DATA 442: Neural Networks & Deep Learning

Dan Runfola – danr@wm.edu

icss.wm.edu/data442/

Generative Models

Input

Output

Given some input of images, generate an output of samples drawn from the same distribution.

Types of Generative Models

NADE/MADE

PixelRNN Explicit Density Estimation

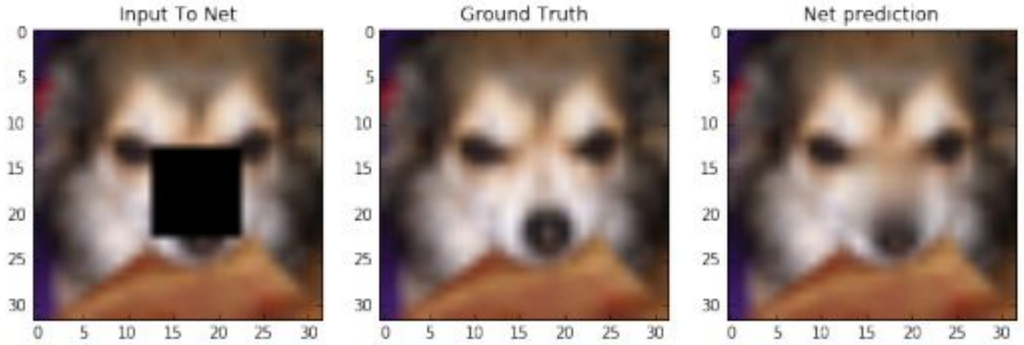
Variational Autoencoder Approximate Density Estimation

Boltzmann Machine

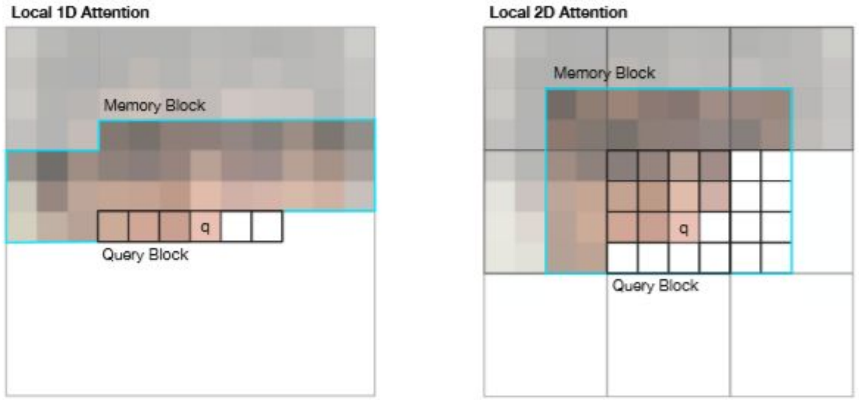
GAN Implicit Density Estimation

GSN

PixelRNN/PixelCNN

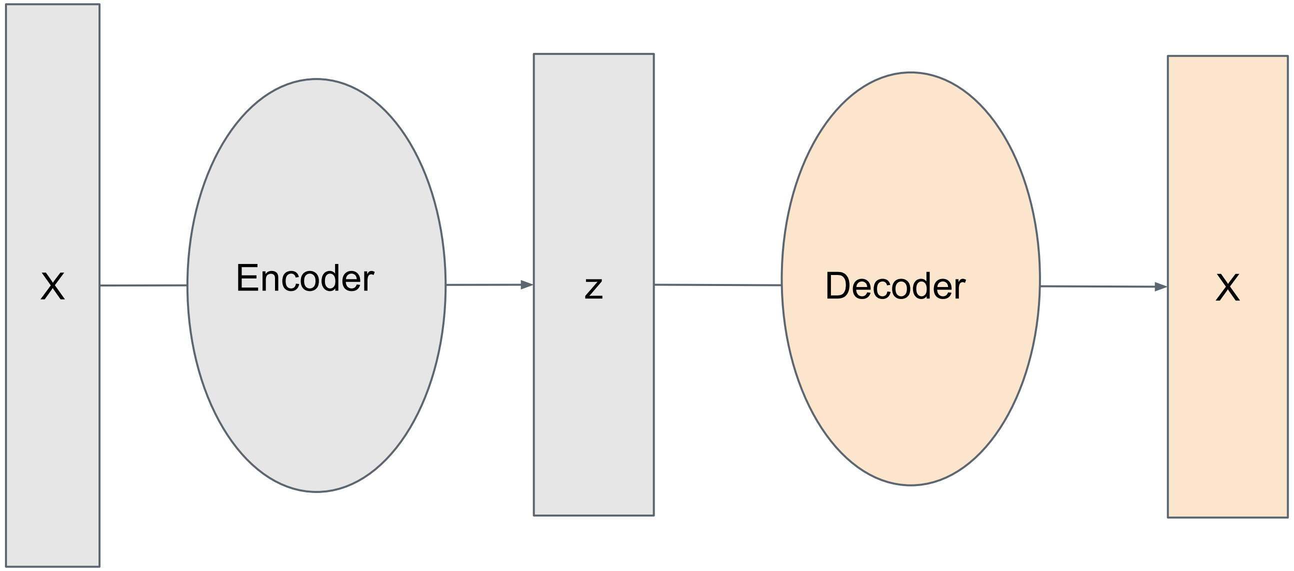


- Explicitly calculates likelihood
  - Can help in understanding model performance
- Relatively Slow
- Makes fairly believable images
- Area of extensive inquiry - PixelCNN+; PixelCNN++, PixelCNN 2.0, and many more.
  - Concept of ‘Attention’

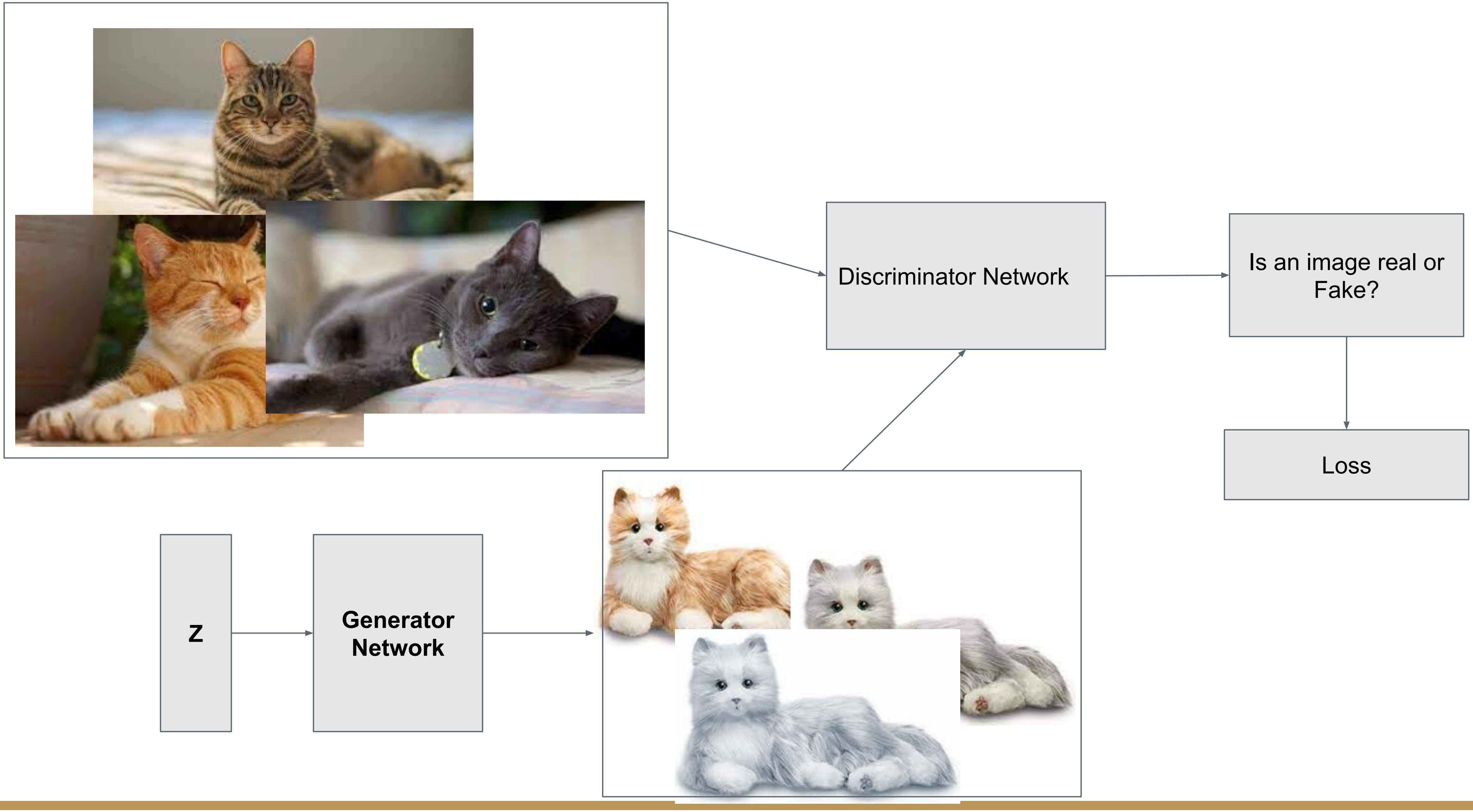


http://cs231n.stanford.edu/reports/2016/pdfs/211_Report.pdf

Variational Autoencoders (VAE)



$$p(image) = \int p(z)p(x|z)dz$$



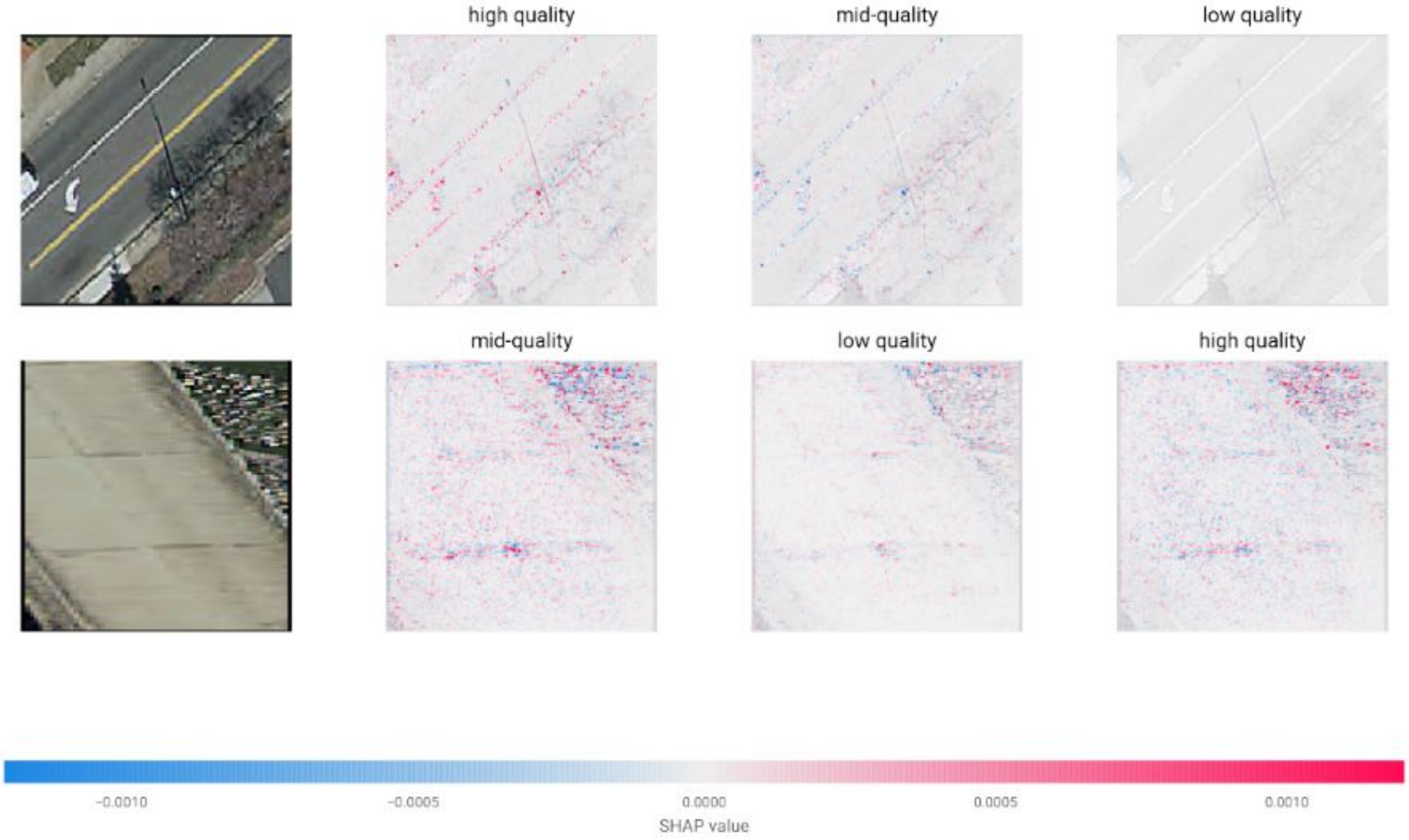
Discriminator Network

Is an image real or Fake?

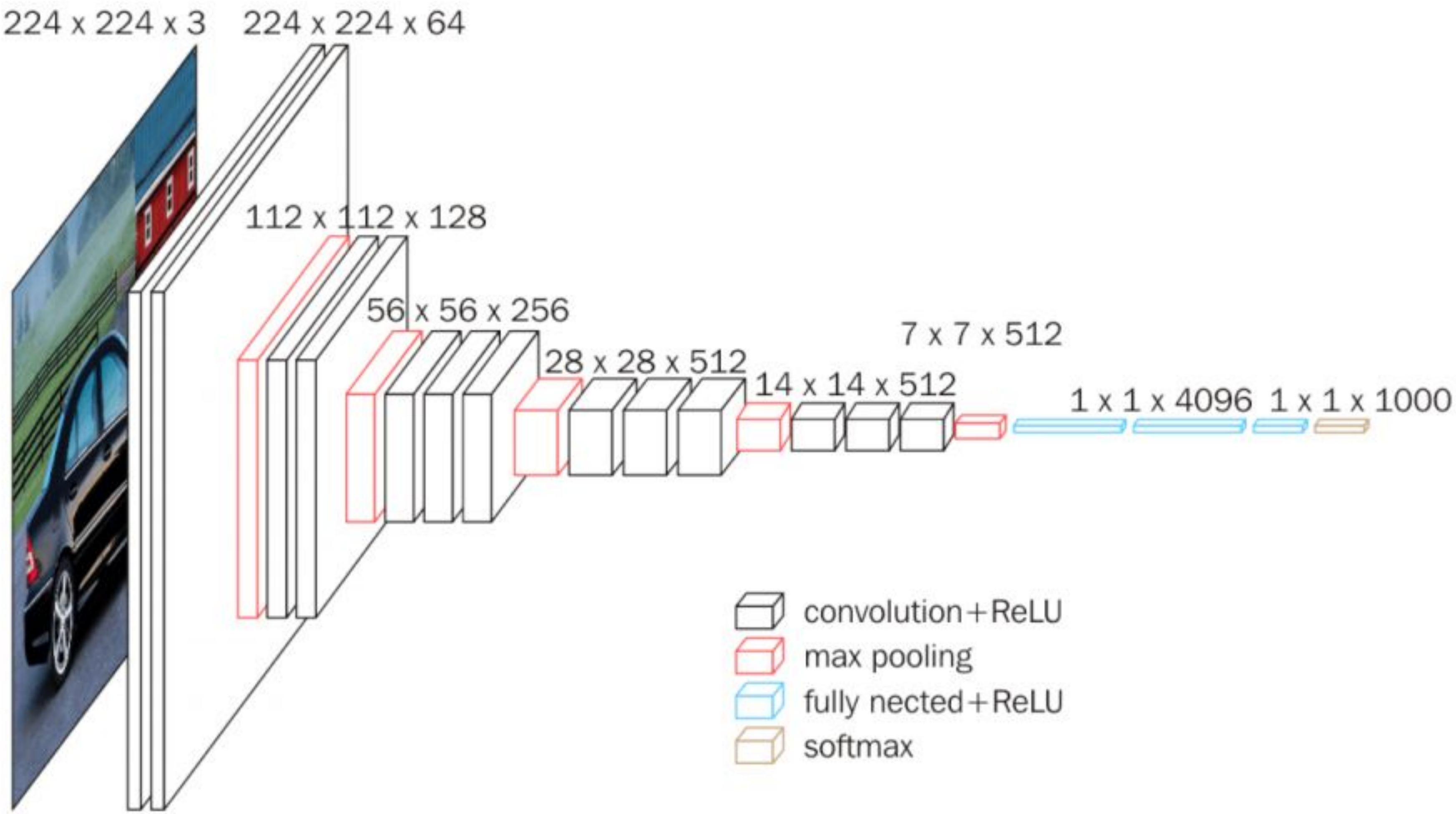
Loss

Z

Generator Network

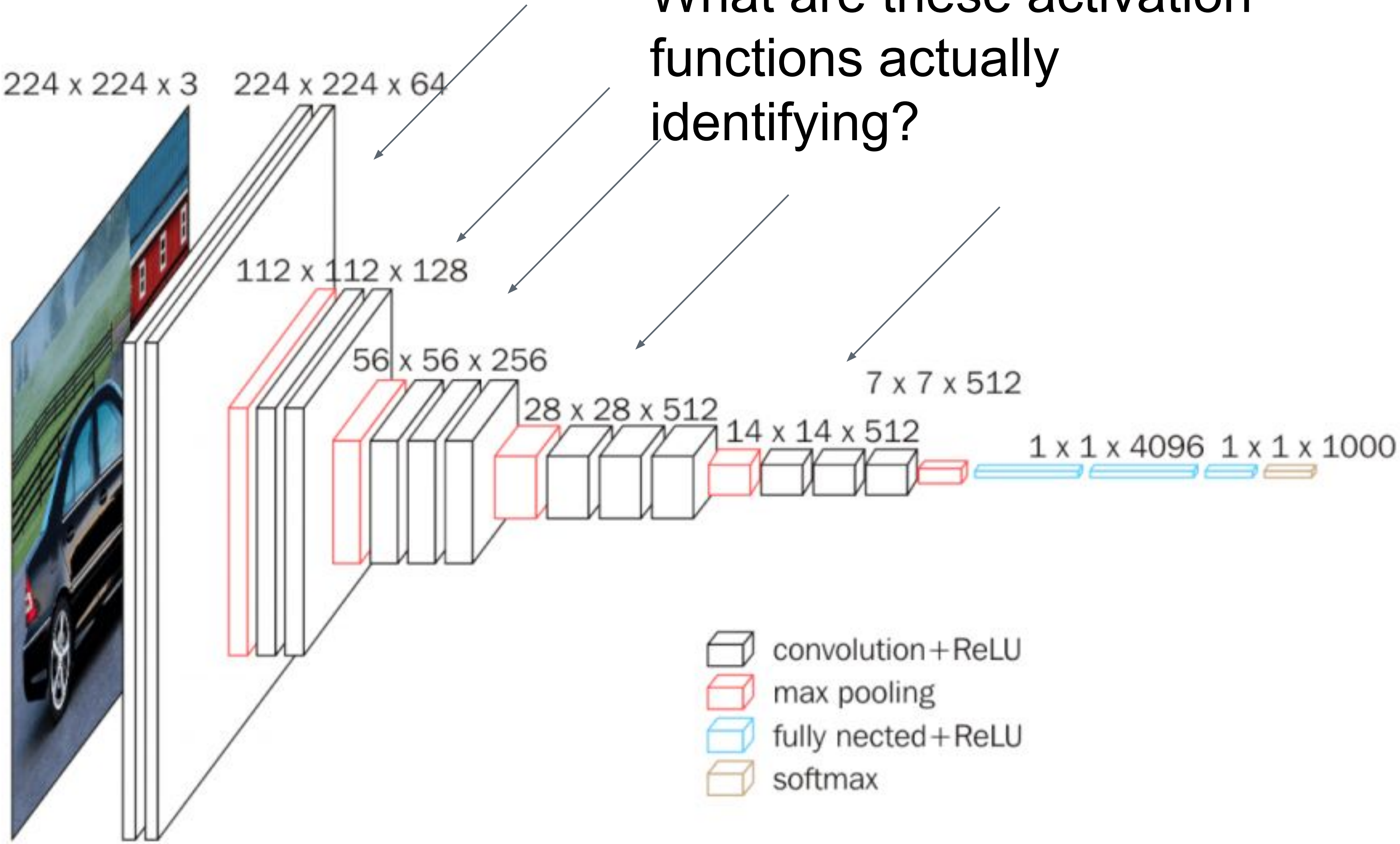


Replicated with Permission. Brewer et al. 2021.



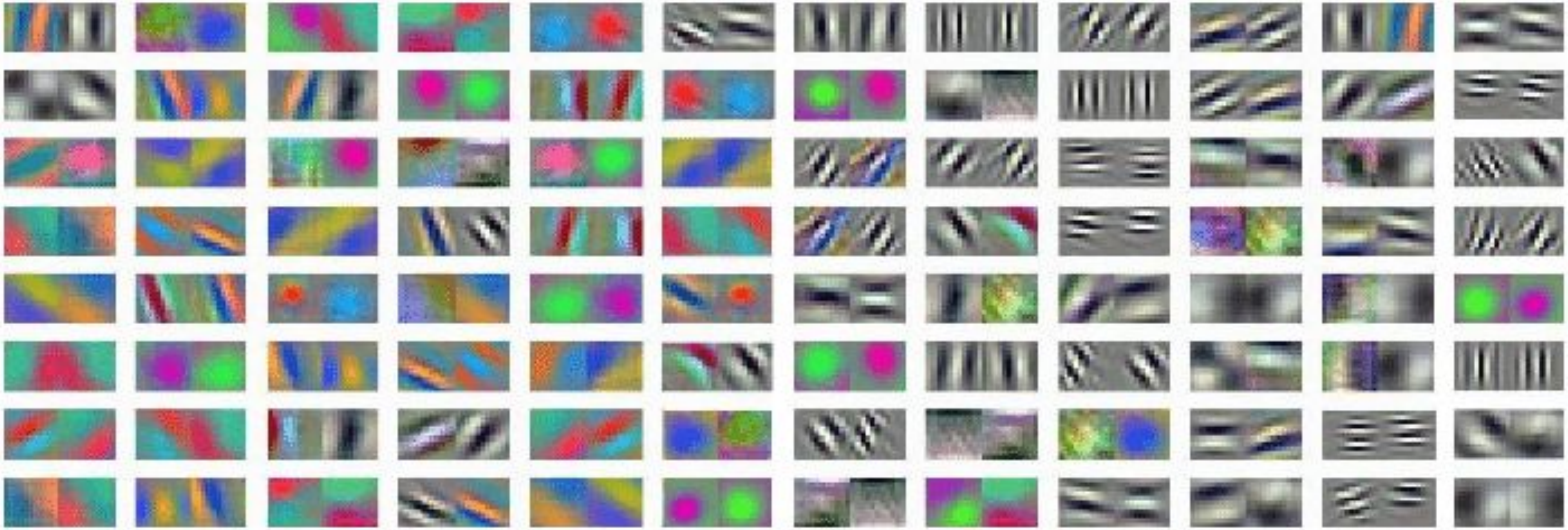
Neurohive.io

What are these activation functions actually identifying?

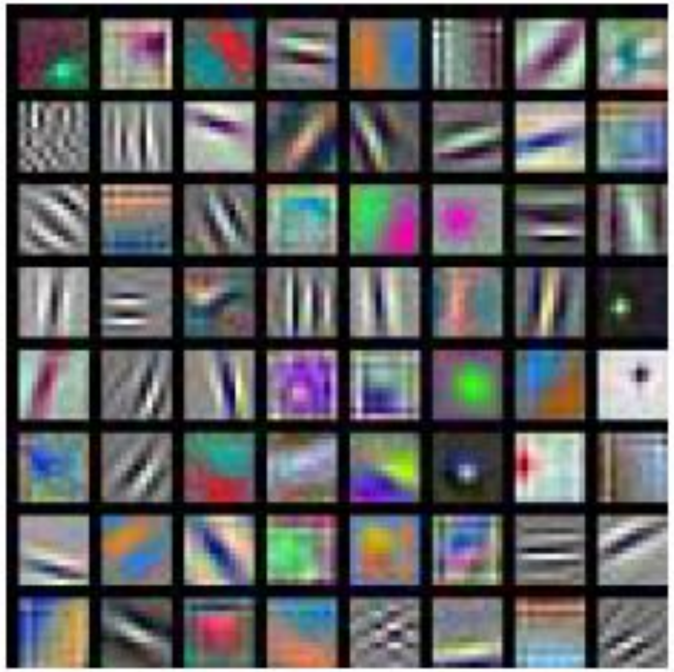


Neurohive.io

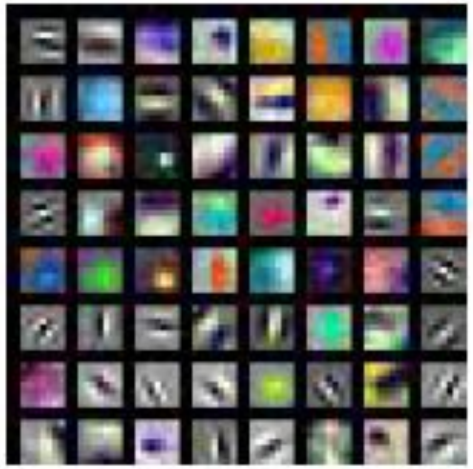
Visualizing Filters



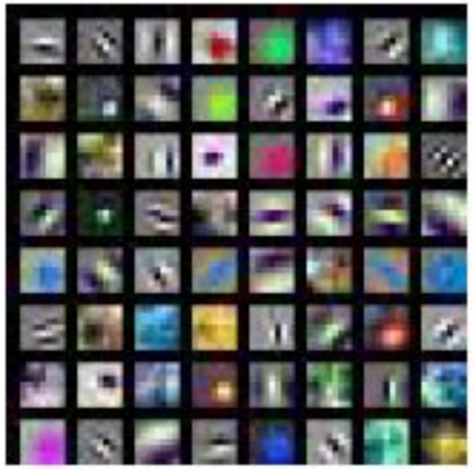
https://www.researchgate.net/publication/324005705_What_Do_We_Understand_About_Convolutional_Networks



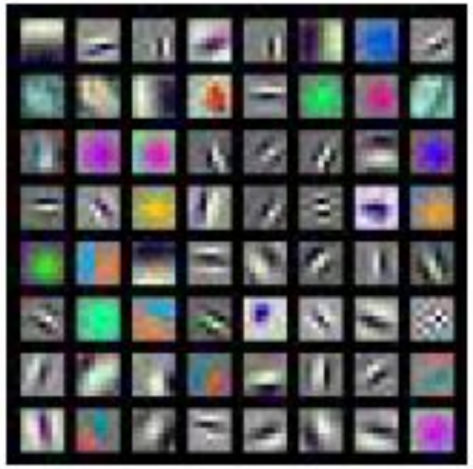
AlexNet:
64 x 3 x 11 x 11



ResNet-18:
64 x 3 x 7 x 7

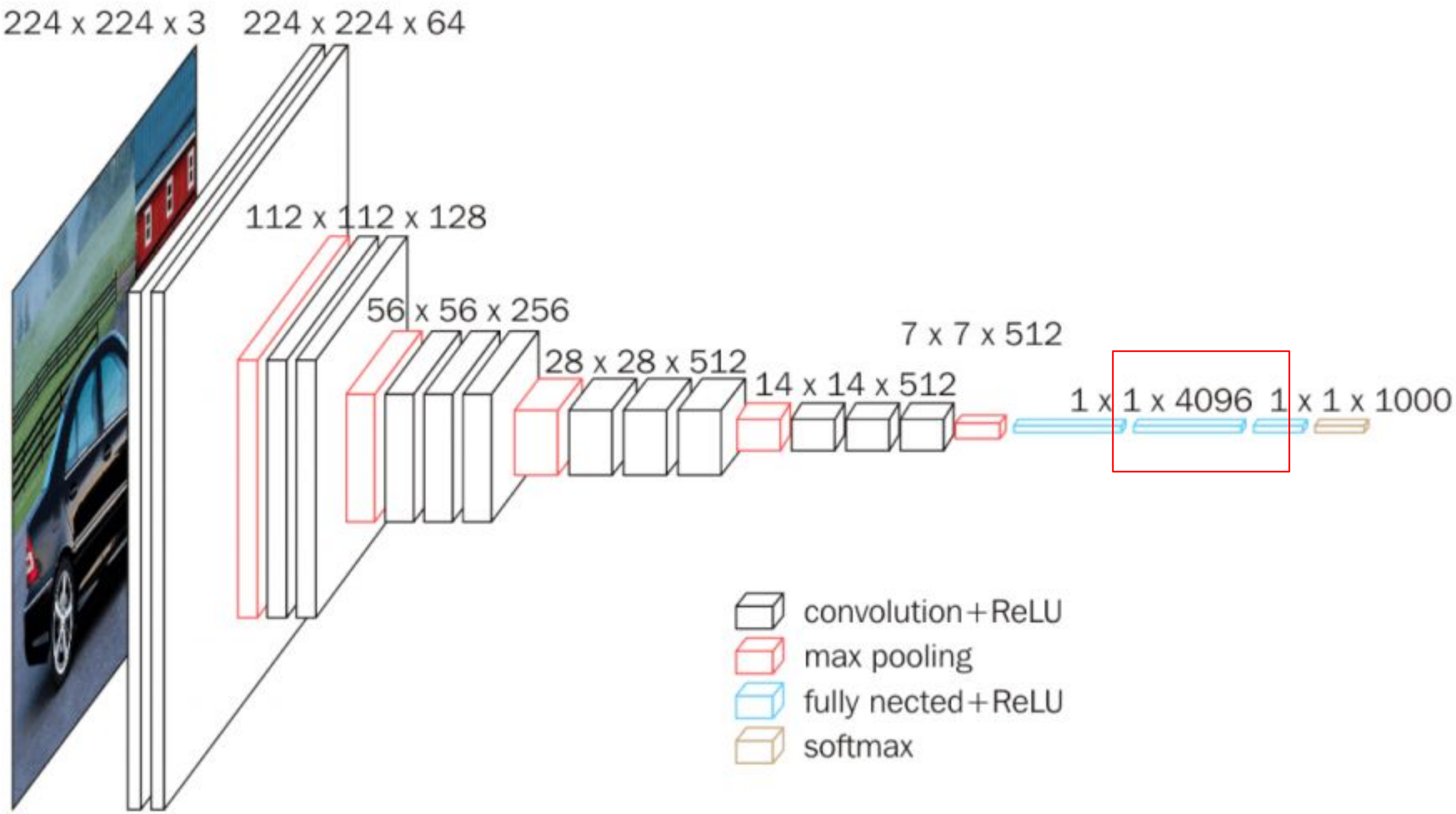


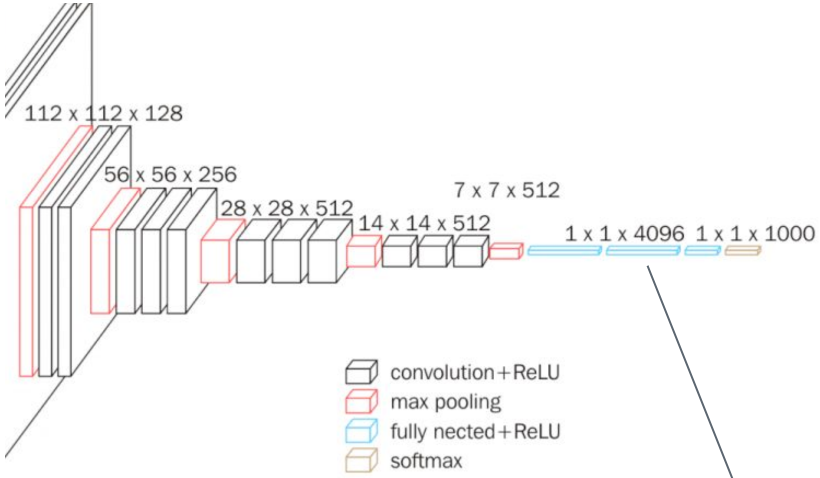
ResNet-101:
64 x 3 x 7 x 7



DenseNet-121:
64 x 3 x 7 x 7

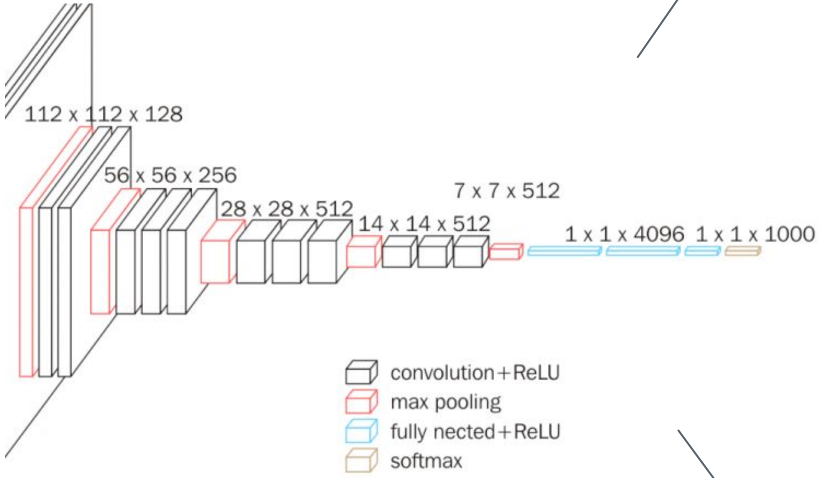
Fei-Fei Li, Justin Johnson, Serena Yeung, 2017


224 x 224 x 3
224 x 224 x 64
112 x 112 x 128
56 x 56 x 256
28 x 28 x 512
14 x 14 x 512
7 x 7 x 512
1 x 1 x 4096
1 x 1 x 1000
convolution+ReLU
max pooling
fully nected+ReLU
softmax

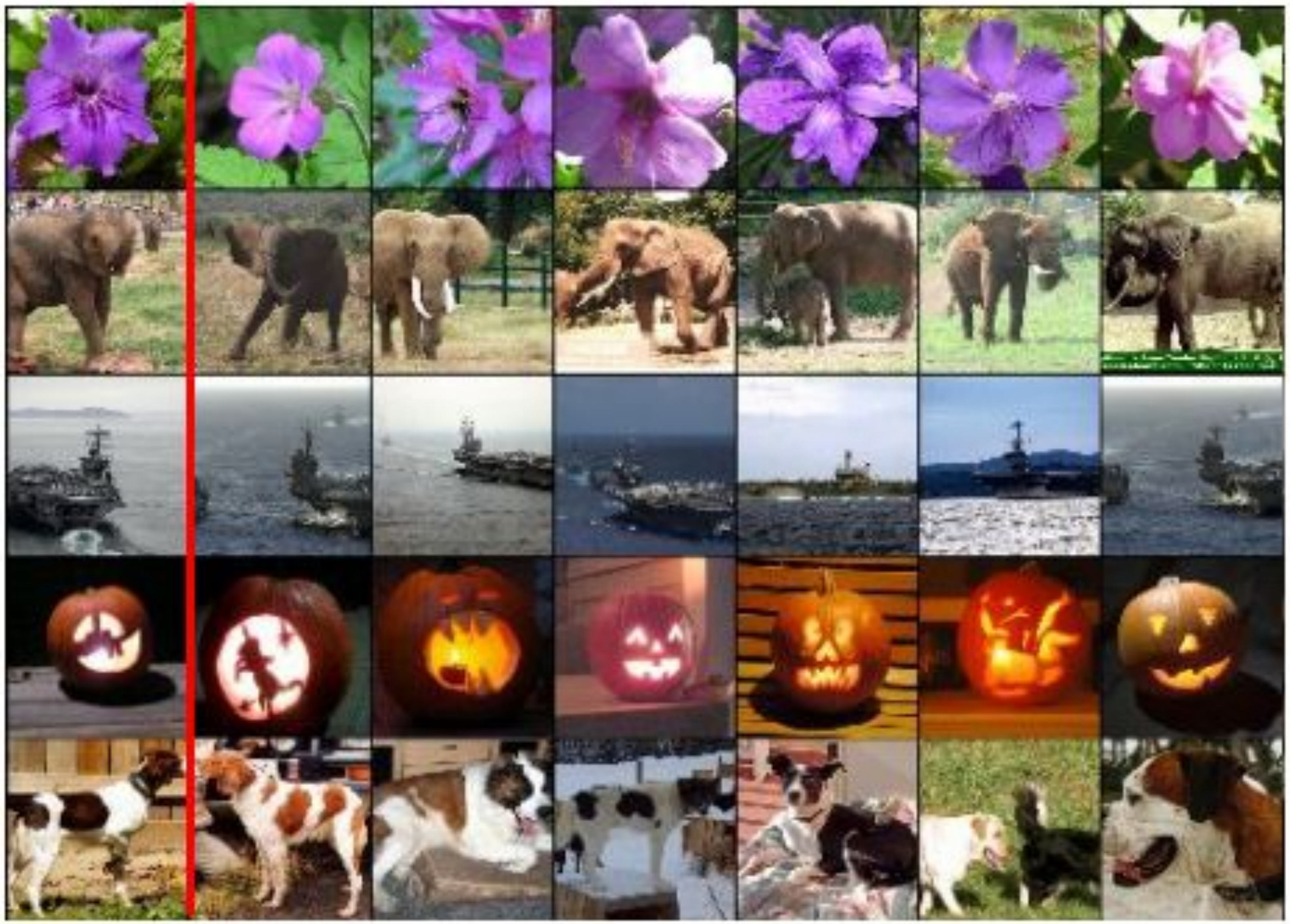


4,096 Numbers

4,096 Numbers

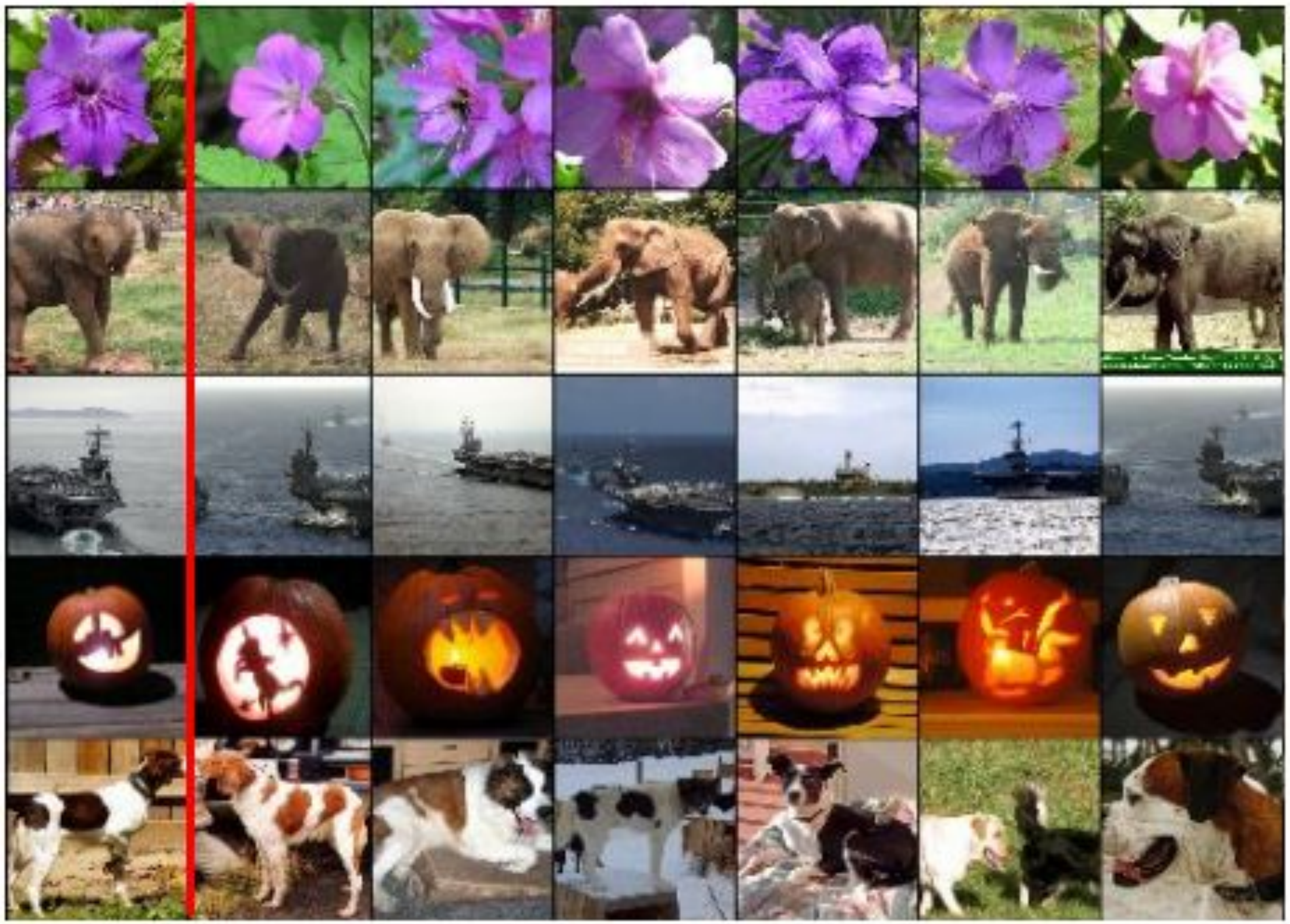


4,096 Numbers

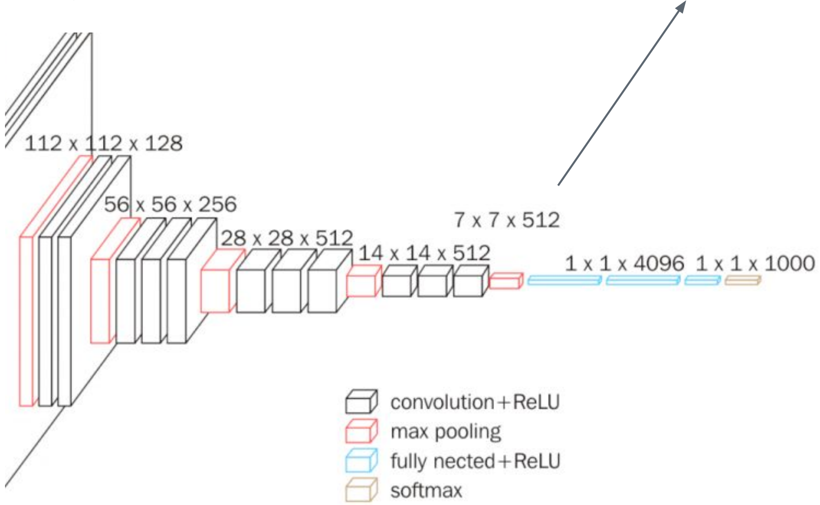


Fei-Fei Li, Justin Johnson, Serena Yeung, 2017

Fei-Fei Li, Justin Johnson, Serena Yeung, 2017



Fei-Fei Li, Justin Johnson, Serena Yeung, 2017



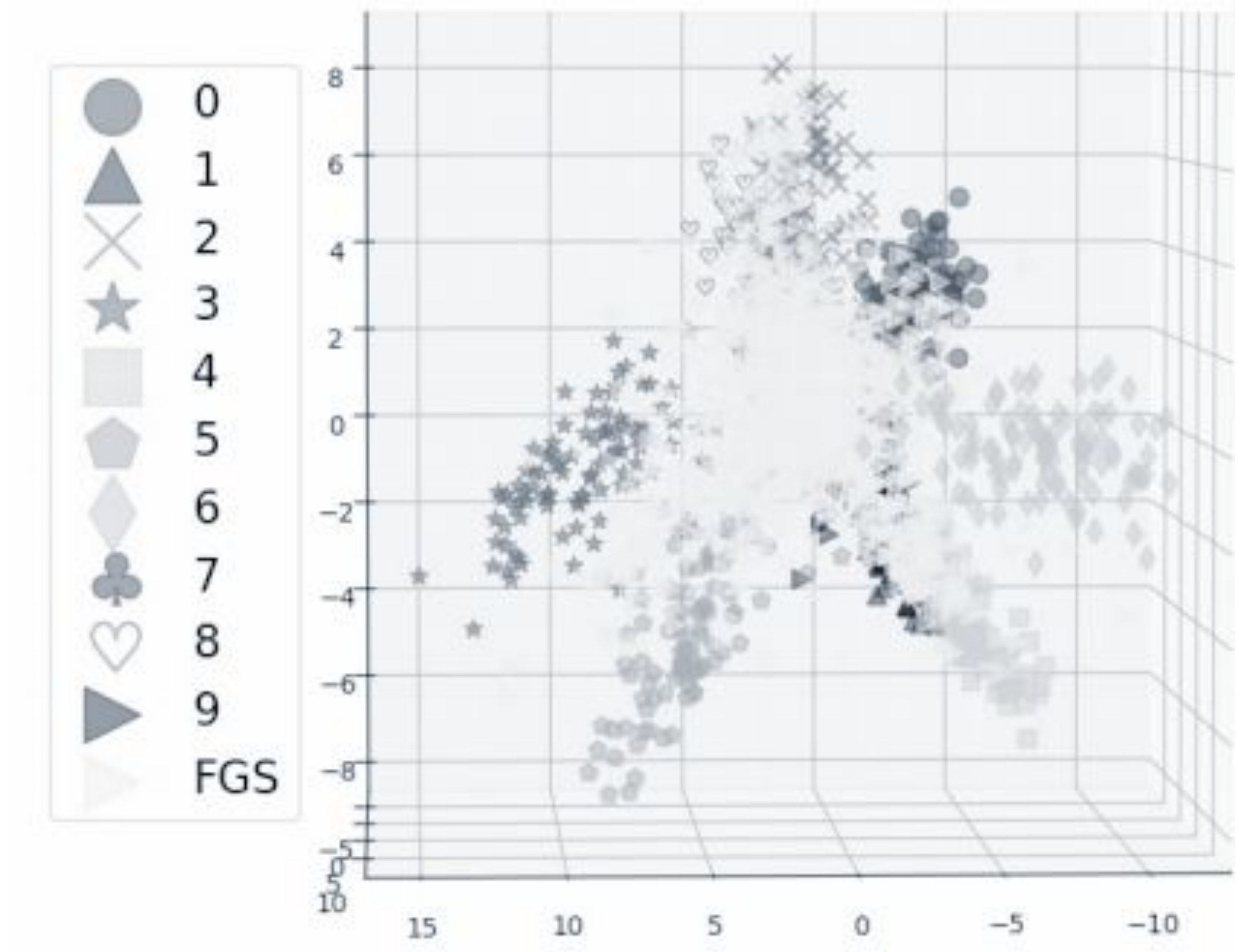
4,096 Numbers

2 Numbers

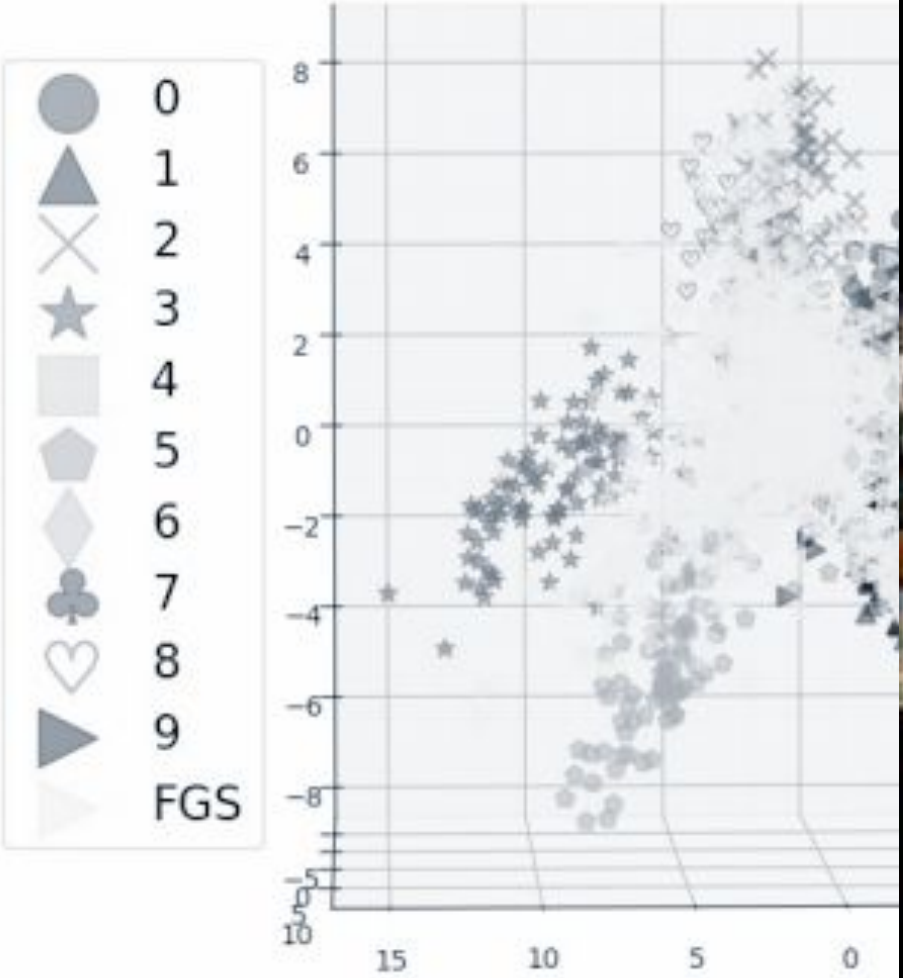
PCA / t-SNE / ??

4,096 Numbers

2 Numbers

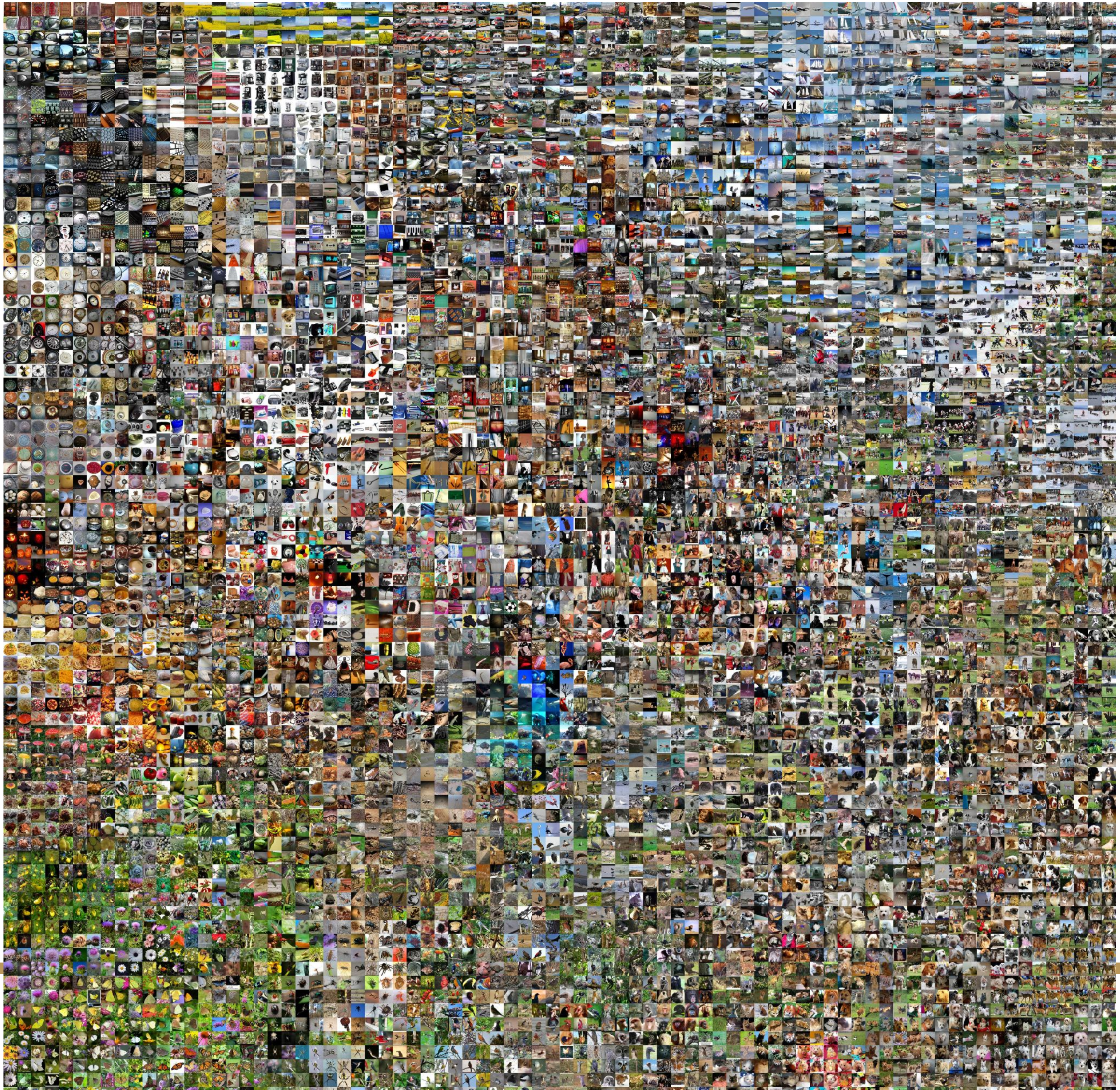


https://arxiv.org/pdf/1802.07124.pdf

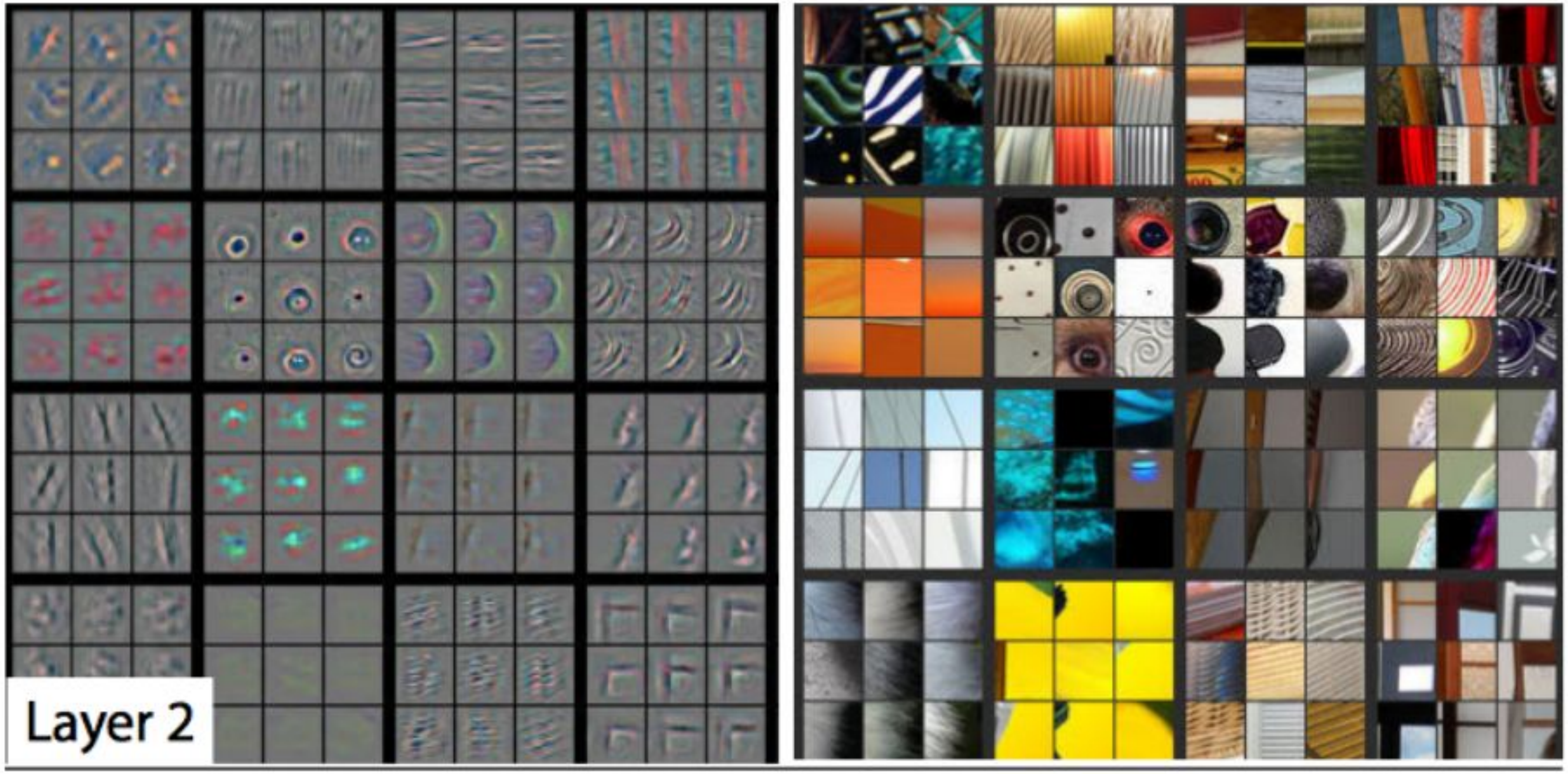


https://arxiv.org/pdf/1802.07124.pdf

https://cs.stanford.edu/people/karpathy/cnnembed/

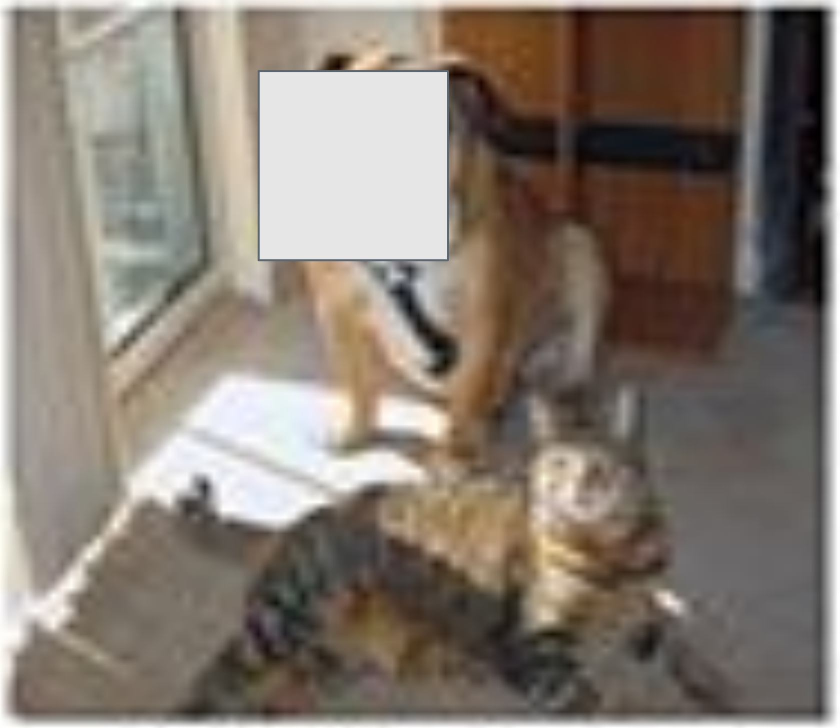


Patch-based Approaches

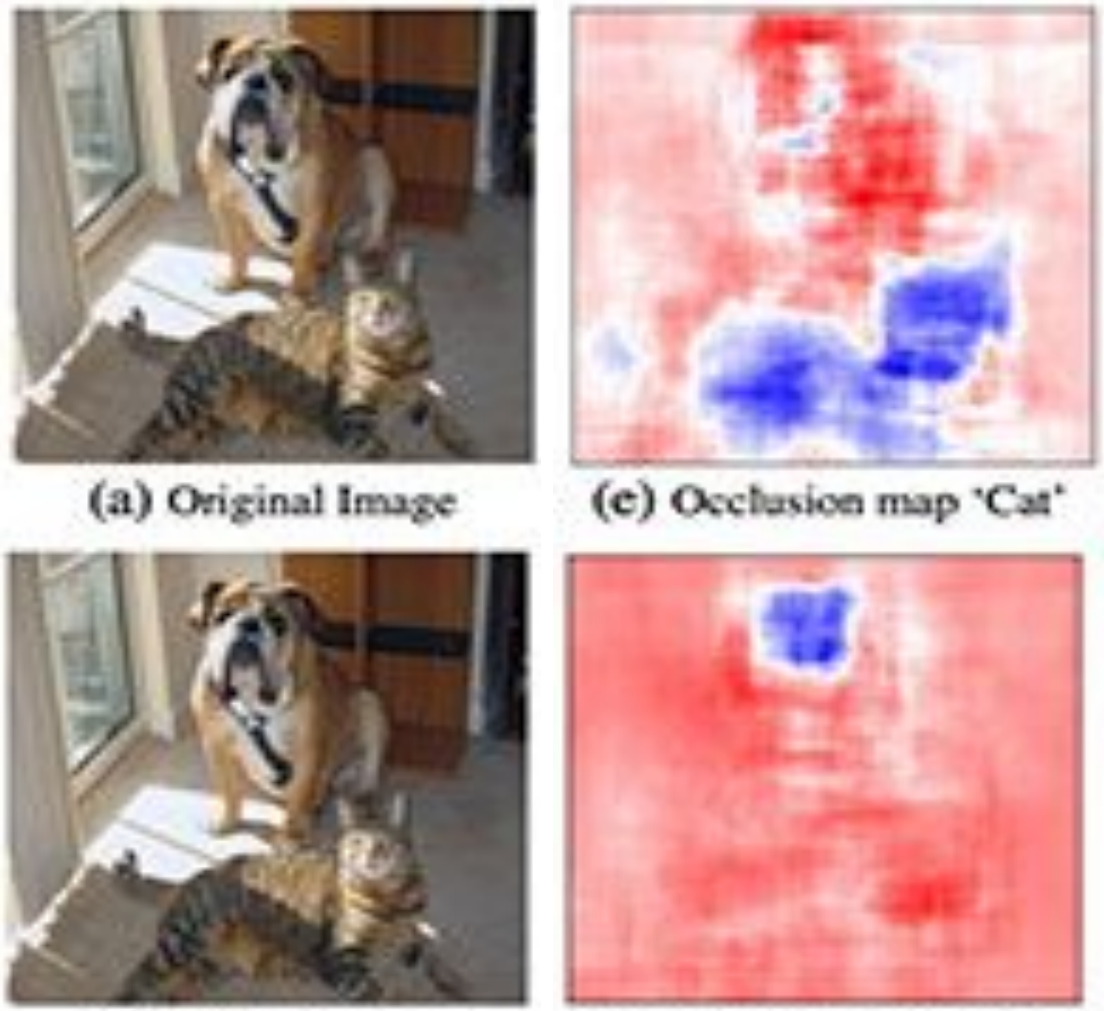


Zeiglar and Furgus, 2013

Occlusion



(g) Original Image



(a) Original Image

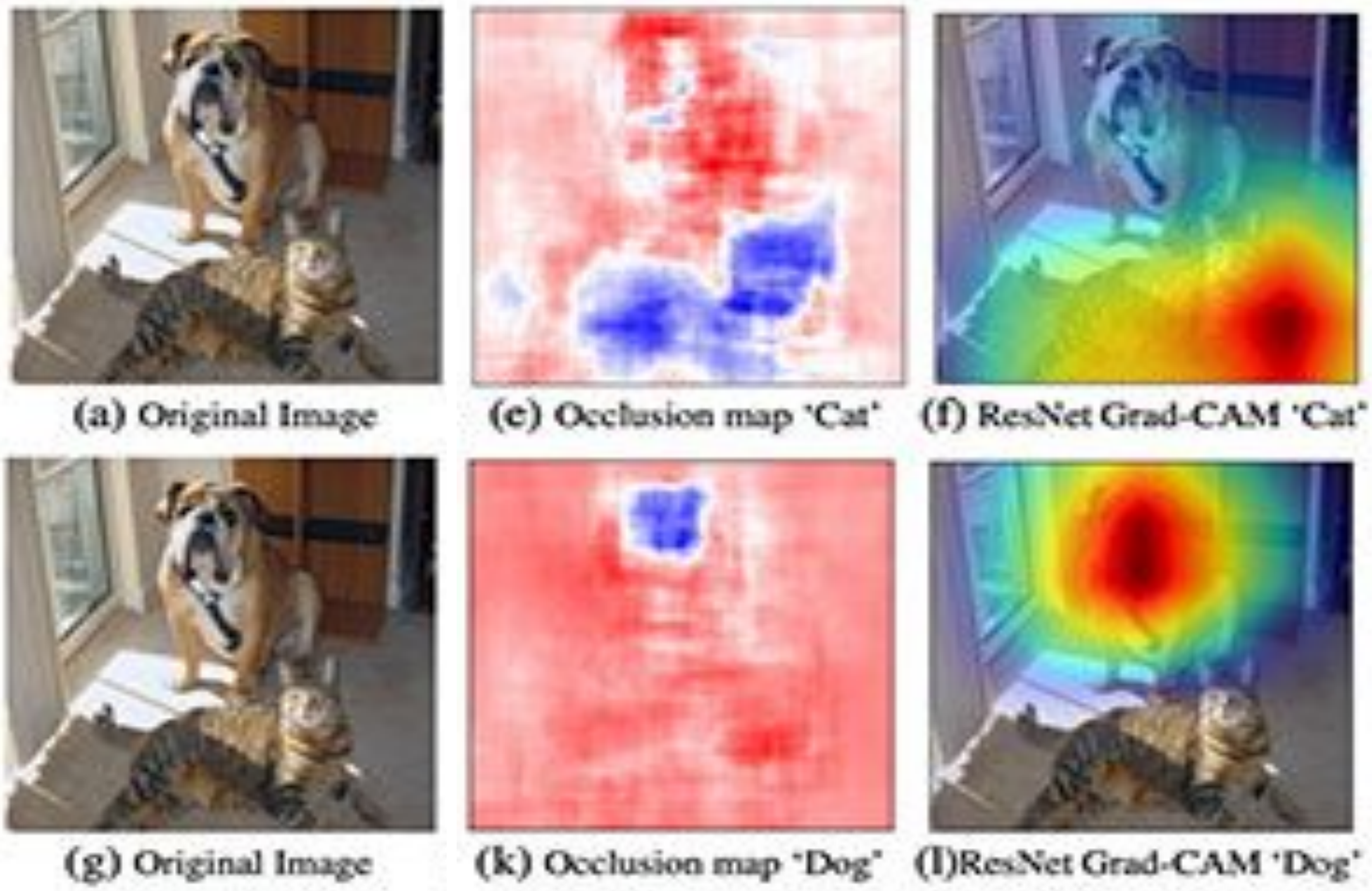
(e) Occlusion map 'Cat'

(g) Original Image

(k) Occlusion map 'Dog'

https://arxiv.org/pdf/1610.02391.pdf

Saliency



(a) Original Image

(e) Occlusion map 'Cat'

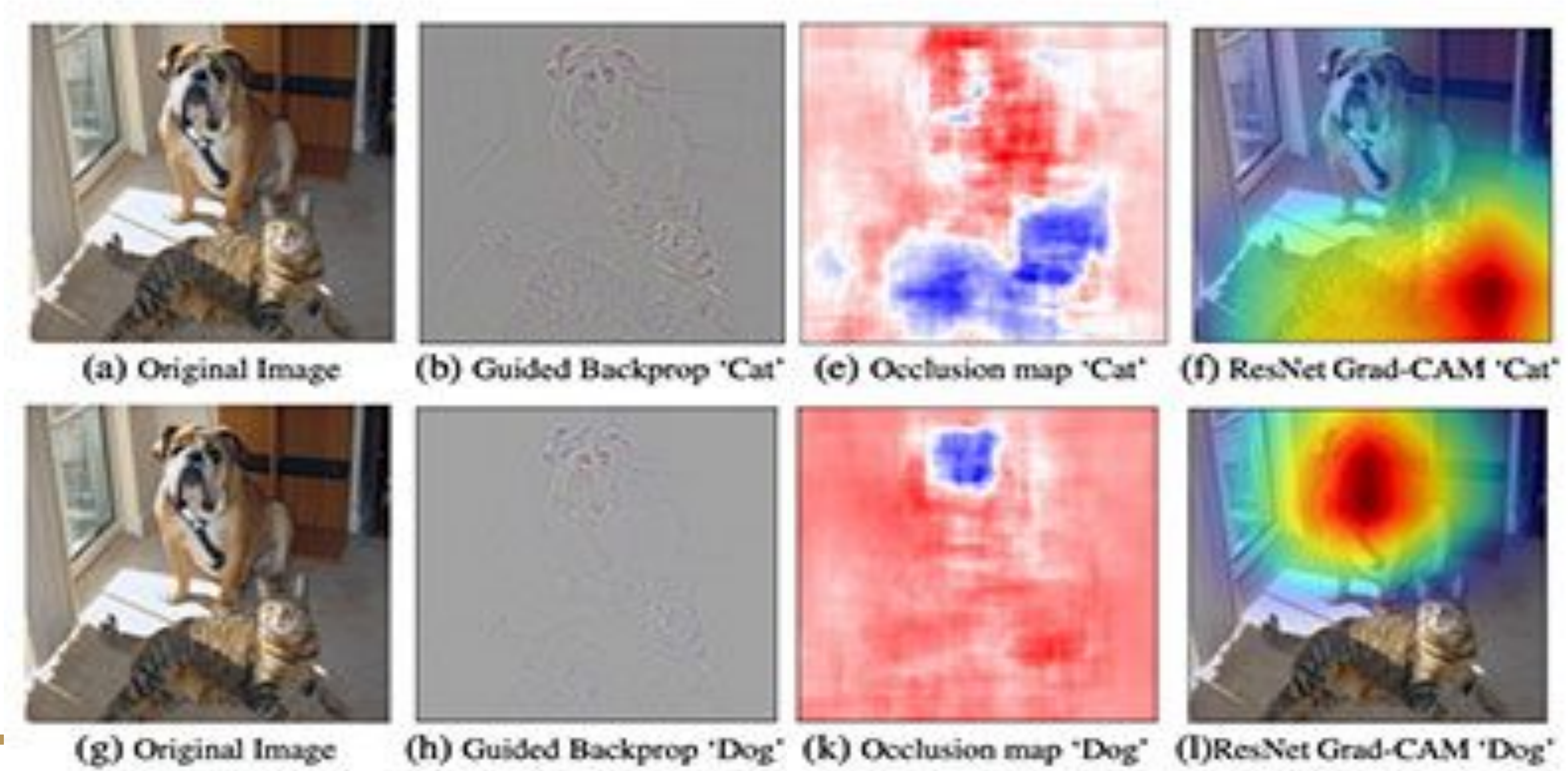
(f) ResNet Grad-CAM 'Cat'

(g) Original Image

(k) Occlusion map 'Dog'

(l)ResNet Grad-CAM 'Dog'

Guided Backpropogation

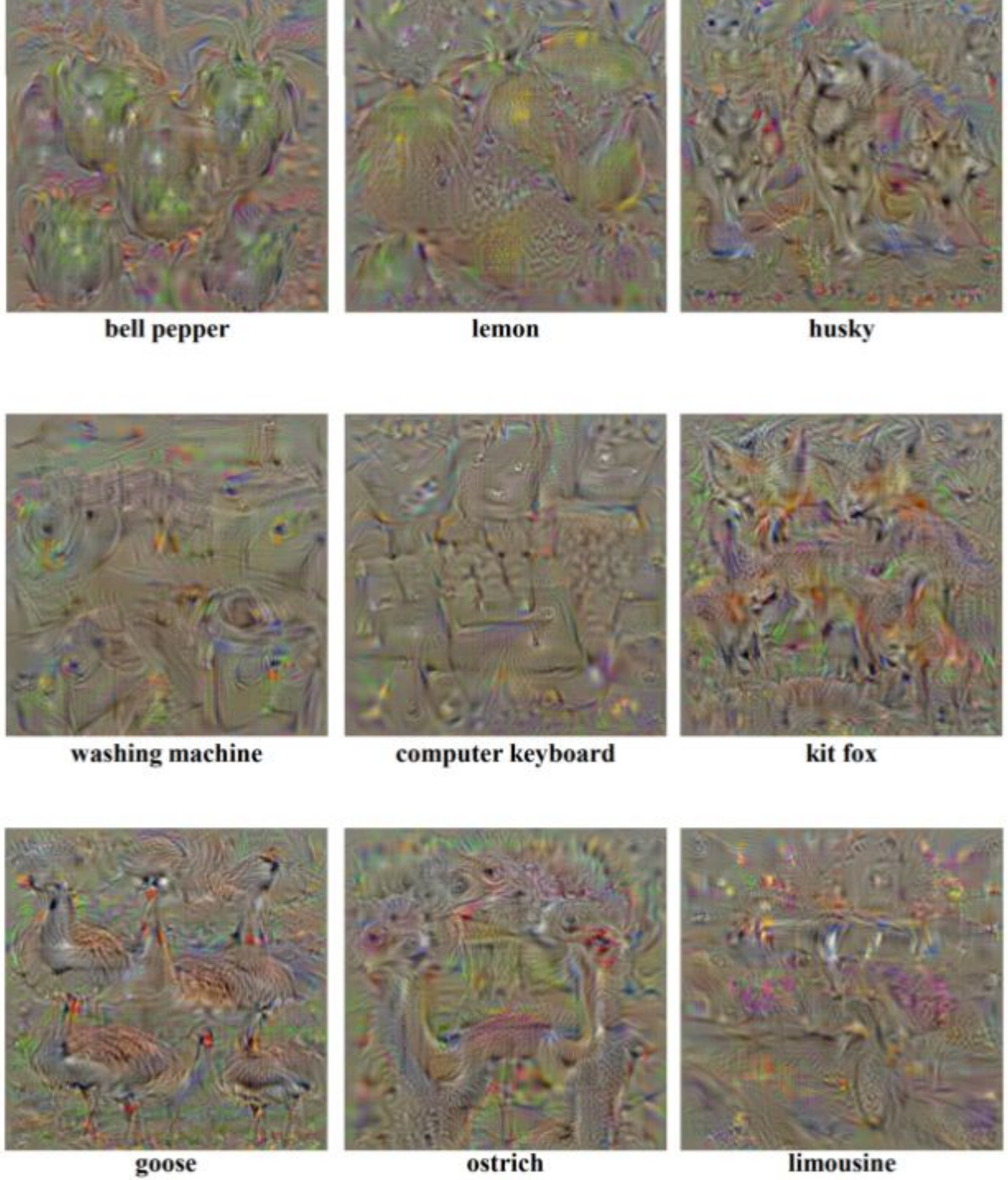


(a) Original Image (b) Guided Backprop 'Cat' (e) Occlusion map 'Cat' (f) ResNet Grad-CAM 'Cat'

(g) Original Image (h) Guided Backprop 'Dog' (k) Occlusion map 'Dog' (l)ResNet Grad-CAM 'Dog'

Zeiler and Fergus, Visualizing and Understanding Convolutional Networks, 2014

Gradient Ascent



bell pepper

lemon

husky

washing machine

computer keyboard

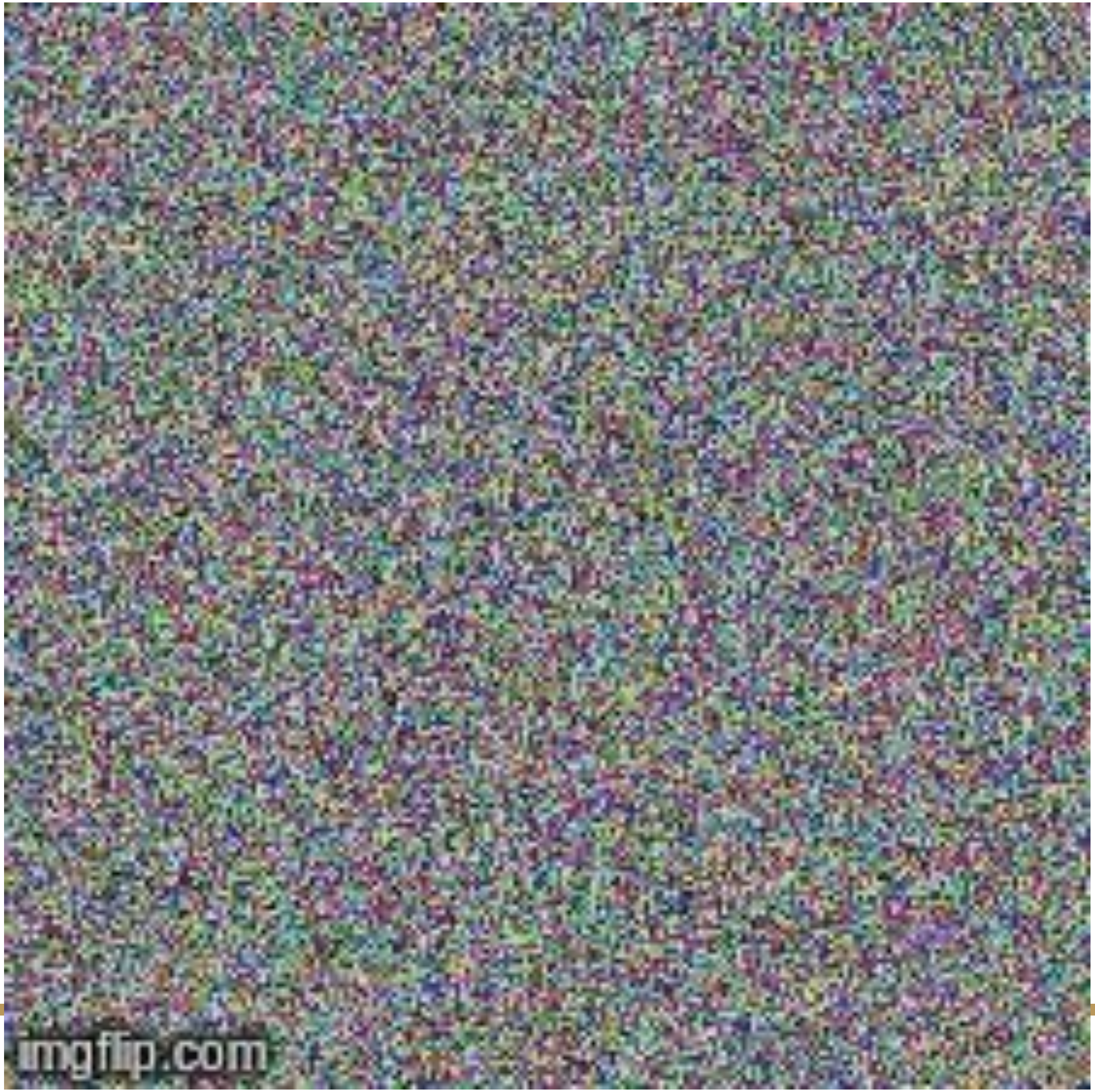
kit fox

goose

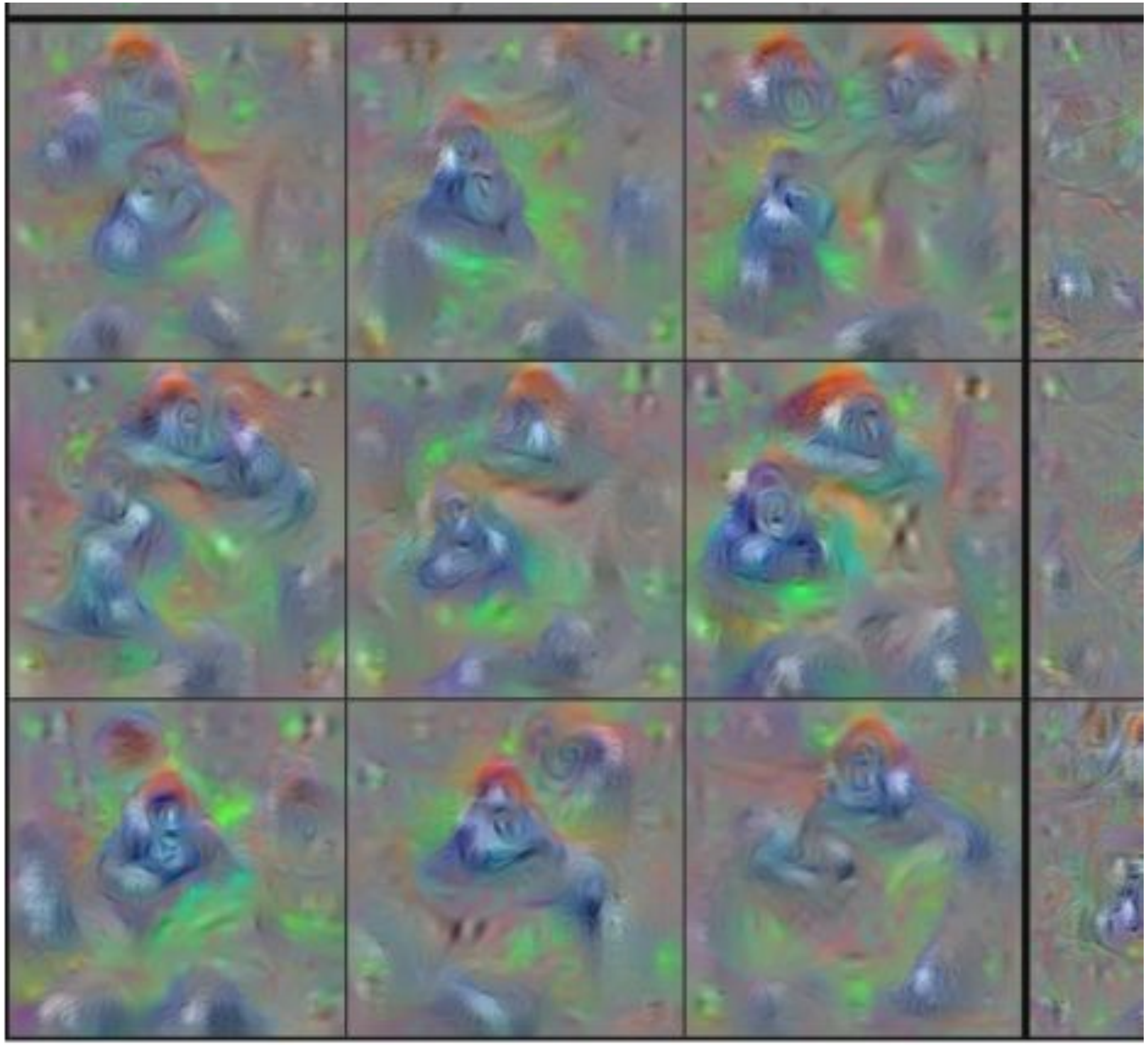
ostrich

limousine

https://arxiv.org/abs/1409.1556



https://glassboxmedicine.com/2019/07/13/class-model-visualization-for-cnns/



Gorilla

https://arxiv.org/pdf/1506.06579.pdf

https://yosinski.com/deepvis

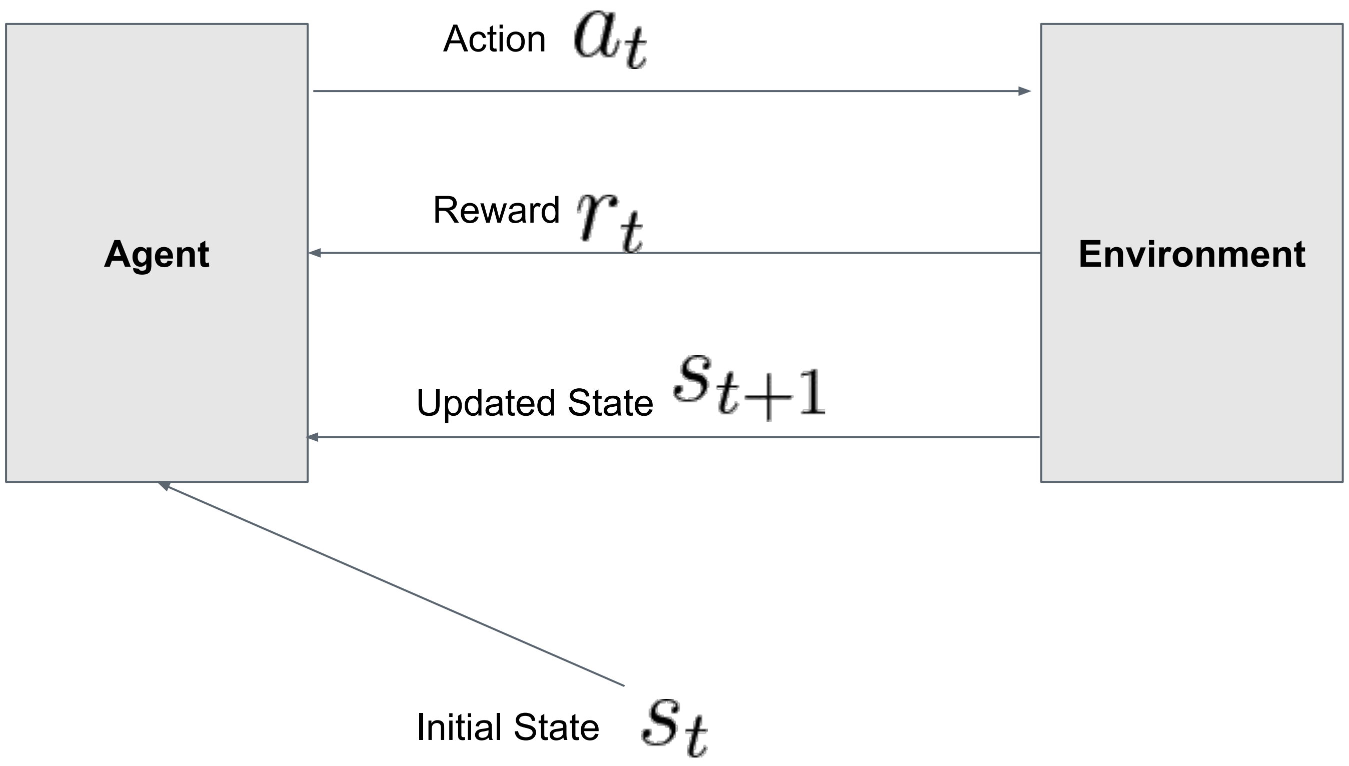
Summary: Understanding CNNs

- Dimensionality Reduction
- Maximal Patches
- Occlusion
- Saliency / Gradient Backpropogation
- Gradient Ascent

Deep Reinforcement Learning

State: Positions of Pieces
Action: Where to move Piece
Reward: 1 if you win, 0 if you lose.

Formalization



Markov Decision Process

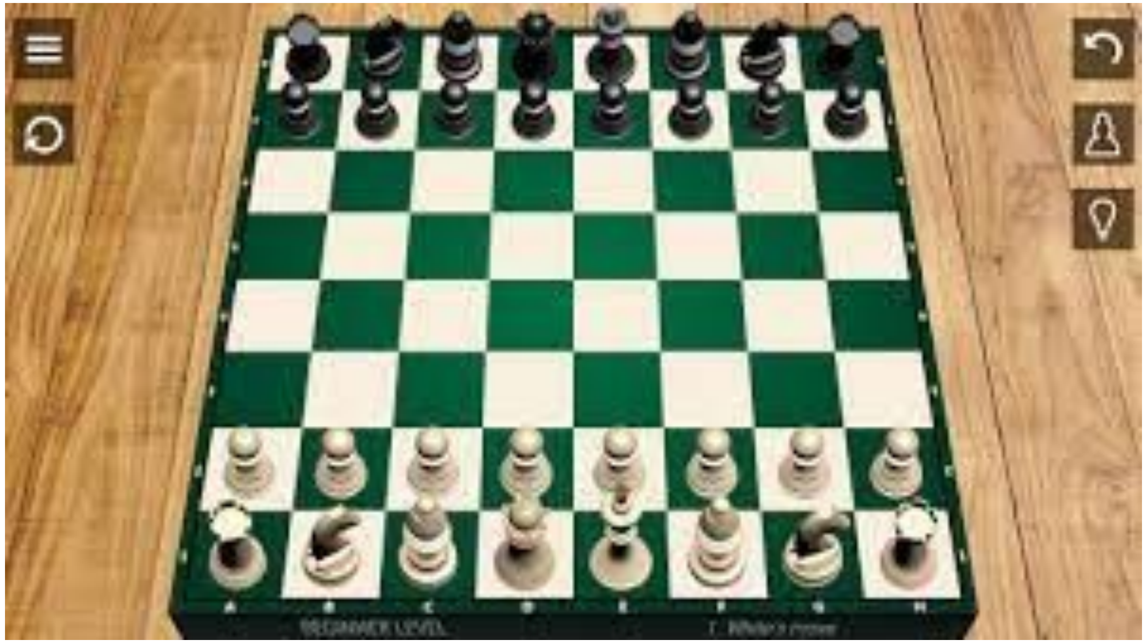
Markov Property: Conditional probability distribution of the future state of a process depends only on the current state.

Markov Decision Process

S	Set {...} of possible states
A	Set {...} of possible actions
R	Rewards for each (State, Action)
$\mathbb{P}$	Probabilities to transition to a new state S given current state and action
γ	Discount

Markov Decision Process

1) At step t=0, the environment is defined as some initial state s_0



Markov Decision Process

1) At step t=0, the environment is defined as some initial state s_0

2) Starting at t=0, and repeating until finished:

A) Agent chooses an action a_t

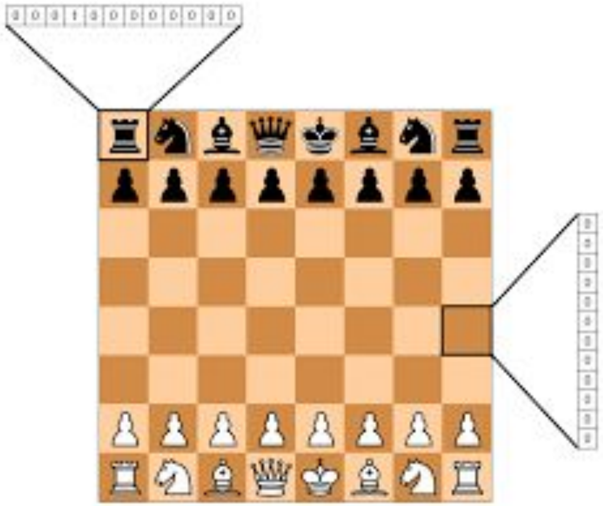
Markov Decision Process

1) At step t=0, the environment is defined as some initial state s_0

2) Starting at t=0, and repeating until finished:

   A) Agent chooses an action a_t

   B) Environment rewards action r_t



Markov Decision Process

1) At step t=0, the environment is defined as some initial state s_0

2) Starting at t=0, and repeating until finished:

- A) Agent chooses an action a_t
- B) Environment rewards action r_t
- C) Environment identifies next state s_{t+1}

Sicilian Defense
8
7
6
5
4
3
2
1
a
b
c
d
e
f
g
h
caissa.com

Markov Decision Process

1) At step t=0, the environment is defined as some initial state s_0

2) Starting at t=0, and repeating until finished:

   A) Agent chooses an action a_t

   B) Environment rewards action r_t

   C) Environment identifies next state s_{t+1}

   D) Agent receives r_t and s_{t+1}

Sicilian Defense
8
7
6
5
4
3
2
1
a
b
c
d
e
f
g
h
caissa.com

Markov Decision Process

A) Agent chooses an action a_t

B) Environment rewards action r_t

C) Environment identifies next state s_{t+1}

D) Agent receives r_t and s_{t+1}

π Policy. Takes in State and Possible Actions, and determines what action to take.

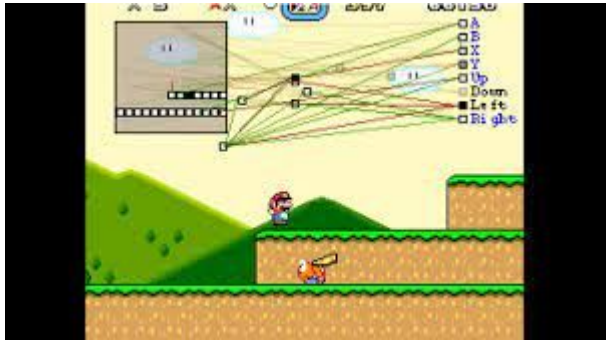
Ω Constraints. Takes in State and Possible Actions, and determines what (if any) actions cannot be taken.

$\sum_t \gamma^t r_t$ Objective Function. We seek to maximize the discounted rewards across all steps t.

Q-learning

MARIO
000000
×00
WORLD
1-1
TIME
375

Agent
Environment


A
B
X
Y
Up
Down
Left
Right

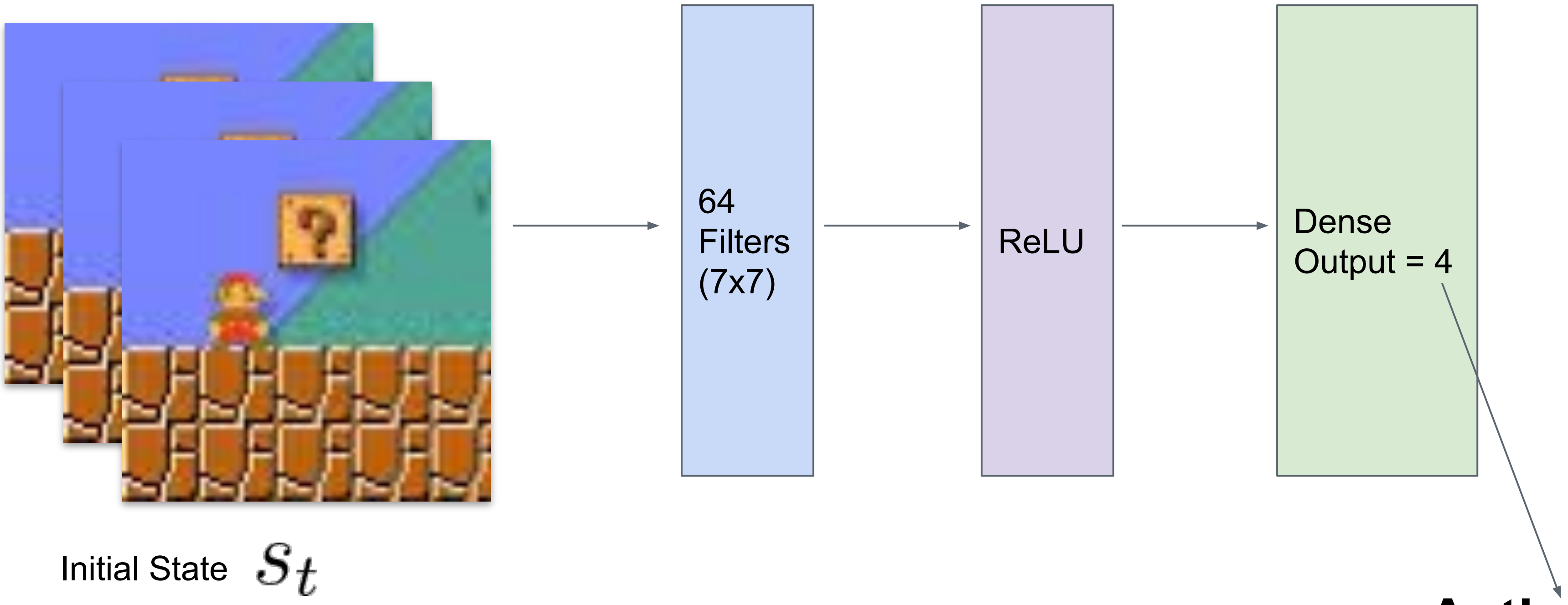
Q-learning

State: Raw pixels of a frame of the game
Actions: {Jump, Left, Right, Wait}
Reward: Score increase for moving right, decrease for left

$$\sum_t \gamma^t r_t$$

Objective Function. We seek to maximize how far we go right, across all steps t.

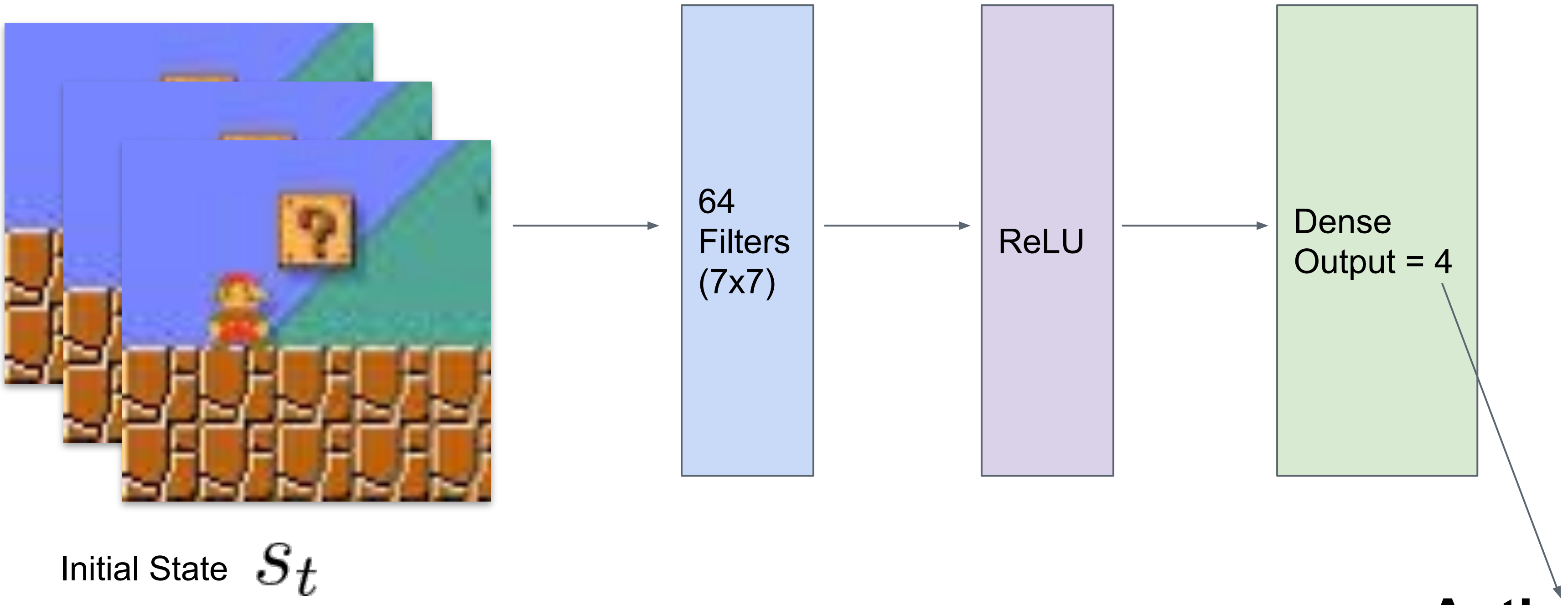
Q-Learning Network Architecture



Initial State s_t

Actions:
{Jump, Left, Right, Wait}

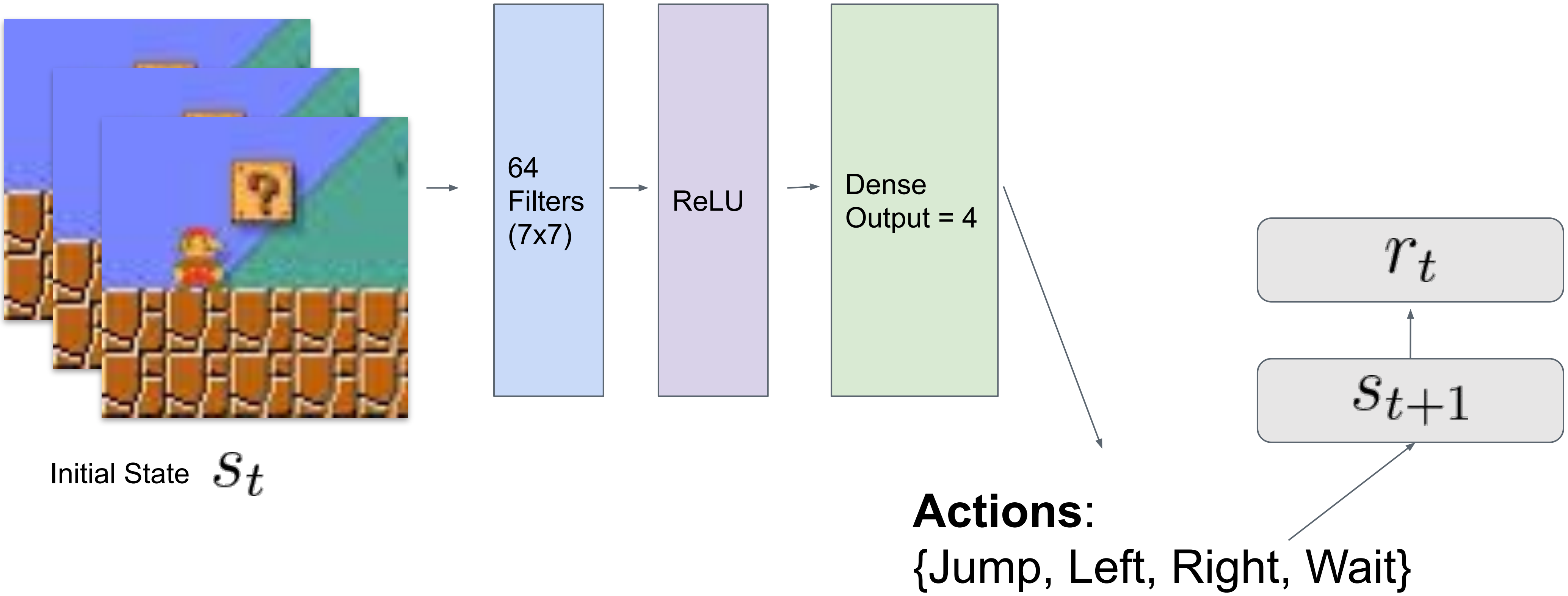
Q-Learning Network Architecture



Initial State s_t

Actions:
{Jump, Left, Right, Wait}

Q-Learning Network Architecture



64 Filters (7x7)

ReLU

Dense Output = 4

r_t

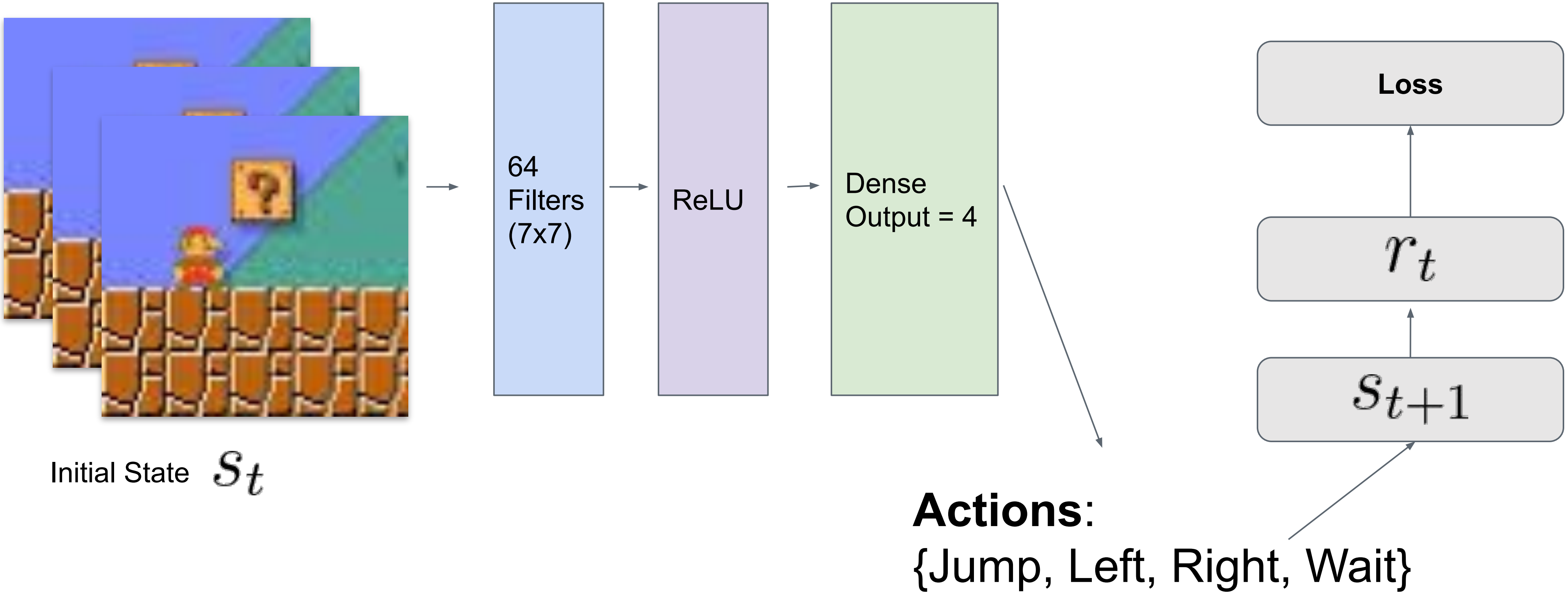
s_{t+1}

Initial State s_t

Actions:
{Jump, Left, Right, Wait}

Q-Learning

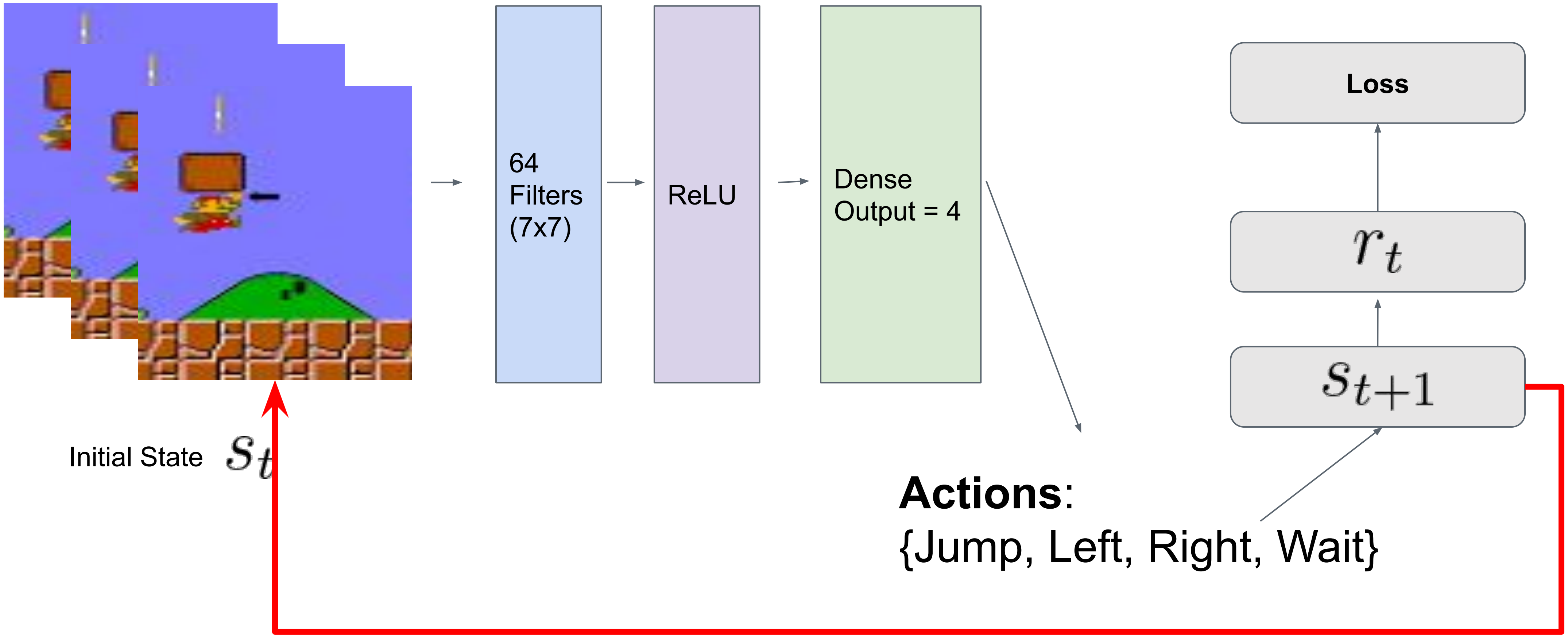
$$E\left[\left(r + \gamma \max_{a'} Q(s',a';\theta_k) - Q(s,a;\theta_k)\right)^2\right]$$



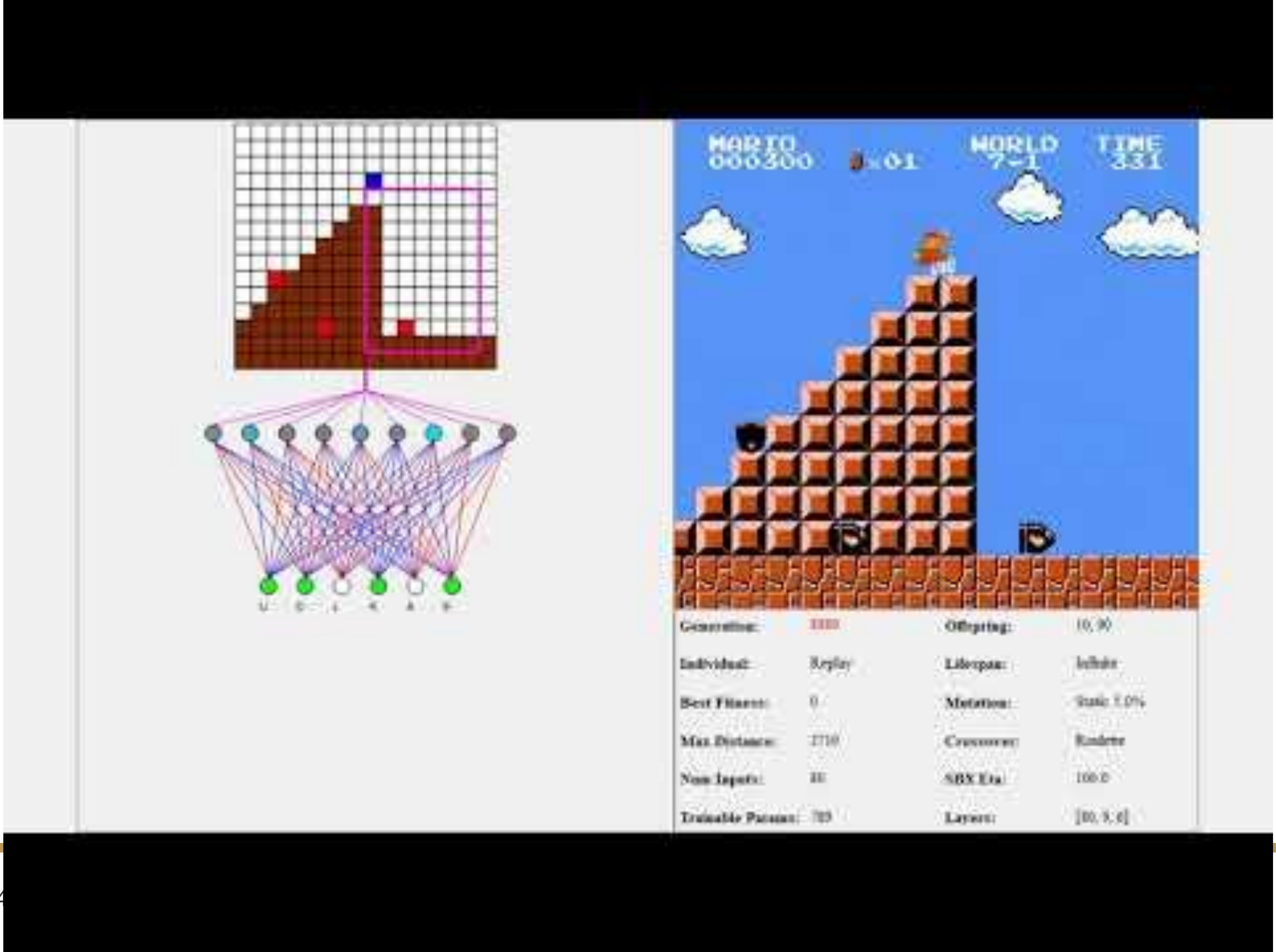
Actions:
{Jump, Left, Right, Wait}

Q-Learning

$$E\left[\left(r+\gamma \max_{a'} Q(s',a';\theta_k)-Q(s,a;\theta_k)\right)^2\right]$$



Actions:
{Jump, Left, Right, Wait}



Generation:	[illegible]	Offspring:	10, 90
Individual:	Replay	Lifespan:	Infinite
Best Fitness:	0	Mutation:	Static 5.0%
Max Distance:	2710	Crossover:	Roulette
Num Inputs:	80	SBX Eta:	100.0
Trainable Params:	789	Layers:	[80, 9, 6]

https://www.youtube.com/watch?v=CI3FRsSAa_U