Pix2Pix

A Preprint

1 Abstract

general-purpose solution to image-to-image translation problems. These networks not only learn the mapping from input image to output image, but also learn a loss function to train this mapping. This makes it possible to apply the same generic approach to problems that traditionally would require very different loss formulations. We demonstrate that this approach is effective at synthesizing photos from label maps, reconstructing objects from edge maps, and colorizing images, among other tasks. Indeed, since the release of the pix2pix software associated with this paper, a large number of internet users (many of them artists) have posted their own experiments with our system, further demonstrating its wide applicability and ease of adoption without the need for parameter tweaking.

2 Main

Image2Image translations usually formulated as per-pixel classification or regression however they suffer from unstructured output space while GAN has structured loss: structured losses penalize the joint configuration of the output. generator we use a "U-Net"-based architecture [50], and for our discriminator we use a convolutional "PatchGAN" classifier, which only penalizes structure at the scale of image patches. A similar PatchGAN architecture was previously proposed in [38] to capture local style statistics.

$$\mathcal{L}_{cGAN} = \mathbb{E}_{x,y} \log(D(x;y)) + \mathbb{E}_{x;z}[\log(1 - D(x; G(x;y)))]$$

where G tries to minimize this objective against an adversarial D that tries to maximize.

Previous research found out that is t better to add L1-L2 type loss because the discrimators job stays the same while generator is forced to go to ground-truth. In the paper also explore this option, using L1 distance rather than L2 as L1 encourages less blurring:

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x;y;z}||y - G(x;z)||$$

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \lambda \mathcal{L}_{L1}(G)$$

3 Architecture

- Generator (U-net)
- Discriminator PatchGAN
- Loss - previously mentioned + L_1

Generator:	$\hat{y} = G(x)$
Discriminator:	$D(x, y), \quad D(x, \hat{y}) = D(x, G(x))$
Adversarial loss (cGAN):	$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_x[\log(1 - D(x, G(x)))]$
L1 loss:	$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y}[\|y - G(x)\|_1]$
Full objective:	$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G)$

Algorithm 1 Pix2Pix Training Pipeline

Input: paired images (x, y), generator G, discriminator D, weight λ
for each training iteration do
    $\hat{y} \leftarrow G(x)$ ▷ Generate output
    $D_{\text{real}} \leftarrow D(x, y)$
    $D_{\text{fake}} \leftarrow D(x, \hat{y})$
    $\mathcal{L}_D \leftarrow -\log D_{\text{real}} - \log(1 - D_{\text{fake}})$
    Update D using $\nabla_D \mathcal{L}_D$
    $\mathcal{L}_{cGAN} \leftarrow -\log D_{\text{fake}}$
    $\mathcal{L}_{L1} \leftarrow \|y - \hat{y}\|_1$
    $\mathcal{L}_G \leftarrow \mathcal{L}_{cGAN} + \lambda \mathcal{L}_{L1}$
    Update G using $\nabla_G \mathcal{L}_G$
end for