DEEP LEARNING LAB(EEE4423-01) Week 13 - Generative Adversarial Nets

Hyeon Si Eun[1], 2019142214
[1] Department of Electric and Electronic Engineering, Yonsei University

1 Introduction

"Generative Adversarial Nets"[1] is a seminal paper by Ian Goodfellow and his colleagues at the University of Montreal, published in 2014. The paper introduces the concept of Generative Adversarial Networks (GANs), a class of machine learning frameworks designed to model high-dimensional distributions of data. The potential of deep learning lies in discovering rich, hierarchical models that represent the probability distributions of various types of data encountered in AI applications, including natural images, speech-embedded audio waveforms, and natural language corpora. Further, most of the notable successes in deep learning are associated with discriminative models that map high-dimensional, rich sensory inputs to class labels. These successes are primarily based on backpropagation and dropout algorithms using piecewise linear units that perform particularly well. Deep generative models have had less impact due to the difficulty of estimating challenging probabilistic computations in maximum likelihood estimation and related areas, and the difficulty of leveraging the advantages of piecewise linear units in the generative context. We propose a new process for estimating generative models that circumvents these difficulties. In the proposed adversarial nets framework, the generator fights adversarially. The discriminator learns to determine whether a sample came from the model's distribution or the data's distribution. The generator can be thought of as a team of counterfeiters making counterfeit money and trying to use it without detection, while the discriminator can be thought of as the police trying to find counterfeit money. In this game, the competition enhances both teams' methods until the counterfeit money is indistinguishable from the real money. This framework can generate special learning algorithms for various types of models and optimization algorithms. In this paper, we explore a special case where the generative model generates samples by passing random noise through a multilayer perceptron. We call these special cases adversarial nets. In this case, we can train the two models only through highly successful backpropagation and dropout algorithms, and we can sample from the generative model only through feed-forward propagation. Rough estimations and Markov chains are unnecessary.

2 Generative Adversarial Nets

The adversarial modeling framework is most simply applied when the model is a multilayer perceptron. To learn the generator's distribution P_g over data x, we define a prior on input noise variables P_z, express a mapping to data space as $G(z,\theta_g)$, where G is a differentiable function represented by a multilayer perceptron with parameters θ_g. We also define a second multilayer perceptron $D(x,\theta_d)$ that outputs a single scalar. $D(x)$ represents the probability that the data came from x rather than P_g. We train D to maximize the probability distribution of assigning the correct labels to both training examples and samples from the generator. At the same time, we train G to minimize $log(1-D(G(z)))$. In other words, D and G play a two-player minmax game with value function $V(G,D)$.

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log(1-D(G(z)))] \quad (1)$$

Here are some of the key components and mathematical formulas discussed in the paper.

1. Discriminator Network: The discriminator network $D(x)$ is a binary classifier that predicts the probability that x came from the real data distribution rather than the generative model.

2. Generator Network: The generator network $G(z)$ maps from a latent space z (random noise) to the data space. Its goal is to fool the discriminator into thinking that the generated samples are real.

Here, $E_{x\sim p_{data}(x)}[\log D(x)]$ represents the expectation of the logarithm of the discriminator outputs for real samples. The discriminator wants to maximize this term to correctly classify real data.

$E_{z\sim p_z(z)}[\log(1-D(G(z)))]$ represents the expectation of the logarithm of (1 - discriminator outputs) for fake samples. The discriminator also wants to maximize this term to correctly classify fake data.

In the next section, we present a theoretical analysis of adversarial networks, essentially showing the training criteria that allow for recovery of the data-generating distribution when G and D are given enough capacity, as in, for instance, the non-parametric limit. For a less formal but more pedagogical explanation, see Figure 1. In practice, we must implement the game using an iterative, numerical approach. Finishing the optimization of D within the inner loop of learning is computationally disadvantageous and can result in overfitting to the limited dataset. Instead, we alternated between optimizing D for k steps and optimizing G for one step. As a result, D was kept close to its optimal solution, and thus G changed sufficiently slowly. This strategy is similar to maintaining samples of the Markov chain from one learning step to the next in the inner loop of SML/PCD training to avoid burning the Markov chain. The process is formally presented in Algorithm 1.

In practice, Equation 1 may not provide sufficient gradients for G to learn well. Early in learning, when G is poor, D can reject samples with high confidence because they are clearly different from the training data. In such cases, $\log(1-D(G(z)))$ saturates. Instead of training G to minimize $\log(1-D(G(z)))$, we can train G to maximize $\log D(G(z))$. While this objective function creates the same fixed point of the G and D dynamics, it can provide much stronger gradients early in learning.

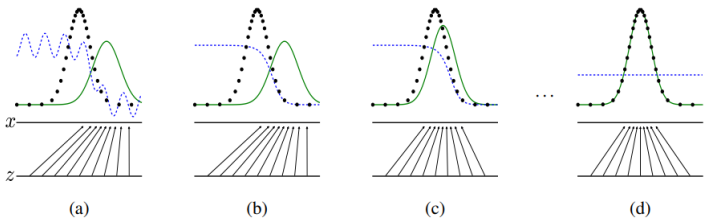

Figure 1: A pictorial representation of the learning process. (a) Initial state of the distributions in learning. (b) It can be seen that D's distribution clearly discerns the data. (c) Once D's learning is somewhat established, G is trained to mimic the distribution of real data, making it hard for D to differentiate. (d) By repeating this process, the distribution created by G becomes nearly identical to the real data distribution, and D exhibits a probability close to 1/2.

- Conditional probability distribution modeled by D: Blue dotted line
- Generative distribution (p_g) modeled by G: Green solid line
- Real data generating distribution (p_x): Black dotted line
- Lower horizontal line: Domain where z is uniformly sampled
- Upper horizontal line: Domain of x
- Upward arrows: Indicate how samples passing through the mapping $x = G(z)$ display the non-uniform p_g

For example, in figure (b), you can see where the maximum value of the green solid line is located. Here, the probability distribution that G models is high, and the real data distribution is low. Therefore, the blue solid line of D's distribution will take a value close to 0, because the probability that the corresponding data is fake is high.

This is an extended abstract. The full paper is available at the this paper.

3 Theoretical Results

Meanwhile, the generator wants to minimize this whole function, meaning that it wants to maximize the probability of the discriminator being fooled. The training procedure for G is to maximize the probability of D making a mistake. The algorithm uses stochastic gradient descent (SGD) to find a saddle point where D is maximized and G is minimized.
If the minmax problem of the GANs presented earlier works properly, p_g should be equal to p_{data} when the minmax problem is at the global optimum, and our proposed algorithm should be able to optimize the equation to have a global optimum. The algorithm for training GANs as outlined in the paper is:

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**
  for k steps **do**
    • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
    • Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
    • Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right].$$

  end for
  • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  • Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for
The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Figure 2: The mini-batch stochastic gradient descent (SGD) training algorithm for Generative Adversarial Nets is as follows. The k applied to the discriminator model is a hyperparameter indicating the number of steps, and we used $k = 1$.

3.1 Global Optimality of $p_g = p_{data}$

Proposition 1 : Consider the optimal discriminator D for any given constructor G. When G is fixed, the optimal discriminator D is

$$D^*_G(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)} \tag{2}$$

Proof : Given G, the training criterion for the discriminator D is to maximize $V(G,D)$. This allows us to find the optimal discriminator D for a given generator G. We derive this by taking the partial derivative of $V(G,D)$ with respect to $D(x)$.

$$V(G,D) = \int_x p_{data}(x) log(D(x)) dx + \int_z p_z(z) log(1 - D(g(z))) dz \tag{3}$$

For any $(a,b) \in \mathbf{R}^2 \setminus \{0,0\}$, the function $y \to alog(y) + blog(1-y)$ achieves its maximum in $[0,1]$ at $\frac{a}{a+b}$. The discriminator does not need to be defined outside of $Supp(p_{data}) \cup Supp(p_g)$, concluding the proof. Note that the training objective for D can be interpreted as maximizing the log-likelihood for estimating the conditional probability $P(Y = y|x)$, where Y indicates whether x comes from p_{data} (with $y = 1$) or from p_g (with $y = 0$). The minimax game in Eq. 1 can now be reformulated as:

$$\begin{aligned} C(G) &= \max_D V(G,D) \\ &= E_{x \sim p_{data}(x)}[\log D^*_G(x)] + E_{z \sim p_z(z)}[\log(1 - D^*_G(G(z)))] \\ &= E_{x \sim p_{data}(x)}[\log D^*_G(x)] + E_{x \sim p_g(x)}[\log(1 - D^*_G(G(x)))] \\ &= E_{x \sim p_{data}(x)}[\log \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}] + E_{x \sim p_g(x)}[\log \frac{p_g(x)}{p_{data}(x) + p_g(x)}] \end{aligned} \tag{4}$$

Theorem 2 : The global minimum of the virtual training criterion $C(G)$ is achieved if and only if $p_g = p_{data}$. At that point, $C(G)$ achieves the value $-log4$.

Proof : For $p_g = p_{data}$, $D^*_G(x) = \frac{1}{2}$, (consider Eq. 2). Hence, by inspecting Eq. 4 at $D^*_G(x) = \frac{1}{2}$, we find $C(G) = \log \frac{1}{2} + \log \frac{1}{2} = -\log 4$. To see that this is the best possible value of $C(G)$, reached only for $p_g = p_{data}$, observe that

$$C(G) = -log(4) + KL\left(p_{data} || \frac{p_{data} + p_g}{2}\right) + KL\left(p_g || \frac{p_{data} + p_g}{2}\right) \tag{5}$$

where KL is the Kullback–Leibler divergence. We recognize in the previous expression the Jensen– Shannon divergence between the model's distribution and the data generating process:

$$C(G) = -log(4) + 2 \cdot JSD(p_{data} || p_g) \tag{6}$$

Since the Jensen–Shannon divergence between two distributions is always non-negative and zero only when they are equal, we have shown that $C^* = -\log(4)$ is the global minimum of $C(G)$ and that the only solution is $p_g = p_{data}$, i.e., the generative model perfectly replicating the data generating process.

3.2 Convergence of Algorithm 1

Proposition 2 : If G and D have enough capacity, and at each step of Algorithm 1, the discriminator is allowed to reach its optimum given G, and p_g is updated so as to improve the criterion

$$E_{x \sim p_{data}(x)}[\log D^*_G(x)] + E_{z \sim p_z(z)}[\log(1 - D^*_G(G(z)))] \tag{7}$$

then p_g converges to p_{data}.
Proof : Consider $V(G,D) = U(p_g, D)$ as a function of p_g as done in the above criterion. Note that $U(p_g, D)$ is convex in p_g. The subderivatives of a supremum of convex functions include the derivative of the function at the point where the maximum is attained. In other words, if $f(x) = \sup_{\alpha \in A} f_\alpha(x)$ and $f_\alpha(x)$ is convex in x for every α, then $\partial f_\beta(x) \in \partial f$ if $\beta = \arg\sup_{\alpha \in A} f_\alpha(x)$. This is equivalent to computing a gradient descent update for p_g at the optimal D given the corresponding G. $\sup_D U(p_g, D)$ is convex in p_g with a unique global optima as proven in Thm 1, therefore with sufficiently small updates of p_g, p_g converges to p_x, concluding the proof.
In practice, adversarial nets represent a limited family of p_g distributions via the function $G(z; \theta_g)$, and we optimize θ_g rather than p_g itself. Using a multilayer perceptron to define G introduces multiple critical points in parameter space. However, the excellent performance of multilayer perceptrons in practice suggests that they are a reasonable model to use despite their lack of theoretical guarantees.

4 Experiments

We trained adversarial nets an a range of datasets including MNIST, the Toronto Face Database (TFD), and CIFAR-10. The generator nets used a mixture of rectifier linear activations and sigmoid activations, while the discriminator net used maxout activations. Dropout was applied in training the discriminator net. While our theoretical framework permits the use of dropout and other noise at intermediate layers of the generator, we used noise as the input to only the bottommost layer of the generator network. If you want to see detailed experimental results, I recommend you to read this paper.

[1] IJ Goodfellow, J Pouget-Abadie, M Mirza, B Xu, D Warde-Farley, S Ozair, and Y Bengio. Generative adversarial networks, 1–9. *arXiv preprint arXiv:1406.2661*, 2014.