Unsupervised Cross-Domain Image Generation

Xinru Hua, Davis Rempe, Haotian Zhang
{huaxinru, drempe, haotianz}@stanford.edu

Introduction and Motivation

• We explore the fundamental problem of general domain transfer by replicating a recent method presented in “Unsupervised Cross-Domain Image Generation” [1]. This method maps a sample from one domain to a similar sample in a different domain using a generative adversarial network (GAN) in an unsupervised fashion.
• We attempt to replicate this method in two visual application areas - digits and faces - and perform additional analysis on various components of the approach. We achieve similar visual results for digits but not faces, finding that the training procedure is crucial to a successful GAN implementation.

Data Collection

• For digit transfer, we use images from the Street View House Numbers (SVHN) dataset [2] and MNIST database of handwritten digits [3].
• For face transfer, we use a subset of the MS-Celeb-1M dataset [4] and our own dataset of emojis we created using Bitmoji [5].
• Digit and face images are resized to (32, 32) and (96, 96) respectively, and all images are normalized to [-1, 1].

Model Architecture and Loss

• \( f \) - encodes an image into a feature vector
• \( g \) - generates an image from feature vector
• \( D \) - classifies each image into three classes

\[
\mathcal{L}_D = - \sum \log D_1(g(f(x))) - \sum \log D_2(g(f(x))) - \sum \log D_3(g(f(x)))
\]

\[
\mathcal{L}_G = \mathcal{L}_{GAN} + \alpha \mathcal{L}_{adv} + \beta \mathcal{L}_{DIV} + \gamma \mathcal{L}_{TV}
\]

\[
\mathcal{L}_{DIV} = - \sum \log D_1(g(f(x))) - \sum \log D_2(g(f(x)))
\]

\[
\mathcal{L}_{DIV} = \sum \| f(x) - f(g(x)) \|_1
\]

Application | Name | Size | Type |
---|---|---|---|
Digit Transfer | SVHN | 531,131 | RGB |
| MNIST | 60,000 | Grayscale |
Face Transfer | MS-Celeb-1M | 912,224 | RGB |
| Bitmoji | 1,000,000 | RGB |

Discussion and Additional Analysis

Best result (digit transfer)

• We trained our best model (Fig. 1 and Fig. 2) for 12 epochs using extra training set of SVHN and training set of MNIST on Google Cloud (8 Intel Broadwell CPUs and 1 NVIDIA Tesla P100 GPU).
• For quantitative evaluation, we judged our transfer results using an MNIST classifier (Table 1).

GAN training strategies

• Balance of discriminator and generator - the methods we tried include: a) Train generator more than discriminator, b) Hyperparameter settings, c) Lower bound on discriminator loss, and d) Model architecture.
• Normalization – standard normal vs. transform to [-1, 1]. Tanh as the last layer of generator.
• Avoid sparse gradients - downsampling with maxpool vs. strided convolution. LeakyReLU vs. ReLU in discriminator.
• Optimization parameters – learning rate schedule, SGD vs. Adam, weight decay.

Effectiveness of \( \mathcal{L}_{const} \)

• Without \( \mathcal{L}_{const} \) - generated digits are clear but obviously lack supervision.
• Cross entropy loss for \( \mathcal{L}_{const} \) - between transferred images and labels of original images. Easily achieved similarly excellent digit transfer performance to the referenced paper, but challenges idea of an “unsupervised” method.

References

[5] [https://www.bitmoji.com/](https://www.bitmoji.com/)