Information Theory III: Advanced Topics for ML

Data Processing Inequality · f-Divergences · ELBO · Information Bottleneck

Recap: What We Have So Far

Entropy $H(p)$ → **Cross-Entropy** $H(p\|q)$ → **KL Divergence** $D_{\mathrm{KL}}(p\|q)$ → **Mutual Info** $I(X;Y)$

We established: cross-entropy loss = MLE, forward KL for supervised learning, reverse KL for variational inference, MI for feature selection.

Today: Four powerful extensions.

1. Data Processing Inequality **2.** f-Divergences & GANs

3. ELBO & VAEs **4.** Information Bottleneck

Data Processing Inequality

Processing data can only **destroy** information, never create it.

Markov Chains and Information Flow

A **Markov chain** $X \to Y \to Z$ means: Z depends on X only through Y.

$$p(x, y, z) = p(x)\, p(y \mid x)\, p(z \mid y)$$

$$X \xrightarrow{p(y|x)} Y \xrightarrow{p(z|y)} Z$$

Examples in ML:

- Raw pixels $\to$ convolutional features $\to$ class prediction
- Original data $\to$ PCA projection $\to$ clustering
- Text $\to$ embedding $\to$ classifier output

The Data Processing Inequality (DPI)

Data Processing Inequality:

If $X \rightarrow Y \rightarrow Z$ is a Markov chain, then:

$$I(X;Z) \leq I(X;Y)$$

"No processing of Y can increase the information that Y contains about X."

The Data Processing Inequality (DPI)

Data Processing Inequality:

If $X \rightarrow Y \rightarrow Z$ is a Markov chain, then:

$$I(X;Z) \leq I(X;Y)$$

"No processing of Y can increase the information that Y contains about X."

Proof sketch:

Chain rule: $I(X;Y,Z) = I(X;Y) + I(X;Z \mid Y) = I(X;Z) + I(X;Y \mid Z)$.

Since $X \rightarrow Y \rightarrow Z$: $I(X;Z \mid Y) = 0$, so $I(X;Y) = I(X;Z) + \underbrace{I(X;Y \mid Z)}_{\geq 0} \geq I(X;Z)$. $\square$

Equality iff Z is a **sufficient statistic** for X w.r.t. Y.

DPI in Neural Networks

A feedforward network with layers $\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_L$:

$$\boxed{\mathbf{X}} \longrightarrow \boxed{\mathbf{h}_1} \longrightarrow \boxed{\mathbf{h}_2} \longrightarrow \cdots \longrightarrow \boxed{\mathbf{h}_L} \longrightarrow \boxed{\hat{Y}}$$

Each layer forms a Markov chain: $\mathbf{X} \to \mathbf{h}_1 \to \mathbf{h}_2 \to \cdots \to \mathbf{h}_L \to \hat{Y}$.

By DPI applied repeatedly:

$$I(\mathbf{X}; \mathbf{h}_1) \ \geq \ I(\mathbf{X}; \mathbf{h}_2) \ \geq \ \cdots \ \geq \ I(\mathbf{X}; \mathbf{h}_L) \ \geq \ I(\mathbf{X}; \hat{Y})$$

Each layer can only lose information about the input.

The network must learn to keep what matters (about Y) and discard what doesn't.

This is exactly the **information bottleneck** idea (coming later).

DPI: Practical Consequences

1. Feature engineering matters: If your features throw away relevant information, no model can recover it. Garbage in, garbage out — **formally**.

2. Dimensionality reduction has a cost: PCA, autoencoders, embeddings — all lose information. The question is whether they keep what matters for your task.

3. Sufficient statistics are special: A statistic $T(\mathbf{X})$ is sufficient for θ iff $I(T;\theta) = I(\mathbf{X};\theta)$. DPI equality case!

4. Post-processing can't help: If model A has less MI with the target than model B, no amount of post-processing of A's output can beat B.

f-Divergences

KL divergence is just one member of a large family.

The right divergence depends on the task.

The f-Divergence Family

f-**Divergence** (Ali–Silvey, Csiszár, 1963/1967):

$$D_f(p\|q) = \sum_x q(x) \quad f\left(\frac{p(x)}{q(x)}\right)$$

where f is convex with $f(1) = 0$.

Key properties (inherited from convexity of f):

- $D_f(p\|q) \geq 0$ always, with equality iff $p = q$
- **Satisfies DPI:** If $X \to Y$, then $D_f(p_Y\|q_Y) \leq D_f(p_X\|q_X)$
- Joint convexity in (p, q)

Different choices of f give different divergences — each with different sensitivities to where p and q disagree.

The Family Members

Name	$f(t)$	**Formula**	**ML use**
KL	$t \log t$	$\sum p \log \frac{p}{q}$	MLE, VI
Reverse KL	$-\log t$	$\sum q \log \frac{q}{p}$	Variational inference
Total Variation	$\frac{1}{2}\lvert t-1 \rvert$	$\frac{1}{2}\sum \lvert p-q \rvert$	Robustness
Chi-squared	$(t-1)^2$	$\sum \frac{(p-q)^2}{q}$	Goodness-of-fit
Jensen–Shannon	(see below)	$\frac{1}{2}D_{\mathrm{KL}}(p \Vert m) + \frac{1}{2}D_{\mathrm{KL}}(q \Vert m)$	**GANs**
Hellinger	$(\sqrt{t}-1)^2$	$\sum(\sqrt{p}-\sqrt{q})^2$	Density estimation

$$\text{JS: } f(t) = -\frac{t+1}{2}\log\frac{t+1}{2} + \frac{t\log t}{2}, \qquad m = \frac{p+q}{2}.$$

Jensen–Shannon Divergence: The Symmetric KL

KL divergence has two problems: it's **asymmetric** and **unbounded**. Jensen–Shannon fixes both:

$$\mathrm{JSD}(p\|q) \quad = \quad \frac{1}{2} D_{\mathrm{KL}}\left(p \,\middle\|\, \frac{p+q}{2}\right) + \frac{1}{2} D_{\mathrm{KL}}\left(q \,\middle\|\, \frac{p+q}{2}\right)$$

Properties:

- ✓ Symmetric: $\mathrm{JSD}(p\|q) = \mathrm{JSD}(q\|p)$
- ✓ Bounded: $0 \leq \mathrm{JSD} \leq \log 2$
- ✓ $\sqrt{\mathrm{JSD}}$ is a true metric
- ✓ Always finite (even when supports differ)

Compare to KL:

- × KL: asymmetric
- × KL: unbounded ($\to \infty$)
- × KL: ∞ when $q(x) = 0, p(x) > 0$

Intuition: Instead of comparing p to q directly, both p and q are compared to their average $m = \frac{p+q}{2}$.

f-Divergences and GANs

The original GAN (Goodfellow et al., 2014) trains a generator G and discriminator D:

$$\min_G \max_D \; \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$$

f-Divergences and GANs

The original GAN (Goodfellow et al., 2014) trains a generator G and discriminator D:

$$\min_G \max_D \; \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))]$$

Key result: For optimal D^*, the GAN objective becomes:

$$\min_G \; 2\,\mathrm{JSD}(p_{\text{data}} \| p_G) - \log 4$$

Training a GAN = minimizing JS divergence between real and generated data.

f-Divergences and GANs

The original GAN (Goodfellow et al., 2014) trains a generator G and discriminator D:

$$\min_G \max_D \; \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]$$

Key result: For optimal D^*, the GAN objective becomes:

$$\min_G \; 2\,\mathrm{JSD}(p_{\text{data}} \| p_G) - \log 4$$

Training a GAN = minimizing JS divergence between real and generated data.

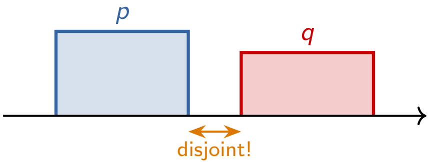
f-**GAN** (Nowozin et al., 2016) generalizes this:

Choose **any** f-divergence $\rightarrow$ get a different GAN variant

KL-GAN, reverse-KL-GAN, Pearson-χ^2-GAN, Hellinger-GAN, ...

When Divergences Differ: Support Mismatch

What happens when p and q have **different supports**?



Divergence	**Disjoint value**	**Gradient?**
$D_{\mathrm{KL}}(p\|q)$	$+\infty$	× undefined
Total Variation	1 (saturated)	× zero
$\mathrm{JSD}(p\|q)$	$\log 2$ (saturated)	× zero

This is why early GANs were hard to train! JS gradients vanish when p_{data} and p_G have little overlap. Fix: **Wasserstein distance** — always gives useful gradients.

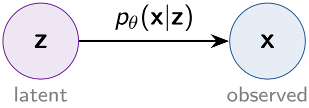
ELBO & Variational Autoencoders

The reverse KL from Lecture 2 leads to
the most important equation in generative modeling.

The Problem: Intractable Posteriors

In a latent variable model, we want $p_\theta(\mathbf{x})$ — the marginal likelihood:

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x} \mid \mathbf{z})\, p(\mathbf{z})\, d\mathbf{z}$$



Two problems:

1. The integral over $\mathbf{z}$ is usually **intractable** (no closed form).
2. The posterior $p_\theta(\mathbf{z} \mid \mathbf{x}) = \frac{p_\theta(\mathbf{x}|\mathbf{z})\, p(\mathbf{z})}{p_\theta(\mathbf{x})}$ requires $p_\theta(\mathbf{x})$ — circular!

Idea: Approximate the intractable posterior $p_\theta(\mathbf{z}|\mathbf{x})$ with a simple distribution $q_\phi(\mathbf{z}|\mathbf{x})$ — using **reverse KL**.

Deriving the ELBO

Start with the log-marginal likelihood and use any distribution $q_\phi(\mathbf{z}|\mathbf{x})$:

$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x}, \mathbf{z})\, d\mathbf{z} = \log \int q_\phi(\mathbf{z}|\mathbf{x})\, \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\, d\mathbf{z}$$

$$\geq \int q_\phi(\mathbf{z}|\mathbf{x})\, \log \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\, d\mathbf{z} \qquad \text{(Jensen's inequality)}$$

Deriving the ELBO

Start with the log-marginal likelihood and use any distribution $q_\phi(\mathbf{z}|\mathbf{x})$:

$$\log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x}, \mathbf{z})\, d\mathbf{z} = \log \int q_\phi(\mathbf{z}|\mathbf{x}) \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\, d\mathbf{z}$$

$$\geq \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p_\theta(\mathbf{x}, \mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\, d\mathbf{z} \qquad \text{(Jensen's inequality)}$$

Evidence Lower BOund (ELBO):

$$\mathcal{L}(\theta, \phi; \mathbf{x}) \quad = \quad \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \tfrac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right] \quad \leq \quad \log p_\theta(\mathbf{x})$$

The **gap** between ELBO and log-evidence is exactly the reverse KL:

$$\boxed{\log p_\theta(\mathbf{x}) \;=\; \underbrace{\mathcal{L}(\theta, \phi; \mathbf{x})}_{\text{ELBO}} + \underbrace{D_{\mathrm{KL}}\big(q_\phi(\mathbf{z}|\mathbf{x}) \,\|\, p_\theta(\mathbf{z}|\mathbf{x})\big)}_{\geq 0}}$$

ELBO = Reconstruction − KL Penalty

Expanding the ELBO gives a beautiful decomposition:

$$\mathcal{L}(\theta, \phi; \mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - D_{\mathrm{KL}}\big(q_\phi(\mathbf{z}|\mathbf{x}) \parallel p(\mathbf{z})\big)$$

Reconstruction: $\mathbb{E}_q[\log p_\theta(\mathbf{x}|\mathbf{z})]$ — **KL penalty:** $D_{\mathbf{KL}}(q \| p(\mathbf{z}))$

"How well can we decode $\mathbf{x}$ from $\mathbf{z}$?"

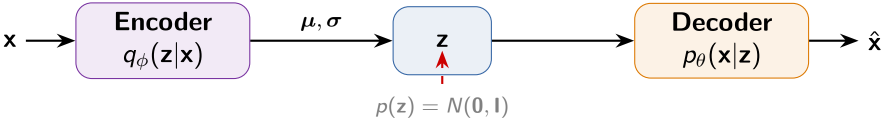
"How close is q to the prior?"

Maximize ELBO = make reconstructions good (high $\log p_\theta(\mathbf{x}|\mathbf{z})$) while keeping $q_\phi(\mathbf{z}|\mathbf{x})$ close to the prior $p(\mathbf{z})$.

The KL penalty acts as a **regularizer** on the latent space.

The Variational Autoencoder (VAE)

The VAE (Kingma & Welling, 2014) implements the ELBO with neural networks:



Encoder $q_\phi(\mathbf{z}|\mathbf{x}) = N(\boldsymbol{\mu}_\phi(\mathbf{x}), \mathrm{diag}(\boldsymbol{\sigma}^2_\phi(\mathbf{x})))$ — outputs $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$

Reparameterization: $\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\varepsilon}, \;\; \boldsymbol{\varepsilon} \sim N(\mathbf{0}, \mathbf{I})$ — enables backprop through sampling

Decoder $p_\theta(\mathbf{x}|\mathbf{z})$ — reconstructs input from latent code

VAE Loss = ELBO in Practice

For Gaussian encoder and prior, the KL term has a **closed form**:

$$D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})) = \frac{1}{2}\sum_{j=1}^{d}\left(\mu_j^2 + \sigma_j^2 - \log\sigma_j^2 - 1\right)$$

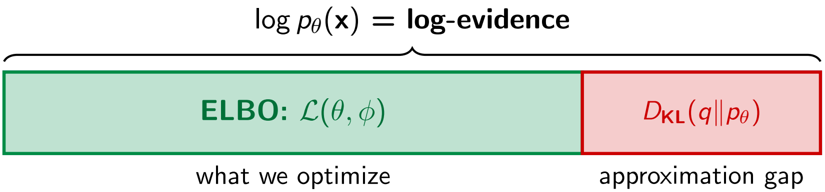
The reconstruction term depends on the output distribution:

Data type	$p_\theta(\mathbf{x}|\mathbf{z})$	**Reconstruction loss**
Continuous	Gaussian	MSE: $\|\mathbf{x} - \hat{\mathbf{x}}\|^2$
Binary / images	Bernoulli	Binary cross-entropy

$$\textbf{VAE loss} = \underbrace{\text{Reconstruction error}}_{\text{MSE or BCE}} + \underbrace{\beta \cdot D_{\mathrm{KL}}(q_\phi \| p)}_{\text{latent regularizer}}$$

$\beta = 1$: standard VAE. $\beta > 1$: β-VAE (disentangled representations).

The ELBO Landscape



Maximize ELBO w.r.t. θ: Pushes up the evidence (better generative model).

Maximize ELBO w.r.t. ϕ: Shrinks the gap (better approximate posterior).

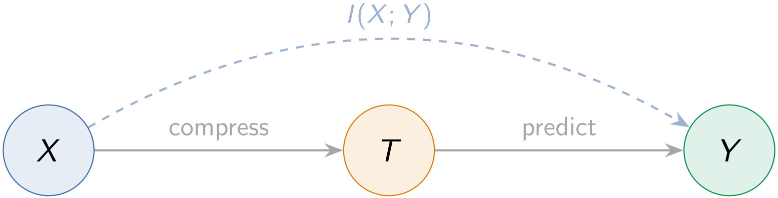
The Information Bottleneck

Compress the input. Preserve what matters for the task.

A principled theory of representation learning.

The Information Bottleneck Principle

Setup: input X, target Y, and a representation T that compresses X.



IB Objective (Tishby et al., 1999):

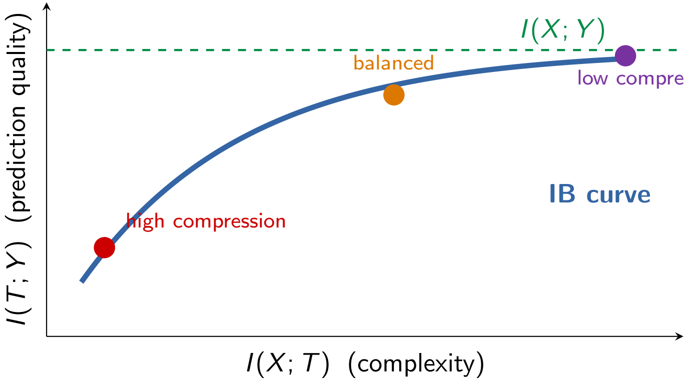
$$\min_{p(t|x)} \quad I(X;T) \; - \; \beta\, I(T;Y)$$

Minimize info kept about X (compress), maximize info kept about Y (predict).

$\beta > 0$ is a Lagrange multiplier controlling the compression–prediction tradeoff.

The Information Plane

Each representation T can be plotted as a point $\big(I(X;T),\, I(T;Y)\big)$:

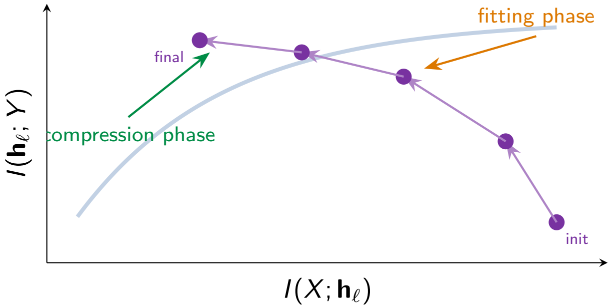


The IB curve traces optimal tradeoffs. Points below it are suboptimal.
β moves you along the curve: small $\beta \to$ more compression, large $\beta \to$ more prediction.

IB and Deep Learning (Tishby & Zaslavsky, 2015)

Claim: Deep networks implicitly optimize the IB tradeoff.

Each hidden layer $\mathbf{h}_\ell$ defines a point in the information plane:



Phase 1 (fitting): $I(\mathbf{h}; Y)$ increases (learning). **Phase 2 (compression):** $I(X; \mathbf{h})$ decreases (forgetting irrelevant details).

IB: The Debate and the Takeaway

Evidence for IB:

- Two-phase behavior observed empirically with saturating activations (tanh)
- Compression correlates with generalization
- IB provides a principled objective for representation learning

Caveats:

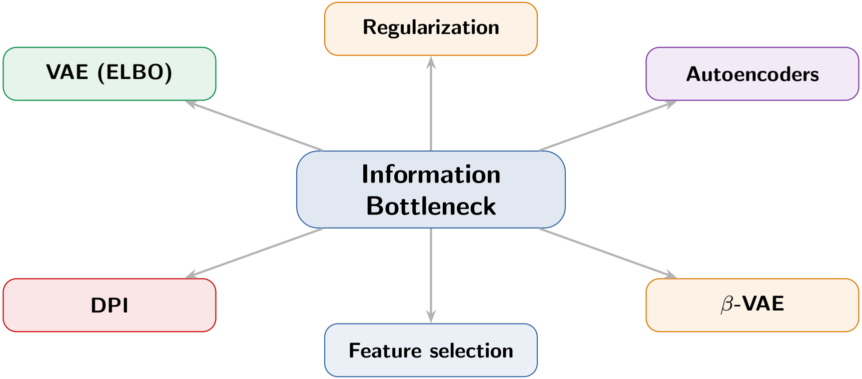
- Compression phase not always observed (e.g., ReLU networks — Saxe et al., 2018)
- MI estimation in high dimensions is hard and noisy
- Deterministic networks: $I(X; \mathbf{h})$ can be infinite

Regardless of the debate, IB gives a valuable conceptual framework:

Good representations compress the input (low $I(X;T)$)
while retaining what's relevant for the task (high $I(T;Y)$).

This intuition underlies autoencoders, bottleneck layers, and regularization.

IB Connects to Everything



IB is a **unifying lens**: many ML techniques can be seen as approximate solutions to the information bottleneck tradeoff.

Today's Toolbox

Concept	Key Idea	ML Application
DPI	$X \to Y \to Z \;\Rightarrow\; I(X;Z) \le I(X;Y)$	Layers lose info, sufficient statistics
f-divergence	$D_f(p\|q) = \sum q\, f(p/q)$	Unifies KL, TV, χ^2, Hellinger
Jensen–Shannon	$\text{JSD} = \frac{1}{2} D_{\text{KL}}(p\|m) + \frac{1}{2} D_{\text{KL}}(q\|m)$	Original GAN objective
ELBO	$\log p(\mathbf{x}) = \text{ELBO} + D_{\text{KL}}(q\|p)$	VAEs, variational inference
VAE loss	Reconstruction $-\beta \cdot D_{\text{KL}}(q\|p(\mathbf{z}))$	Generative modeling
Info Bottleneck	$\min I(X;T) - \beta\, I(T;Y)$	Representation learning

The common thread: Information theory gives us the language to reason about
what neural networks learn, what they forget, and how to train them.

DPI says info is lost; IB says lose the right info; ELBO says how to do it in practice.

Homework

1. **DPI application.** Suppose $X \to Y \to Z$ with $I(X;Y) = 2$ bits.
   (a) What is the maximum possible value of $I(X;Z)$?
   (b) Give an example where $I(X;Z) = I(X;Y)$ (hint: sufficient statistic).
   (c) Give an example where $I(X;Z) = 0$.
2. **f-divergences.** Show that the total variation distance $\mathrm{TV}(p,q) = \frac{1}{2}\sum |p(x) - q(x)|$ is an f-divergence with $f(t) = \frac{1}{2}|t-1|$. Verify f is convex and $f(1) = 0$.
3. **ELBO derivation.** Starting from $\log p_\theta(\mathbf{x})$, derive the ELBO by writing $\log p_\theta(\mathbf{x}) = \log p_\theta(\mathbf{x}) \cdot \int q_\phi(\mathbf{z}|\mathbf{x})\, d\mathbf{z}$ and applying Jensen's inequality. Show that the gap is $D_{\mathrm{KL}}(q_\phi \| p_\theta(\mathbf{z}|\mathbf{x}))$.
4. **IB tradeoff.** A 10-class classifier uses 128-dim features from a bottleneck layer.
   (a) What is the maximum $I(T;Y)$? (Hint: $H(Y) \leq \log_2 10$.)
   (b) If we reduce to 2-dim features, how does the IB tradeoff change?

Questions?