Generative Adversarial Networks (GANs)

Ian Goodfellow, Research Scientist
MLSLP Keynote, San Francisco 2016-09-13
Generative Modeling

- Density estimation
- Sample generation

Training examples  Model samples

(Goodfellow 2016)
Conditional Generative Modeling

SO, I REMEMBER WHEN THEY CAME HERE
Semi-supervised learning

SO, I REMEMBER WHEN THEY CAME HERE
Maximum Likelihood

\[ \theta^* = \arg\max_{\theta} \mathbb{E}_{x \sim p_{\text{data}}} \log p_{\text{model}}(x \mid \theta) \]
Taxonomy of Generative Models

Maximum Likelihood

Explicit density

- Tractable density
  - Fully visible belief nets
  - NADE / MADE
  - PixelRNN / WaveNet
  - Change of variables models (nonlinear ICA)

Implicit density

Approximate density

- Variational density
  - Variational autoencoder

- Markov Chain density
  - Boltzmann machine

Markov Chain

Direct

GAN
Fully Visible Belief Nets

• Explicit formula based on chain rule:
  \[ p_{\text{model}}(\mathbf{x}) = p_{\text{model}}(x_1) \prod_{i=2}^{n} p_{\text{model}}(x_i \mid x_1, \ldots, x_{i-1}) \]

(Frey et al, 1996)

• Disadvantages:
  • \(O(n)\) non-parallelizable steps to sample generation
  • No latent representation

PixeCNN elephants
(van den Oord et al 2016)
WaveNet

Amazing quality
Sample generation slow
(Not sure how much is just research code not being optimized and how much is intrinsic)

I quoted this claim at MLSLP, but as of 2016-09-19 I have been informed it in fact takes 2 minutes to synthesize one second of audio.
GANs

- Have a fast, parallelizable sample generation process
- Use a latent code
- Are often regarded as producing the best samples
  - No good way to quantify this
Generator Network

\[ x = G(z; \theta^{(G)}) \]

- Must be differentiable
- In theory, could use REINFORCE for discrete variables
- No invertibility requirement
- Trainable for any size of \( z \)
- Some guarantees require \( z \) to have higher dimension than \( x \)
- Can make \( x \) conditionally Gaussian given \( z \) but need not do so

(Goodfellow 2016)
Training Procedure

• Use SGD-like algorithm of choice (Adam) on two minibatches simultaneously:
  
  • A minibatch of training examples
  
  • A minibatch of generated samples
  
• Optional: run $k$ steps of one player for every step of the other player.
Minimax Game

\[ J^{(D)} = -\frac{1}{2} \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) - \frac{1}{2} \mathbb{E}_z \log (1 - D(G(z))) \]
\[ J^{(G)} = -J^{(D)} \]

- Equilibrium is a saddle point of the discriminator loss
- Resembles Jensen-Shannon divergence
- Generator minimizes the log-probability of the discriminator being correct
Non-Saturating Game

\[ J(D) = -\frac{1}{2} \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) - \frac{1}{2} \mathbb{E}_{z} \log (1 - D(G(z))) \]

\[ J(G) = -\frac{1}{2} \mathbb{E}_{z} \log D(G(z)) \]

- Equilibrium no longer describable with a single loss
- Generator maximizes the log-probability of the discriminator being mistaken
- Heuristically motivated; generator can still learn even when discriminator successfully rejects all generator samples

(Goodfellow 2016)
Maximum Likelihood Game

\[ J^{(D)} = -\frac{1}{2} \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) - \frac{1}{2} \mathbb{E}_z \log (1 - D(G(z))) \]

\[ J^{(G)} = -\frac{1}{2} \mathbb{E}_z \exp (\sigma^{-1} (D(G(z)))) \]

When discriminator is optimal, the generator gradient matches that of maximum likelihood

**Discriminator Strategy**

Optimal $D(x)$ for any $p_{\text{data}}(x)$ and $p_{\text{model}}(x)$ is always

$$D(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_{\text{model}}(x)}$$

A cooperative rather than adversarial view of GANs: the discriminator tries to estimate the ratio of the data and model distributions, and informs the generator of its estimate in order to guide its improvements.

---

(Goodfellow 2016)
DCGAN Architecture

Most “deconvs” are batch normalized

(Radford et al. 2015)
DCGANs for LSUN Bedrooms

(Radford et al 2015)
Vector Space Arithmetic

Figure 15.9: A generative model has learned a distributed representation that disentangles the concept of gender from the concept of wearing glasses. If we begin with the representation of the concept of a man with glasses, then subtract the vector representing the concept of a man without glasses, and finally add the vector representing the concept of a woman without glasses, we obtain the vector representing the concept of a woman with glasses. The generative model correctly decodes all of these representation vectors to images that may be recognized as belonging to the correct class. Images reproduced with permission from Radford et al. (2015).

It is common is that one could imagine learning about each of them without having to see all the configurations of all the others. Radford et al. (2015) demonstrated that a generative model can learn a representation of images of faces, with separate directions in representation space capturing different underlying factors of variation. Figure 15.9 demonstrates that one direction in representation space corresponds to whether the person is male or female, while another corresponds to whether the person is wearing glasses. These features were discovered automatically, not fixed a priori. There is no need to have labels for the hidden unit classifiers: gradient descent on an objective function of interest naturally learns semantically interesting features, so long as the task requires such features. We can learn about the distinction between male and female, or about the presence or absence of other features by examples covering all of these combinations of values. This form of statistical separability is what allows one to generalize to new configurations of a person's features that have never been seen during training.
Mode Collapse

- Fully optimizing the discriminator with the generator held constant is safe.

- Fully optimizing the generator with the discriminator held constant results in mapping all points to the argmax of the discriminator.

- Can partially fix this by adding nearest-neighbor features constructed from the current minibatch to the discriminator ("minibatch GAN")

(Salimans et al 2016)
Minibatch GAN on CIFAR

Training Data

Samples

(Salimans et al 2016)
Minibatch GAN on ImageNet

(Salimans et al 2016)
Cherry-Picked Samples
Conditional Generation: Text to Image

Output distributions with lower entropy are easier

- this small bird has a pink breast and crown, and black primaries and secondaries.
- this magnificent fellow is almost all black with a red crest, and white cheek patch.
- the flower has petals that are bright pinkish purple with white stigma
- this white and yellow flower have thin white petals and a round yellow stamen

(Reed et al 2016)
## Semi-Supervised Classification

### MNIST (Permutation Invariant)

<table>
<thead>
<tr>
<th>Model</th>
<th>Number of incorrectly predicted test examples for a given number of labeled samples</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>20</td>
</tr>
<tr>
<td>DGN [21]</td>
<td></td>
</tr>
<tr>
<td>Virtual Adversarial [22]</td>
<td></td>
</tr>
<tr>
<td>CatGAN [14]</td>
<td></td>
</tr>
<tr>
<td>Skip Deep Generative Model [23]</td>
<td></td>
</tr>
<tr>
<td>Ladder network [24]</td>
<td></td>
</tr>
<tr>
<td>Auxiliary Deep Generative Model [23]</td>
<td></td>
</tr>
<tr>
<td>Our model</td>
<td>1677 ± 452</td>
</tr>
<tr>
<td>Ensemble of 10 of our models</td>
<td>1134 ± 445</td>
</tr>
</tbody>
</table>

(Salimans et al 2016)
Semi-Supervised Classification

**CIFAR-10**

<table>
<thead>
<tr>
<th>Model</th>
<th>Test error rate for a given number of labeled samples</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1000</td>
</tr>
<tr>
<td>Ladder network [24]</td>
<td>20.40±0.47</td>
</tr>
<tr>
<td>CatGAN [14]</td>
<td>19.58±0.46</td>
</tr>
<tr>
<td>Our model</td>
<td>21.83±2.01</td>
</tr>
<tr>
<td>Ensemble of 10 of our models</td>
<td>19.22±0.54</td>
</tr>
</tbody>
</table>

**SVHN**

<table>
<thead>
<tr>
<th>Model</th>
<th>Percentage of incorrectly predicted test examples for a given number of labeled samples</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>500</td>
</tr>
<tr>
<td>DGN [21]</td>
<td>36.02±0.10</td>
</tr>
<tr>
<td>Virtual Adversarial [22]</td>
<td>24.63</td>
</tr>
<tr>
<td>Auxiliary Deep Generative Model [23]</td>
<td>22.86</td>
</tr>
<tr>
<td>Skip Deep Generative Model [23]</td>
<td>16.61±0.24</td>
</tr>
<tr>
<td>Our model</td>
<td>18.44 ± 4.8</td>
</tr>
<tr>
<td>Ensemble of 10 of our models</td>
<td>5.88 ± 1.0</td>
</tr>
</tbody>
</table>

(Salimans et al 2016)
Optimization and Games

Optimization: find a minimum:

\[ \theta^* = \arg\min_{\theta} J(\theta) \]

Game:

- Player 1 controls \( \theta^{(1)} \)
- Player 2 controls \( \theta^{(2)} \)
- Player 1 wants to minimize \( J^{(1)}(\theta^{(1)}, \theta^{(2)}) \)
- Player 2 wants to minimize \( J^{(2)}(\theta^{(1)}, \theta^{(2)}) \)
- Depending on \( J \) functions, they may compete or cooperate.
Other Games in AI

- Robust optimization / robust control
  - for security/safety, e.g. resisting adversarial examples
- Domain-adversarial learning for domain adaptation
- Adversarial privacy
- Guided cost learning
- Predictability minimization
- ...
Conclusion

• GANs are generative models that use supervised learning to approximate an intractable cost function

• GANs may be useful for text-to-speech and for speech recognition, especially in the semi-supervised setting

• Finding Nash equilibria in high-dimensional, continuous, non-convex games is an important open research problem