\begin{aligned} I(x; z) - \log N &= \mathbb{E}_S \left[ \log \frac{p(x^* | z^*)}{p(x^*)} \right] - \log N \\ &= \mathbb{E}_S \left[ \log \left( \frac{p(x^* | z^*)}{p(x^*)} \frac{1}{N} \right) \right] \\ &= \mathbb{E}_S \left[ \log \left( \frac{1}{\frac{p(x^*)}{p(x^* | z^*)} N} \right) \right] \\ &\geq \mathbb{E}_S \left[ \log \frac{1}{1 + \frac{p(x^*)}{p(x^* | z^*)} (N-1)} \right] \\ &= \mathbb{E}_S \left[ -\log \left( 1 + \frac{p(x^*)}{p(x^* | z^*)} (N-1) \right) \right] \\ &= \mathbb{E}_S \left[ -\log \left( 1 + \frac{p(x^*)}{p(x^* | z^*)} (N-1) \mathbb{E}_{S \sim [x^*]} \left[ \frac{p(x | z^*)}{p(x)} \right] \right) \right] \\ &= \mathbb{E}_S \left[ -\log \left( 1 + \frac{p(x^*)}{p(x^* | z^*)} \sum_{j=1}^{N-1} \frac{p(x_j | z^*)}{p(x_j)} \right) \right] \\ &= \mathbb{E}_S \left[ -\log \left( 1 + \frac{\sum_{j=1}^{N-1} \frac{p(x_j | z^*)}{p(x_j)}}{\frac{p(x^*)}{p(x^* | z^*)}} \right) \right] \\ &= \mathbb{E}_S \left[ \log \frac{\frac{p(x^* | z^*)}{p(x^*)}}{\frac{p(x^* | z^*)}{p(x^*)} + \sum_{j=1}^{N-1} \frac{p(x_j | z^*)}{p(x_j)}} \right] \\ &\approx \mathbb{E}_S \left[ \log \frac{h_\theta(x^*, z^*)}{h_\theta(x^*, z^*) + \sum_{j=1}^{N-1} h_\theta(x_j, z^*)} \right] \\ &= \mathbb{E}_S \left[ \log \frac{h_\theta(x^*, z^*)}{\sum_{x \in S} h_\theta(x, z^*)} \right], \end{aligned}$$

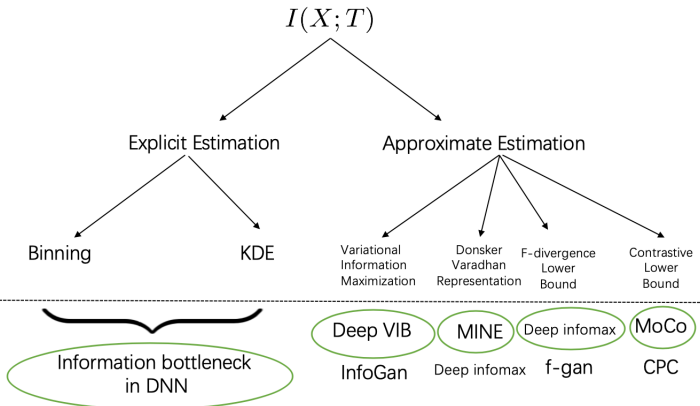
# MoCo (MI as objective function)

Comparison with MoCo and other self-supervised methods.

method	architecture	#params (M)	accuracy (%)
Exemplar [17]	R50w3 $\times$	211	46.0 [38]
RelativePosition [13]	R50w2 $\times$	94	51.4 [38]
Jigsaw [45]	R50w2 $\times$	94	44.6 [38]
Rotation [19]	Rv50w4 $\times$	86	55.4 [38]
Colorization [64]	R101*	28	39.6 [14]
DeepCluster [3]	VGG [53]	15	48.4 [4]
BigBiGAN [16]	R50	24	56.6
	Rv50w4 $\times$	86	61.3
<i>methods based on contrastive learning follow:</i>			
InstDisc [61]	R50	24	54.0
LocalAgg [66]	R50	24	58.8
CPC v1 [46]	R101*	28	48.7
CPC v2 [35]	R170* <sub>wider</sub>	303	65.9
CMC [56]	R50 <sub>L+ab</sub>	47	64.1 <sup>†</sup>
	R50w2 $\times$ <sub>L+ab</sub>	188	68.4 <sup>†</sup>
AMDIM [2]	AMDIM <sub>small</sub>	194	63.5 <sup>†</sup>
	AMDIM <sub>large</sub>	626	68.1 <sup>†</sup>
<b>MoCo</b>	R50	24	60.6
	RX50	46	63.9
	R50w2 $\times$	94	65.4
	R50w4 $\times$	375	<b>68.6</b>

# MI in Deep Learning

Estimating MI (Mutual Information):



## 1 Mutual Information

- What is mutual information?
- How to estimate mutual information?

## 2 MI in Deep Learning

- Information Bottleneck (MI as visualization tool)
- Deep VIB (MI as regularizer)
- MINE (MI as regularizer)
- Deep Infomax (MI as objective function)
- MoCo (MI as objective function)

## 3 Apply MI in Your Own Research



# Apply MI in Your Own Research

- When your research (problem) involves calculating the similarity (correlation) between two random variables, consider using MI as a criterion.
- Choosing a proper estimator is essential. You may try different estimators and select the best.

- Opening the Black Box of Deep Neural Networks via Information. Arxiv:1703.00810
- On the Information Bottleneck Theory of Deep Learning. ICLR 2018
- Learning deep representations by mutual information estimation and maximization. ICLR 2019
- MINE: Mutual Information Neural Estimation. ICML 2018
- Learning deep representations by mutual information estimation and maximization. ICLR 2019
- Momentum Contrast for Unsupervised Visual Representation Learning. CVPR 2020
- InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. NIPS 2016
- f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization. NIPS 2016
- <http://karangrewal.ca/blog/mutual-information-objectives/>

# Thanks!