Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks

authors: Google Brain

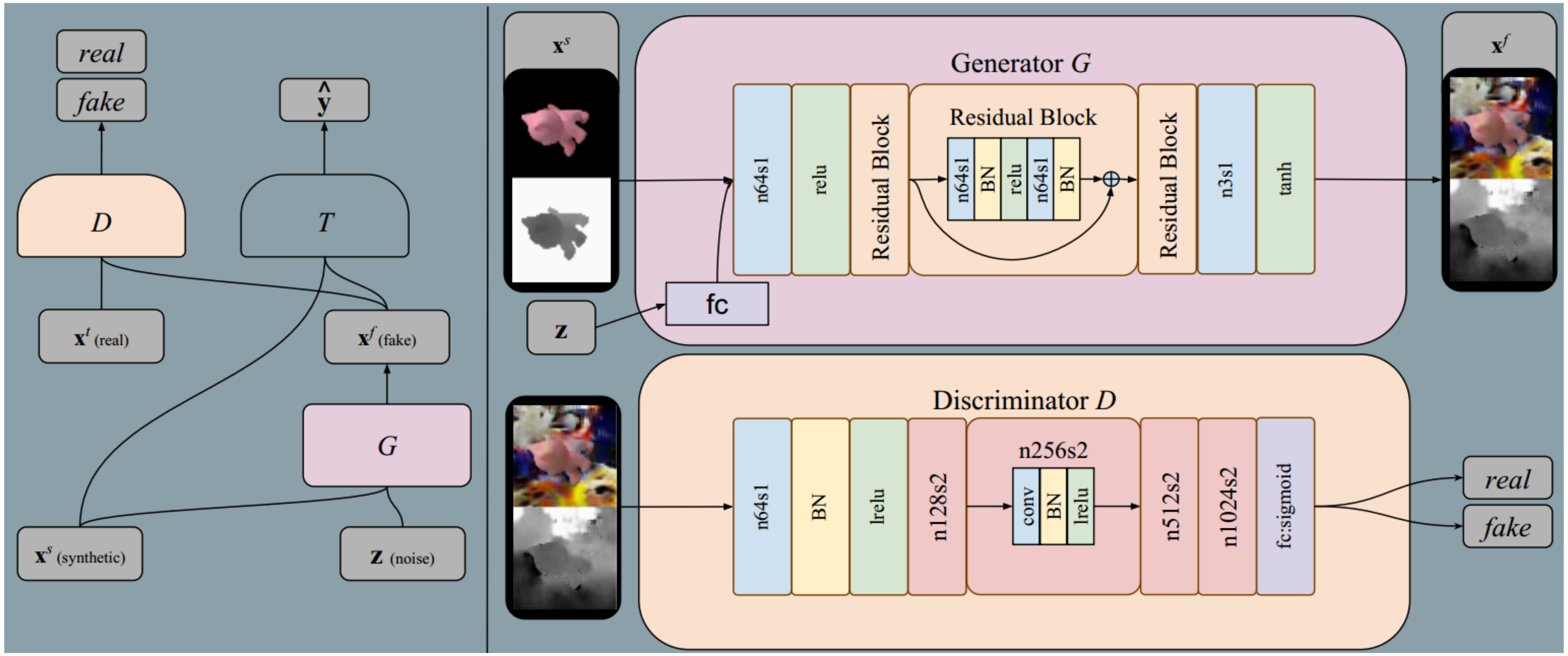
published: 2016-12

The main puporses of this paper is to solve the domian adaption problems. This paper using the sourse domian samples to generate target domain samples via Generative Network.

> We would like to transfer knowledge learned from a source domian, for which we have labeled data, to a target domain, for which we have no ground truth labels.

> We employ a generative adversarial objective to encourage G to produce images that are similar to the target domain images. $G(x^s, z; \theta_G) \rightarrow x^f$ maps a source image x^s and a noise vector z to an adapted image x^f.

> The discriminator tries to distinguish between 'fake' images X^f produced by the generator, and 'real' images from the target domain X^t.



main formulations

$$\min_{\theta_G,\theta_T}\max_{\theta_D}\alpha L_D(D,G)+\beta L_t(G,T)+\gamma L_c(G)$$

GAN Loss

$$L_d(D,G) = E_{x^t}[\log D(x^t;\theta_D)] + E_{x^s,z}[\log(1 - D(G(x^s,z;\Theta_G);\Theta_D))]$$

Task Specific Loss
L_t is a task=specific loss, and in the case of classification we use a typical softmax cross-entropy loss:

$$L_t(G,T) = E_{x^s,y^s,z}[-y^{s^T} \log T(G(x^s,z;\Theta_G);\Theta_T) - y^{s^T} \log T(x^s);\Theta_T]$$

When training T only on adapted images, it's possible to achieve similar performance, but doing so many require many runs with different initializations due to the instability of the model.

> During the first stpe, we update the discriminator and task-specific parameters θ_D,θ_T, while keeping the generator parameters $\theta_G fixed$. During the second step we fix θ_D, θ_T and update θ_G.

Content-similarity loss
We may expect the hues of the source and adapted images to be the same. This prior knowledge can be formalized via the use of an additional loss that penalizes large differences between source and generated images fro foreground pixels only.

$L_c(G = E_{x^s,z}[$