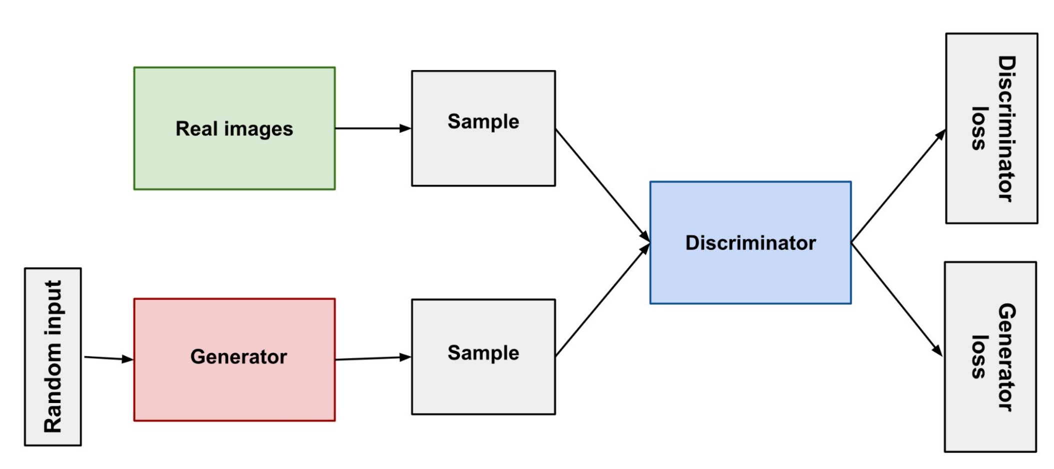
GAN and Cycle-GAN

Qingyun Wang



Generative Adversarial Nets

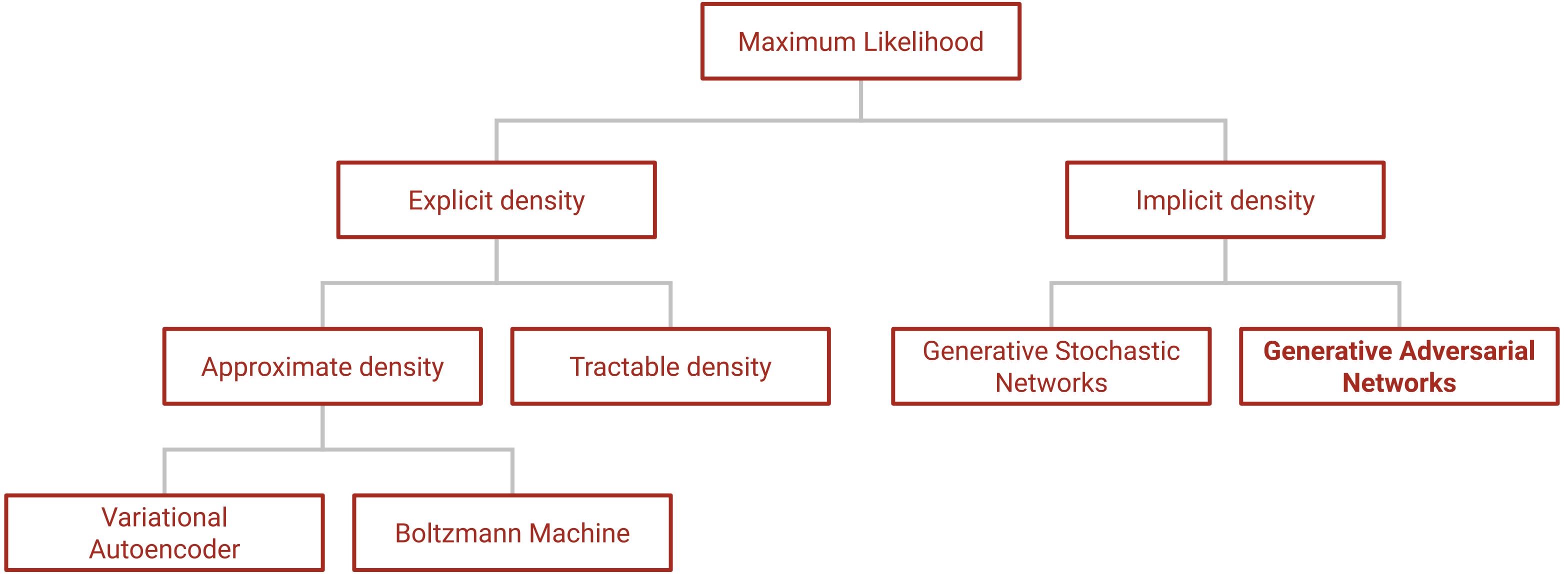
Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio

Why do we need generative models?

- Test our ability to use high-dimensional, complicated probability distributions
- Simulate possible futures for planning or simulated RL
- Handle missing data especially in semi-supervised learning
- Work with multi-modal outputs

Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

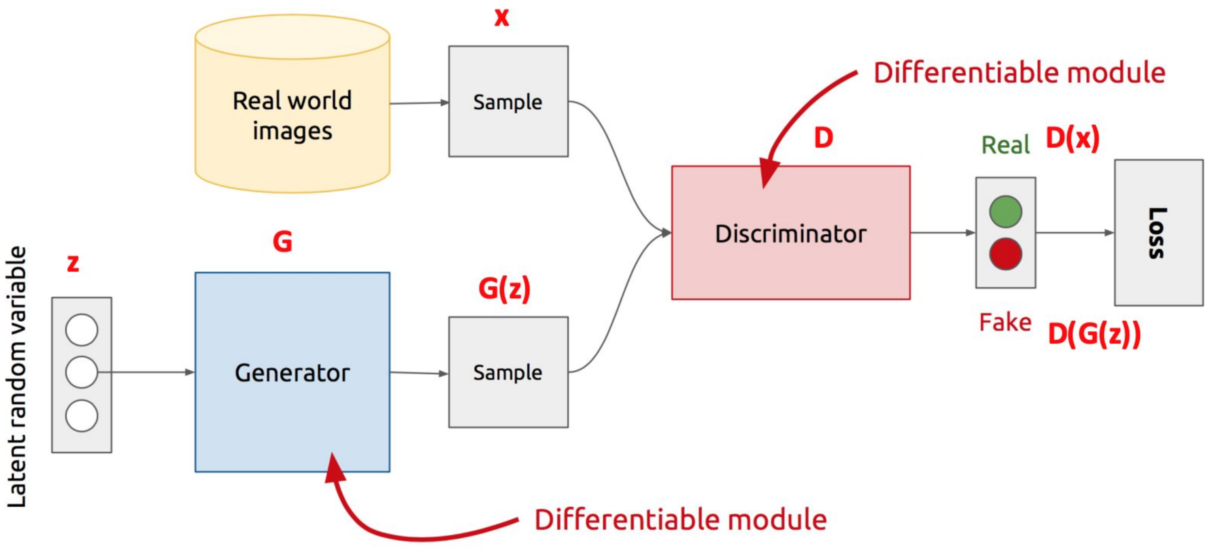
GAN compared with other generative models



Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

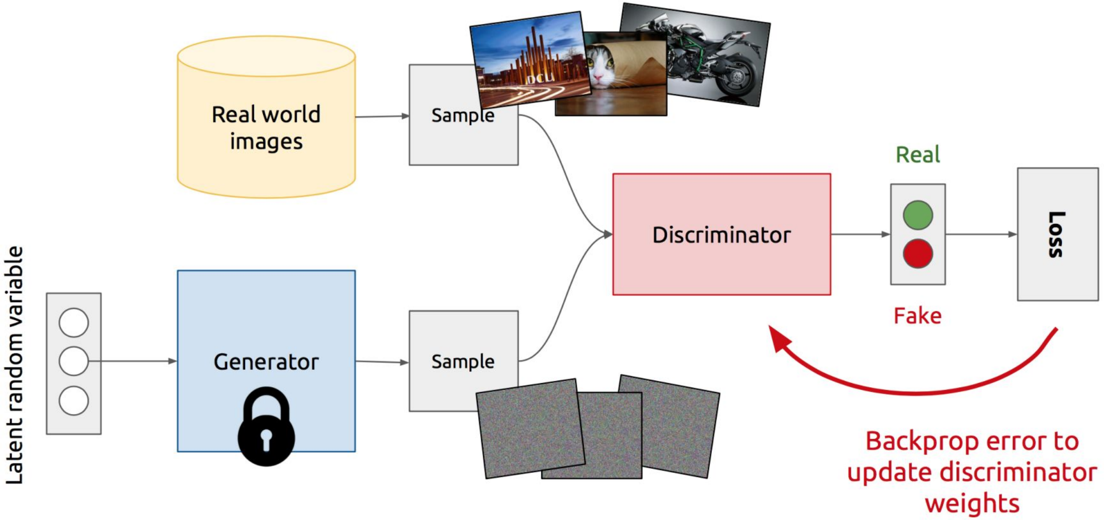
Overview

- Use a latent code
- No Markov Chain
- No variational bound
- Play a minmax game



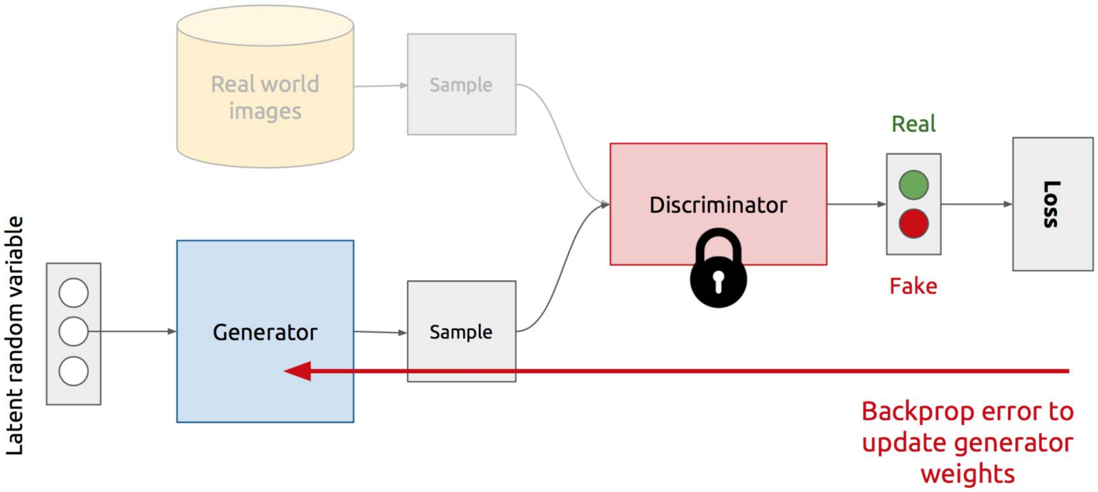
https://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-generative-models-and-adversarial-training-upc-2016

Training Discriminator



https://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-generative-models-and-adversarial-training-upc-2016

Training Generator



https://www.slideshare.net/xavigiro/deep-learning-for-computer-vision-generative-models-and-adversarial-training-upc-2016

MinMax Game

- The Discriminator is trying to maximize its reward
- The Generator is trying to minimize Discriminator's reward

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

Discriminator pushes up

Discriminator's ability to recognize data as being real

Discriminator's ability to recognize generator samples as being fake

Generator pushes down

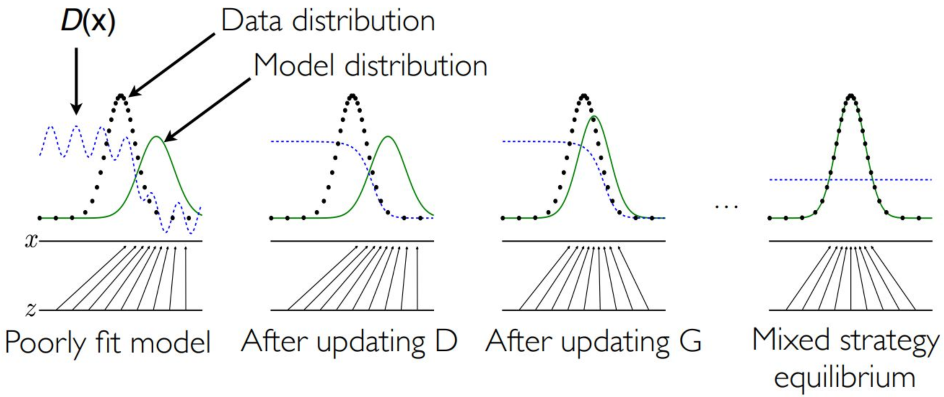
Discriminator Strategy

- Optimal strategy for any $p_{model}(x)$ is always

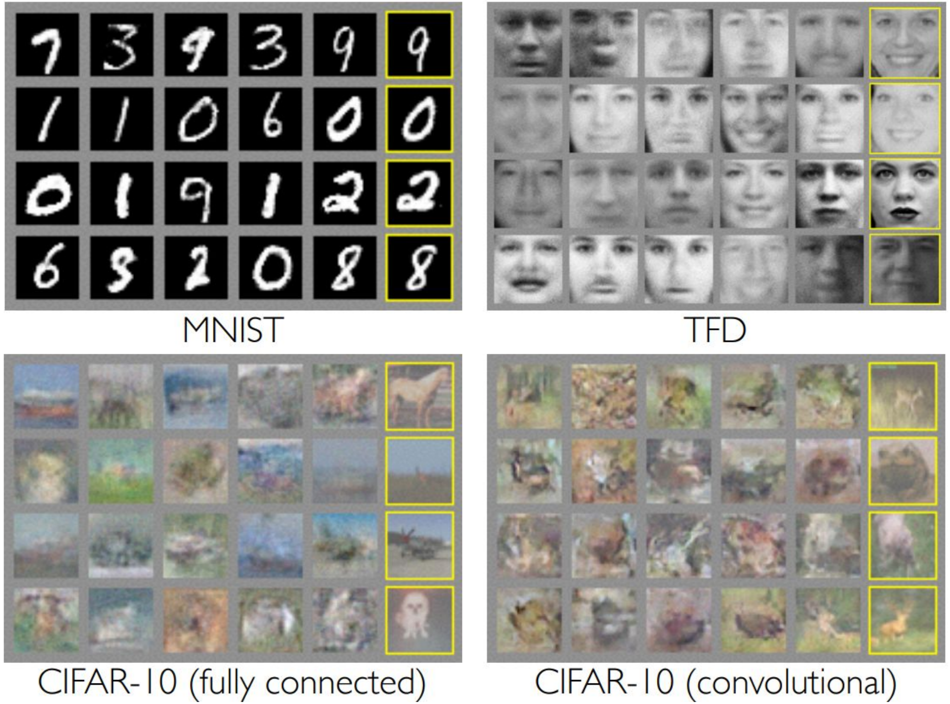
$$D(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}}(x) + p_{\text{model}}(x)}$$

Theoretical properties

- Given infinite data, infinite model capacity, direct updating of generator's distribution
  - Unique global optimum
  - Optimum corresponds to data distribution ($p_{data} = p_g$)
  - Convergence to optimum guaranteed



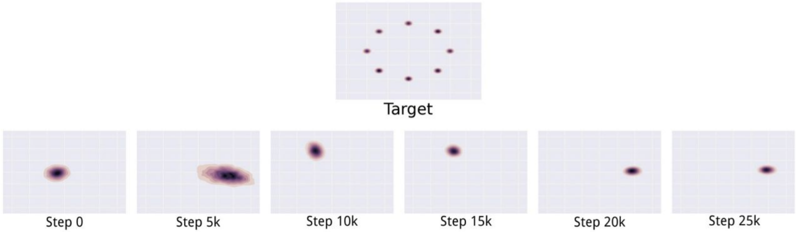
Experiment Results



Advantages

- Sampling (or generation) is straightforward.
- Training doesn't involve Maximum Likelihood estimation.
- Robust to overfitting since Generator never sees the training data.
- Empirically, GANs are good at capturing the modes of the distribution.

Disadvantages



- Probability distribution is implicit
    - Not straightforward to compute P(X)
- Training is Hard
    - Non-Convergence
        - Optimization algorithms often approach a saddle point or local minimum rather than a global minimum
        - Game solving algorithms may not approach an equilibrium at all
    - Mode-Collapse
        - Generator learns to map several different input z values to the same output point
        - Generator makes multiple images that contain the same color or texture themes, or multiple images containing different views of the same dog

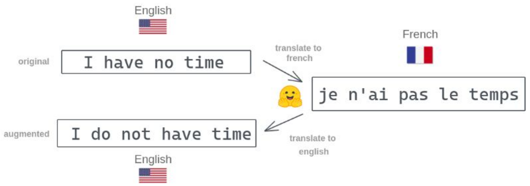
Metz, L., Poole, B., Pfau, D., and Sohl-Dickstein, J. (2016). Unrolled generative adversarial networks. arXiv preprint arXiv:1611.02163 .

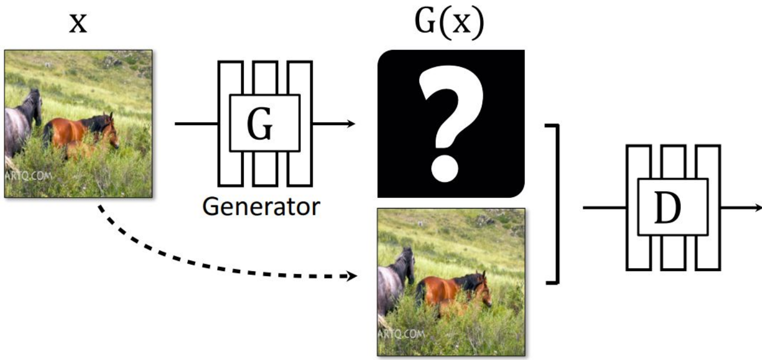
Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

Jun-Yan Zhu* Taesung Park* Phillip Isola Alexei A. Efros

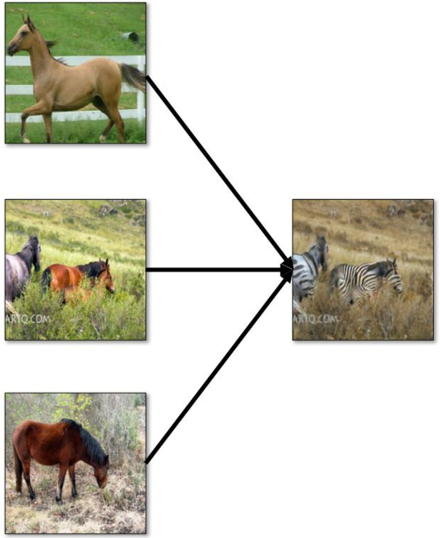
https://junyanz.github.io/CycleGAN/

Motivation

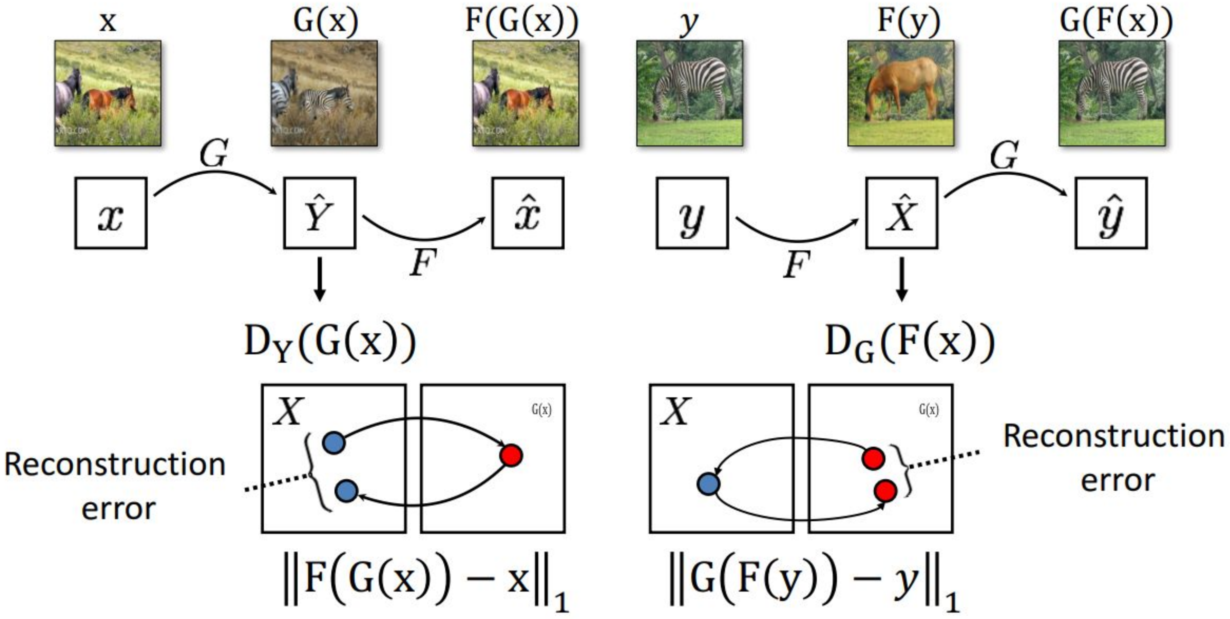




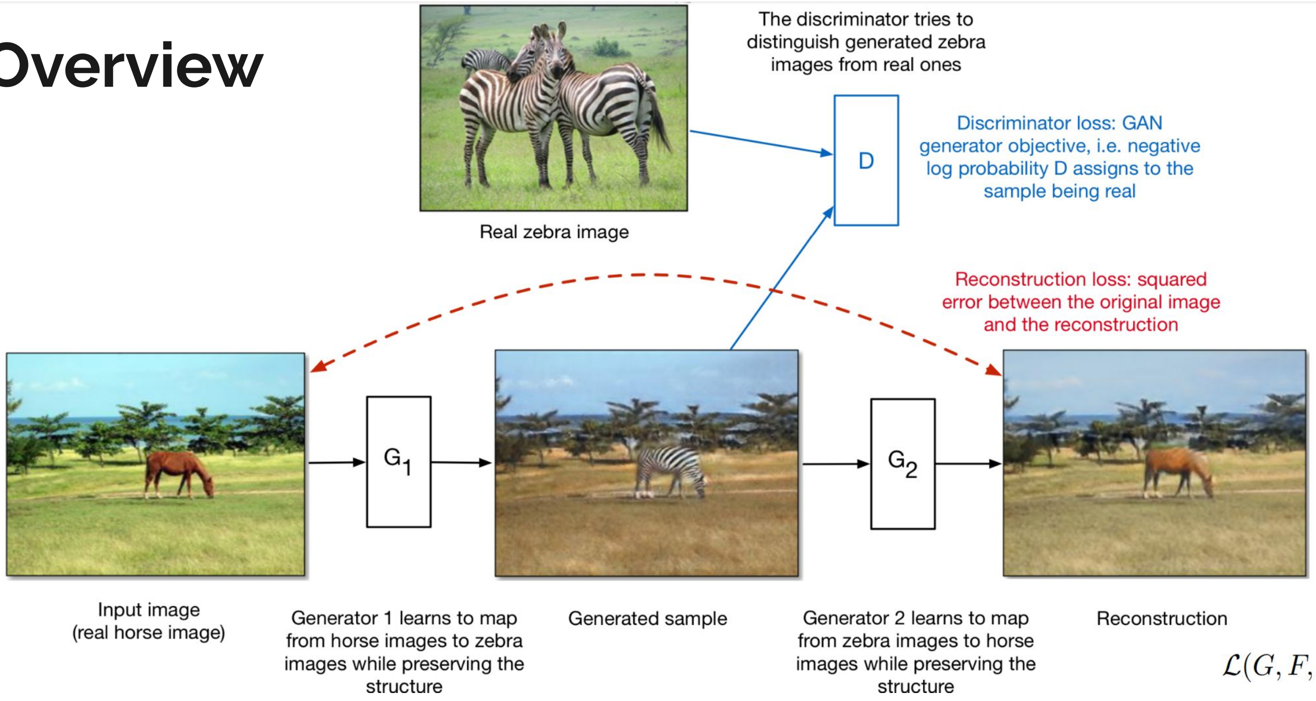
- Paired training data are not available for all tasks
- GAN suffer from mode collapse
- Introduce a two-step transformation of source domain image
  - Inspired by back-translation, e.g. a sentence from English to French, and then translate it back from French to English
  - Cycle-consistent
    - Mapping from style 1 to style 2 and back again should give you almost the original image



Cycle Consistency Loss



Overview



$$
\begin{aligned}
\mathcal{L}(G, F, D_X, D_Y) = & \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\
& + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\
& + \lambda \mathcal{L}_{\text{cyc}}(G, F),
\end{aligned}
$$

Experiment Results

Loss	Map → Photo % Turkers labeled *real*	Photo → Map % Turkers labeled *real*
CoGAN [30]	0.6% ± 0.5%	0.9% ± 0.5%
BiGAN/ALI [8, 6]	2.1% ± 1.0%	1.9% ± 0.9%
SimGAN [45]	0.7% ± 0.5%	2.6% ± 1.1%
Feature loss + GAN	1.2% ± 0.6%	0.3% ± 0.2%
CycleGAN (ours)	**26.8% ± 2.8%**	**23.2% ± 3.4%**

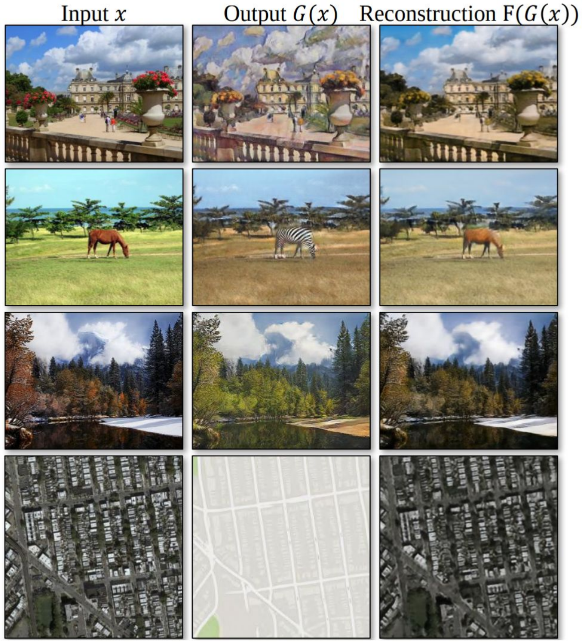
AMT 'real vs fake' test on maps ↔ aerial

Loss	Per-pixel acc.	Per-class acc.	Class IOU
CoGAN [30]	0.40	0.10	0.06
BiGAN/ALI [8, 6]	0.19	0.06	0.02
SimGAN [45]	0.20	0.10	0.04
Feature loss + GAN	0.06	0.04	0.01
CycleGAN (ours)	**0.52**	**0.17**	**0.11**

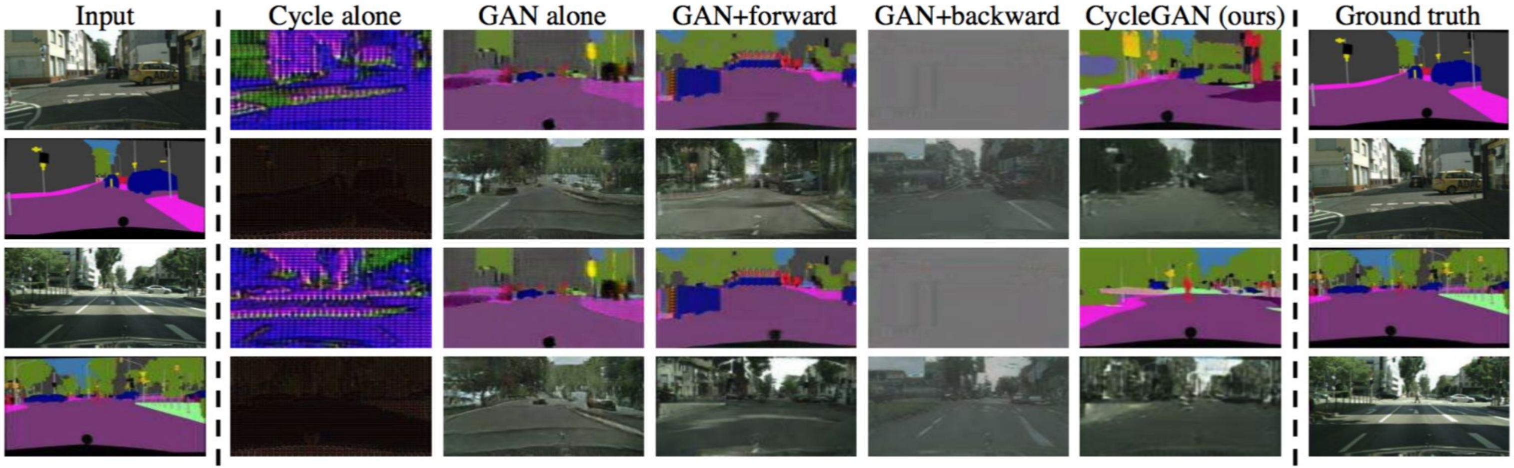
FCN scores on cityscapes labels→ photos

Loss	Per-pixel acc.	Per-class acc.	Class IOU
CoGAN [30]	0.45	0.11	0.08
BiGAN/ALI [8, 6]	0.41	0.13	0.07
SimGAN [45]	0.47	0.11	0.07
Feature loss + GAN	0.50	0.10	0.06
CycleGAN (ours)	**0.58**	**0.22**	**0.16**

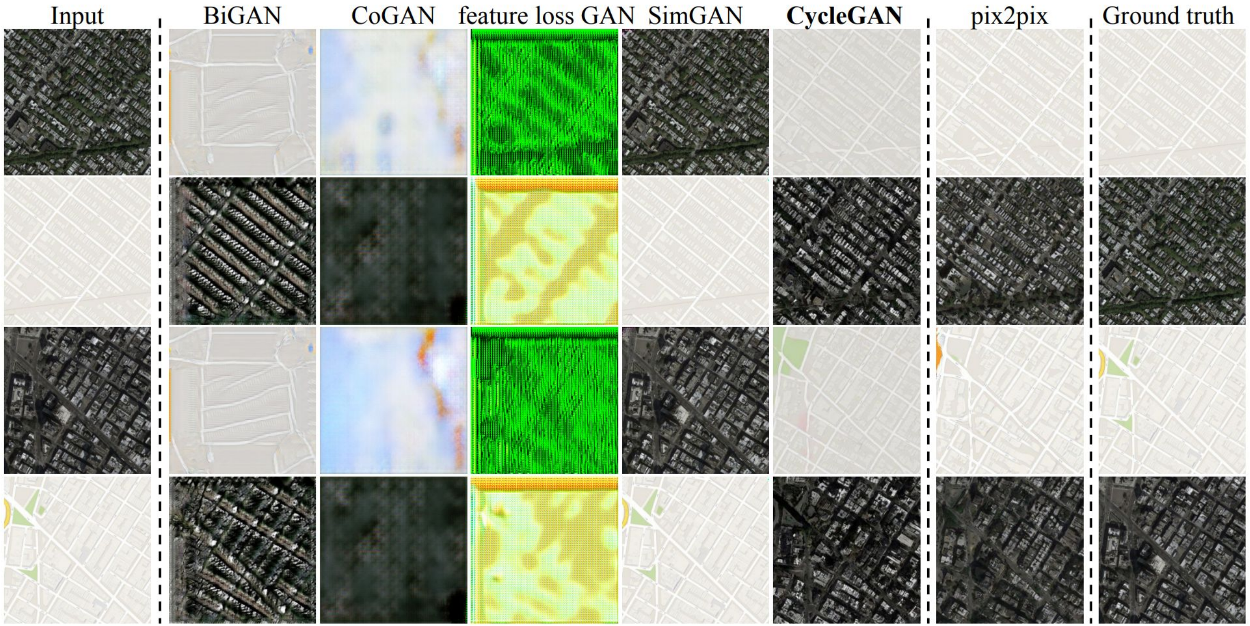
Classification performance of photo→labels



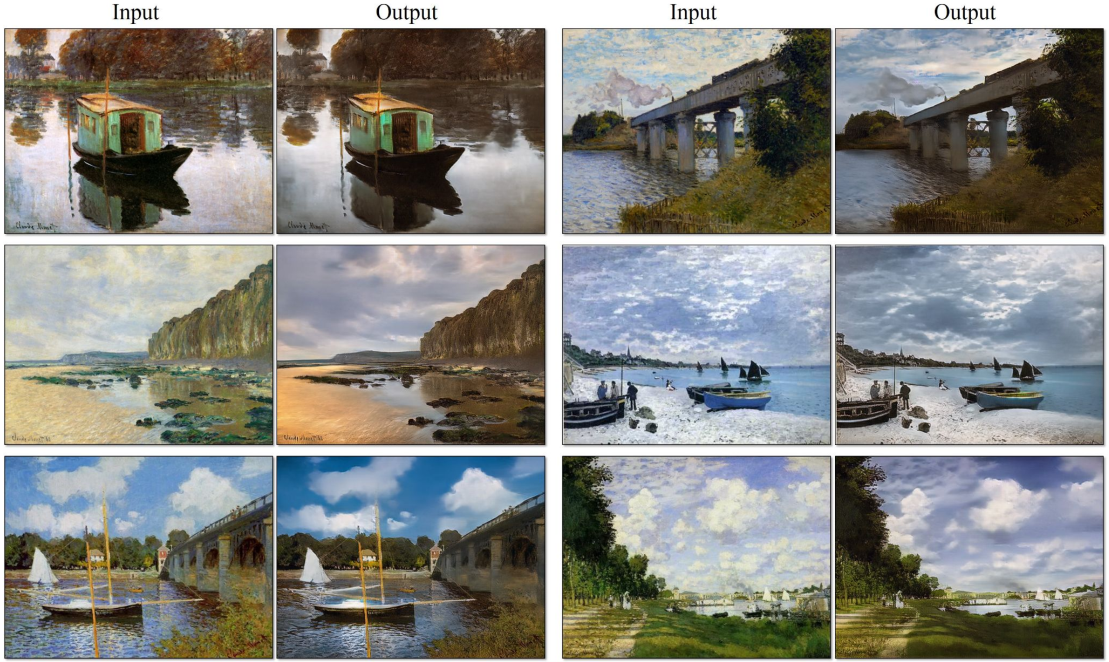
Cityscapes


Input
Cycle alone
GAN alone
GAN+forward
GAN+backward
CycleGAN (ours)
Ground truth

Google Maps

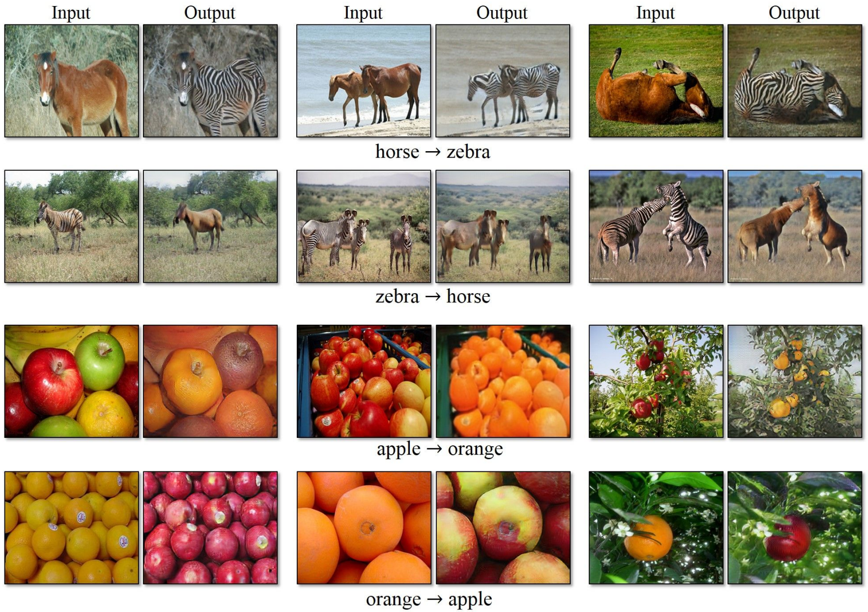

Input
BiGAN
CoGAN
feature loss GAN
SimGAN
CycleGAN
pix2pix
Ground truth

Monet Paintings → Photos

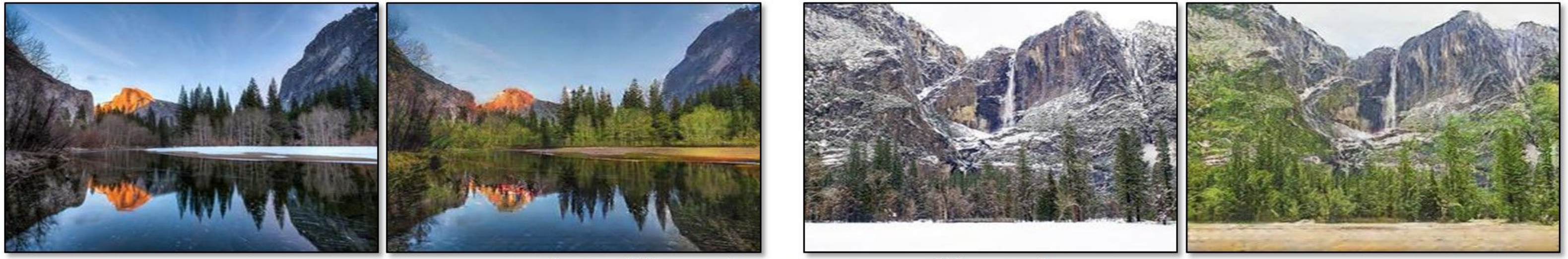


Collection Style Transfer

Object Transfiguration



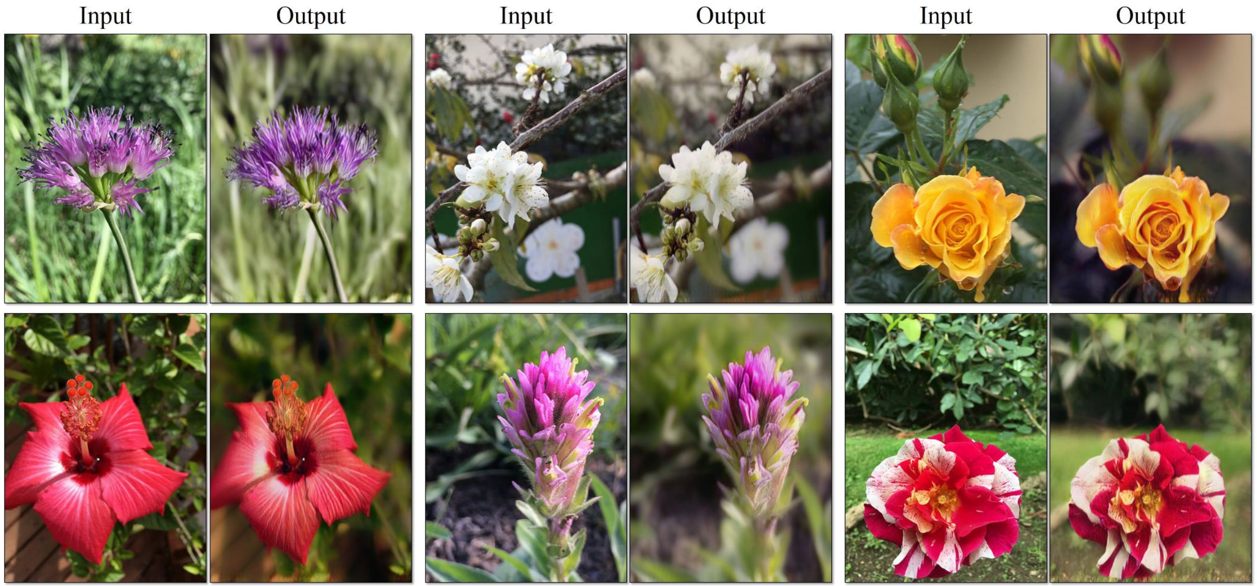
Season transfer



winter Yosemite → summer Yosemite

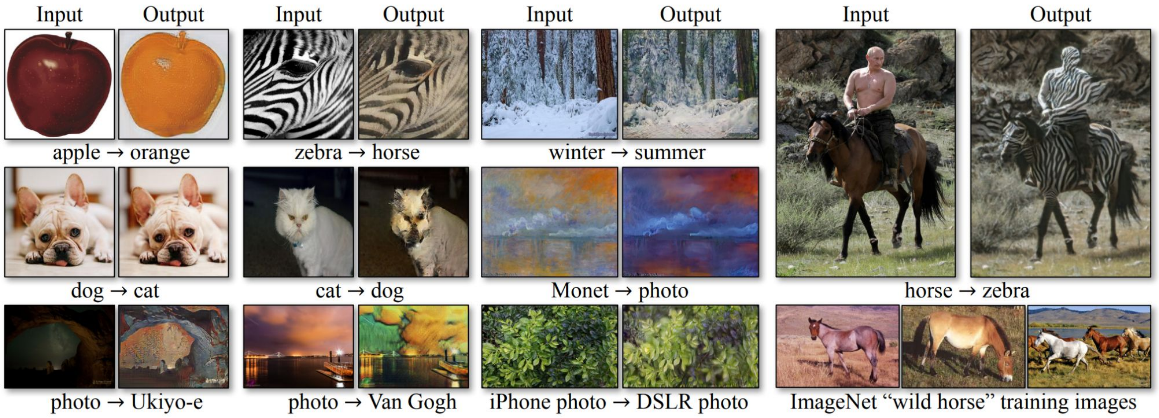
summer Yosemite → winter Yosemite

Photo Enhancement



Limitations

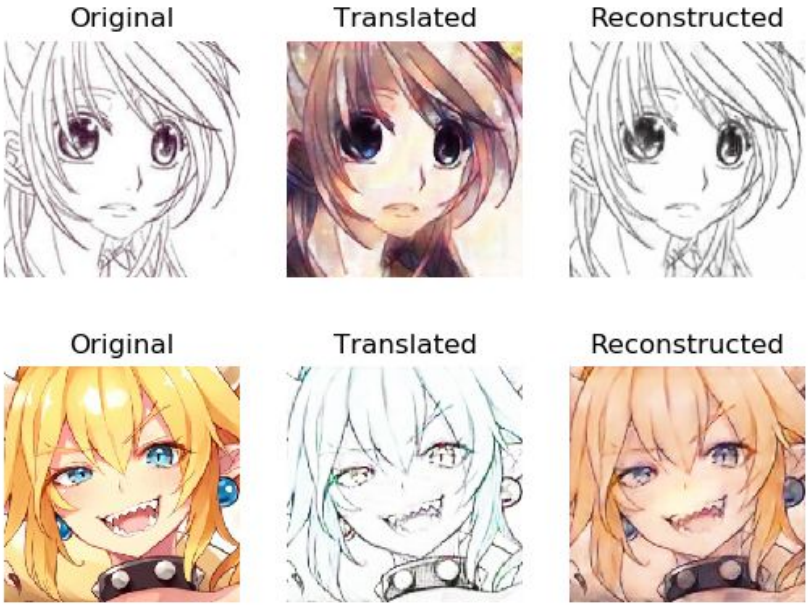
- Works well for translation tasks involving color and texture changes
- Failed for tasks that require substantial geometric changes to the image, such as cat-to-dog translations because of the generator architecture which is trained to perform appearance changes in the image



Potential Improvement

- CycleGAN lacks the straightforward description of the target domain
  - Adding additional regularization term to enforce similar image content in the source space to also be similar in the target space (Harmonic GAN)
  - Translating both an image and the corresponding set of instance attributes while maintaining the permutation invariance property of the instances (InstaGAN)
  - Disentangling structured information (Cross-domain disentanglement networks)
  - Introducing a semantic content loss to cope with substantial style variation and an edge-promoting adversarial loss for preserving clear edges (CartoonGAN)

Color filling

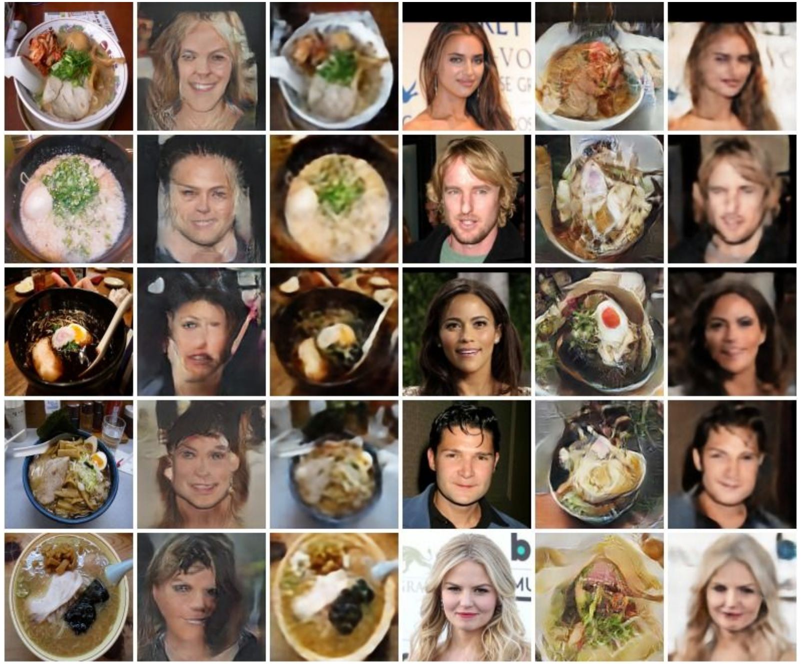


https://github.com/RikoLi/cyclegan-line2color

Converting Monet into Thomas Kinkade

https://web.eecs.umich.edu/~fouhey//fun/monet/index.html

Face to Ramen



Thank you!