Information theory for and by deep learning

Yoshua Bengio

JULY 11TH, 2019
ISIT 2019, PARIS
Learning Multiple Levels of Abstraction

*Bengio & LeCun 2007*

- The big payoff of deep learning is to allow learning higher levels of abstraction.

- Higher-level abstractions **disentangle the factors of variation**, which allows much easier generalization and transfer.
Invariance and Disentangling

- Invariant features
- Which invariances?
- Alternative: learning to disentangle factors

Good disentangling $\rightarrow$
avoid the curse of dimensionality:

Dependencies are “simple” when the data is projected in the right abstract space of high-level concepts
Disentangling from unsupervised objective 
(Glorot, Bordes & Bengio ICML 2011)

- Early deep learning research already is looking for possible disentangling arising from unsupervised learning of representations
- Experiments on stacked denoising auto-encoders: features specialize to known underlying factors
How to Discover Good Disentangled Representations

• How to discover abstractions?
• What is a good representation? *(Bengio et al 2013)*
• Need clues (= priors) to help disentangle the underlying factors, such as
  – Spatial & temporal scales
  – Marginal independence
  – Simple & sparse dependencies between factors
    • Consciousness prior
  – Causal / mechanism independence
    • Controllable factors
  – Multiple spatial and temporal scales
    • Coarse high-level factors explain lower-level details
Latent Variables and Abstract Representations to Disentangle Manifolds

- Encoder/decoder view: maps between low & high-levels
- Encoder does inference: interpret the data at the abstract level
- Decoder can generate new configurations
- Encoder flattens and disentangles the data manifold
Why Latent Space Generative Models?

Discovery

• Learn relevant factors

“What I cannot create, I do not understand”
- Richard Feynman
Why Latent Space Generative Models?

Discovery

• Learn relevant factors
• Inference

“In what I cannot create, I do not understand”
- Richard Feynman
Why Latent Space Generative Models?

**Discovery**
- Learn relevant factors
- Inference
- Semi-supervised learning

“What I cannot create, I do not understand”
- Richard Feynman
What’s wrong with standard maximum likelihood?

• Pay a huge price for not putting probability mass at even a single training example, even if the data manifold and model manifold are very close.
What’s wrong with standard maximum likelihood?

1. Pay a huge price for not putting probability mass at even a single training example, even if the data manifold and model manifold are very close.
   - So MLE makes the model distribution very fat and conservative

2. Another problem is that MLE measures error bits in pixel space whereas humans really care about errors in abstract space, so we would like loss measured in learned latent space
Classifiers for modeling distributions

• We were inspired by the work of Gutmann & Hyvarinen using probabilistic classifiers to estimate energy functions
  Gutmann & Hyvarinen 2012, Noise-Contrastive Estimation

• In high dimension, more relevant then density is whether you are in-support vs out-of-support

• A classifier of in-support vs out-of-support pays a *constant* price (rather than huge) for not putting support at a training example
Generative adversarial networks (GANs): a two player game with neural networks

Givens:
Samples from a target distribution $P$
(Simple) prior $Q_z$

[Goodfellow et. al., 2014]
Generative adversarial networks (GANs): a two player game with neural networks

**Givens:**
- Samples from a **target distribution** $\mathbb{P}$
- **(Simple) prior** $Q_z$

**Player 1: Generator**
A neural network with parameters, $\theta$, whose samples **fool the discriminator**

---

[Goodfellow et. al., 2014]
Generative adversarial networks (GANs): a two player game with neural networks

Givens:
- Samples from a target distribution $\mathbb{P}$
- (Simple) prior $Q_z$

**Player 1: Generator**
A neural network with parameters, $\theta$, whose samples fool the discriminator

**Player 2: Discriminator**
Distinguish (classify) real and fake correctly

[Goodfellow et. al., 2014]
Generative adversarial networks (GANs): a two player game with neural networks

Givens:
Samples from a target distribution $P$
(Simple) prior $Q_z$

Player 1: Generator
A neural network with parameters, $\theta$, whose samples fool the discriminator

Player 2: Discriminator
Distinguish (classify) real and fake correctly

Minimax on value function
$$\mathcal{V}(P, Q_\theta, D_\phi) = E_P \left[ \log D_\phi(x) \right] + E_{Q_z} \left[ \log(1 - D_\phi(G_\theta(z))) \right]$$
$$(\hat{\theta}, \hat{\phi}) = \arg\min_{\theta} \arg\max_{\phi} \mathcal{V}(P, Q_\theta, D_\phi)$$

Fine print: Continuous data only

[Goodfellow et. al., 2014]
A closer look at the discriminator

• The discriminator defines a lower-bound

\[ 2 \star D_{JSD}(P||Q_\theta) - \log 4 \geq \mathcal{V}(P, Q_\theta; T_\phi) \]
A closer look at the discriminator

- The discriminator defines a lower-bound
  \[ 2 \cdot \mathcal{D}_{JSD}(\mathbb{P}||\mathbb{Q}_\theta) - \log 4 \geq \mathcal{V}(\mathbb{P}, \mathbb{Q}_\theta; T_\phi) \]

- \(f\)-divergence
  \[ \mathcal{D}_f(\mathbb{P}||\mathbb{Q}_\theta) = \mathbb{E}_{\mathbb{Q}_\theta} \left[ f \left( \frac{p(x)}{q_\theta(x)} \right) \right] \]

[f-GAN. Nowozin et. al., 2017]
A closer look at the discriminator

• The discriminator defines a lower-bound
  \[ 2 \cdot D_{JS}(\mathbb{P} || Q_{\theta}) - \log 4 \geq \mathcal{V}(\mathbb{P}, Q_{\theta}; T_{\phi}) \]

• \( f \)-divergence
  \[ D_f(\mathbb{P} || Q_{\theta}) = \mathbb{E}_{Q_{\theta}} \left[ f \left( \frac{p(x)}{q_{\theta}(x)} \right) \right] \]

• Convex dual using neural networks
  \[ D_f(\mathbb{P} || Q_{\theta}) \geq \mathbb{E}_{\mathbb{P}}[T_{\phi}(x)] - \mathbb{E}_{Q_{\theta}}[f^*(T_{\phi}(x))] \]
  \[ = \mathcal{V}_f(\mathbb{P}, Q_{\theta}; T_{\phi}) \]

\[ [f\text{-GAN. Nowozin et. al., 2017}] \]
A closer look at the discriminator

• The discriminator defines a lower-bound
  \[ 2 \cdot D_{JS}(\mathbb{P} \parallel \mathbb{Q}_\theta) - \log 4 \geq \mathcal{V}(\mathbb{P}, \mathbb{Q}_\theta; T_\phi) \]

• \( f \)-divergence
  \[ D_f(\mathbb{P} \parallel \mathbb{Q}_\theta) = \mathbb{E}_{\mathbb{Q}_\theta} \left[ f \left( \frac{p(x)}{q_\theta(x)} \right) \right] \]

• Convex dual using neural networks
  \[
  D_f(\mathbb{P} \parallel \mathbb{Q}_\theta) \geq \mathbb{E}_\mathbb{P}[T_\phi(x)] - \mathbb{E}_{\mathbb{Q}_\theta}[f^*(T_\phi(x))] \\
  = \mathcal{V}_f(\mathbb{P}, \mathbb{Q}_\theta; T_\phi)
  \]

• Estimate using samples

• **Other Examples**
  KL, Jensen-Shannon, Squared Hellinger, Pearson \( \chi^2 \)

• **GANS are a convex dual optimization with a classifier**

[f-GAN. Nowozin et. al., 2017]
MI estimation is classification

Positive sample: sample from the joint distribution (e.g., \((X_{\text{image}}, Y_{\text{image}}))\)

Negative sample: sample from the product of marginals (e.g., \((X_{\text{other}}, Y_{\text{image}}))\)
Using a discriminator to optimize independence, mutual information or entropy

- The GAN discriminator is trained to estimate a similarity function between two distributions.

- Two independent r.v A & B have the property that $P(A,B) = P(A)P(B)$

- Given samples from $P(A,B)$ you can obtain samples from $P(A)P(B)$, e.g. by shuffling A values within a minibatch.

Train a discriminator to separate between pairs (A,B) coming from $P(A,B)$ and pairs coming from $P(A)P(B)$

Brakel & Bengio ArXiv:1710.05050
Using a discriminator to optimize independence, mutual information or entropy

Brakel & Bengio ArXiv:1710.05050

• Train a discriminator to separate between pairs (A,B) coming from P(A,B) and pairs coming from P(A) P(B)

• Generalize this to measuring independence of all the outputs of a representation function (encoder). Maximize independence by backprop independence score into encoder \(\rightarrow\) NON-LINEAR ICA.
Using a discriminator to optimize independence, mutual information or entropy

**MINE: Mutual Information Neural Estimator**
Belghazi et al ArXiv:1801.04062

Same architecture, but with a twist in the training objective which provides an asymptotically correct estimator of mutual independence

- Note that
  \[
  MI(A, B) = H[A] - H[B|A]
  \]
**Mutual information neural estimator (MINE)**

Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, R Devon Hjelm

**Mutual information**: measure of dependence between two variables

\[
I(X; Z) = D_{KL}(\mathbb{P}_{X,Z} \| \mathbb{P}_X \otimes \mathbb{P}_Z) = E_{\mathbb{P}_{X,Z}} \left[ \log \left( \frac{p(x,z)}{p(x)p(z)} \right) \right]
\]

**Fenchel convex dual ($f$-GAN): MINE-$f$**

\[
D_{KL}(\mathbb{P}_{X,Z} \| \mathbb{P}_X \otimes \mathbb{P}_Z) \geq E_{\mathbb{P}_{X,Z}} [T_\phi(x)] - E_{\mathbb{P}_X \otimes \mathbb{P}_Z} [e^{T_\phi(x)} - 1]
\]

**Donsker-Varadhan (tighter): MINE**

\[
D_{KL}(\mathbb{P}_{X,Z} \| \mathbb{P}_X \otimes \mathbb{P}_Z) \geq E_{\mathbb{P}_{X,Z}} [T_\phi(x)] - \log E_{\mathbb{P}_X \otimes \mathbb{P}_Z} [e^{T_\phi(x)}]
\]

[Belghazi et. al., 2018]
Demonstration of estimation

[Belghazi et. al., 2018]
Demonstration of estimation

[Belghazi et. al., 2018]
Mutual Information for Representation Learning

Mutual Information

$I(\text{input}; \text{representation})$

Extract Information

Mutual Information

a.k.a, the “Infomax principle”
Maximizing mutual information: avoid GAN mode dropping by max MI(X,Z)

[Belghazi et. al., 2018]
Maximizing mutual information (stacked MNIST)

| Model            | Modes (max 1000) | $\mathcal{D}_{KL}(P_Y || Q_Y)$ |
|------------------|------------------|---------------------------------|
| DCGAN            | 99               | 3,4                             |
| ALI              | 16               | 5,4                             |
| Unrolled GAN     | 48,7             | 4,32                            |
| VEEGAN           | 150              | 2,96                            |
| PacGAN           | 1000             | 0,6                             |
| DCGAN+MINE       | 1000             | 0,5                             |
We don’t necessarily need generation in pixel space

Generative models (in principle) care about all the pixels
Self-supervision in the wild

• **Question:** What direction is the video running?

![Image of horses]

• **Question:** Do these image patches go together (context prediction)?

![Image of patches]

• **Question:** Where does this patch go (jigsaws)?

![Image of puzzle pieces]

• **Question:** Which sentence follows this first one (Quick-thoughts)?

1. To be or not to be.
2. That is the question.
3. I want a hot dog.
4. I can't do that, Dave.

* e.g., see Logeswaran et al. 2018, Doersch et al. 2015, 2017, Wei et al. 2018
Mutual Information for Self-Supervised Representation Learning

Partial observation / view Representation

\[ I(X; Y) \]
Mutual Information

a.k.a, the “Infomax principle”

Ask questions about the data at the representation / feature level
Maximizing mutual information in encoders

Network encodes the input

The discriminator estimates mutual information (batch-wise)

Estimate is used to maximize the mutual information between encoder input and output

\[ \hat{I}_\omega(X;Y) \]

\[ T_\omega \]

\[ E_\psi \]

\[ Y \]

Input Image

Encoder output

Denoising auto-encoders reconstruction error:
weaker lower bound on \( MI(input, \text{representation}) \)

(\text{Vincent et al ICML 2008})
Using local structure is crucial

Deep Info Max: DIM

Learning Deep Representations by Estimating and Maximizing Mutual Information.
Hjelm, Foderov, Lavoie, Grewal, Bachman, Trischler, and Bengio. ICLR 2019.
Using local structure is crucial

Maximizes the average mutual information across locations

Order of importance
Evaluating the representations

- Linear / Nonlinear classifier on global features
- Linear / Nonlinear classifier on local features
- Measuring mutual information (MINE)
- Measuring dependence (NDM)
- Measuring reconstruction (MS-SSIM)

Hjelm et. al., ICLR 2019
Classification evaluation results

<table>
<thead>
<tr>
<th>Model</th>
<th>CIFAR10</th>
<th>CIFAR100</th>
<th>Tiny Imagenet</th>
<th>STL10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fully supervised</td>
<td>75.39 %</td>
<td>42.27 %</td>
<td>36.60</td>
<td>68.7</td>
</tr>
<tr>
<td>VAE</td>
<td>60.71 %</td>
<td>37.21 %</td>
<td>18.63</td>
<td>58.27</td>
</tr>
<tr>
<td>AAE</td>
<td>59.44 %</td>
<td>36.22 %</td>
<td>18.04</td>
<td>59.54</td>
</tr>
<tr>
<td>ALI/BiGAN</td>
<td>62.57 %</td>
<td>37.59 %</td>
<td>24.38</td>
<td>71.53</td>
</tr>
<tr>
<td>NAT</td>
<td>56.19 %</td>
<td>29.18 %</td>
<td>13.70</td>
<td>64.32</td>
</tr>
<tr>
<td>DIM(MINE)</td>
<td>72.66 %</td>
<td>48.52 %</td>
<td>30.35</td>
<td>69.15</td>
</tr>
<tr>
<td>DIM(JSD)</td>
<td>73.25 %</td>
<td>48.13 %</td>
<td>33.54</td>
<td>72.86</td>
</tr>
<tr>
<td>DIM(infoNCE)</td>
<td>75.21 %</td>
<td>49.74 %</td>
<td>34.21</td>
<td>72.57</td>
</tr>
</tbody>
</table>

Hjelm et. al., ICLR 2019
Deep InfoMax extends easily to other types of data.
Spatio-Temporal DIM (STDIM) for Atari

Anand, Racah, Ozair et. al., ICML workshop on self-supervised learning, 2019
Thanks!

Code for Deep InfoMax: https://github.com/rdevon/DIM

Code for Deep Graph InfoMax: https://github.com/PetarV-/DGI

Some Works covered:

• *Learning Independent Features with Adversarial Nets for Non-linear ICA.*

• *Mutual Information Neural Estimation.*
  Belghazi, Baratin, Ozair, Rajeswar, Bengio, Courville, and Hjelm.
  ICML 2018.

• *Learning Deep Representations by Estimating and Maximizing Mutual Information.*
  Hjelm, Foderov, Lavoie, Grewal, Bachman, Trischler, and Bengio.
  ICLR 2019.

• *Deep Graph Infomax (DGI).*
  Veličković, Fedus, Hamilton, and Hjelm.
  ICLR 2019.

• *Unsupervised State Representation Learning in Atari.*
  Workshop for Self-supervised Learning, ICML 2019.