GAN and its Application

Iman Khani Jazani
Generative Adversarial Networks (GANs) for Text Mining

Iman Khani Jazani

October, 2017
• Introduction
• What is GAN
• Application in document modeling
Introduction
Why GANs

Richard Feynman:

“What I cannot create, I do not understand.”

California Institute of Technology
quantum mechanics, quantum electrodynamics
Keine bestätigte E-Mail-Adresse
Why GANs

Yann LeCun:

“GANs, and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion.”

Director of AI Research at Facebook & Silver Professor at the Courant Institute, New York University
AI, machine learning, computer vision, robotics, image compression
Some Application in GANs

- Image, video generation
- SuperResolution
- Interactive image generate
- Text to image
- Speech enhancement
- Document modeling
- Document modeling
- Texture transformation
- Image to image translation
- ...

Ian Goodfellow  Yoshua Bengio
Generator and discriminator in machine learning

![Diagram of neural network layers](image1)

![Diagram of GAN architecture](image2)
How to combine G and D?
What is GAN
Simple explanation for Adversarial Algorithm
In the language of GANs

- Counterfeit == Generator
- Police == Discriminator
Loss function

\[
\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{data}(x)} \left[ \log D(x) \right] + \mathbb{E}_{z \sim p_z(z)} \left[ \log (1 - D(G(z))) \right]
\]
How to train

Training Algorithm

for number of training iterations do
  for k steps do
    • Sample minibatch of m noise samples \( \{z^{(1)}, \ldots, z^{(m)}\} \) from noise prior \( p_g(z) \).
    • Sample minibatch of m examples \( \{x^{(1)}, \ldots, x^{(m)}\} \) from data generating distribution \( p_{data}(x) \).
    • Update the discriminator by ascending its stochastic gradient:
      \[
      \nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[ \log D \left( x^{(i)} \right) + \log \left( 1 - D \left( G \left( z^{(i)} \right) \right) \right) \right].
      \]
  end for
  • Sample minibatch of m noise samples \( \{z^{(1)}, \ldots, z^{(m)}\} \) from noise prior \( p_g(z) \).
  • Update the generator by descending its stochastic gradient:
    \[
    \nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log \left( 1 - D \left( G \left( z^{(i)} \right) \right) \right).
    \]
end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.
Application in document modeling
Motivation

• Representation learning has been a hot topic in recent years, in part driven by the desire to apply the impressive results of Deep Learning models on supervised tasks to the areas of unsupervised learning and transfer learning.
• There are a large variety of approaches to representation learning in general, but the basic idea is to learn some set of features from data, and then using these features for some other task.
• In many domains there may be an abundance of unlabeled data available to us, while supervised data is difficult and/or costly to acquire.
• Some people also feel that we will never be able to build more generally intelligent machines using purely supervised learning.
Other approaches

- LDA
- Replicated Softmax
- DocNADE
- DAE
Autoencoder GAN
Objective Function

\[ f_D(x, z) = D(x) + \max(0, m - D(G(z))) \]

\[ f_G(z) = D(G(z)) \]
Train and Test set

- 20 Newsgroups1 corpus
- 20 Newsgroups consists of 18,786 documents (postings) partitioned into 20 different newsgroups, where each document is assigned to a single topic.
- The data is split into 11,314 training and 7,532 test documents.
- set the vocabulary size to 2000 with preprocessing.
Model details

• In order to make a direct comparison, set the representation size $hd$ (the size of the DAE hidden state) to 50.
• The generator input noise vector $hg$ is also set to be the same size.
• The generator is a 3-layer feedforward network, with ReLU activations in the first 2 layers and a sigmoid nonlinearity in the output layer. Layers 1 and 2 are both of size 300, with the final layer being the same size as the vocabulary. Layers 1 and 2 use batch normalization.
• The discriminator encoder consists of a single linear layer followed by a leaky ReLU nonlinearity (with a leak of 0.02). The decoder is a linear transformation back to the vocabulary size.
• Optimize both $G$ and $D$ using Adam with an initial learning rate of 0.0001. The DAE corruption process is to randomly zero 40% of the input values, and we use a margin size $m$ of 5% of the vocabulary size.
Result
Source code


Thanks for your attention

Imankhanijazani@gmail.com