

Label-base/Feature-specification General Adversarial Networks(LGAN) for Plant Image Generation



Group 39

Shiwei Zhou
s1zhou@eng.ucsd.edu

Bosi Cheng
bocheng@eng.ucsd.edu

Sixuan Feng
sfeng@eng.ucsd.edu

Yifan Hou
yihou@eng.ucsd.edu

Abstraction

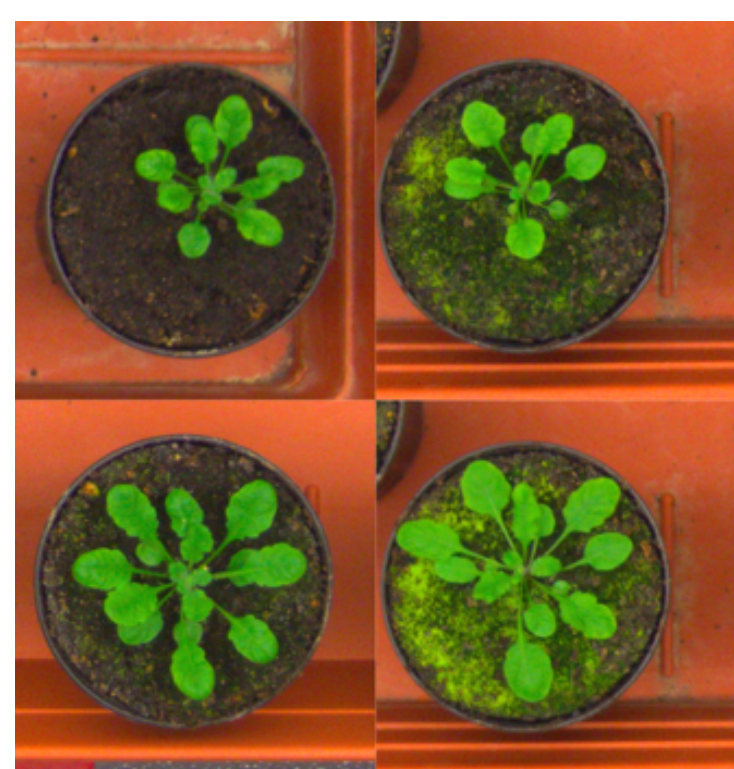
Generative Adversarial Networks (GAN) are a relatively new concept in Machine Learning, introduced for the first time in 2014. The goal of GAN is to synthesize artificial samples, such as images, that are indistinguishable from authentic images. However, most GAN models only focus on improving realism of synthesized samples, which can result in synthesized samples with uncontrolled features and huge unpredictability. To further improve our manipulation of the output, we decide to structurally modify the existing GAN model to enable it generate output with certain feature labels. By doing so, GAN models can be used to generate more meaningful output other than random output.

Prediction

Our model intends to extend GAN's basic functionality by enabling it to generate image with specific features or even the features that does not exist in the original data set. In our case, we expect our LGAN model generating plant image with specified growing stage. The output should be plant image with feature converging to our requirement as the iteration increases.

Dataset

The dataset was collected from A1 data section from LSC(leaf segmentation challenge) CVPPP 2015 dataset. It has high consistency of image structure and great variation of image details. The dataset is unlabeled since it is used for unsupervised learning. Here shows some examples of the dataset.



Feature

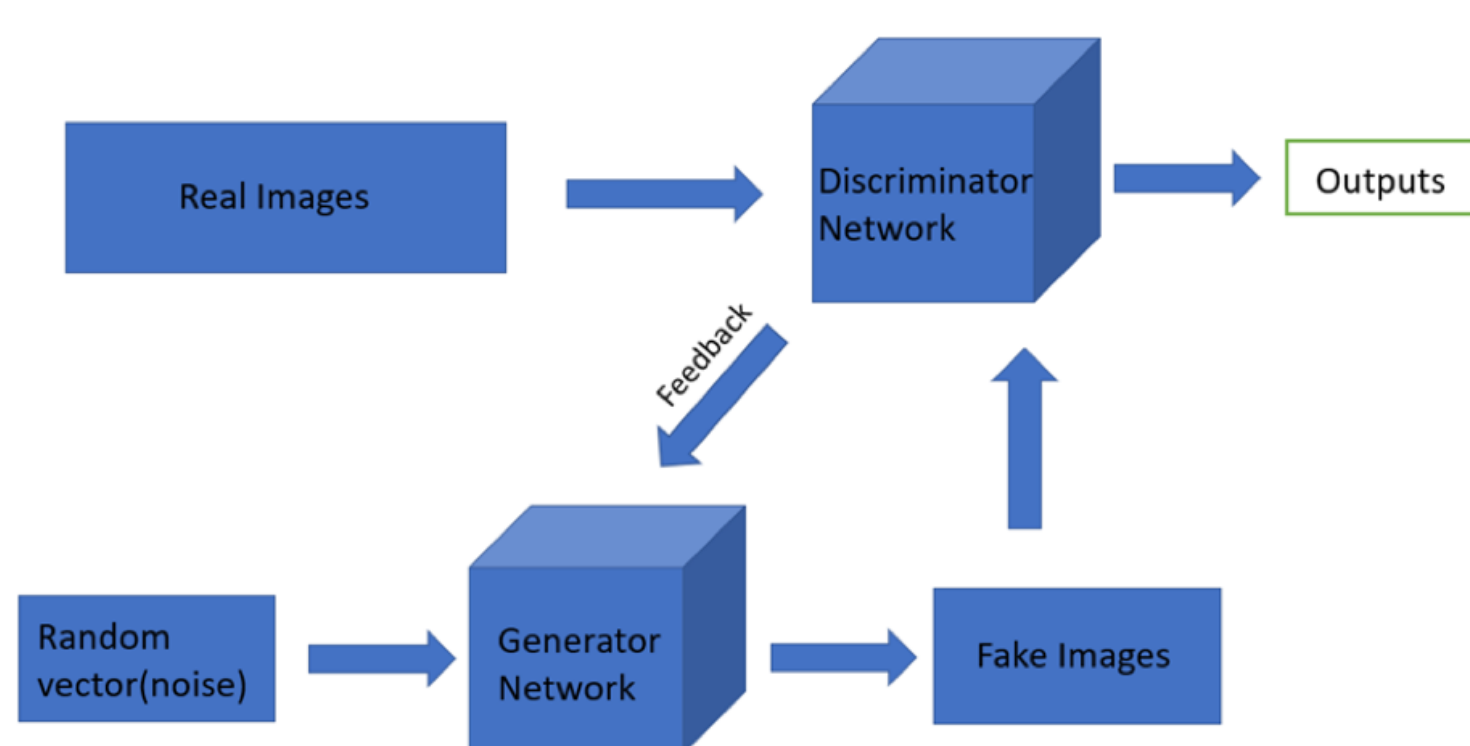
The model is implemented with no conceptual features to reduce the workload. The input features are image pixels. In this case, the features will be 452*480 pixels. No other features are derived due to existing high-complexity of this model. This model can be further improved with task specific features. In this image set, conceptual features can be leaf area, leaf number and etc.

Model

A DCGAN model is used as basic model then an image filter (labeling) block is added to the model to create LGAN model. An LGAN is not limited to be implemented on DCGAN but we use DCGAN as a demonstration here. A comparison will be taken between DCGAN and LGAN in our experiment to compare their performance.

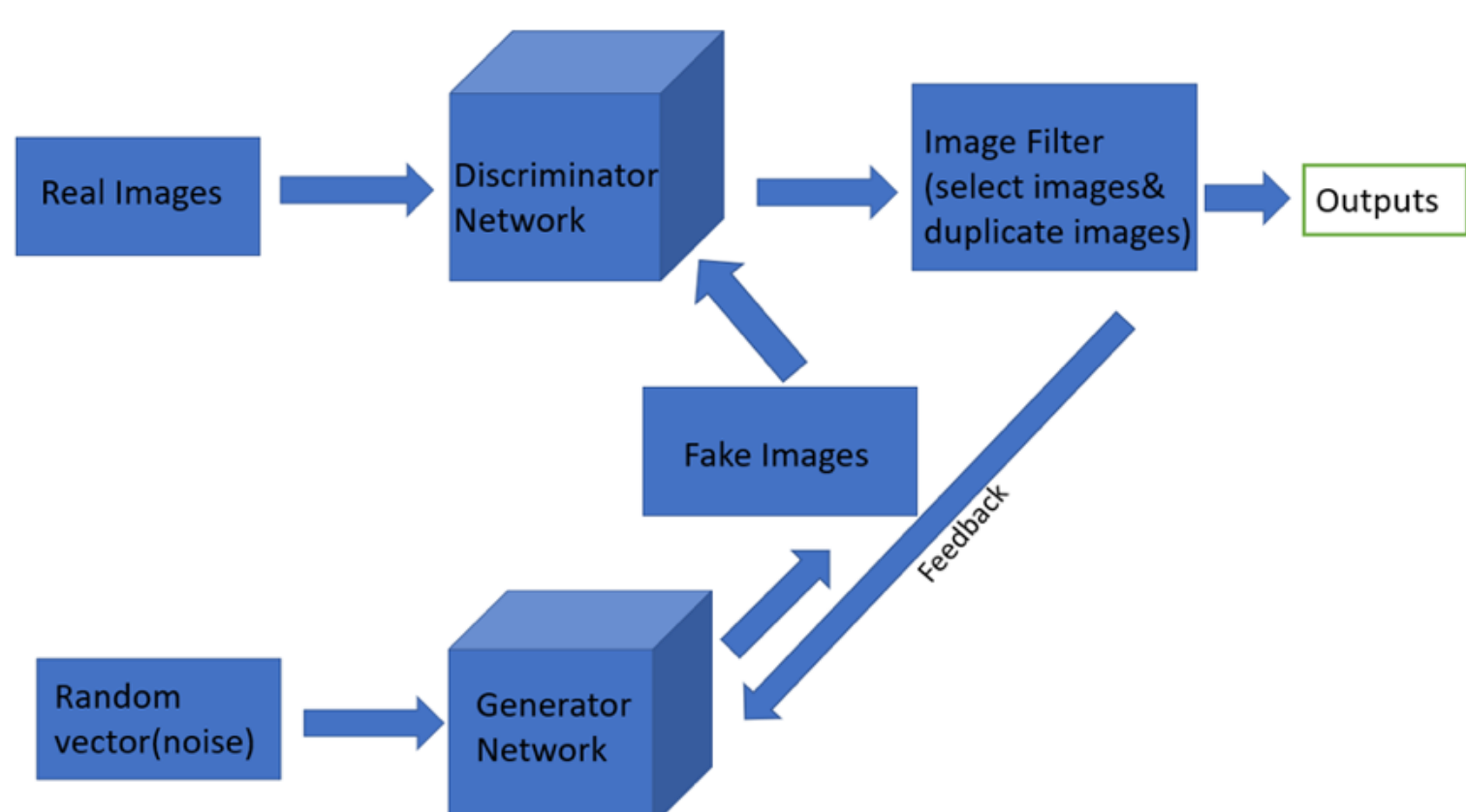
1. DCGAN(Deep Convolutional Generative Adversarial Networks)

Use generator and discriminator two networks to compete with each other, generating fake images assemble realism as much as possible.



2. LGAN(labeled General Adversarial Networks)

Add an image filter(label block) after the discriminator. The Image filter selects specific images that satisfy the tolerance range of our feature requirement and randomly duplicates the images to maintain sample population. Now, each generation of output images has to go through both discriminator and label block, which makes the results converge to both realism and feature requirements.



Model

Mathematically it is a min-max problem which targets to minimize loss functions from both discriminator and generator to reach optimal for both networks.

Train jointly in **minimax game**

Discriminator outputs likelihood in (0,1) of real image

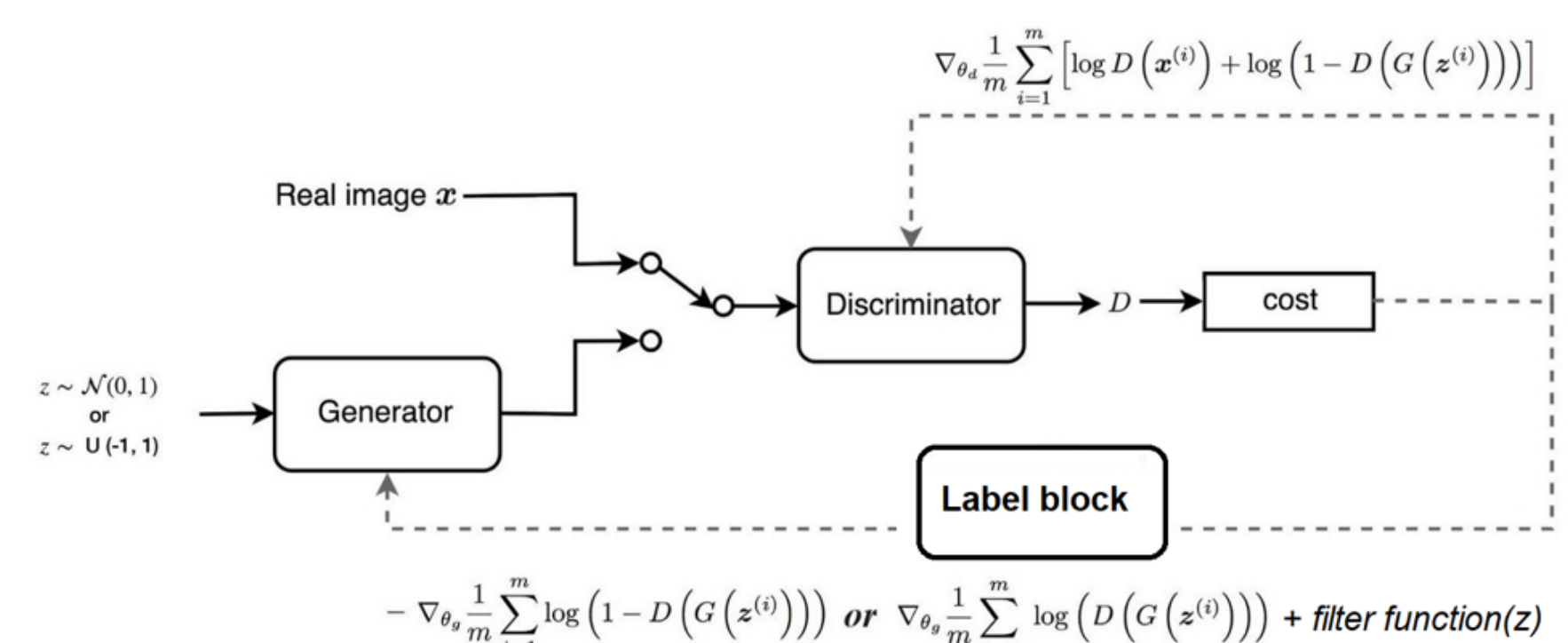
Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Discriminator output for real data x
Discriminator output for generated fake data G(z)

- Discriminator (θ_d) wants to **maximize objective** such that $D(x)$ is close to 1 (real) and $D(G(z))$ is close to 0 (fake)
- Generator (θ_g) wants to **minimize objective** such that $D(G(z))$ is close to 1 (discriminator is fooled into thinking generated $G(z)$ is real)

The image filter(label block) should be implemented according to the feature specification. It is not limited to filter function but can be any processing function potentially, but here it is implemented as simple filter for simplicity of demonstration. The label block will not running time complexity.



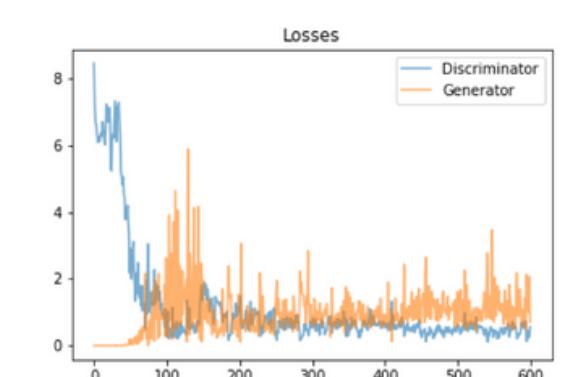
Result

In this example, 160 real images are used as initial input. The label we use will be green area, which is a simplified representation of growing stage of the plantation and can be useful to generate fake plant image with certain growing stage we desire. The label, green factor, is measured as green pixel portion in the whole image, which ranges from 0.0 to 1.0. The comparison of the results is taken after a certain level of loss function convergence between generator and discriminator.

As shown below, DCGAN will generate plant image of random growing stage and unpredictable features, but LGAN(Label-based General Adversarial Networks) will generate plant image close to the label we provide as the generation goes on. Some output images even have green area that exceeds the maximum of original input or smaller than the minimum of the original input.

D C G A N	Generation 230~270	Generation 290~300
L G A N	Label: Green factor = 0.2±0.1 Generation 230~270	Label: Green factor = 0.7±0.1 Generation 230~270
	Generation 290~300	Generation 290~300

There are 300 generations/iterations and the loss function of generator and discriminator converges after 200 generations, hence we assume that we have reached the limitation of the optimal number of iteration. The precision of the output graph is limited by compressed input image and reduced features due to the heavy workload of the model, but can be further improved.



Discussion & Future works

Our model offers a linear modification to all existing GAN models, which is easy to implement and efficiently reuses existing codes and models. The performance from example reaches our expectation and validates the assumption we proposed.

For the future work, the model can be improved by: more diversified and sophisticated label blocks to improve performance; more tests on other GAN models to testify versatility; use of other data sets to improve generality of the model.

Reference

1. Ren, Mengye, and Richard S. Zemel. "End-to-end instance segmentation with recurrent attention." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.
2. Romera-Paredes, Bernardino, and Philip Hilaire Sean Torr. "Recurrent instance segmentation." *European conference on computer vision*. Springer, Cham, 2016.
3. Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." *arXiv preprint arXiv:1511.06434* (2015).
4. Goodfellow, Ian, et al. "Generative adversarial nets." *Advances in neural information processing systems*. 2014.
5. Karras, Tero, Samuli Laine, and Timo Aila. "A style-based generator architecture for generative adversarial networks." *arXiv preprint arXiv:1812.04948* (2018).
6. Mirza, Mehdi, and Simon Osindero. "Conditional generative adversarial nets." *arXiv preprint arXiv:1411.1784* (2014).