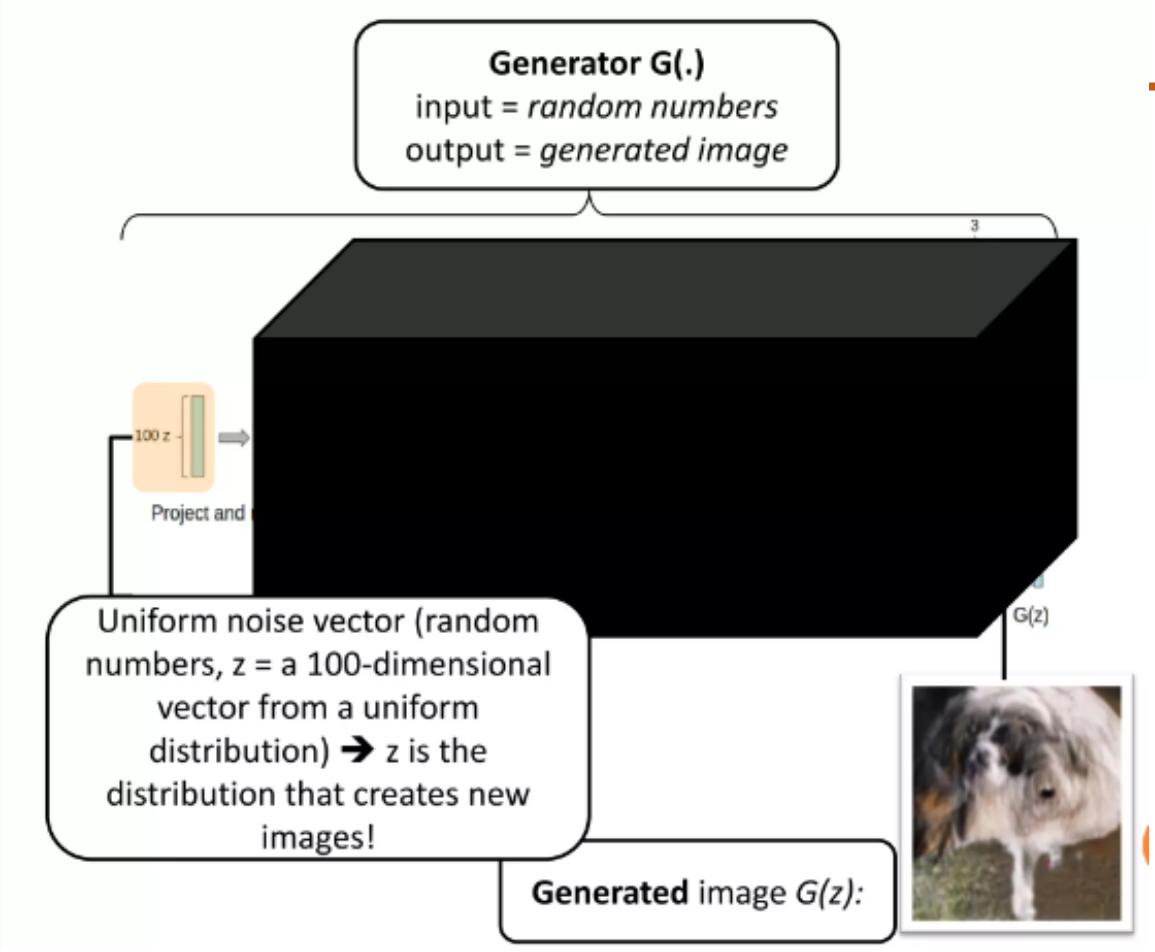
## DCGAN Components



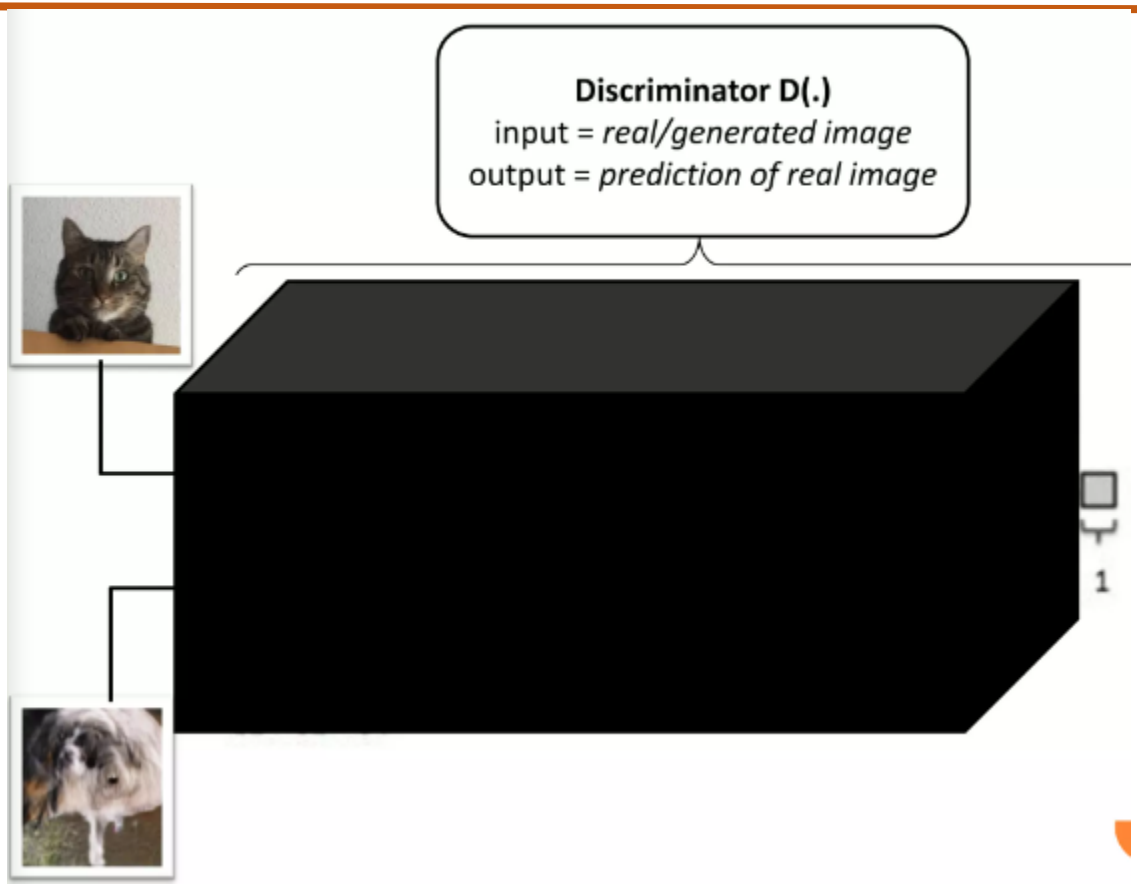
- The discriminator is made up of strided convolution layers, batch norm layers, and LeakyReLU activations.
- The input is a 3x64x64 input image and the output is a scalar probability that the input is from the real data distribution.
- The generator is comprised of convolutional-transpose layers, batch norm layers, and ReLU activations.
- The input is a latent vector, z, that is drawn from a standard normal distribution and the output is a 3x64x64 RGB image.
- The strided conv-transpose layers allow the latent vector to be transformed into a volume with the same shape as an image.

DCGAN

Can be thought of as two separate networks.

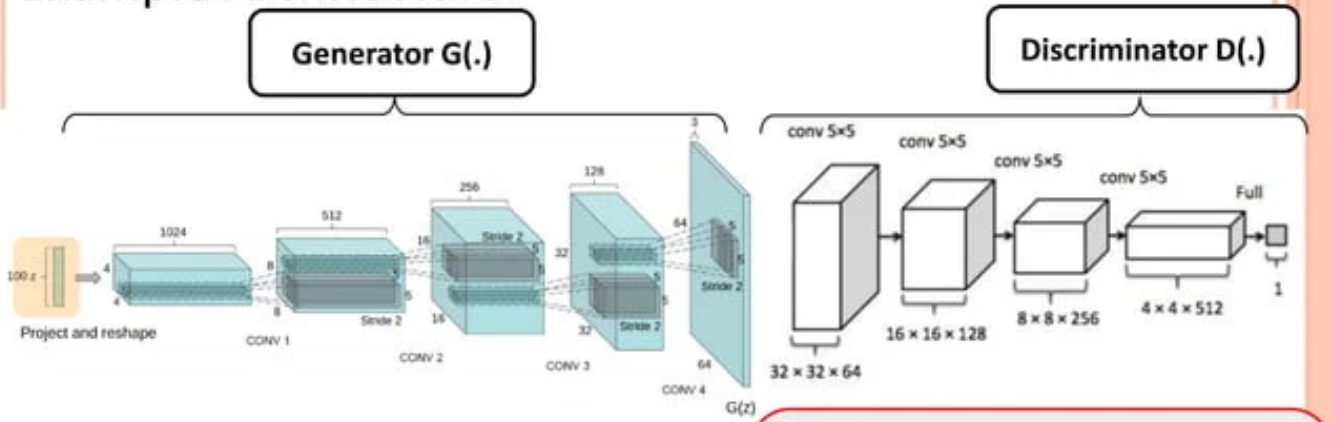


## DCGAN


Discriminator D(.)
input = real/generated image
output = prediction of real image
1

DCGAN

## Example Architecture:



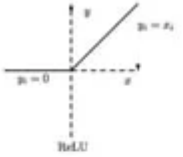

**Generator Goal:** Fool *D(G(z))*
i.e., generate an image *G(z)* such that *D(G(z))* is wrong, i.e., *D(G(z)) = 1*

**Discriminator Goal**: discriminate between **real** and **generated** images *i.e., D(x)=1,* where *x* is a **real** image *D(G(z))=0,* where *G(z)* is a **generated** image.

- **Conflicting** goals.
- Both goals are **unsupervised.**
- Optimal when D(.)=0.5 (i.e., cannot tell the difference between real and generated images) and G(z)=learns the training images distribution.

DCGAN

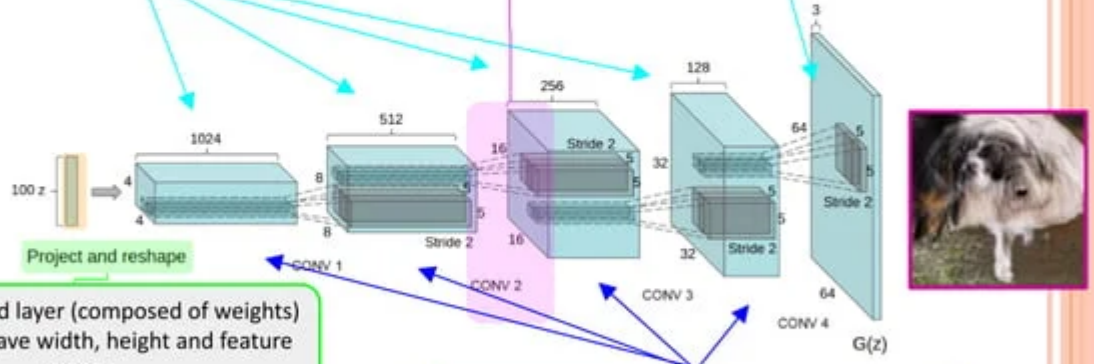
## DCGAN Generator:




Uses ReLU activation functions

**Fractionally-strided convolutions:** 8x8 input, 5x5 conv window = 16x16 output

Uses Tanh to scale generated image output between -1 and 1

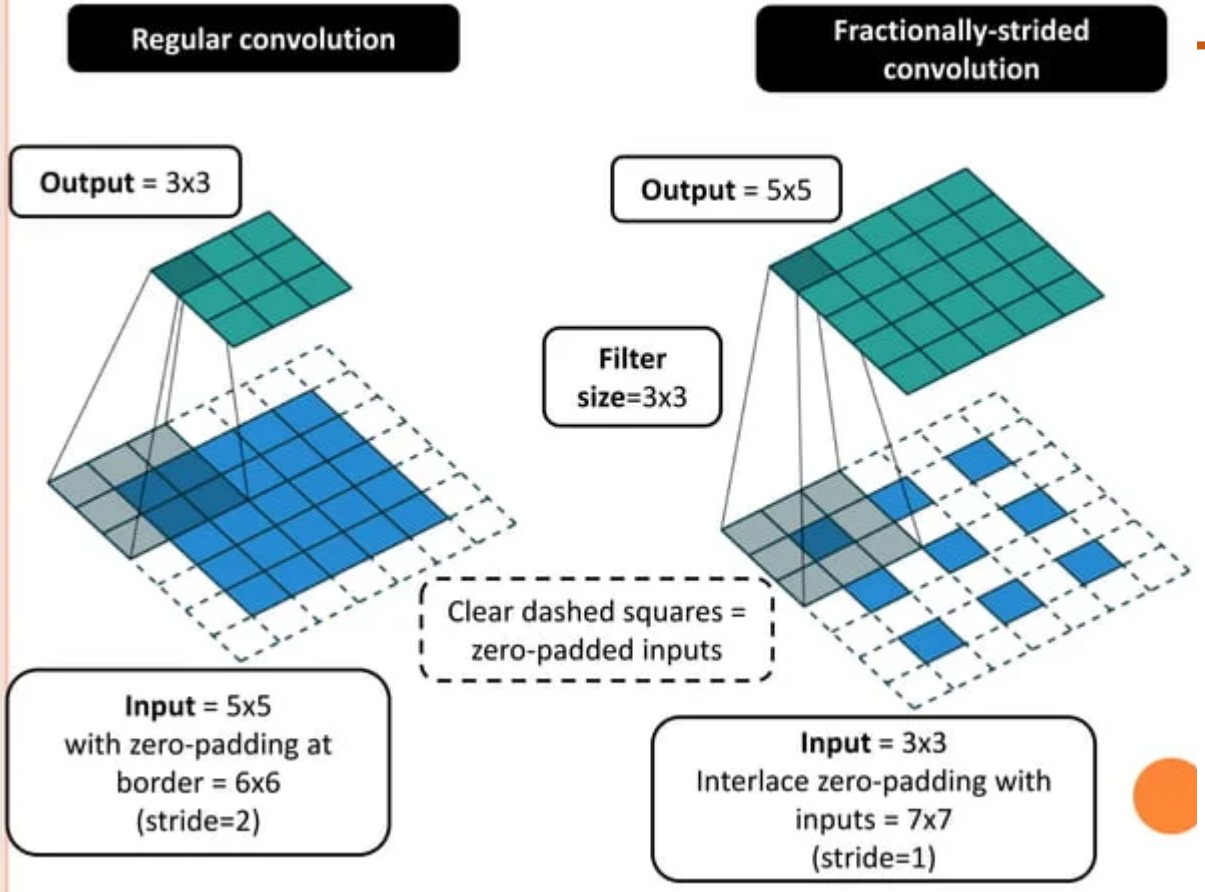



Fully-connected layer (composed of weights) reshaped to have width, height and feature maps

**No max pooling!** Increases spatial dimensionality through fractionally-strided convolutions

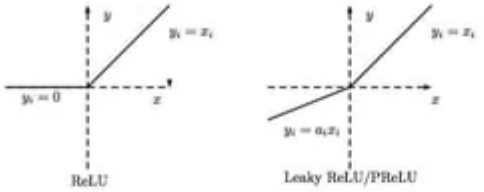
**Batch Normalization**: normalize responses to have zero mean and unit variance over the entire mini-batch, but not in last layer (to prevent sample oscilation and model instability)

DCGAN


Regular convolution
Fractionally-strided convolution
Output = 3x3
Output = 5x5
Filter size=3x3
Clear dashed squares = zero-padded inputs
Input = 5x5 with zero-padding at border = 6x6 (stride=2)
Input = 3x3 Interlace zero-padding with inputs = 7x7 (stride=1)

DCGAN

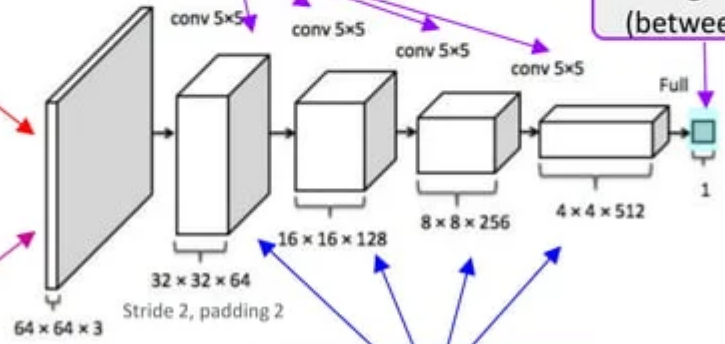
## DCGAN Discriminator:




**No max pooling!** Reduces spatial dimensionality through strided convolutions

Uses LeakyReLU activation functions






Sigmoid (between 0-1)

Batch Normalization

PES UNIVERSITY

# Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator.
- Use LeakyReLU activation in the discriminator.

## References


- https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html
- https://www.analyticsvidhya.com/blog/2021/07/deep-convolutional-generative-adversarial-network-dcgan-for-beginners/


# Thank You

**Dr. Shylaja S S**
Director of Cloud Computing & Big Data (CCBD), Centre for Data Sciences & Applied Machine Learning (CDSAML)
Department of Computer Science and Engineering
**shylaja.sharath@pes.edu**

**Ack:Divija L**
**Teaching Assistant**