Generative Adversarial Nets

Two models

- **Generative model G**

Generator takes a role of creating data in such a way that it can fool the discriminator

- **Discriminative model D**

Discriminator takes the role of distinguishing between actual and generated(fake) data

Parameters and Variables

$$D = Discriminator$$
$$G = Generator$$
$$\theta_d = Parameters\ of\ discriminators$$
$$\theta_g = Parameters\ of\ generator$$
$$P_z(z) = Input\ noise\ distribution$$
$$P_{data}(x) = Original\ data\ distribution$$
$$P_g(x) = Generated\ distribution$$

3. Adversarial nets

When the models are both multilayer perceptrons,
And to learn the generator's distribution p_g over data

$p_z(x)$: a prior on input noise variables

Generator $G(z;\theta_g)$: a multilayer perceptron
a mapping to data space (where G is a differentiable function)

Discriminator $D(x;\theta_d)$: a multilayer perceptron
outputs a single scalar(probability),
$D(x)$ represents the probability that x came from the data rather than p_g

Train D to maximize the probability of assigning the correct label to both training examples and samples from G.
Simultaneously train G to minimize $\log(1 - D(G(z))$

Therefore, value function $\mathrm{V}(G, D)$ becomes:

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

Binary cross-entropy loss formula

$$L(\hat{y}, y) = [y.log\hat{y} + (1 - \underset{\substack{\uparrow \\ \text{Original data}}}{\boxed{y}}).log(1 - \underset{\substack{\uparrow \\ \text{Reconstructed data}}}{\boxed{\hat{y}}})]$$

Discriminator loss formula

$$L^{(D)} = \max[log(D(x)) + log(1 - D(G(z)))]$$

Generator loss formula

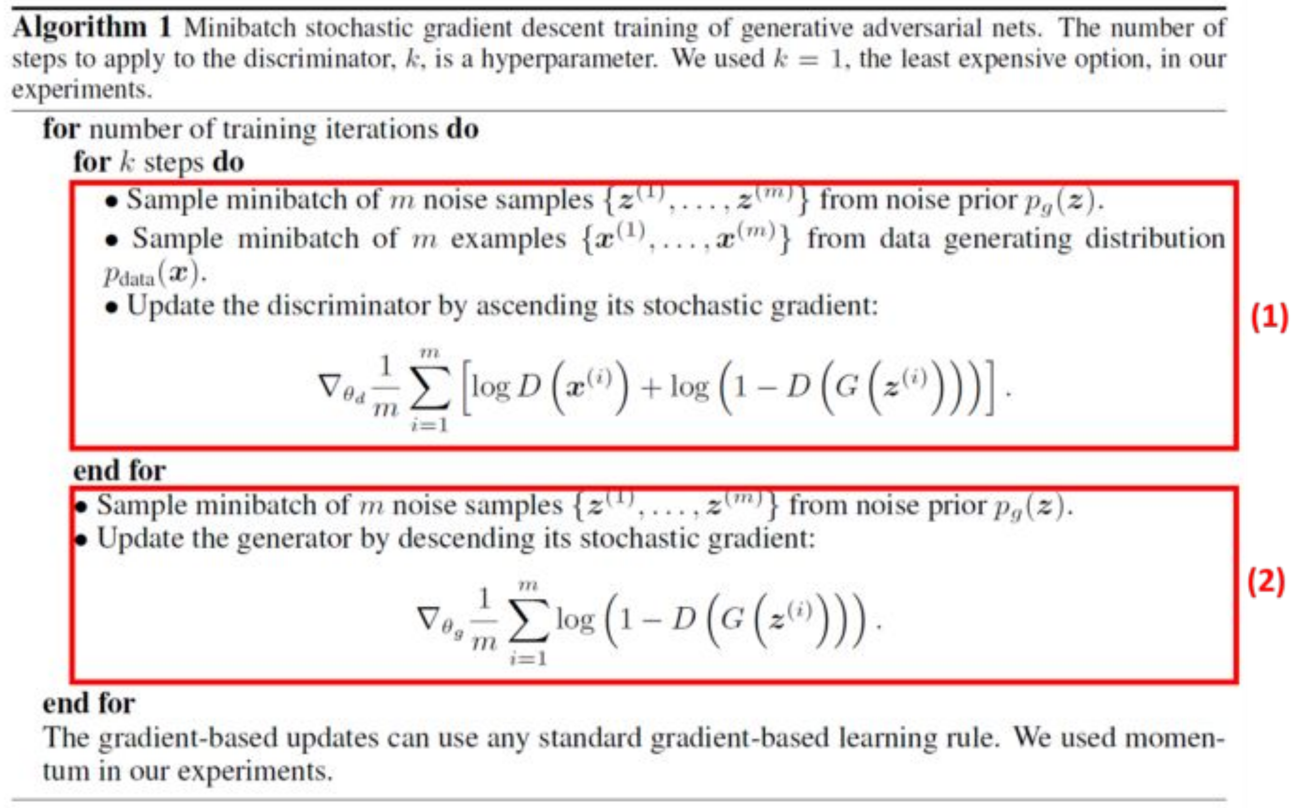
$$L^{(G)} = \min[log(D(x)) + log(1 - D(G(z)))]$$

Combined loss function

$$\min_{G} \max_{D} V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]. \qquad (1)$$

4. Theoretical Results

Algorithm 1



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 $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
        • Sample minibatch of $m$ examples $\{x^{(1)}, \dots, x^{(m)}\}$ from data generating distribution $p_{\text{data}}(x)$.
        • 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                                                                                    (1)
    • Sample minibatch of $m$ noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
    • Update the generator by descending its stochastic gradient:
          $\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right).$
                                                                                               (2)
end for
```

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

- Generator and discriminator are trained separately

- In the first section,

Real data and fake data are inserted into the discriminator with correct labels

Gradients are propagated keeping generator fixed.

The discriminator is updated by ascending its stochastic gradient

- In the second section,

Fake data with fake labels are passed to fool the discriminator keeping the discriminator fixed

The generator is updated by descending its stochastic gradient

References

- https://arxiv.org/pdf/1406.2661.pdf
- https://towardsdatascience.com/the-math-behind-gans-generative-adversarial-networks-3828f3469d9c
- https://jaejunyoo.blogspot.com/2017/01/generative-adversarial-nets-1.html