Adversarial Learning

Kacper

February 21, 2016

Learning density

The aim is to learn distribution p. We want to learn a generative model G, such that for a random variable Z, $G(Z) \sim p$. Or, $p = p_G$, where p_G is distribution of $G(Z)$.

Generative Adversarial Networks

artificial labels

- Random variable X, distributed according to distribution p.
- Some noise random variable Z.
- A generative model G, which transforms Z to $X' = G(Z)$, in other words $X' \sim p_G$.

Consider an artificial label $Y \in \{-1, 1\}$, distributed according to Bernoulli distribution. Consider a random vector W, such that $W_1 = Y$.

if $Y = 1$ then $W_2 = X \sim p$.

if $Y = -1$ then $W_2 = X' \sim p_G$.

I probably could have written $W = (1, X) | (-1, X')$.

Classifier

Let D be discriminative function which is suppose to figure out if W_2 comes from p or p_G (if $W_2 = X'$ or $W_2 = X$).

For a random variable V, $D(W_2)$ a probability that $W_1 = 1$ (label is one).

Learning density

Consider an expression

$$E\left(\log(D(W_2))|Y=1\right)+E\left(\log(1-D(W_2))|Y=-1\right).$$

It can be written using X,$G(Z)$, as a function of D, G

$$V(D,G)=E\log(D(X))+E\log(1-D(G(Z)))$$

For each generator G there exists an optimal discriminator D that maximizes the above function (we will show).

$$C(G)=\sup_D V(D,G)$$

We are going to show that G that minimizes $C(G)$ is the best generator, i.e. $p_G=p$.

Optimal discriminator

For each generator G there exists an optimal discriminator D that maximizes the above function (as we will show).

$$
\begin{aligned}
V(D,G) =& E\left[\log D(X) + \log(1 - D(G(Z)))\right] && (1)\\
=& E\left[\log D(X) + \log(1 - D(X'))\right] && (2)\\
=& \int p(x)\log D(x) + p_G(x)\log(1 - D(x)) && (3)
\end{aligned}
$$

which, for a fixed G, $V(D,G)$ reaches maximum at $D = \frac{p}{p+p_g}$

Experiments

All the experimatal reasults will be discussed at the end.

Optimal generator

For a fixed, optimal $D = \frac{p}{p+p_g}$, V is a divergence

$$V(D,G) = E\left(\log D(X) + \log(1 - D(X'))\right) \tag{4}$$

$$E\left(\log \frac{p(X)}{p(X)+p_g(X)} + \log \frac{p_g(X')}{p(X')+p_g(X')}\right) = \tag{5}$$

$$D_{KL}(p||\frac{p+p_g}{2}) + D_{KL}(p_g||\frac{p+p_g}{2}) \tag{6}$$

Training generative neural networks via Maximum Mean Discrepancy optimization

Relating to distance between measures

The discriminator presented in the previous paper was $C(G) = \sup_D V(D, G)$. This paper suggests to put

$$C(G) = MMD(G(Z), X)$$

and the rest remains the same, we solve

$$\arg\min C(G) \tag{7}$$

Deatils

Suppose θ is parameter of the network G. The gradient of MMD w.r.t. to θ on the smallest minibatch batch $y_1 = G(z_1), y_2 = G(z_2), x_1, x_2$, is

$$
\begin{aligned}
\nabla MMD \sim & \frac{\partial k(y_1, y_2)}{\partial(y_1, y_2)}(\frac{\partial y_1}{\partial \theta}, \frac{\partial y_2}{\partial \theta})+ \\
& - \frac{\partial k(x_1, y_1)}{2\partial y_1}\frac{\partial y_1}{\partial \theta} - \frac{\partial k(x_2, y_1)}{2\partial y_1}\frac{\partial y_1}{\partial \theta}+ \\
& - \frac{\partial k(x_1, y_2)}{2\partial y_1}\frac{\partial y_1}{\partial \theta} - \frac{\partial k(x_2, y_2)}{2\partial y_1}\frac{\partial y_1}{\partial \theta}
\end{aligned}
$$

Training algortihm

Algorithm 1 Stochastic gradient descent for MMD nets.

Initialize M, θ, α, k
Randomly divide training set X into N_{mini} mini batches
for $i \leftarrow 1$, number-of-iterations **do**
  Regenerate noise inputs $\{w_i\}_{i=1,\ldots,M}$ every r iterations
  for $n_{\text{mini}} \leftarrow 1, N_{\text{mini}}$ **do**
    for $m \leftarrow 1, M$ **do**
      $y_m \leftarrow G_\theta(w_m)$
    end for
    compute the n'th minibatch's gradient
    update learning rate α (e.g., RMSPROP)
    $\theta \leftarrow \theta - \alpha \nabla C_n$
  end for
end for

Comments

Arthur noticed that varaince of MMD should be taken into account in the objective i.e.

$$\frac{\overline{MMD}}{\sqrt{var(\overline{MMD})}}.$$

Are random variables suitable model for images? 'It is well known that, for any distribution p and any continuous distribution p_z on sufficiently regular spaces (...), there is a function G, such that $G(Z)$'

Deep Mean Maps, digression

Not excatly what I've expected

Use random features as a layer in a network, L^i is a layer for i-th picture

$$L^i \in R^{H,W}$$

H, W is number of (super) pixels. For frequencies $\omega_1, \cdots, \omega_d, \cdots, \omega_D$, use feature in network

$$\mu_d^i = \sum_{w=1,h=1}^{H,W} \cos(\omega_d L_{h,w}^i + b))$$

Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks

The conditional generative adversarial net

D receives additional information L as input. This might contain, say, information about the class of the training example X.

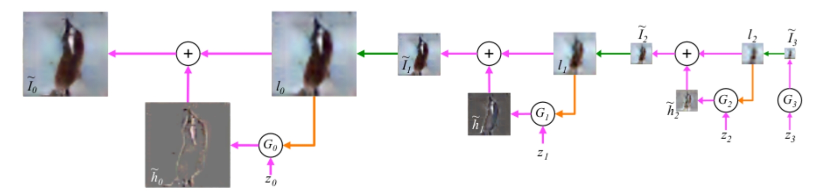
$$\arg\min_G \sup_D E \log(D(X, L)) + E \log(1 - D(G(Z), L))$$

M. Mirza and S. Osindero. Conditional generative adversarial nets.

Laplacian Pyramid

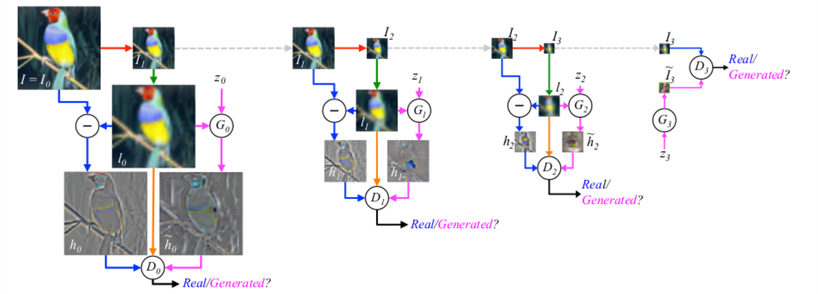
d is downsampling operation which blurs and decimates a $j \times j$ image I to $d(I)$, which is $j/2 \times j/2$. u is upsampling opertation that which smooths and expands. $I_k = d^k(I)$ h_k are residuals i.e $h_k = I_k - u(d(I_k))$. They aim to lear residuals.

Sampling



$\tilde{I}_0$
+
l_0
$\tilde{h}_0$
G_0
z_0
$\tilde{I}_1$
l_1
$\tilde{h}_1$
G_1
z_1
$\tilde{I}_2$
l_2
$\tilde{I}_3$
$\tilde{h}_2$
G_2
G_3
z_2
z_3

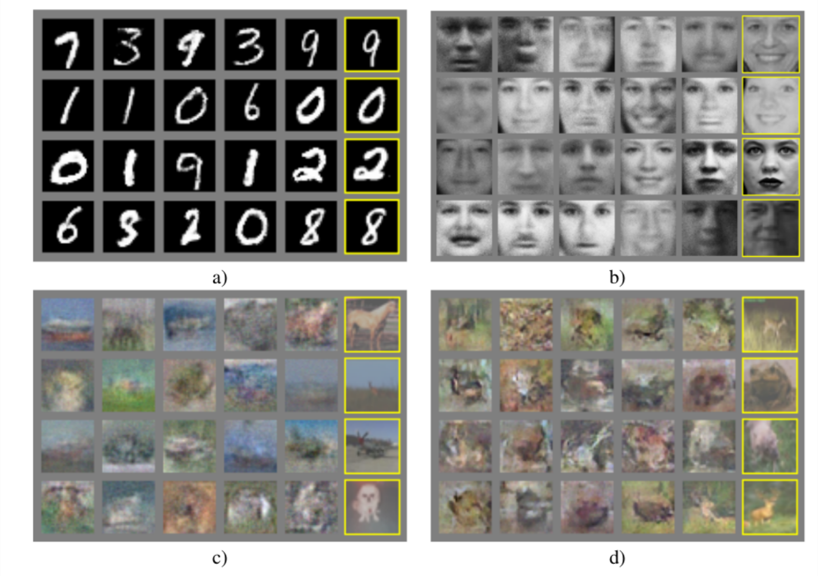
Learning



$I = I_0$
I_1
z_0
G_0
l_0
h_0
$\tilde{h}_0$
D_0
Real/Generated?
I_1
I_2
z_1
G_1
l_1
h_1
$\tilde{h}_1$
D_1
Real/Generated?
I_2
I_3
z_2
l_2
G_2
h_2
$\tilde{h}_2$
D_2
Real/
Generated?
I_3
$\tilde{I}_3$
D_3
Real/
Generated?
G_3
z_3

Resluts

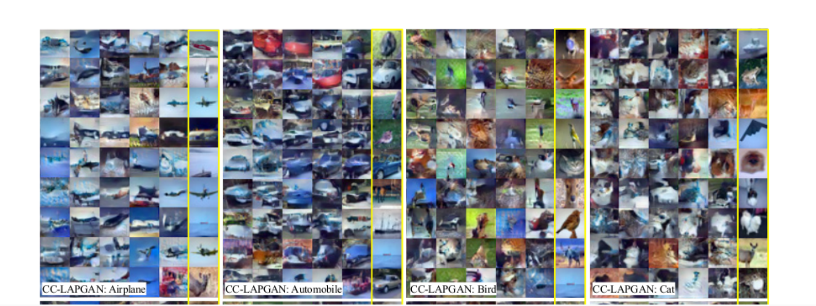
GAN



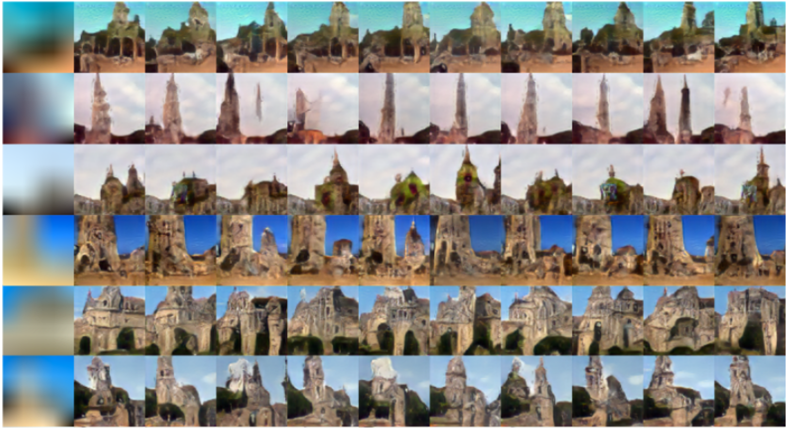
a) MNIST b) TFD c) CIFAR-10 (fully connected model) d)

MNIST

LAPGAN



CIFAR 10



LSUN