Generative Adversarial Nets

Cheng Guan

July 24, 2018

1. Related Work

Today, I continue to read the paper about Generative Adversarial Nets. After reading the paper, I know that Noise-contrastive estimation (NCE) [1] involves training a generative model by learning the weights that make the model useful for discriminating discriminating data from a fixed noise distribution. Using a previously trained models as 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 network 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 back-propagate through both densities.

Some previous work has used the general concept of having two neural networks compete. The most relevant work is predictability minimization [2].

In predictability minimization, 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 of the 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 networks' outputs are compared, with one network trying to make the outputs similar and the other trying to make the outputs different.

2. Adversarial Nets

The adversarial modeling framework is most straightforward to apply when the models are both multilayer perceptrons. To learn the generators distribution p_g over data x, they define a prior on input noise variables $p_z(z)$, then represent 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 . The authors also define a second multilayer perceptron $D(x;\theta_d)$ that outputs a single scalar. $D(x)$ represents the probability that x came from the data rather than p_g . They train D to maximize the probability of assigning the correct label to both training examples and samples from G. They simultaneously train G to minimize $\log(1-D(G(z)))$. In other words, D and G play the following two-player minimax game with value function $V(G,D)$ as shown in Eq.1:

$$\min_G \max_D \mathbb{E}_{x\sim P_{data}(x)} \log[D(x)] + \mathbb{E}_{z\sim P_z(z)} \log[1-D(G(z))] \tag{1}$$

In the next section, the authors present a theoretical analysis of adversarial nets, essentially showing that the training criterion allows one to recover the data generating distribution as G and D are given enough capacity, *i.e.*, in the non-parametric limit. See Fig. 2 for a less formal, more pedagogical explanation of the approach. In practice, they must implement the game using an iterative, numerical approach. Optimizing D to completion in the inner loop of training is computationally prohibitive, and on finite datasets would result in overfitting.

3. Theoretical Results

The generator G implicitly defines a probability distribution p_g as the distribution of the samples $G(z)$ obtained when $z \sim p_z$. Therefore, They would like algorithm to converge to a good estimator of p data , if given enough capacity and training time. The results of this section are done in a non-parametric setting, *e.g.* they represent a model with infinite capacity by studying convergence in the space of probability density functions.

In practice, Eq.1 may not provide sufficient gradient 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 this case, $\log(1-D(G(z)))$ saturates. Rather than training G to minimize $\log(1-D(G(z)))$ they can train G to maximize $\log D(D(G(z)))$. This objective function results in the same fixed point of the dynamics of G and D but provides much stronger gradients early in learning.

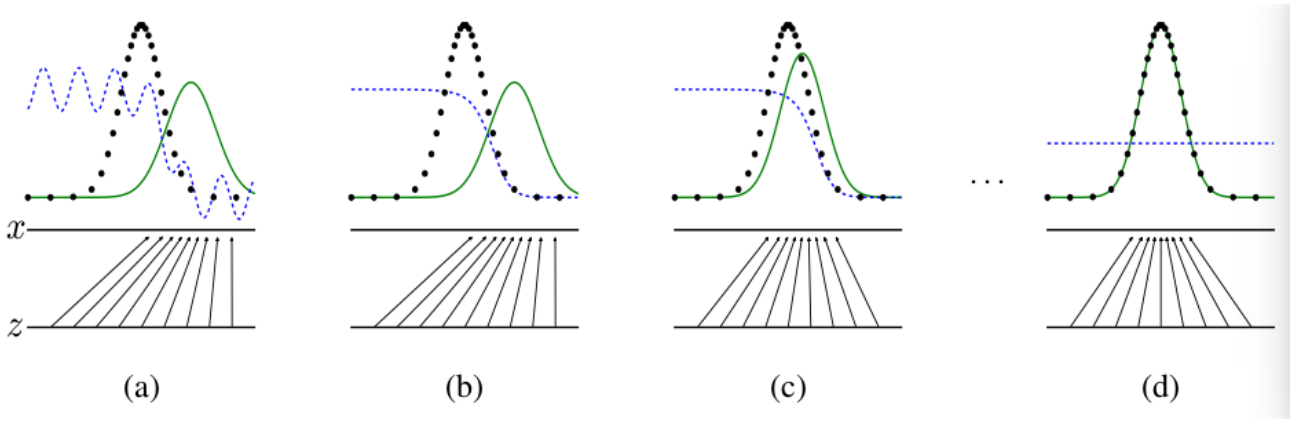

Figure 1. (a) Consider an adversarial pair near convergence: p_g is similar to p data and D is a partially accurate classifier. (b) In the inner loop of the algorithm D is trained to discriminate samples from data, converging to $D^*(x) = \frac{P_{data}(x)}{P_{data}(x)+P_g(x)}$. (c) After an update to G, gradient of D has guided $G(z)$ to flow to regions that are more likely to be classified as data. (d) After several steps of training, if G and D have enough capacity, they will reach a point at which both cannot improve because $p_g = p_{data}$.

References

[1] M. Gutmann and A. Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010. 1

[2] J. Schmidhuber. Learning factorial codes by predictability minimization. *Neural Computation*, 1992. 1