Multilevel Optimization for Generative Models, Games and Robotics

David Pfau
A Whirlwind Tour of Generative Models
Why Generative Models

Move beyond associating inputs to outputs

Understand and imagine how the world evolves

Recognize objects in the world and their factors of variation

Detect surprising events in the world

Establish concepts as useful for reasoning and decision making

Imagine and generate rich plans for the future

Part of a suite of complementary learning systems
Types of Generative Models

**Fully-observed models**
Model observed data directly without introducing any new unobserved local variables.

![Diagram showing fully-observed models](image)

**Implicit Models**
Simulator-based models; models data as a transformation of an unobserved noise source using a parameterized function.

![Diagram showing implicit models](image)

**Latent variable models**
Introduce an unobserved random variable for every observed data point to explain hidden causes.

![Diagram showing latent variable models](image)
Generative Adversarial Networks

\[ \mathcal{L}(\theta, \phi) = \mathbb{E}_{p^*(x)}[\log D_\theta(x)] + \mathbb{E}_{q_\phi(x)}[\log(1 - D_\theta(x))] \]


Minimax optimisation

\[ \min_{\theta} \max_{\phi} \mathcal{L}(\theta, \phi) \]

Optimal Discriminator

\[ D_{\theta^*}(x) = \frac{p^*(x)}{p^*(x) + q_\phi(x)} \]

\[ \mathcal{L}(\theta^*, \phi) = 2JS(p^* || q_\phi) - \log(4) \]
Generative Adversarial Networks

Progressively-Growing Generative Adversarial Networks

Generative Adversarial Networks

GANs are Poor at Matching Statistics of True Distribution

F. Carminati et al (2017)
Generative Models Beyond GANs

Fully-observed models

- Boltzmann Machines

- PixelCNN
  A. van den Oord, N. Kalchbrenner and K. Kavukcuoglu (2016)

- WaveNet
  A. van den Oord et al (2016)

- Normalizing Flows
  D. Rezende and S. Mohamed (2015)

- Real Non-volume-preserving Flows

Latent variable models

- Helmholtz Machines

- Variational Autoencoders
  D. Rezende and S. Mohamed (2014)
  D. Kingma and M. Welling (2014)

- Importance Weighted Autoencoders
  Y. Burda, R. Grosse, R. Salakhutdinov (2016)

- Variational Inference for Monte Carlo Objectives
  A. Mnih and D. Rezende (2016)

- Memory, RL and Inference Networks
  G. Wayne et al. (2018)
Generative Adversarial Networks and Bilevel Optimization
Adversarial Learning

**Generative Adversarial Networks**

**GAN Value Function**

\[
\mathcal{L}(\theta, \phi) = \mathbb{E}_{p^*(x)}[\log D_\theta(x)] + \mathbb{E}_{q_\phi(x)}[\log(1 - D_\theta(x))]
\]


\[z \sim p(z)\]
\[x^{\text{gen}} = f_\phi(z)\]

Minimax optimisation

\[
\min_{\theta} \max_{\phi} \mathcal{L}(\theta, \phi)
\]

Optimal Discriminator

\[
D_{\theta^*}(x) = \frac{p^*(x)}{p^*(x) + q_\phi(x)}
\]

\[
\mathcal{L}(\theta^*, \phi) = 2JS(p^* || q_\phi) - \log(4)
\]
Generative Adversarial Networks

**Discriminator Loss**
\[ \mathcal{L}_D(\theta, \phi) = \mathbb{E}_{p^*}(x)[\log D_\theta(x)] + \mathbb{E}_{q_\phi}(x)[\log(1 - D_\theta(x))] \]

**Generator Loss**
\[ \mathcal{L}_G(\theta, \phi) = -\mathbb{E}_{q_\phi}(x)[\log D_\theta(x)] \]


Optimal Discriminator
\[ D^*(x) = \frac{p^*(x)}{p^*(x) + q_\phi(x)} \]

Optimal Generator
\[ \theta^*(\phi) = \arg \min_{\theta} \mathcal{L}_D(\theta, \phi) \]

Multilevel optimisation
\[ \min_{\phi} \mathcal{L}_G(\theta^*, \phi) \]

\[ \mathcal{L}(\theta^*, \phi) = 2JS(p^* || q_\phi) - \log(4) \]
Bilevel Optimization

\[ \phi^* = \arg \min_{\phi} F(\theta^*, \phi) \]

\[ \theta^*(\phi) = \arg \min_{\theta} f(\theta, \phi) \]

- Also appears under the name “two time scale optimization”
- Naturally arises in adversarial settings in operations research (B. Colson, P. Marcotte and G. Savard 2005)
- Appearance of parameters in both problems couples solutions
- NP-Hard even for bilevel linear programming (P. Hansen, B. Jaumard and G. Savard 1992)
- Little formal study of optimization for deep learning
Bilevel Optimization

\[
\phi^* = \arg \min_{\phi} F(\theta^*, \phi) \\
\theta^*(\phi) = \arg \min_{\theta} f(\theta, \phi)
\]

\[
\vec{v}(\phi, \theta) = \begin{pmatrix}
\nabla_{\phi} F(\theta, \phi) \\
\nabla_{\theta} f(\theta, \phi)
\end{pmatrix}
\]

\[
\vec{v}(\phi, \theta) \neq \nabla \rho(\phi, \theta)
\]

No scalar function whose gradient is the vector field.
Bilevel Optimization

Non-conservative vector field $\mathbf{v}$

No scalar function whose gradient is the vector field.

M. C. Escher
• More than just GANs!

• Many reinforcement learning architectures have similar issues -
  • Actor-critic methods: Deep Deterministic Policy Gradients
  • Inverse reinforcement learning: Guided Cost Learning
  • Imitation learning: Generative Adversarial Imitation Learning

• All these models have close architectural similarities with GANs
Reinforcement Learning and Actor-Critic Methods
Key Idea: Rather than passively observe data, goal is to "pick an action" that maximizes the long-term reward when only the short term reward can be observed at each time step.

Policy: The probability distribution over actions given a state.

Value: The expected long-term reward of following a policy from a given state.

\[ V^*(s) = \max_a \left[ R(s, a) + \gamma \sum_{s'} p(s' | s, a) V^*(s') \right] \]
Reinforcement Learning

Human-level performance on Atari Games

Reinforcement Learning

**Policy Gradient Theorem**

Expected Discounted Reward

\[ \eta(\pi_\theta) = \mathbb{E}_{\pi_\theta} \left[ \sum_{t=1}^{\infty} \gamma^{t-1} r_t \bigg| s_0 \right] \]

Policy Gradient

\[ \frac{\partial \eta}{\partial \theta} = \sum_s d^{\pi_\theta}(s) \sum_a \frac{\partial \pi_\theta(s, a)}{\partial \theta} Q^{\pi_\theta}(s, a) \]

Action-State Value Function

\[ Q^{\pi_\theta}(s, a) = \mathbb{E}_{\pi_\theta} \left[ \sum_{k=1}^{\infty} \gamma^{k-1} r_{t+k} \bigg| s_t = s, a_t = a \right] \]

Expected Discounted State Distribution

\[ d^{\pi_\theta}(s) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\pi_\theta} \left[ s_t \big| s_0 \right] \]
Actor-Critic Methods

Classic Policy Gradient Algorithms

**REINFORCE** (R. J. Williams 1992):
Use real returns to approximate Q(s,a)
- Unbiased but high variance
- Can approximate Q(s,a) to reduce variance
- Scale policy gradient update by TD error

**Actor-Critic Methods:**
Directly plug Q(s,a) estimate into policy gradient
- Simple but often unstable if Q(s,a) not accurate
- Highly biased gradients
- Can provide unbiased gradients if Q is linear function of gradients of policy (R. S. Sutton et al 1999)
Actor-Critic Methods

Deep Deterministic Policy Gradients

$$\frac{\partial \eta}{\partial \theta} = \mathbb{E}_{\pi_{\theta}} \left[ \frac{\partial Q_{\phi}}{\partial a} \frac{\partial \pi_{\theta}}{\partial \theta} \right]$$

- Deterministic policy network
- Alternate between updates to Q by TD learning and policy by gradient descent
- Policy network only sees gradients of TD error wrt action, not rewards
- Similar architecture to NFQCA, SVG(o)

GANs as Actor-Critic

Generative Adversarial Networks

Deep Deterministic Policy Gradients

\[ \mathcal{L}_D(\theta, \phi) = \mathbb{E}_{p^*}(x)[\log D_\theta(x)] + \mathbb{E}_{q_\phi}(x)[\log(1 - D_\theta(x))] \]

\[ \mathcal{L}_G(\theta, \phi) = -\mathbb{E}_{q_\phi}(x)[\log(D_\theta(x))] \]

\[ \mathcal{L}_Q(\theta, \phi) = \mathbb{E}_{\pi_\theta}[\|\mathbb{E}_{\pi_\theta}[r_t + \gamma Q_\phi(s_{t+1}, \pi_\theta(s_{t+1}))] - Q_\phi(s_t, \pi_\theta(s_t))\|^2] \]

\[ \mathcal{L}_\pi(\theta, \phi) = -\mathbb{E}_{\pi_\theta}[Q_\phi(s_0, \pi_\theta(s_0))] \]

D. Pfau and O. Vinyals (2016)
GANs as Actor-Critic

- Actor cannot see state - otherwise it could just copy environment
- Reward is not a function of action - actor and critic become adversarial

D. Pfau and O. Vinyals (2016)
**Generative Adversarial Imitation Learning**

- Replace generator with policy that acts on environment
- Can optimize policy directly (eg with TRPO) or *add actor-critic inside*, making it trilevel optimization problem

---

**Stochastic Value Gradients**

- Learn policy, value function *and model* simultaneously
- Policy update combines 3 gradients simultaneously

---

*J. Ho and S. Ermon (2016)*

*N. Heess, G. Wayne, D. Silver, T. Lillicrap, T. Erez, Y. Tassa (2015)*
Stabilizing Multilevel Optimization
Stabilizing Strategies

**Generative Adversarial Networks**
- Batch normalization
- Historical averaging (Salimans et al 2016)
- Minibatch discrimination (Salimans et al 2016)
- Instance noise (Sønderby et al 2016)
- Weight clipping or approximate Wasserstein loss (Arjovsky et al 2017)
- Unrolling SGD (Metz et al 2017)
- Consensus optimization (Mescheder et al 2017)
- Symplectic Gradient Adjustment (Balduzzi et al 2018)

**Actor-Critic and Inverse RL**
- Compatibility (Sutton et al 1999)
- Polyak averaging (Konda and Tsitsiklis 2003)
- Entropy regularization
- Target networks (Lillicrap et al 2015)
- Replay buffers (Lillicrap et al 2015)
Fixing the GAN Loss

Generative Adversarial Networks

Discriminator Loss
\[ \mathcal{L}_D(\theta, \phi) = \mathbb{E}_{p^*(x)}[\log D_\theta(x)] + \mathbb{E}_{q_\phi(x)}[\log(1 - D_\theta(x))] \]

Generator Loss
\[ \mathcal{L}_G(\theta, \phi) = -\mathbb{E}_{q_\phi(x)}[\log(D_\theta(x))] \]

\[
\begin{align*}
\min_{\phi} \mathcal{L}_G(\theta^*, \phi) \\
\theta^*(\phi) &= \arg\min_{\theta} \mathcal{L}_D(\theta, \phi)
\end{align*}
\]

Can we directly optimize wrt the optimal value?
Fixing the GAN Loss

**Generative Adversarial Networks**

**Discriminator Loss**
\[ \mathcal{L}_D(\theta, \phi) = \mathbb{E}_{p^*(x)}[\log D_\theta(x)] + \mathbb{E}_{q_\phi(x)}[\log(1 - D_\theta(x))] \]

**Generator Loss**
\[ \mathcal{L}_G(\theta, \phi) = -\mathbb{E}_{q_\phi(x)}[\log(D_\theta(x))] \]

\[ \min_{\phi} \mathcal{L}_G(\theta^*, \phi) \]

\[ \theta^*(\phi) = \arg \min_{\theta} \mathcal{L}_D(\theta, \phi) \]

**Unrolled Optimization**

\[ \theta^*(\phi) \approx \theta^K(\phi, \theta) \]

\[ \theta^K(\phi, \theta) = \theta^{K-1}(\phi, \theta) - \epsilon \nabla_\theta \mathcal{L}_D(\theta^{K-1}, \phi) \]

Approximate with a few steps of SGD!
Unrolled GANs

\[
\phi \leftarrow \phi - \epsilon \nabla_\phi \mathcal{L}_G(\theta^K, \phi) \\
\theta \leftarrow \theta - \epsilon \nabla_\theta \mathcal{L}_D(\theta, \phi)
\]

\[H(\theta, \varphi) \rightarrow H(\theta^1, \varphi) \rightarrow H(\theta^2, \varphi) \rightarrow H(\theta^3, \varphi) \rightarrow H(\theta^4, \varphi)\]

\[\theta \rightarrow \text{SGD} \rightarrow \varphi \rightarrow \text{SGD} \rightarrow \varphi \rightarrow \text{SGD} \rightarrow \varphi \rightarrow \text{SGD} \rightarrow \varphi\]

L. Metz, B. Poole, D. Pfau and J. Sohl-Dickstein (2017)

https://github.com/poolio/unrolled_gan
Mode Collapse

Under review as a conference paper at ICLR 2017

Figure 2: Unrolling the discriminator stabilizes GAN training on a toy 2D mixture of Gaussians dataset. Columns show a heatmap of the generator distribution after increasing numbers of training steps. The final column shows the data distribution. The top row shows training for a GAN with 10 unrolling steps. Its generator quickly spreads out and converges to the target distribution. The bottom row shows standard GAN training. The generator rotates through the modes of the data distribution. It never converges to a fixed distribution, and only ever assigns significant probability mass to a single data mode at once.

Figure 3: Unrolled GAN training increases stability for an RNN generator and convolutional discriminator trained on MNIST. The top row was run with 20 unrolling steps. The bottom row is a standard GAN, with 0 unrolling steps. Images are samples from the generator after the indicated number of training steps.

3.2 Pathological Model with Mismatched Generator and Discriminator

To evaluate the ability of this approach to improve trainability, we look to a traditionally challenging family of models to train – recurrent neural networks (RNNs). In this experiment we try to generate MNIST samples using an LSTM (Hochreiter & Schmidhuber, 1997). MNIST digits are 28x28 pixel images. At each timestep of the generator LSTM, it outputs one column of this image, so that after 28 timesteps it has output the entire sample. We use a convolutional neural network as the discriminator. See Appendix C for the full model and training details. Unlike in all previously successful GAN models, there is no symmetry between the generator and the discriminator in this task, resulting in a more complex power balance. Results can be seen in Figure 3. Once again, without unrolling the model quickly collapses, and rotates through a sequence of single modes. Instead of rotating spatially, it cycles through proto-digit like blobs. When running with unrolling steps the generator disperses and appears to cover the whole data distribution, as in the 2D example.

Recurrent Generator

<table>
<thead>
<tr>
<th>Unrolled GAN</th>
<th>Vanilla GAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 0</td>
<td>Step 0</td>
</tr>
<tr>
<td>Step 5k</td>
<td>Step 5k</td>
</tr>
<tr>
<td>Step 10k</td>
<td>Step 10k</td>
</tr>
<tr>
<td>Step 15k</td>
<td>Step 15k</td>
</tr>
<tr>
<td>Step 20k</td>
<td>Step 20k</td>
</tr>
<tr>
<td>Step 25k</td>
<td>Step 25k</td>
</tr>
<tr>
<td>Target</td>
<td>Target</td>
</tr>
</tbody>
</table>

Unrolled GAN

Vanilla GAN

10k steps | 20k steps | 50k steps | 100k steps
Image Reconstruction
Consensus Optimization

\[ \vec{v}(\phi, \theta) = \begin{pmatrix} \nabla_\phi F(\theta, \phi) \\ \nabla_\theta f(\theta, \phi) \end{pmatrix} \]

Non-conservative vector field \( \vec{v} \)

\( \phi \)

\( \theta \)

Courtesy Ferenc Huszar

L. Mescheder, S. Nowozin, A. Geiger (2017)
Consensus Optimization

L. Mescheder, S. Nowozin, A. Geiger (2017)

\[ \vec{v}(\phi, \theta) = \left( \nabla_\phi F(\theta, \phi), \nabla_\theta f(\theta, \phi) \right) \]

\[ L(\phi, \theta) = \frac{1}{2} ||\vec{v}||^2 \]

All extrema become minima!

Courtesy Ferenc Huszar
Consensus Optimization

L. Mescheder, S. Nowozin, A. Geiger (2017)

Combined vector field $\nu - 0.6 \nabla L$

$$\nu(\phi, \theta) = \left( \nabla_\phi F(\theta, \phi), \nabla_\theta f(\theta, \phi) \right)$$

$$L(\phi, \theta) = \frac{1}{2} ||\nu||^2$$

**Update:**

$$-\nu - \gamma \nabla L$$

Courtesy Ferenc Huszar
Symplectic Gradient Adjustment


\[ \vec{v}(\phi, \theta) = \begin{pmatrix} \nabla_{\phi} F(\theta, \phi) \\ \nabla_{\theta} f(\theta, \phi) \end{pmatrix} \]

\[ H(\phi, \theta) = \begin{pmatrix} \nabla_{\phi} \nabla_{\phi} F & \nabla_{\phi} \nabla_{\theta} F \\ \nabla_{\phi} \nabla_{\theta} f & \nabla_{\phi} \nabla_{\theta} f \end{pmatrix} \]

\[ A = \frac{1}{2} (H - H^{T}) \]

**Update:**

\[ -\vec{v} - \gamma A^{T} \vec{v} \]
Conclusions

• GANs are an instance of multilevel deep learning, which also include actor-critic methods, imitation learning and multi-agent games.

• Multilevel optimization is a significantly harder problem than normal optimization, and new algorithms are being developed to handle this complexity

• Unrolled optimization, consensus optimization and symplectic gradient adjustment are all recently-developed algorithms for multilevel problems

• If you just want a generative model, there are better models than GANs. If you can fix multilevel optimization, the horizon of possible models becomes much larger than just GANs.
Many thanks to the IML Organizers, as well as David Balduzzi, Ferenc Huszar, Shakir Mohamed, Oriol Vinyals, Josh Merel, Luke Metz, Ben Poole and Jascha Sohl-Dickstein
References

• R. Sutton and A. Barto (1998). Reinforcement Learning
References