Generative Adversarial Nets

Disclaimer: Portions of text in this review are directly taken from the research paper linked above. I am in no way claiming ownership of text or claiming originality of it. The notes here are simply for study, personal and research purposes.

Authors

- Authors from University of Montreal, MILA. Include Ian Goodfellow and Yoshua Bengio (Turing Award Winner)

Abstract:

- Proposal of a new framework for estimating generative models via an adversarial process.
- Two data models are simultaneously trained, a *Generative Model G*, that captures the data distribution and a *Discriminative Model D* that estimates the probability that a sample came from the training data rather than G.
- The training procedure for **G is to maximize the probability of D making a mistake**.
- This framework corresponds to a minimax two player game.
- In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D equal to ½ everywhere.
- In the case where G and D are defined by multilayer perceptrons, the entire system can be trained with backpropagation.
- There is no need for any Markov chains or unrolled approximate interference networks during either training or generation of samples.
- Experiments demonstrate the potential of the framework through qualitative and quantitative evaluation of generated samples.

Introduction

- Current state-of-the-art (before this paper in 2014-15) deep learning models possess the capability to map high-dimensional, rich sensory input to a class label (Classification Models). These successes were based on backpropagation and dropout algorithms, using piecewise Rectified Linear Units which have a particularly well behaved gradient.
- Deep Generative models on the other hand have had less of an impact due to the difficulty in approximating many intractable probabilistic computations that arise in maximum likelihood estimation and related strategies, and due to difficulty in leveraging the benefits of piecewise linear units in the generative context.
- In the proposed Adversarial Nets framework, the generative model is pitted against an adversary: a discriminative model that learns to determine whether a sample is from the model distribution or data distribution.
- The generative model can be thought of as analogous to a team of counterfeiters, trying to produce fake currency and use it without detection, while the discriminative model is the police, trying to detect the counterfeit currency.

- Competition in this game drives both teams to improve their methods until the counterfeits are indistinguishable from the genuine articles.
- This framework can yield specific training algorithms for many kinds of models and optimization algorithm.
- In this article, we explore the special case when the generative model generates samples by passing random noise through a multilayer perceptron, and the discriminative model is also a multilayer perceptron. We refer to this case as adversarial nets.
- In this case we can train both models using only highly successful back propagation and dropout algorithms and sample from the generative model using only forward propagation. No approximate inference or Markov chains are necessary.

Related Works

- Before this paper, most generative models were focused on providing a parametric specification of a probability distribution function. The model is then trained by maximizing the log likelihood.
- In such models probably the most successful was Boltzamann machine. Such models generally have intractable likelihood functions and therefore require numerous approximations to the likelihood gradient.
- These difficulties motivated the development of "generative machines" - models that do not explicitly represent the likelihood, yet are able to generate samples from the desired distribution.
- Generative Stochastic Networks are an example of a generative machine that can be trained with exact back propagation rather than numerous approximations by the Boltzmann machines.
- This work extends the idea of Generative machines by eliminating the Markov Chains used in Generative Stochastic Networks.
- Our work backpropagates derivatives through generative processes by using the observation that:
- $$\lim_{\sigma \to 0} \nabla_{\boldsymbol{x}} \mathbb{E}_{\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})} f(\boldsymbol{x} + \epsilon) = \nabla_{\boldsymbol{x}} f(\boldsymbol{x}).$$
    - This mathematical expression is related to the concept of a gradient of a function and its behavior under a small perturbation.
        - $\mathbb{E}_{\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})}$: This denotes the expectation (average) over the random variable epsilon, which is drawn from a multivariate normal distribution with mean 0 and covariance matrix σ2 I, where I is the identity matrix.
        - $f(\boldsymbol{x} + \epsilon)$: This is the function f evaluated at x + epsilon where epsilon is a small perturbation.
        - ∇x : This is the gradient operator with respect to x.
    - The expression is stating that as the perturbation epsilon becomes very small (sigma -> 0), the gradient of the expected value of f(x + epsilon) approaches the gradient of f(x).

- Kingma and Welling (Auto-Encoding Variational Bayes Paper) and Rezende (Stochastic backpropagation and approximate inference in deep generative models) had developed more general stochastic backpropagation rules, allowing one to backpropagate through Gaussian Distributions with finite variance, and to backpropagate to the covariance parameter as well as the mean.
- These backpropagation rules could allow one to learn the conditional variance of the generator, which we treated as a hyperparameter in this work.
- Kingma and Welling and Rezende use Stochastic backpropagation to train variational autoencoders.
- Like GANs, variational autoencoders pair a differentiable generator network in a VAE is a recognition model that performs approximate inference.
- GANs require differentiation through the visible units, and thus cannot model discrete data, while VAEs require differentiation through hidden units, and thus cannot model discrete data, while VAEs require differentiation through the hidden units, and thus cannot have discrete latent variables.
- Other previous works have also taken the approach of using a discriminative criterion to train a generative model. These approaches use criteria that are intractable for deep generative models. These methods are difficult even to approximate for deep models because they involve ratios of probabilities which cannot be approximated using variational approximations that lower bound the probability.
- Noise-contrastive estimation (NCE) involves training a generative model by learning the weights that make the model useful for discriminating data from a fixed noise distribution. Using a previously trained model as the noise distribution allows training a sequence of models of increasing quality.
- This can be seen as an informal competition mechanism similar in spirit to the formal competition used in the adversarial networks game. The key limitation of NCE is that its discriminator is defined by the ratio of the probability densities of the noise distribution and the model distribution, and thus requires the ability to evaluate and backpropagate through both densities.
- Some other previous works have used the idea of having two neural networks compete. Like Predictability Minimization, in which each hidden unit in a neural network, is trained to be different from the output of a second network, which predicts the value of that hidden unit given the value of all other hidden units. This work differs from predictability minimization in three important ways:
- 1. In this work, the competition between the networks is the sole training criterion, and is sufficient on its own to train the network. Predictability minimization is only a regularizer that encourages the hidden units of a neural network to be statistically independent while they accomplish some other task; it is not a primary training criterion.
- 2. The nature of the competition is different. In predictability minimization, two network's outputs are compared, with one network trying to make outputs similar and the other trying to make the outputs different. The output in question is single scalar. In GANs, one network produces a rich, high dimensional vector that is used as the input to another network, and attempts to choose an input that the other network does not know how to process.

- 3. The specification of the learning process is different. Predictability minimization is described as an optimization problem, and have a value function that one agent seeks to maximize and the other seeks to minimize. The game terminates at a saddle point that is a minimum with respect to one player's strategy and a maximum with respect to the other player's strategy.

Body (subtopics being discussed)

Adversarial Nets

- The adversarial modeling framework is the most straightforward to apply when the models are both multilayer perceptrons.
- Prior: A prior distribution in Bayesian analysis is the assumed probability distribution of an uncertain quantity before any evidence is taken into account. It represents our prior belief about the value of a parameter.
- Components of the GAN:
    - Generator (G):
        - Input: A noise vector z sampled from a prior distribution pz(z).
        - Output: A data sample G(z; θ_g) that aims to mimic the real data distribution.
        - Parameters: θ_g are the weights of the multilayer perceptron (MLP) representing the generator.
    - Discriminator (D):
        - Input: A data sample x.
        - Output: A scalar D(x; θ_d) representing the probability that x came from the real data distribution rather than the generator's distribution.
        - Parameters: θ_d are the weights of the MLP representing the discriminator.
- Training Process: The training process involves a two-player minimax game where the generator and discriminator are trained simultaneously but with opposing objectives.
    - Discriminator's objective:
        - Maximize the probability of correctly classifying real data as real and generated data as fake.
        - This is achieved by maximizing the value function V(D, G):
        - $$\max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))].$$
    - Generator's objective:
        - Minimize the probability that the discriminator correctly classifies generated data as fake.
        - This is achieved by minimizing log(1 - D(G(z)):
        - $$\min_G V(D,G) = \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))].$$
- Value Function: The Value function V(D, G) combines both objectives:

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

Value Function in Detail:

- Real Data Term:
- $$\mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)].$$
    - Interpretation: This term encourages the discriminator to output high probabilities (close to 1) for real data samples x drawn from the true data distribution $p_{data}(x)$
    - Maximization: By maximizing log D(x), the discriminator learns to correctly classify data as real.
- Fake Data Term:
- $$\mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))].$$
    - Interpretation: This term encourages the discriminator to output low probabilities (close to 0) for fake data samples G(z) generated by the generator from noise z drawn from the prior distribution pz(z).
    - Maximization: By maximizing log(1 - D(G(z)), the discriminator learns to correctly classify generated data as fake.

Generator's Objective:

- The generator's objective is to minimize the value function V(D,G). Specifically the generator aims to minimize the second term of the value function (Fake Data Term).
-

Conclusions

Citations in Paper