

CNNs, RNNs

Bryan Cardenas, MSc
Maxwell Cai, PhD



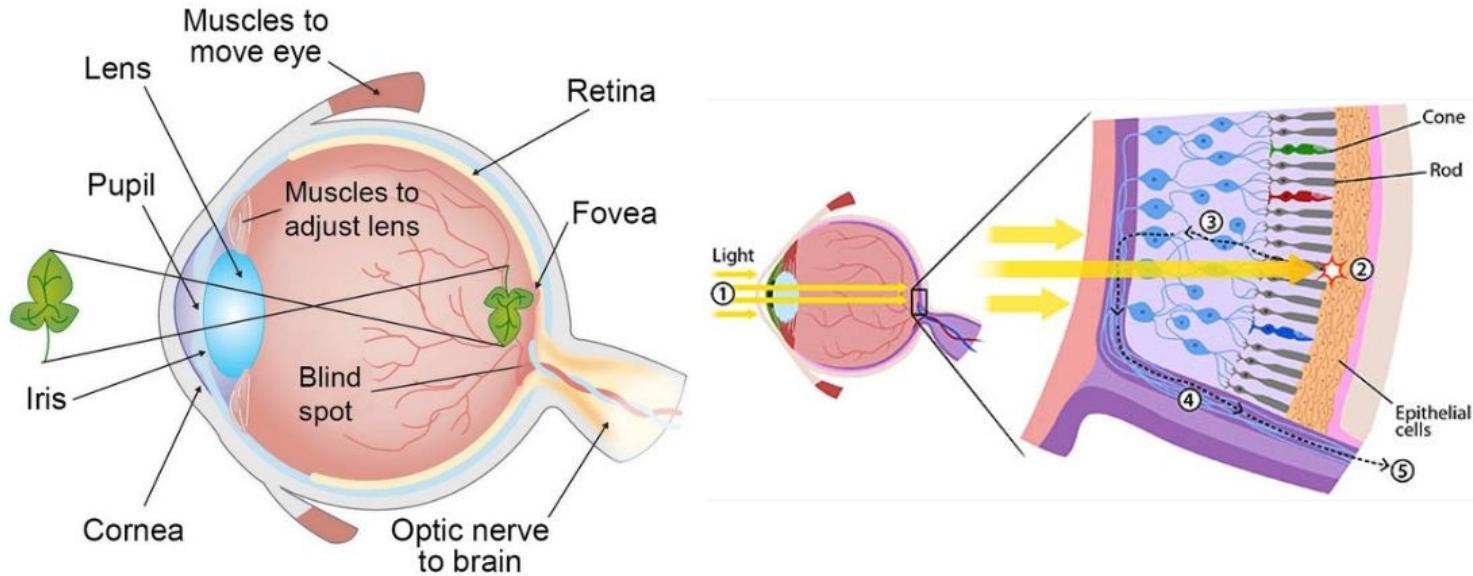
Background Prerequisites

Welcome to the world of computer vision!

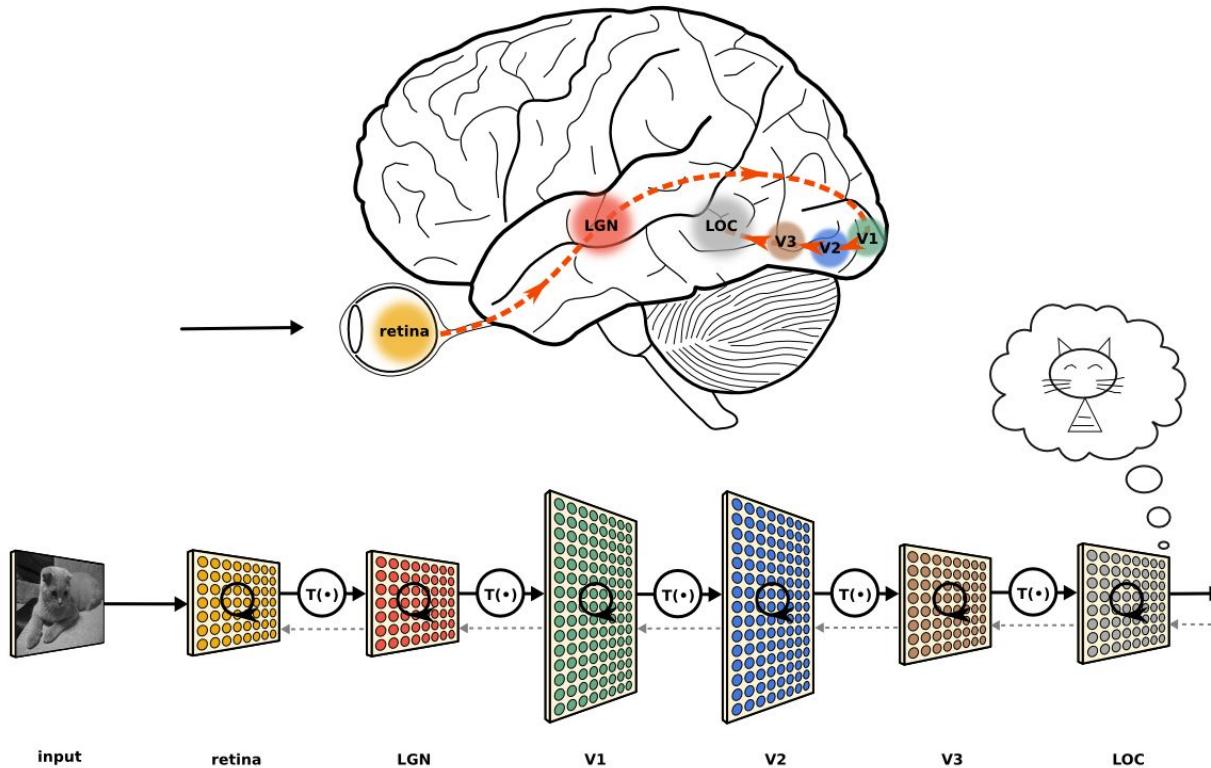
How do we let computers see something?

How do we see something

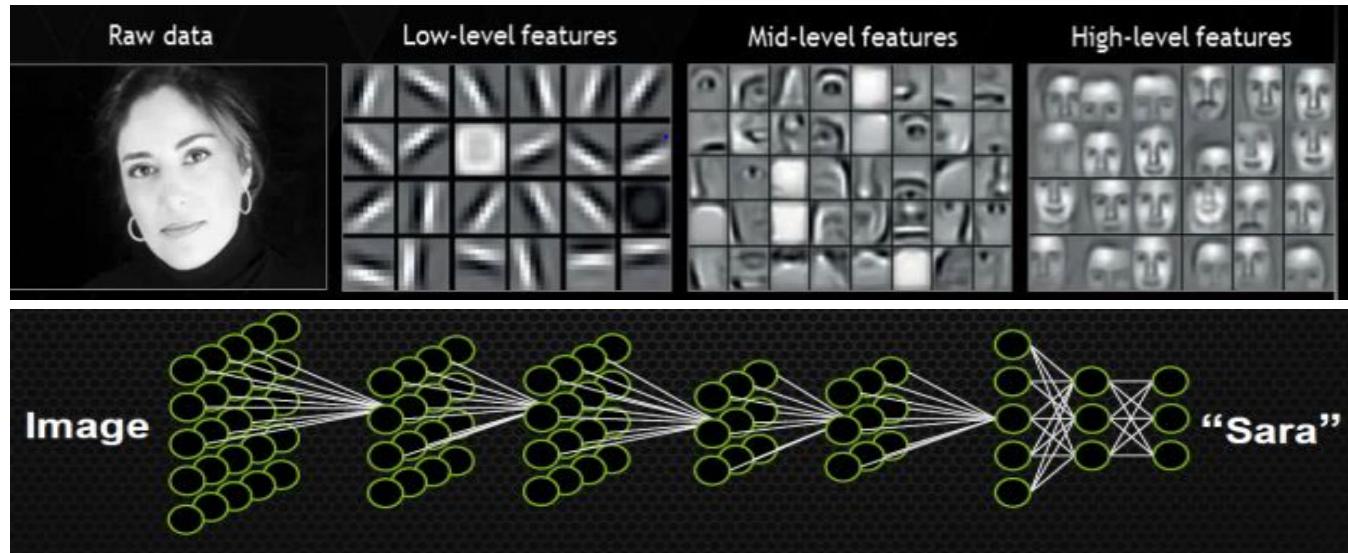
Human eyes



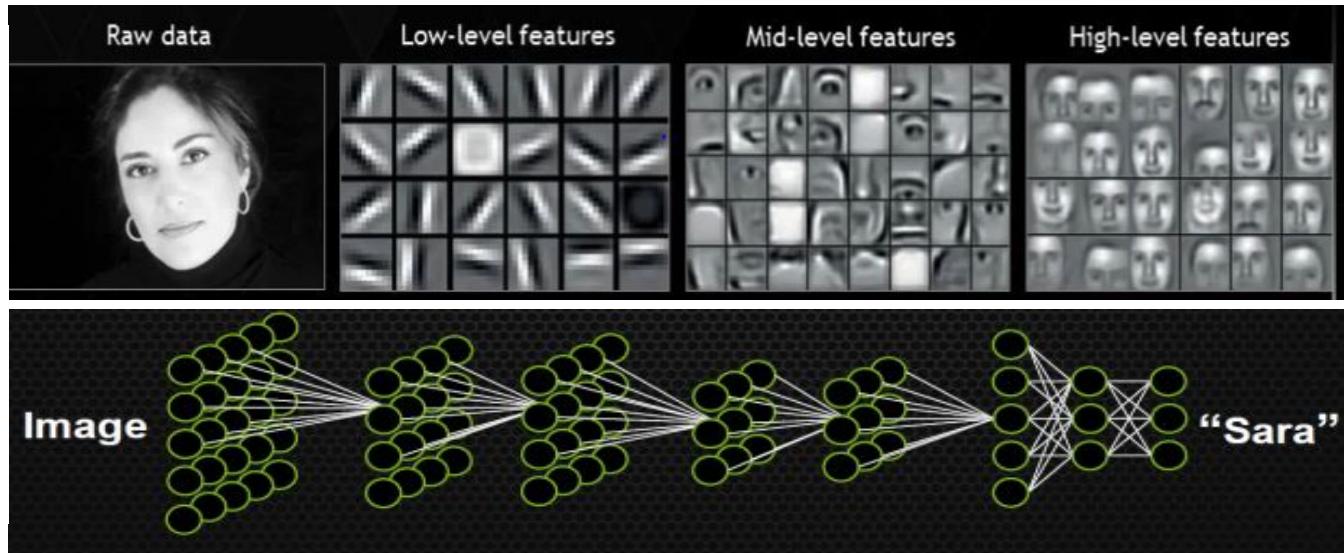
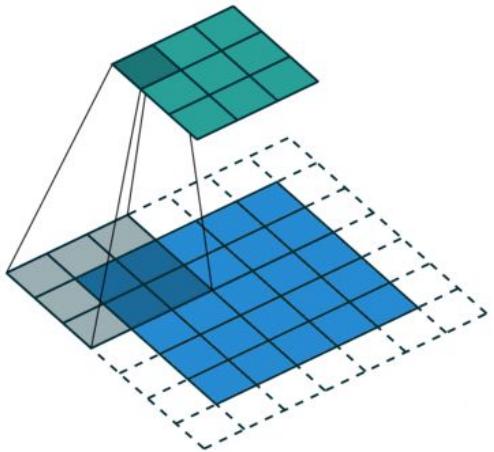
Human eyes



Convolutional Neural Network



Convolutional Neural Network



Representation of images in computers?

An **image**

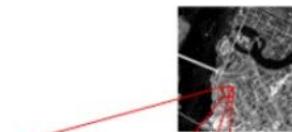
1. Is a matrix/array of intensity values
2. Usually consists of integers [0, 255] or float points [0,1]
3. Each element in the matrix is a **pixel**
4. Can have 1 greyscale channel or multiple colour
channels: RGB

Representation of images in computers?

An **image**

1. Is a matrix/array of intensity values
2. Usually consists of integers [0, 255] or float points [0,1]
3. Each element in the matrix is a **pixel**
4. Can have 1 greyscale channel or multiple colour **channels: RGB**

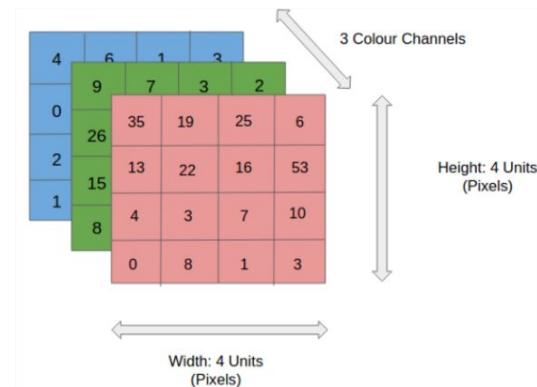
3D Array. Shape: (width, height, channel)



2D Array. Shape = (width, height)

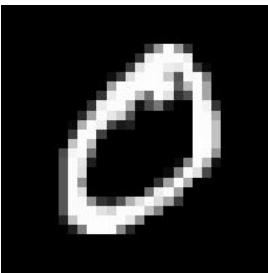
A 4x6 grid of grayscale pixel values represented as integers:

170	238	85	255	221	0
68	136	17	170	119	68
221	0	238	136	0	255
119	255	85	170	136	238
238	17	221	68	119	255
85	170	119	221	17	136

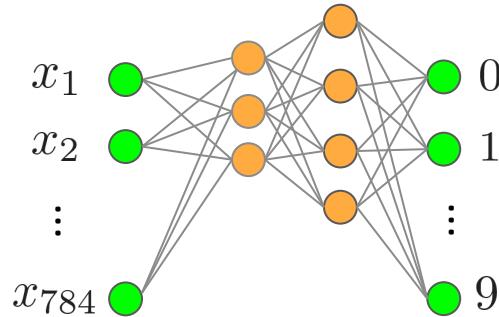


Representation of images in computers?

We will use a neural network to recognize
Hand written digits

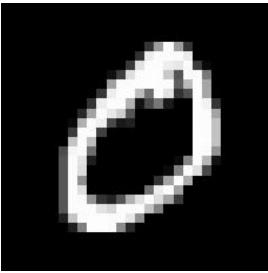


images of size $28 \times 28 = 784$ pixels



Representation of images in computers?

We will use a neural network to recognize
Hand written digits

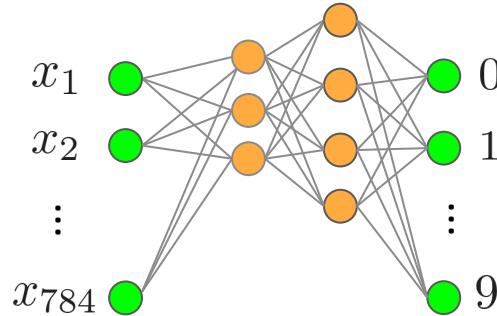


images of size $28 \times 28 = 784$ pixels

Inefficient: Every pixel is connected to
everything

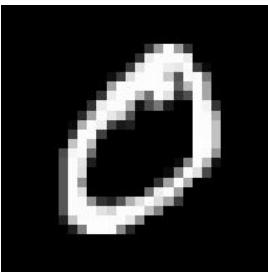
Regions in Images share features

Not shift invariant



Representation of images in computers?

We will use a neural network to recognize
Hand written digits

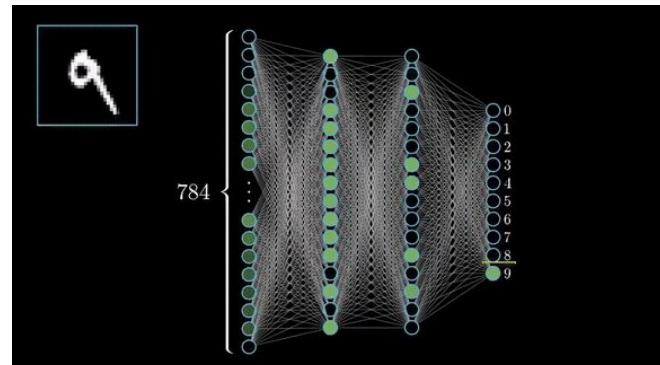
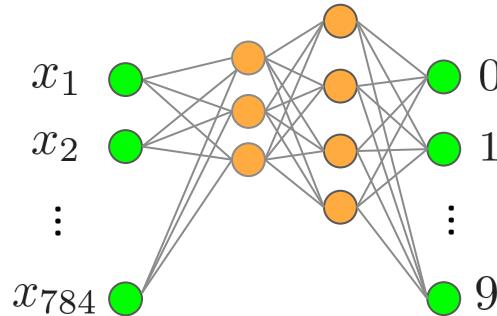


images of size $28 \times 28 = 784$ pixels

Inefficient: Every pixel is connected to
everything

Regions in Images share features

Not shift invariant

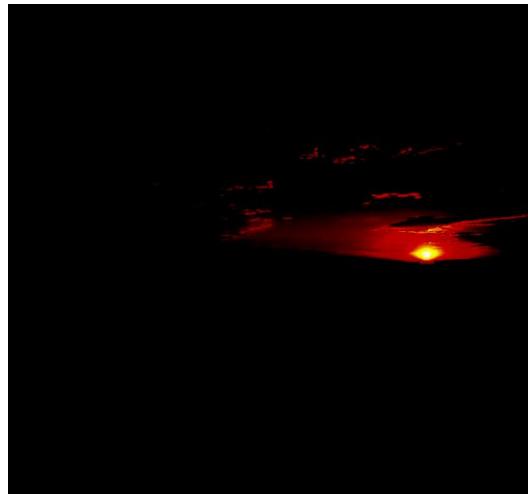


Convolutions

Filters



f



Convolutions

Filters



f



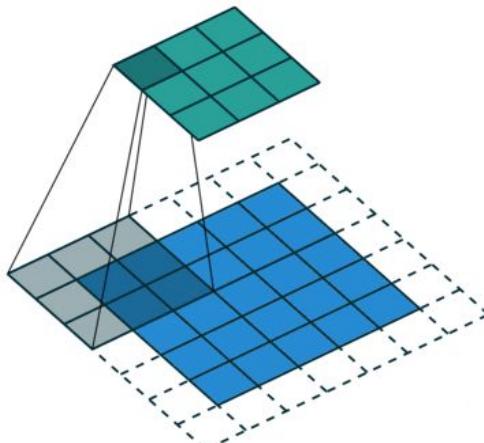
Convolutions

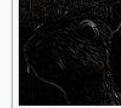
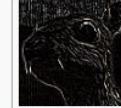
A filter applies a **convolution** operation upon the original signal

Kernel (filter): small matrix that we use to convolve an image

Convolution:

An operation that “blends” one function with another.



Operation	Kernel	Image result
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

Convolutions

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5

Source layer (image)

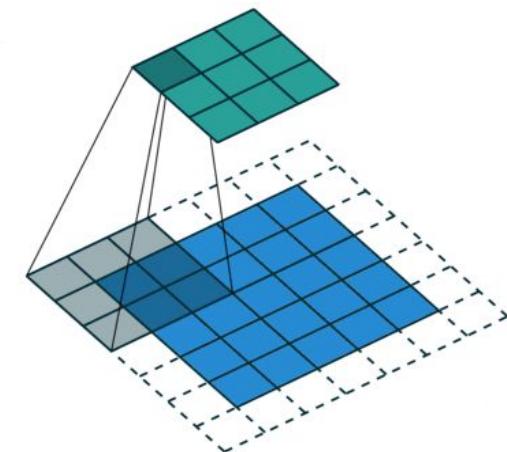
$$(-1 \times 2) + (0 \times 2) + (1 \times 6) + (-2 \times 4) + (0 \times 3) + (2 \times 4) + (-1 \times 3) + (0 \times 9) + (1 \times 4) = 5$$

Convolutional
Kernel (filter)

-1	0	1
-2	0	2
-1	0	1

Destination layer

5



Convolutions

1 x1	1 x0	1 x1	0	0
0 x0	1 x1	1 x0	1	0
0 x1	0 x0	1 x1	1	1
0	0	1	1	0
0	1	1	0	0

Image

4		

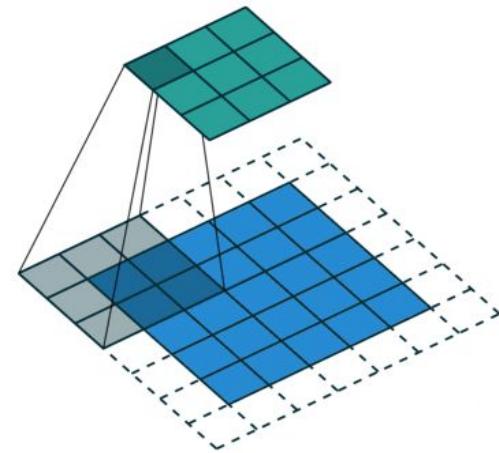
Convolved Feature

- The kernel is **shifted** over the image with a step size, and computes the output for each position.
- The step size is called **stride**.
- Output has **smaller** dimension than input: $\text{dim}(\text{output}) = \text{dim}(\text{input}) - (\text{dim}(\text{kernel}) - 1)$
- Padding** is used to solve this problem, which artificially make the image “bigger” by adding synthesis data (typically 0-padding)

0	0	0	0	0	0	0
0	2	2	6	8	2	0
0	4	3	4	5	1	0
0	3	9	4	4	7	0
0	1	3	4	6	8	0
0	8	4	6	2	3	0
0	0	0	0	0	0	0

Original array

Padded array



Convolutions

Source layer (image)

2	2	6	8	2	0	1	2
4	3	4	5	1	9	6	3
3	9	4	4	7	7	6	9
1	3	4	6	8	2	2	1
8	4	6	2	3	1	8	8
5	8	9	0	1	0	2	3
9	2	6	6	3	6	2	1
9	8	8	2	6	3	4	5



W ₁	W ₂	W ₃
W ₄	W ₅	W ₆
W ₇	W ₈	W ₉

Kernel (filter)

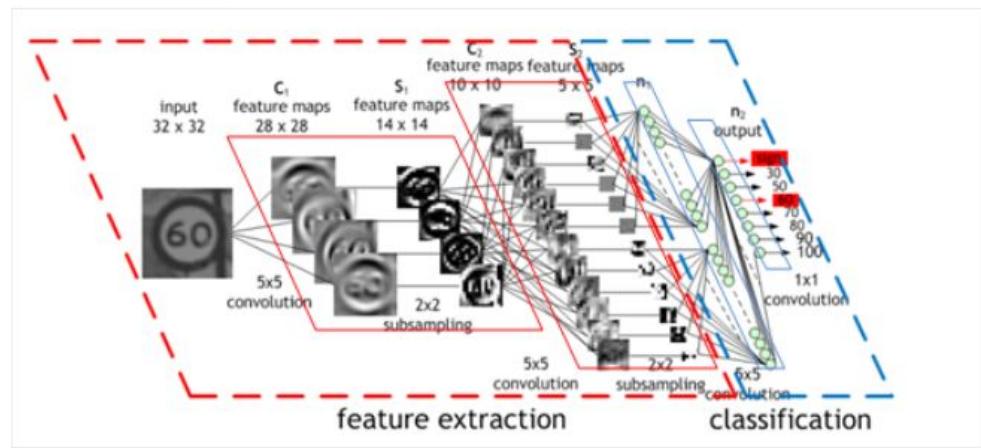
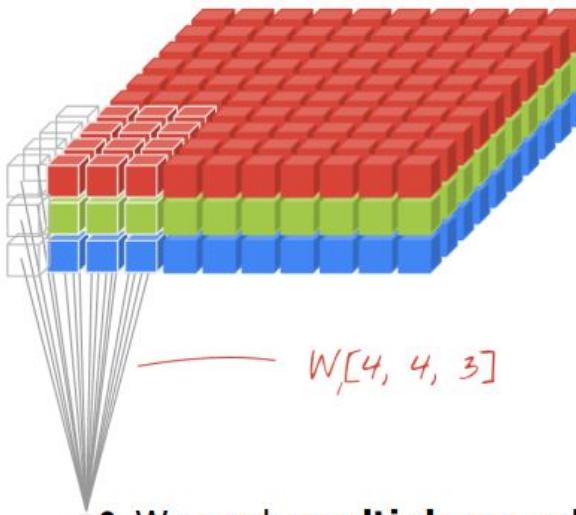
Feature map (activation map)

5								

How do we know which kernels to use?

- Kernels are learnt!
- Initialized **randomly**
- Backward propagation; the CNN learns which features to detect

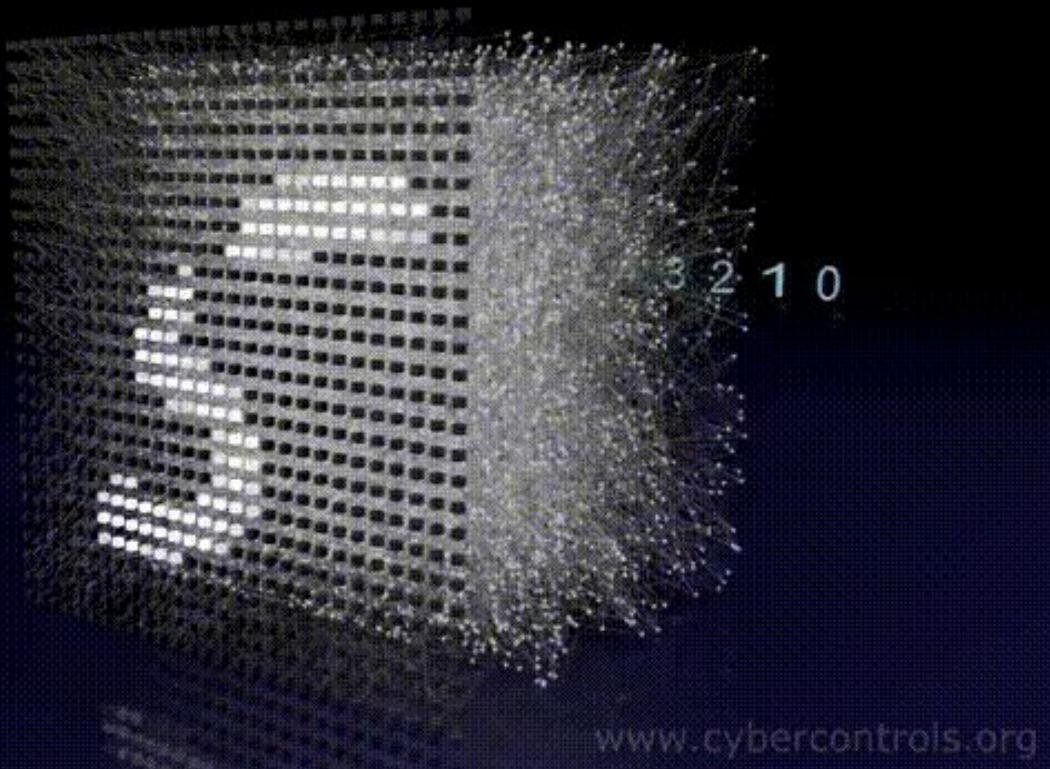
Convolutions



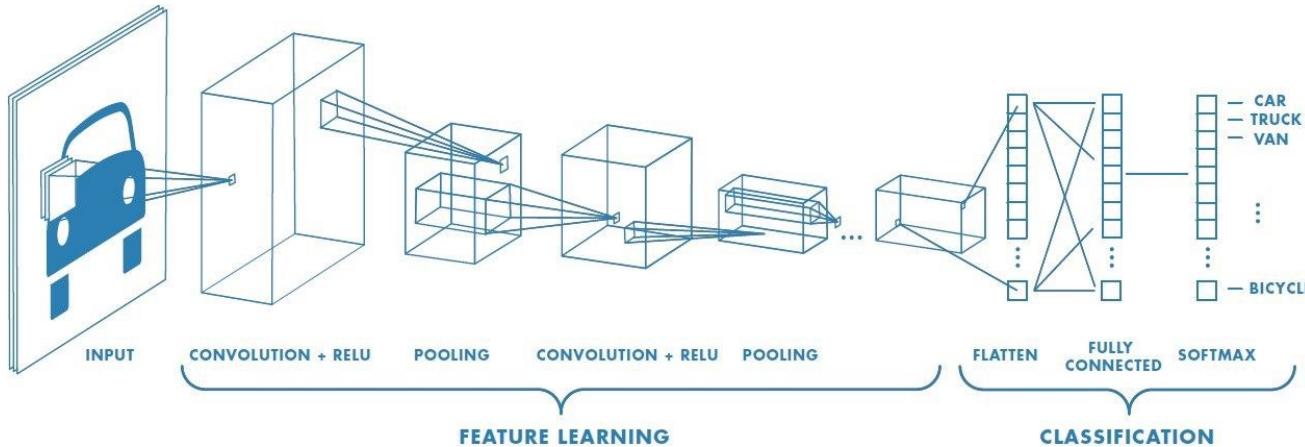
- We apply **multiple** convolutional **filters/kernels** to the **same** image.
- Each filter results in one convolutional **channel**.
- By learning the same image from different channels, one can detect complex patterns.

Convolutions

Type: ML Perceptron
Data Set: MNIST
Hidden Layers: 3
Hidden Neurons: 10000
Synapses: 24864180
Synapses shown: 1284
Learning: BP



CNN Form



Two parts:

1. Feature Learning
2. Classification (fully connected layers)

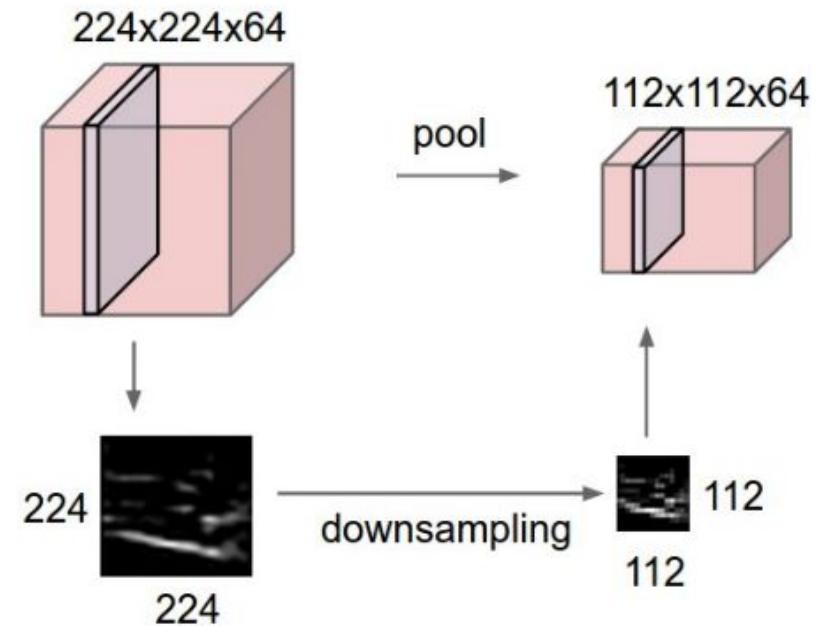
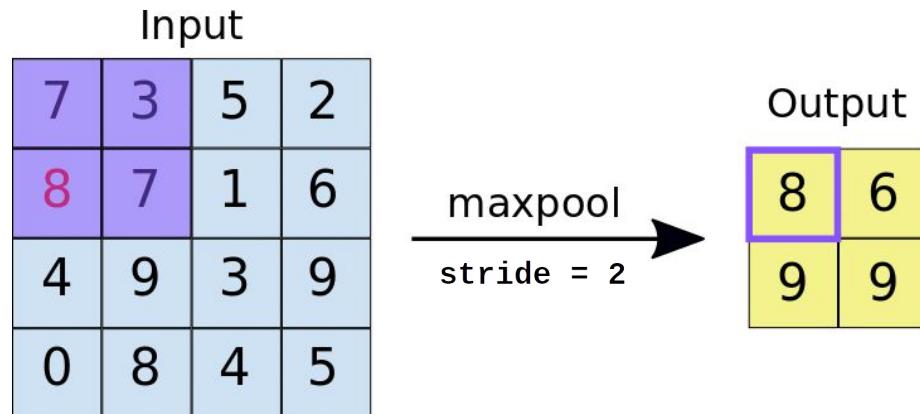
Images become smaller and number of filters increases as we go deeper
Complex Images -> we need more filters
Easy images -> we need fewer filters

Pooling (downsampling)

We need to **discard** information **gradually**.

Pooling is a way of information **abstraction**.

Pooling is usually applied **after** convolutions.



Max pooling: keep the **strongest** signal

Average pooling: use the **local average** as the signal

Transfer Learning



Trained

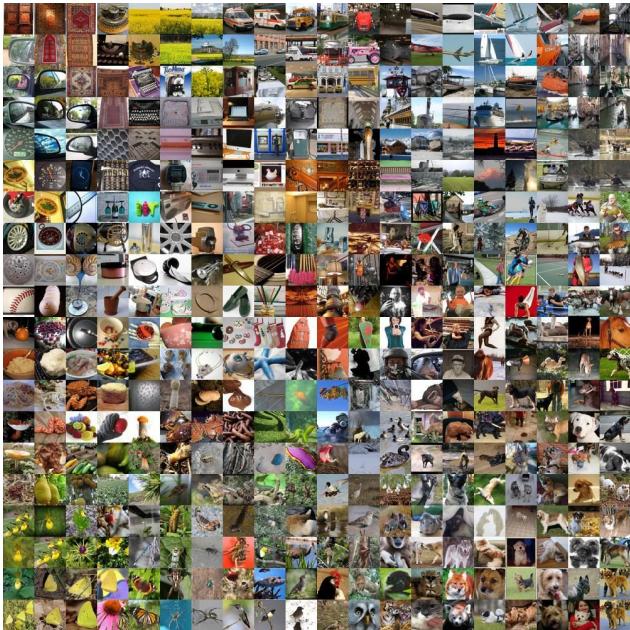


Continue
Training



CNN reuses its knowledge; its features

Transfer Learning



Trained



Continue
Training



CNN reuses its knowledge; its features

Generative Models

How do we generate images?

What I cannot build, I do not understand

Generative Models

Classification: mapping a distribution to a certain output

Generation: mapping a distribution to a certain output



Happy cat



Angry cat



Relaxed cat



"Cats"

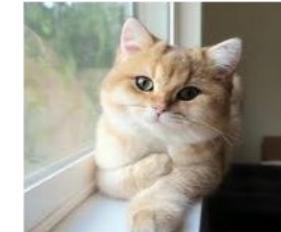
"Cats"



Happy cat

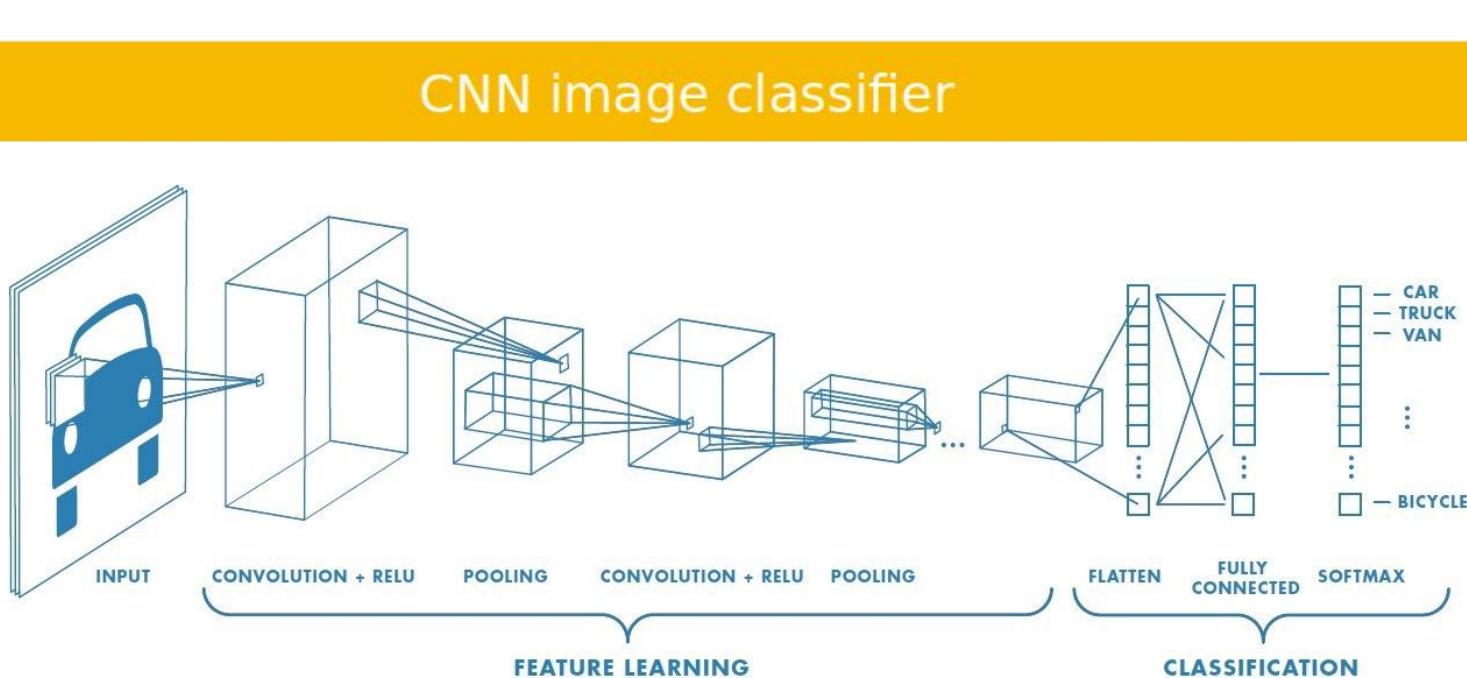


Angry cat



Relaxed cat

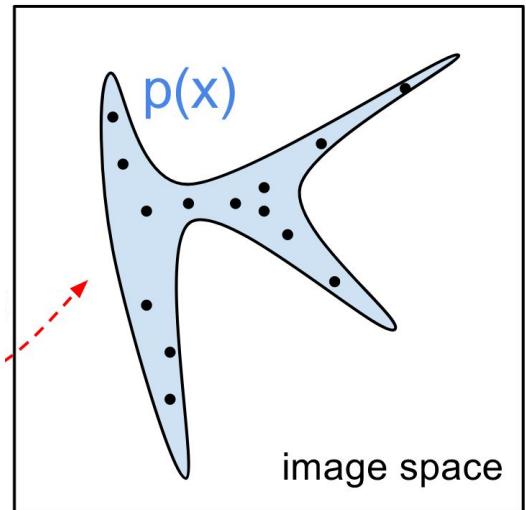
Generative Models



Generative models?

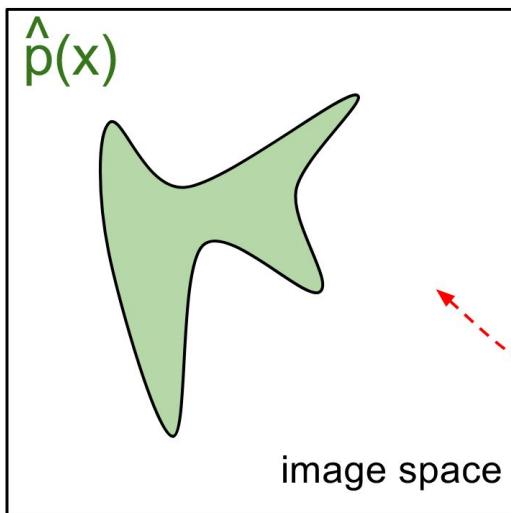
Generative Models

true data distribution

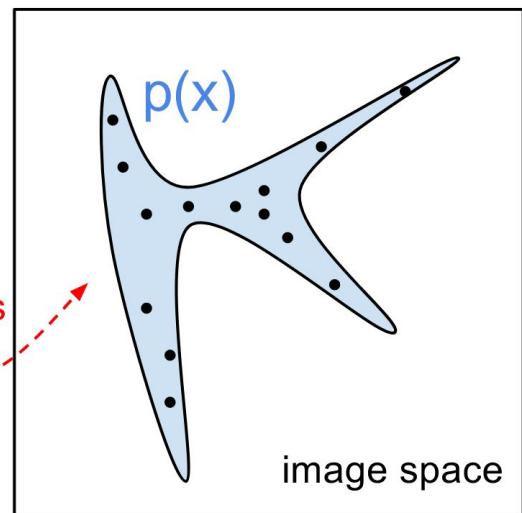


Generative Models

generated distribution



true data distribution



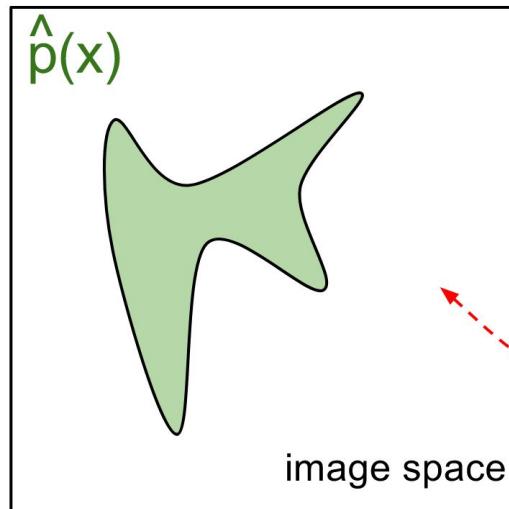
loss

Generative Models

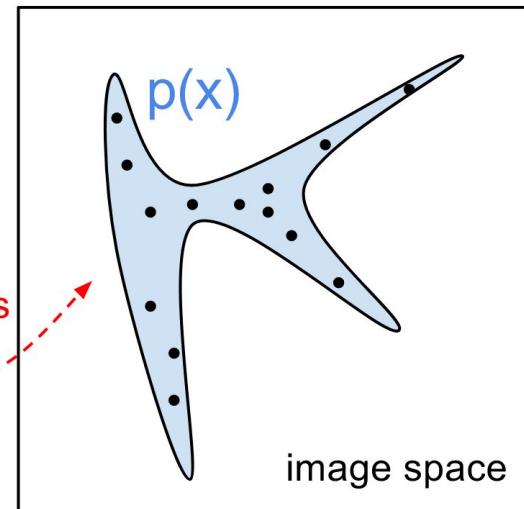
A generative model

1. Learns the data distribution instead
2. Allows uncertainties to be captures
3. Allows for (condition) density estimation

generated distribution

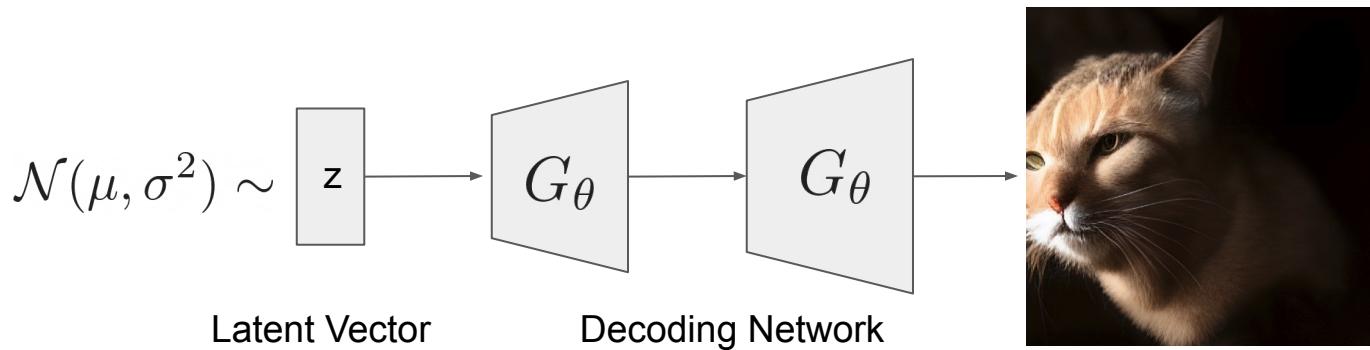


true data distribution



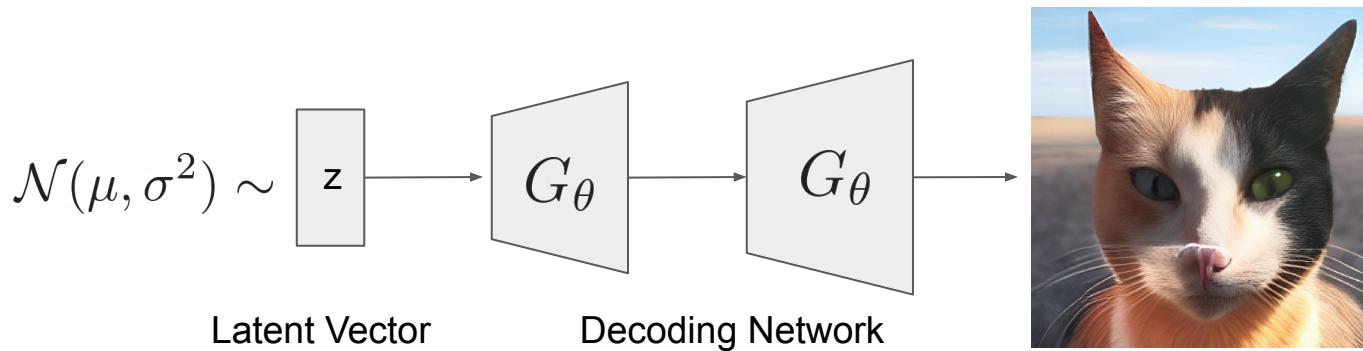
Generative Models

Generator



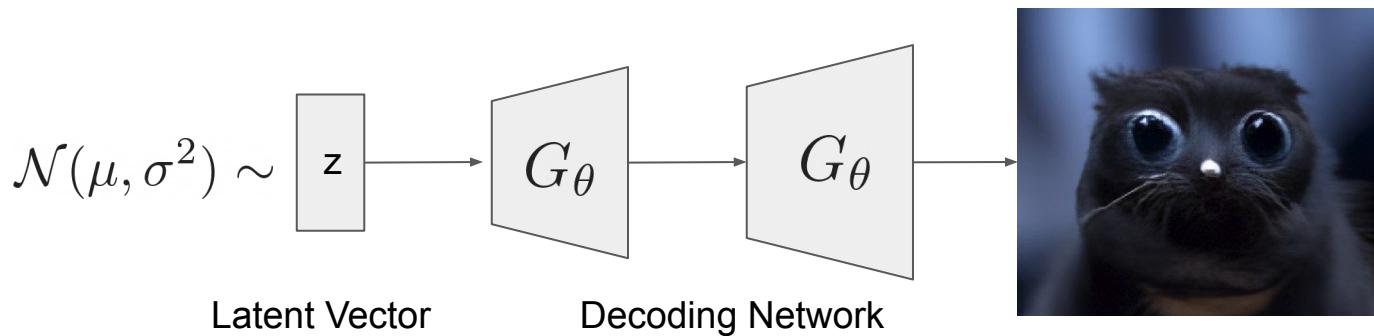
Generative Models

Generator



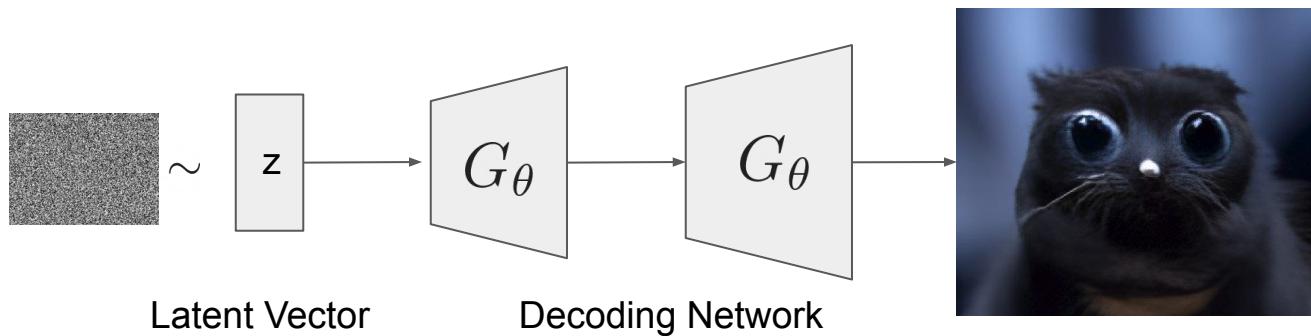
Generative Models

Generator

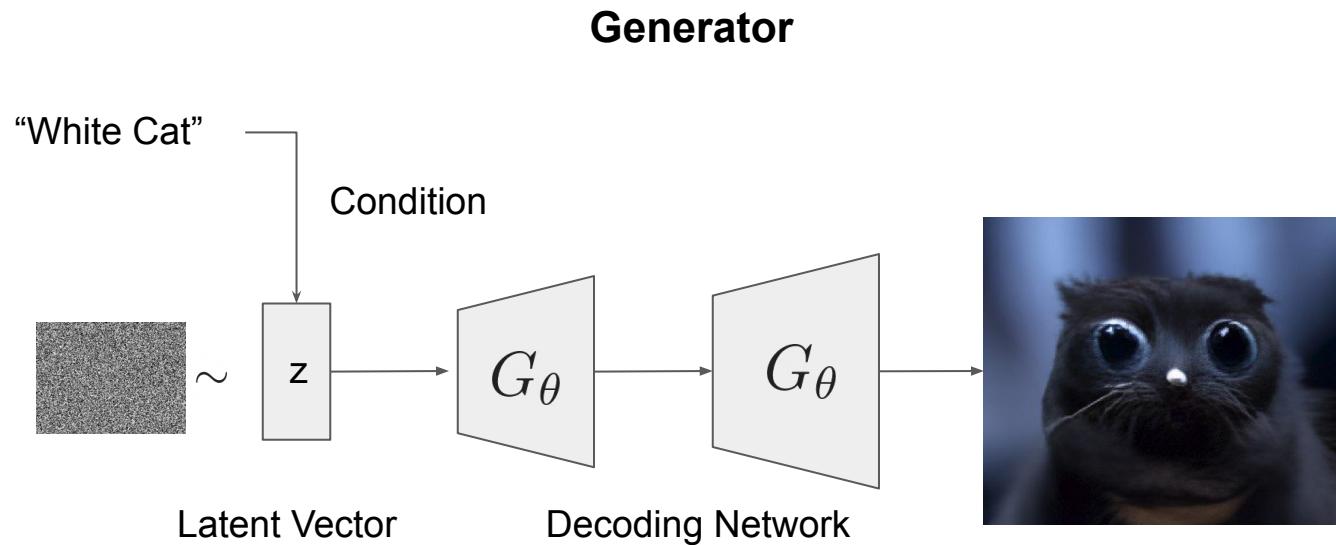


Generative Models

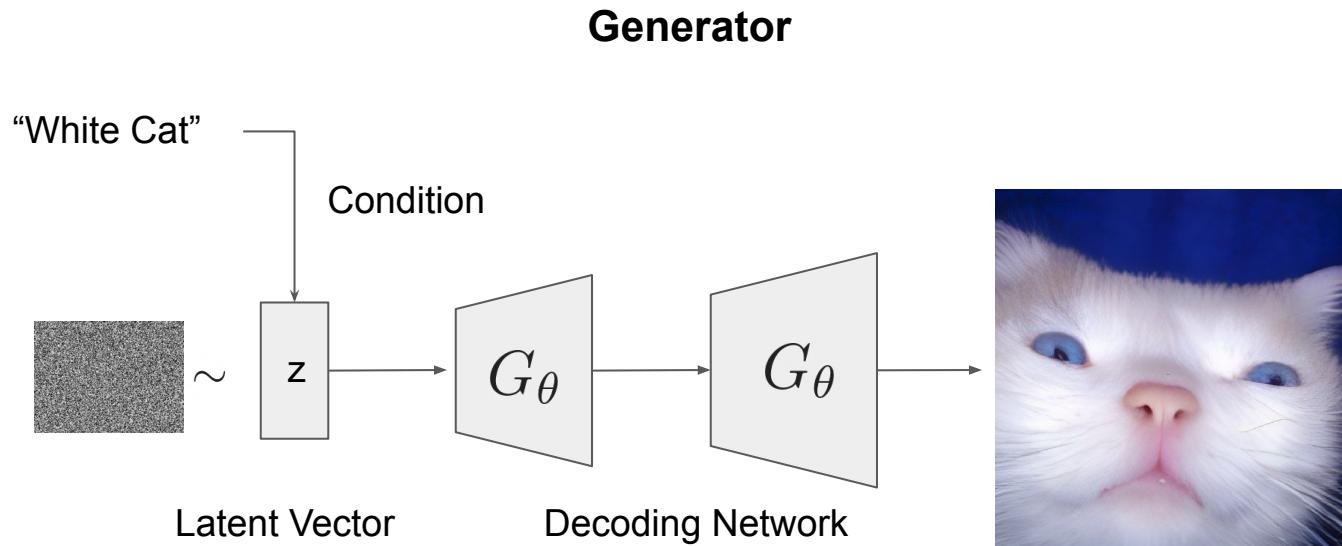
Generator



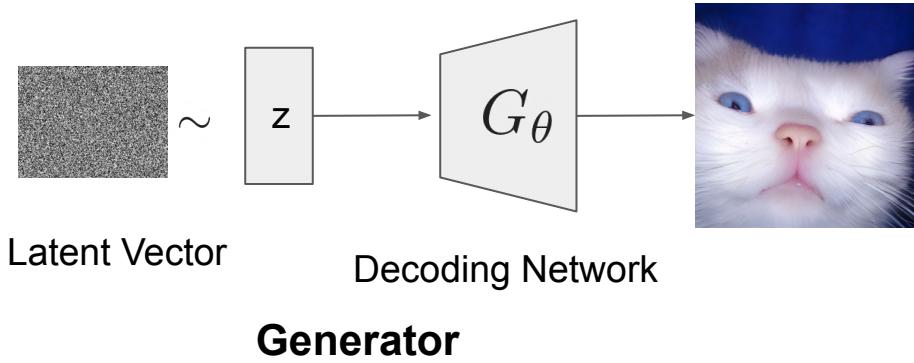
Generative Models



Generative Models

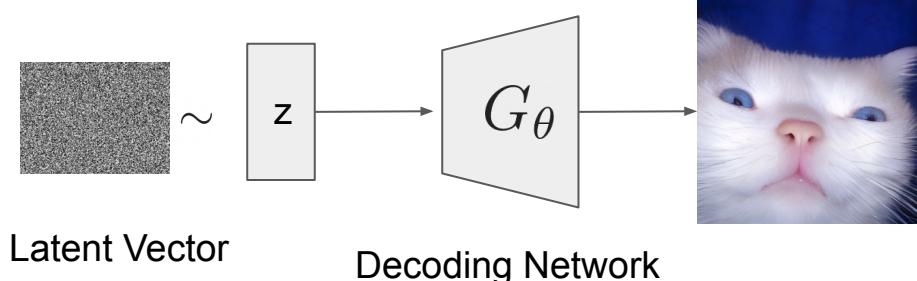


Adversarial Training



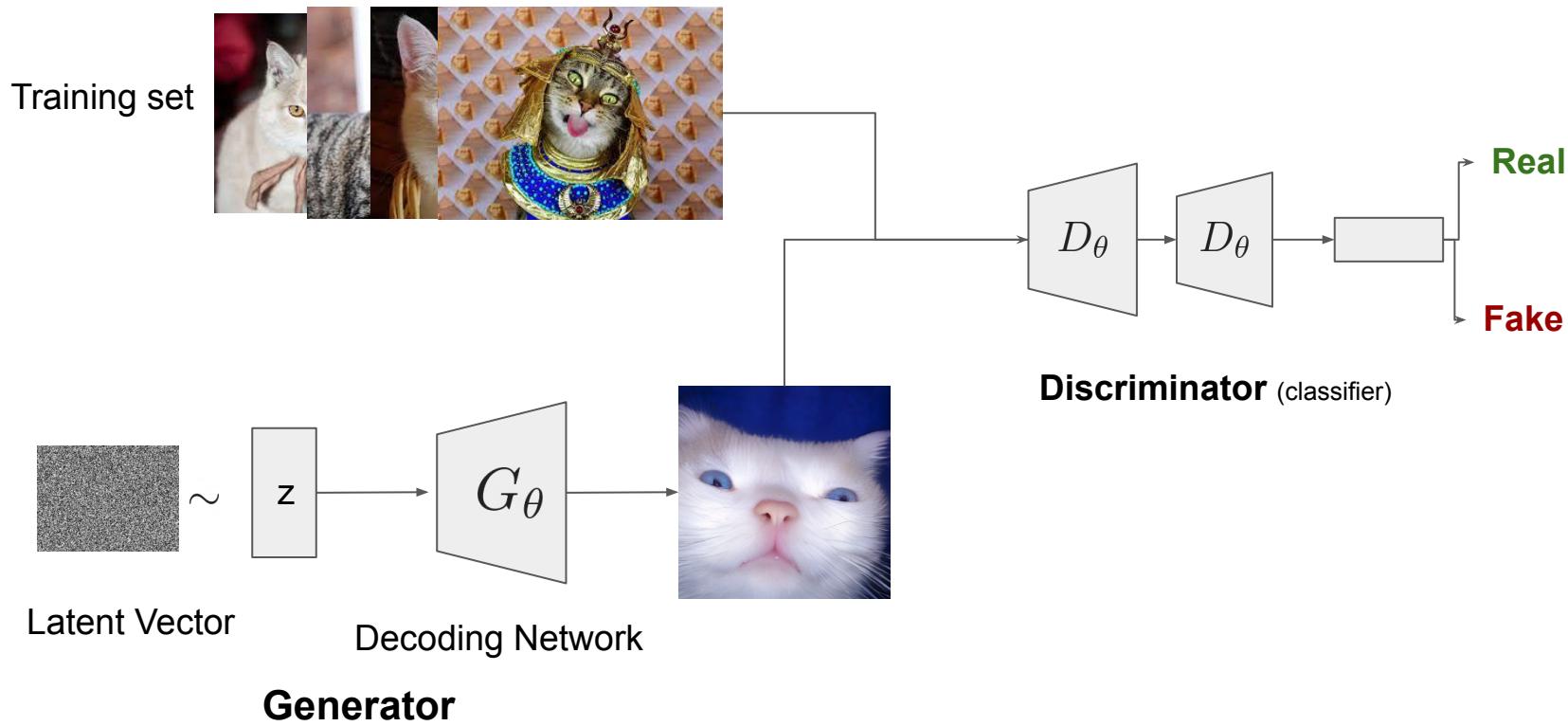
Adversarial Training

Training set

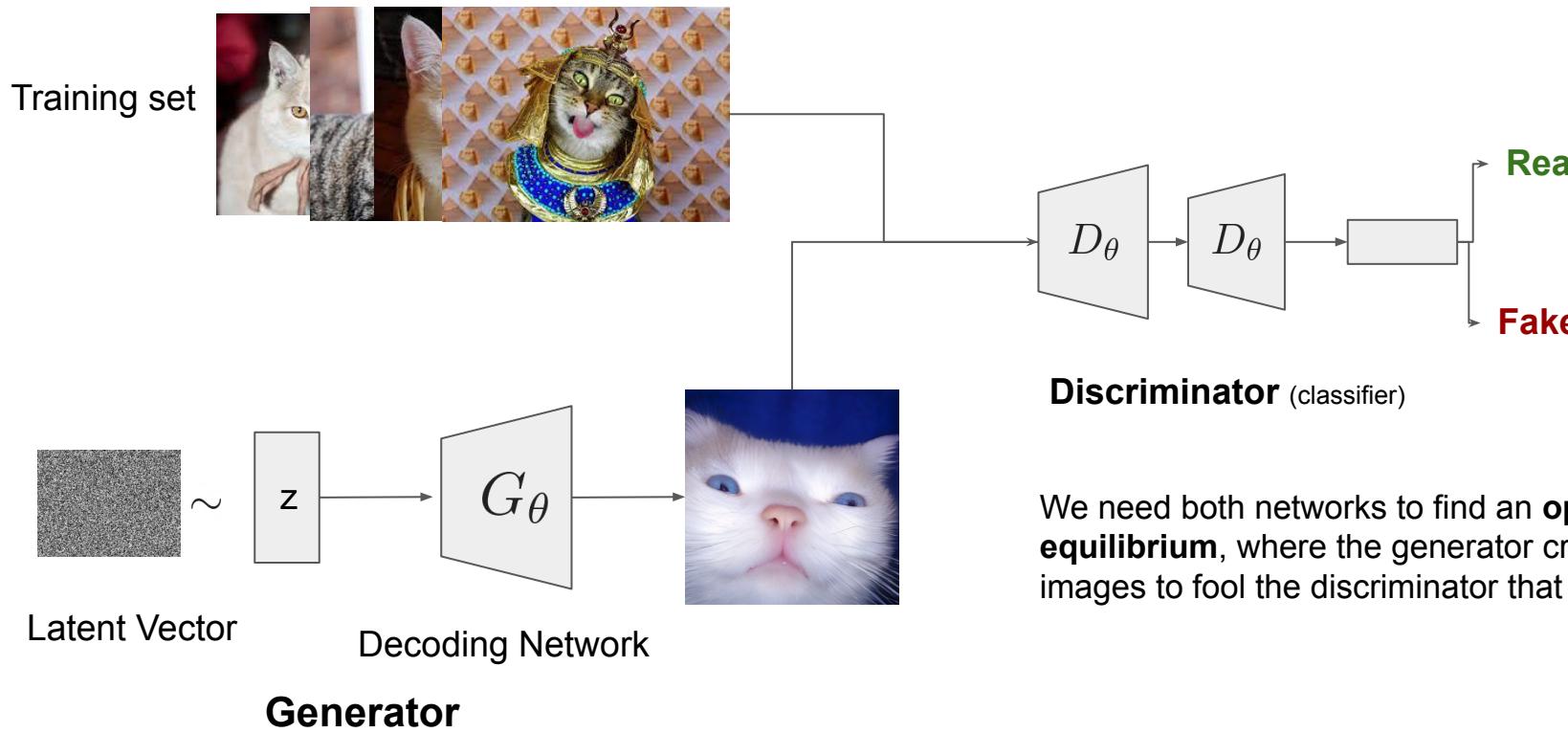


Generator

Adversarial Training

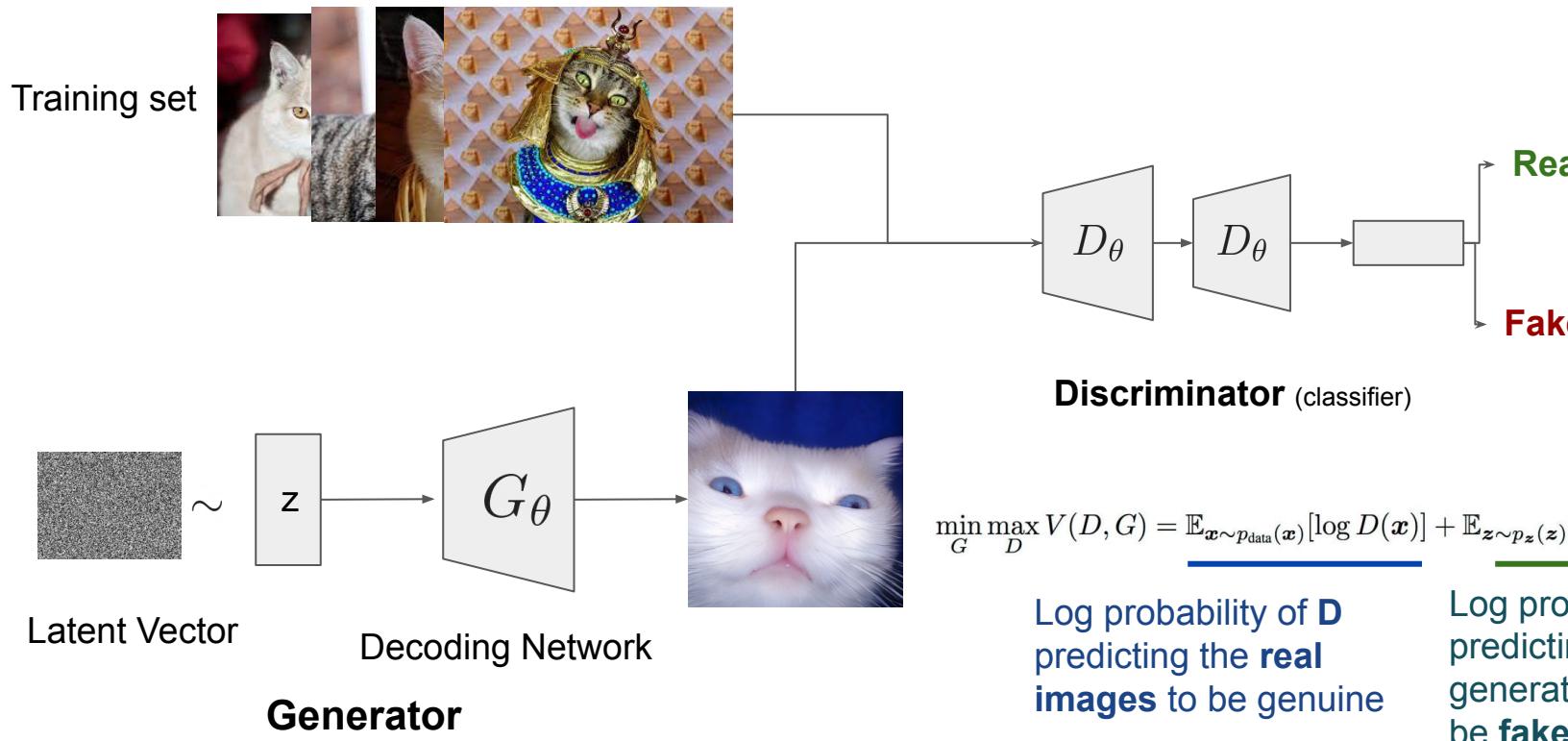


Adversarial Training



We need both networks to find an **optimal equilibrium**, where the generator creates good images to fool the discriminator that they are real

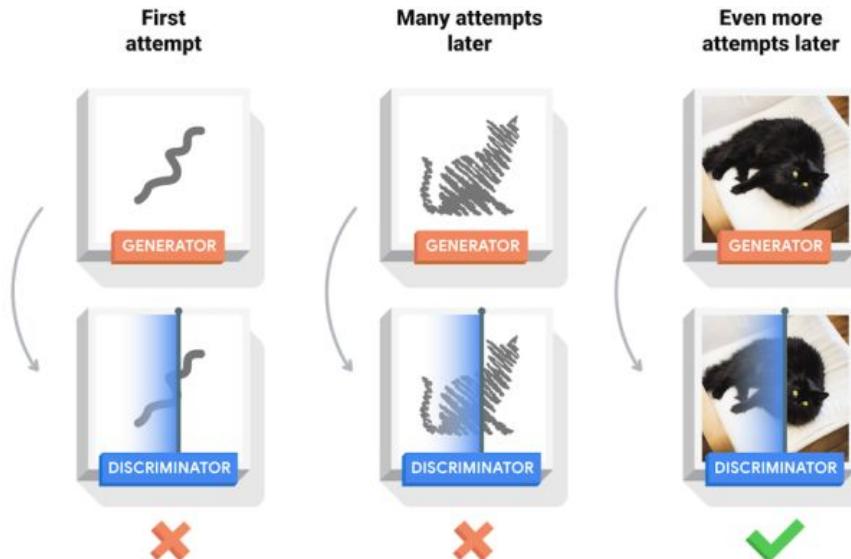
Adversarial Training



Adversarial Training

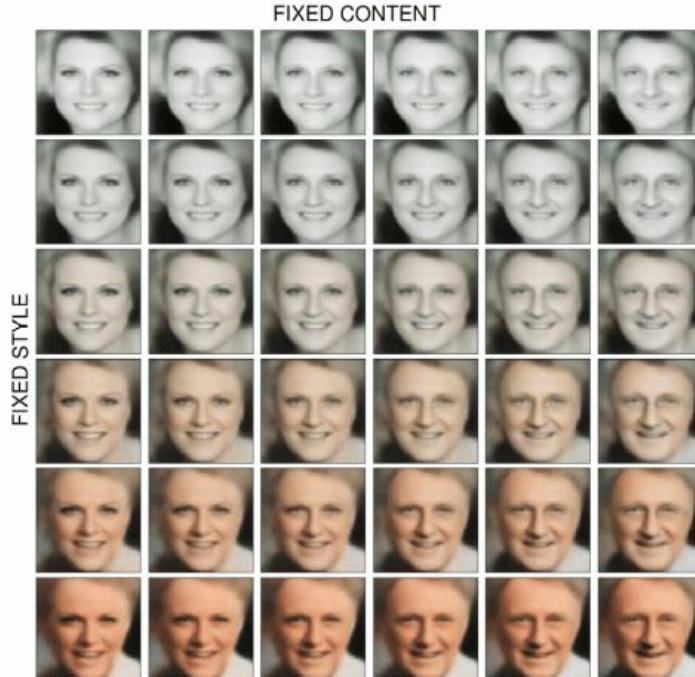
During the training:

- The **generator** progressively **becomes better** in creating images that looks real;
- The **discriminator** progressively **becomes better** in telling them apart.
- The process reaches an **equilibrium** when the discriminator can no longer distinguish real images from fakes.



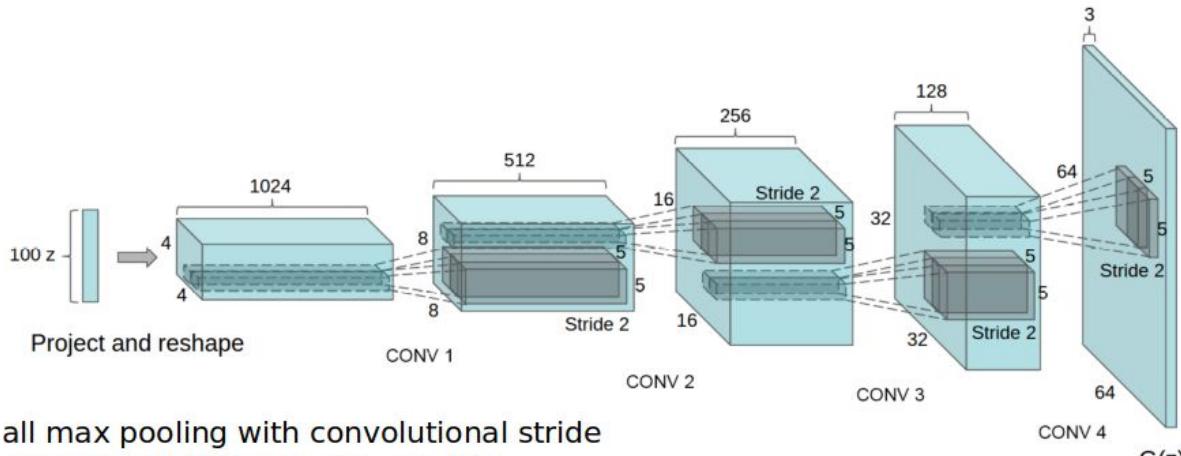
Latent Vector Interpolation

6 6 6 6 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
4 4 4 4 2 2 2 2 2 0 0 0 0 0 0 0 0 0 0 0 0 2
9 2 2 2 2 2 2 2 8 5 5 5 5 0 0 0 0 0 0 0 0 2
9 4 2 2 2 2 2 2 3 3 3 3 5 5 5 5 0 0 0 0 0 0 2
9 9 4 2 2 2 2 2 3 3 3 3 3 5 5 5 5 5 5 5 3 3
9 9 9 4 2 2 2 2 3 3 3 3 3 3 5 5 5 5 5 5 3 3
9 9 9 9 9 2 2 2 3 3 3 3 3 3 5 5 5 5 5 5 3 3
9 9 9 9 9 8 3 3 3 3 3 3 3 5 5 5 5 8 8 7
9 9 9 9 9 8 3 3 3 3 3 3 3 8 8 8 8 8 8 8 7
9 9 9 9 9 8 3 8 8 8 8 8 8 8 8 8 8 8 8 8 7
9 9 9 9 9 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 7
9 9 9 9 9 8 8 8 8 8 8 6 6 6 6 6 6 5 5 7
9 9 9 9 9 9 8 8 8 8 8 6 6 6 6 6 6 6 5 5 7
9 9 9 4 9 9 9 9 9 5 5 5 6 6 6 6 6 6 6 5 5 7
9 9 4 4 4 4 9 9 9 5 5 6 6 6 6 6 6 6 6 5 5 7
9 9 4 4 4 4 9 9 9 5 5 6 6 6 6 6 6 6 6 6 6 7
9 9 4 4 4 4 9 9 9 5 5 6 6 6 6 6 6 6 6 6 6 7
9 9 9 9 9 7 7 7 1 1 1 1 1 1 1 1 1 1 1 1 1 1
9 9 9 9 9 7 7 7 1 1 1 1 1 1 1 1 1 1 1 1 1 1



Case Study: DCGAN

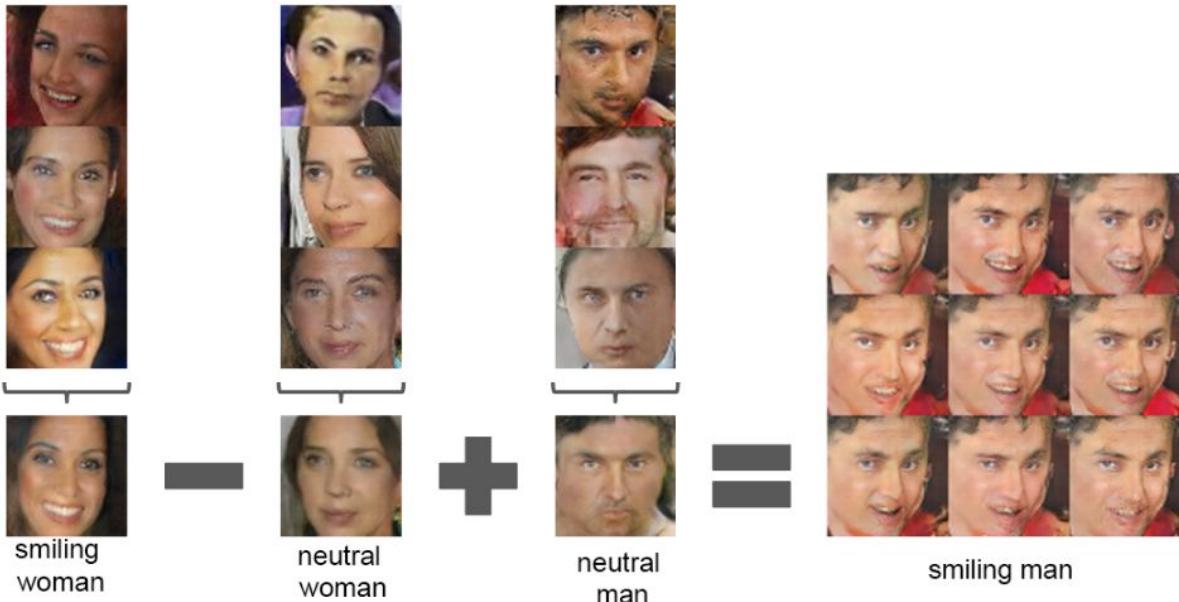
DCGAN: Deep Convolutional Generative Adversarial Network



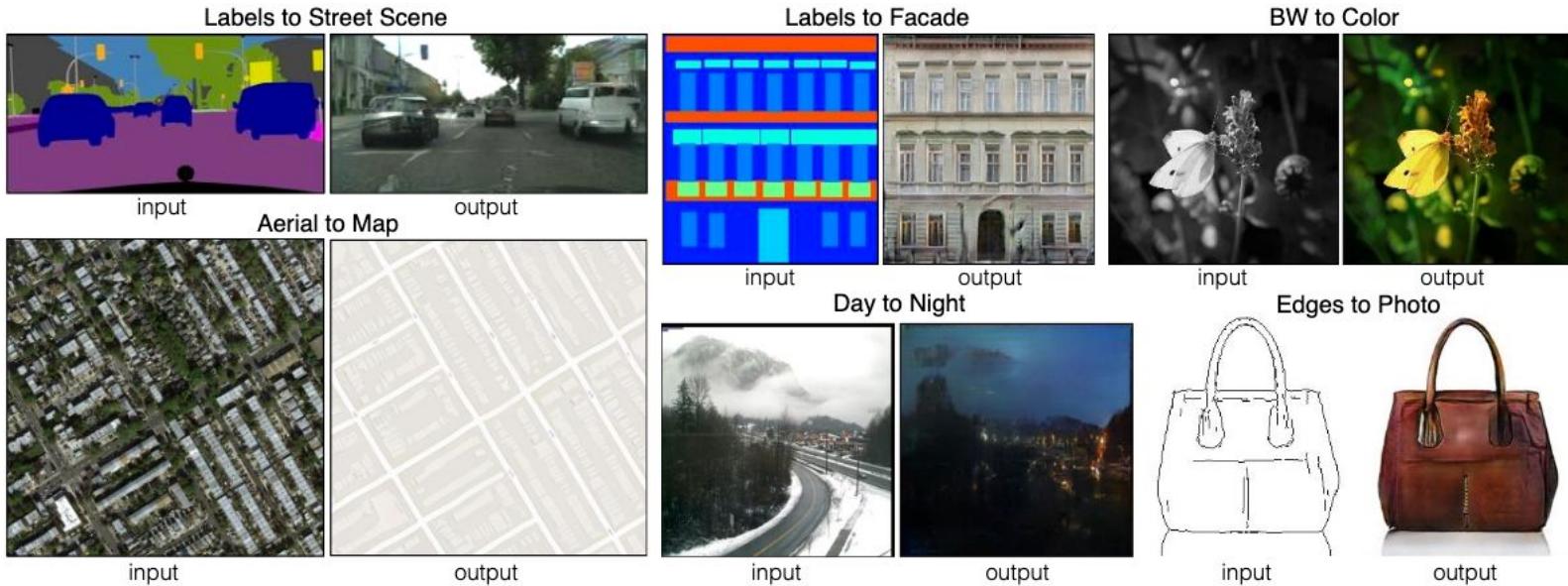
- Replaces all **max pooling** with convolutional stride
- Use transposed convolution for upsampling
- Eliminate fully connected layers
- Use batch normalization except the output layer for the generator
- Use batch normalization for the input layer of the discriminator
- Use ReLU in the generator except for the output (which uses tanh)
- Use LeakyReLU in the discriminator.

Case Study: DCGAN

Latent Space Arithmetic



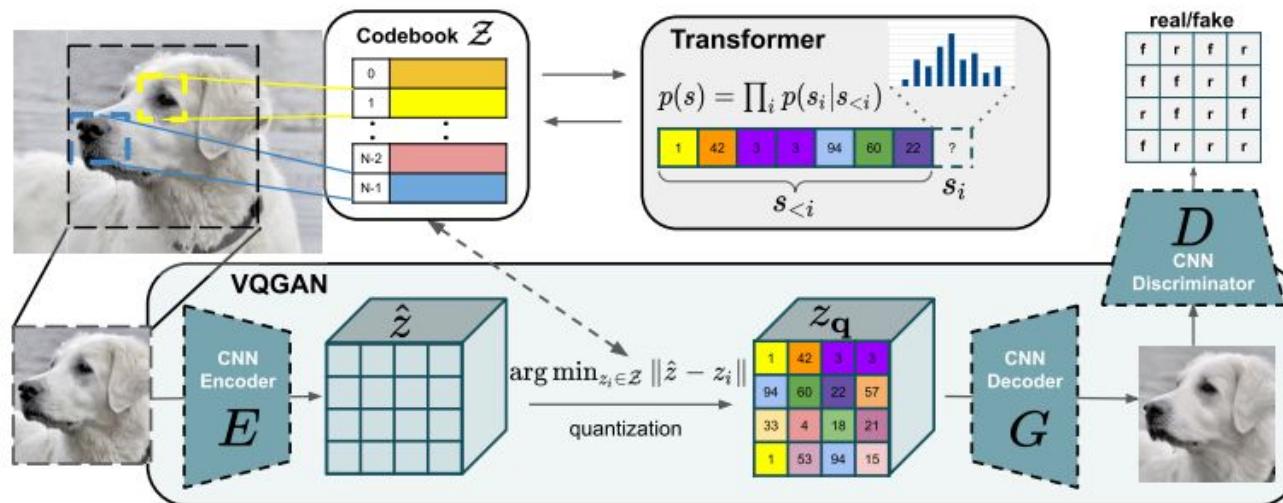
Case Study: CGAN



Quality of GAN-Generated images



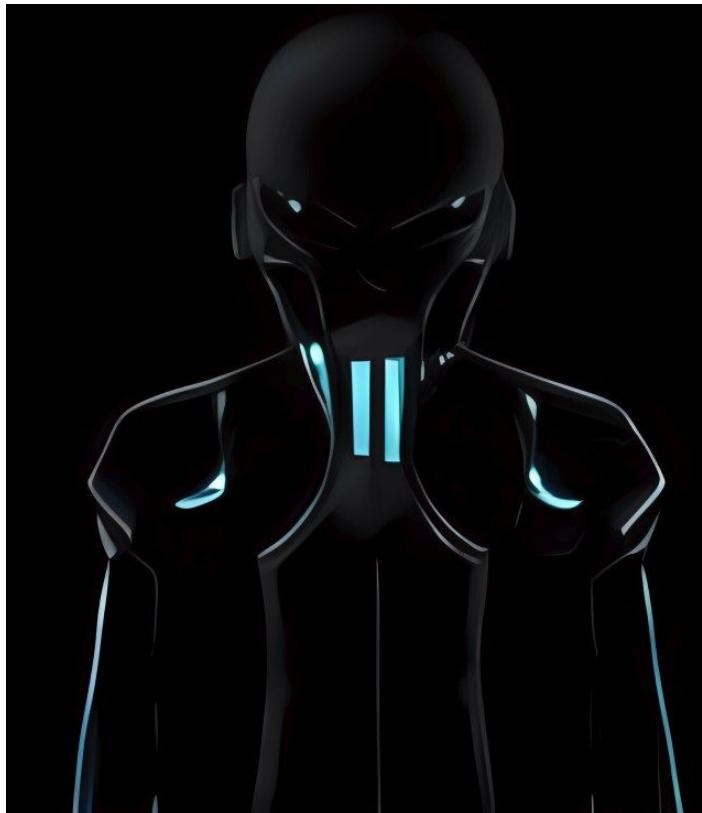
Quality of GAN-Generated images



Quality of GAN-Generated images



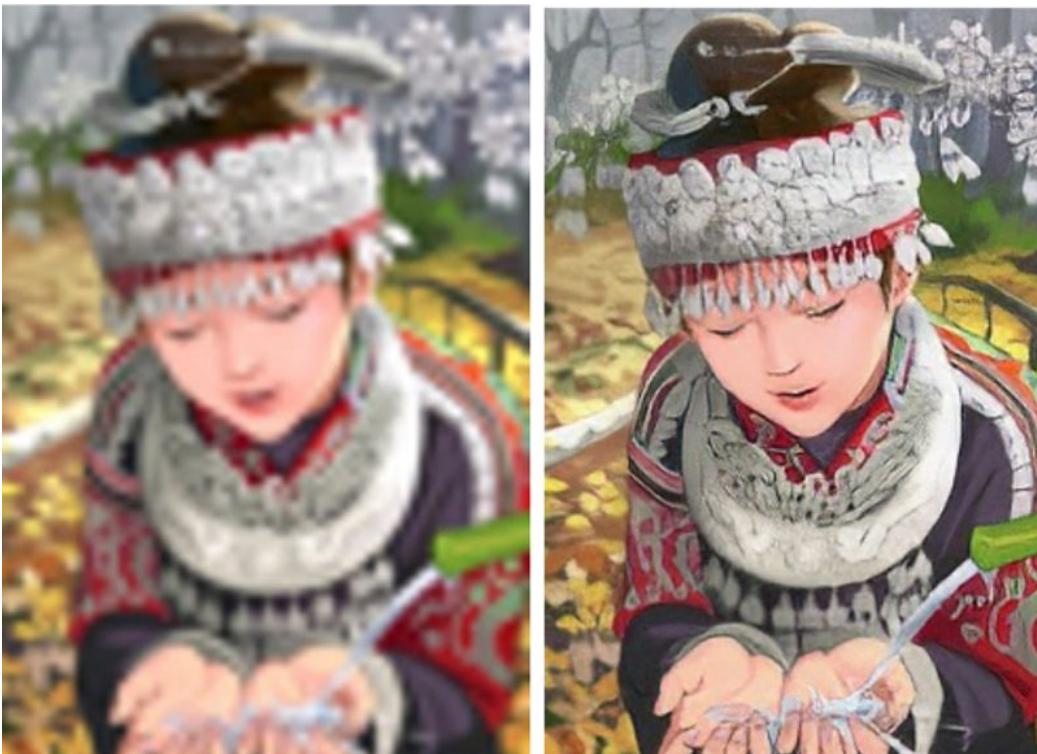
Quality of GAN-Generated images



Quality of GAN-Generated images



Super Resolution



Change Style: Young to Old



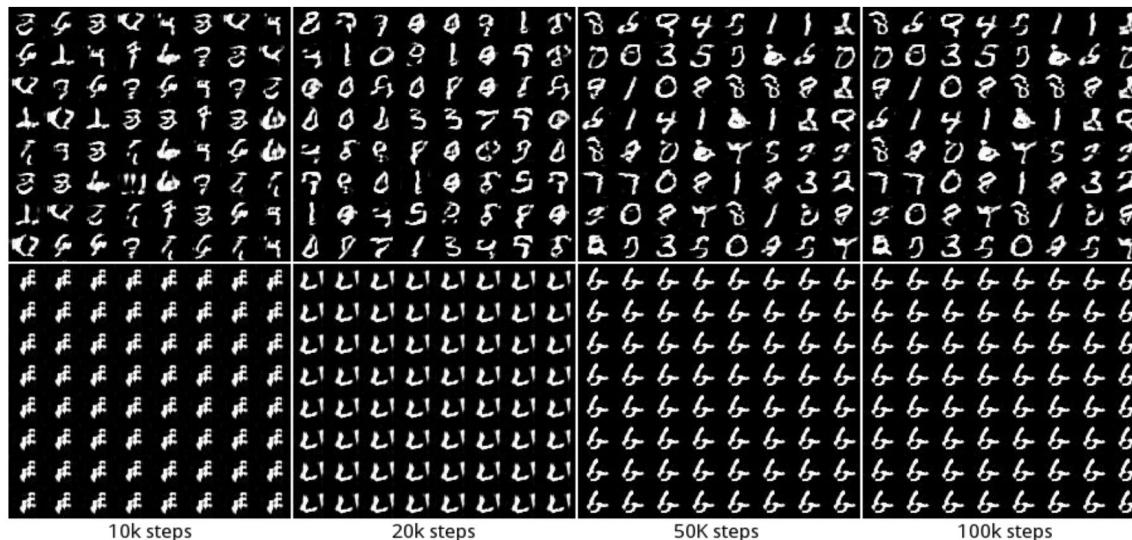
<https://yuval-alaluf.github.io/SAM/>

Problems with GANs

Convergence is not guaranteed: Learning is unstable due to adversarial nature -> no convergence

Mode Collapse: Limited variation learnt

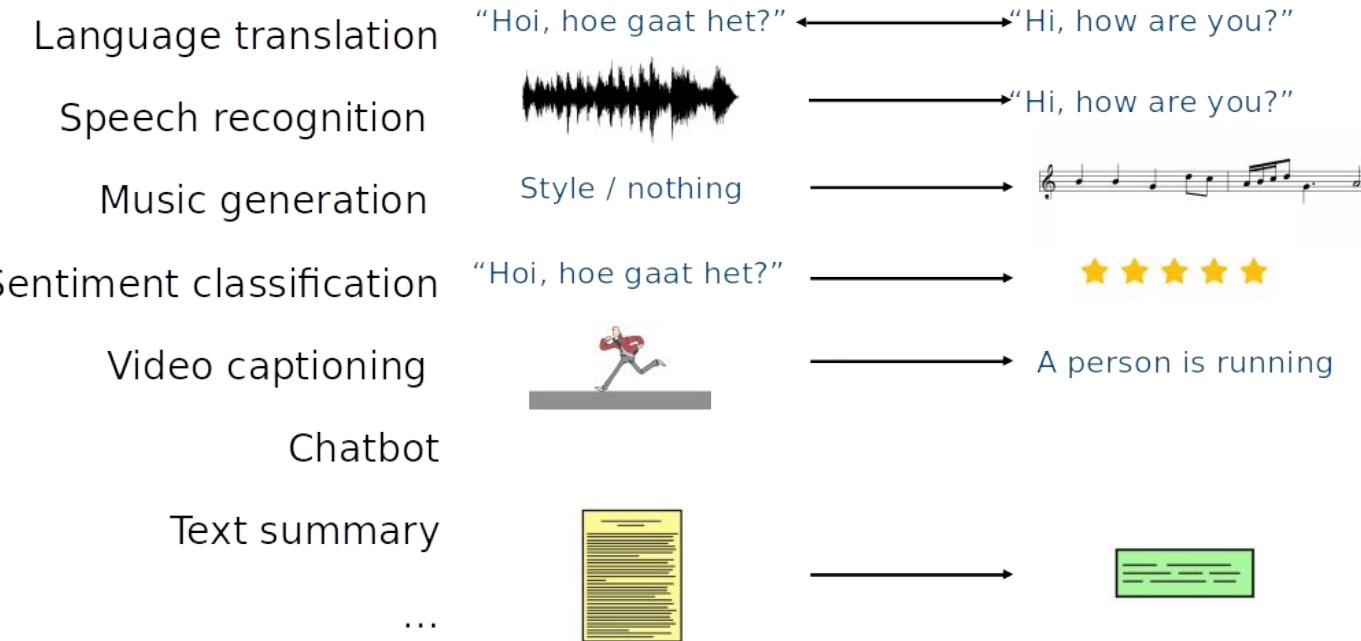
Diminished Gradient: The discriminator is too good telling fake from real \rightarrow the generator learns nothing



Modelling Sequences

How to work with data that has a temporal dimension?

Modelling Sequences



Sequential Data

Sequential Data are represented with 1d or 2d arrays

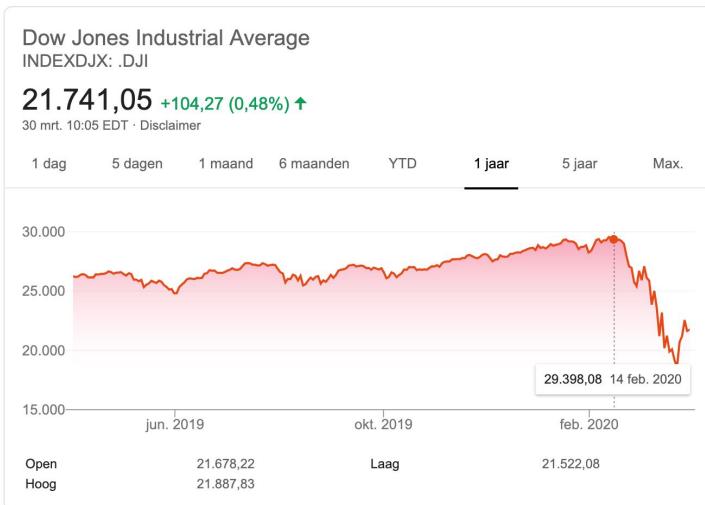


Time	24/3	25/3	26/3	27/3	28/3	...
Value	19,722	21,050	21,188	22,532	21,678	...

Sequential Data

Sequential Data are represented with 1d or 2d arrays

“The cat is black.”



Value	42	82	16	4	0	...
-------	----	----	----	---	---	-----

Time	24/3	25/3	26/3	27/3	28/3	...
Value	19,722	21,050	21,188	22,532	21,678	...

Sequential Data

Sequential Data are represented with 1d or 2d arrays

“The cat is black.”



Value	42	82	16	4	0	...
-------	----	----	----	---	---	-----

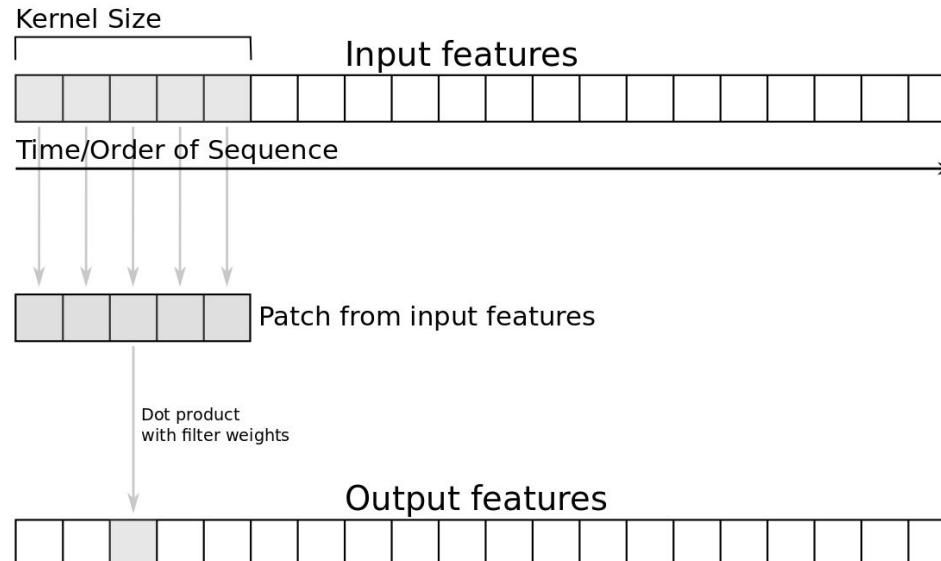
In Natural Language Processing (NLP)

- Words converted into tokens
- Token is represented numerically
- Sentence is a sequence of tokens
- Tokens may be logically related to each other

Time	24/3	25/3	26/3	27/3	28/3	...
Value	19,722	21,050	21,188	22,532	21,678	...

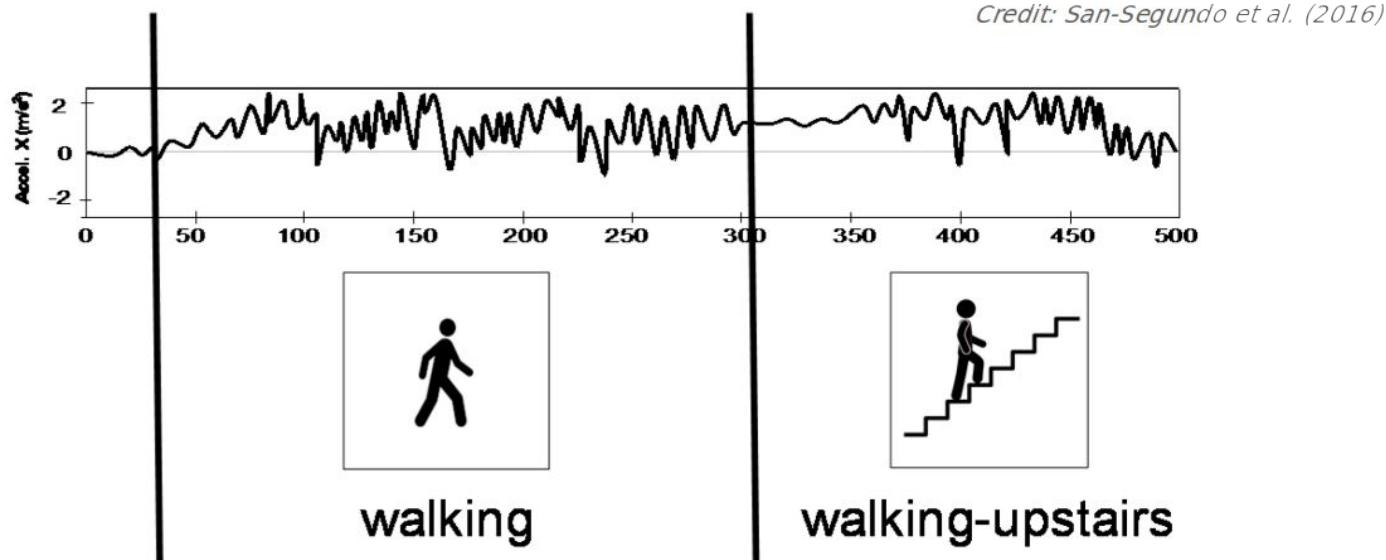
Sequence modelling with 1D convolutions

- CNNs work here because they take into account **nearby** tokens



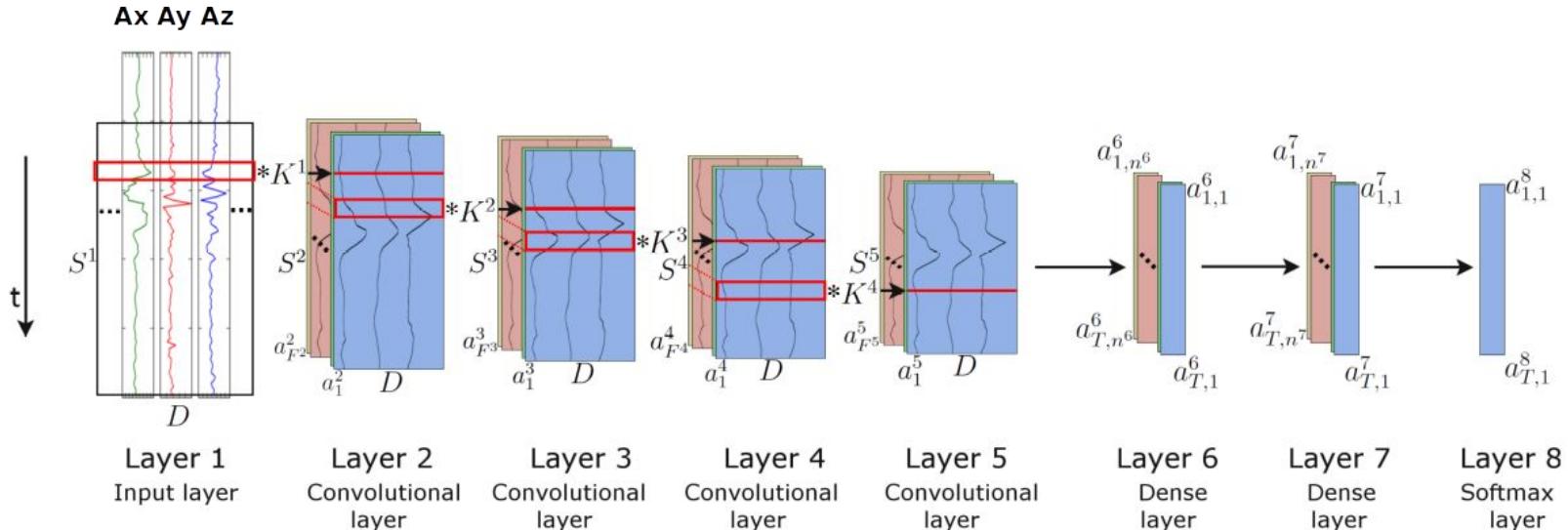
Human Activity Modelling with CNNs

- CNNs work here because they take into account **nearby** tokens

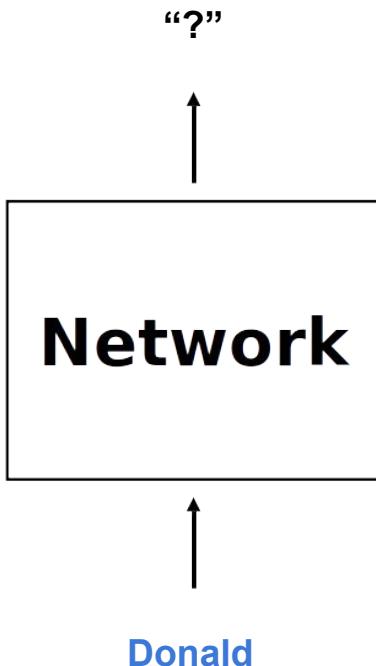


The DNN can be trained with **existing** open datasets!
e.g., <https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones>

Human Activity Modelling with CNNs

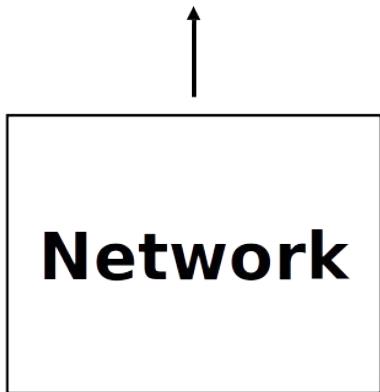


Natural Language Processing



Natural Language Processing

“?”



How about modeling just the next word?

*The president of the United States is Donald Trump.
Wherever Donald Trump goes, security is tight.*

The comic Donald Duck is very famous.

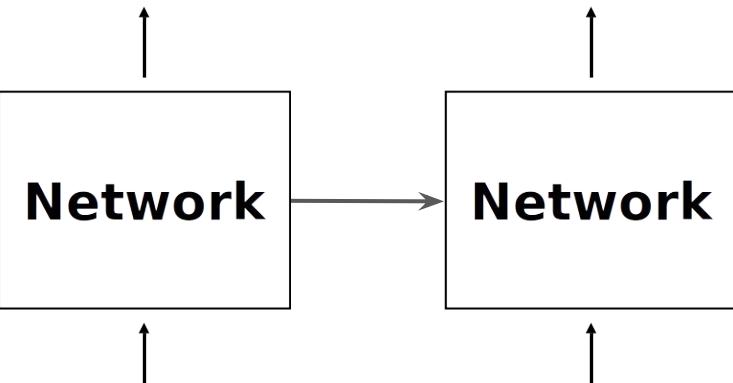
Context matters to the prediction!

Donald

Natural Language Processing

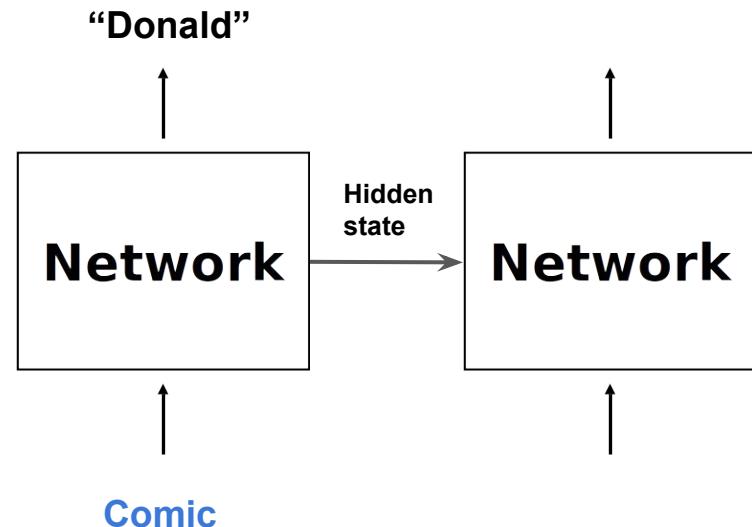
“Donald”

The meaning of the sentence depends on the
context



Comic

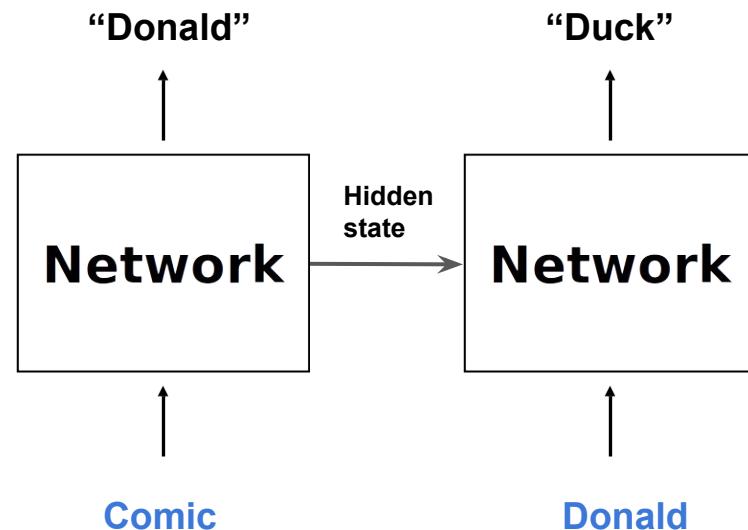
Natural Language Processing



The meaning of the sentence depends on the **context**

The context is propagated to the next time step via the **hidden state**.

Natural Language Processing

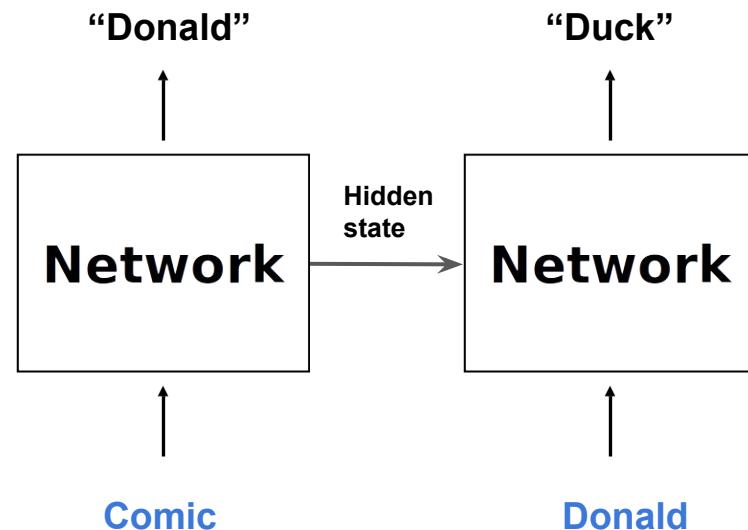


The meaning of the sentence depends on the **context**

The context is propagated to the next time step via the **hidden state**.

The next prediction is then modelled with both the hidden state and the next word

Natural Language Processing

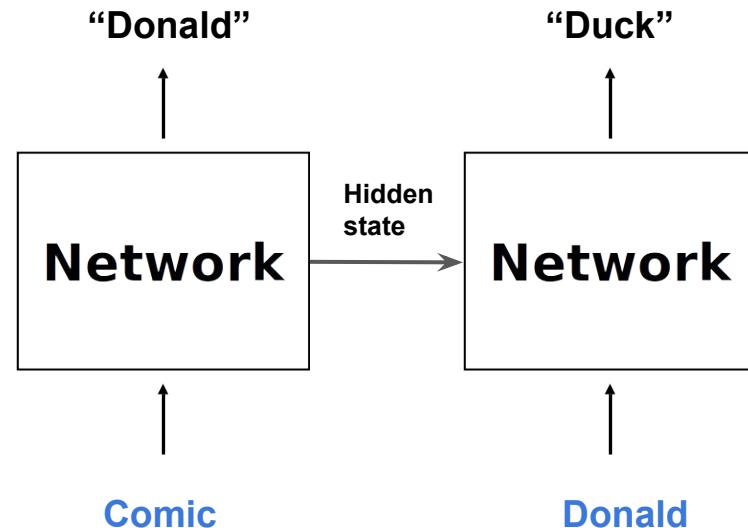


The meaning of the sentence depends on the **context**

The context is propagated to the next time step via the **hidden state**.

The next prediction is then modelled with both the hidden state and the next word

Natural Language Processing



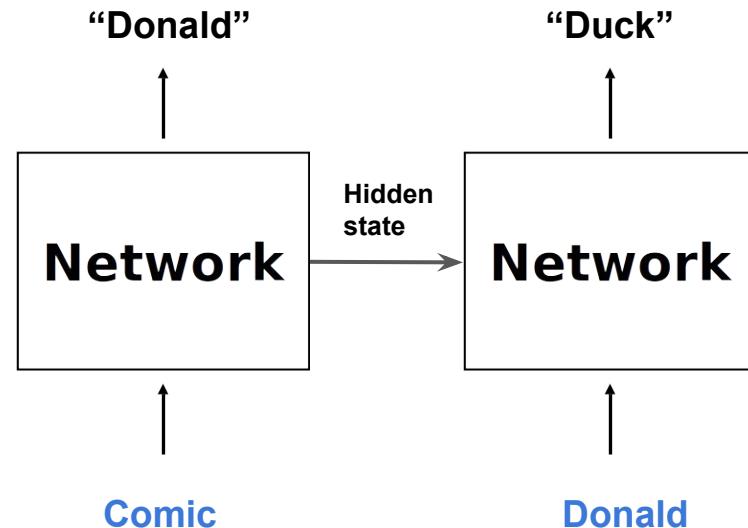
The meaning of the sentence depends on the **context**

The context is propagated to the next time step via the **hidden state**.

The next prediction is then modelled with both the hidden state and the next word

The hidden state allows the network to have **memory**

Natural Language Processing



The meaning of the sentence depends on the **context**

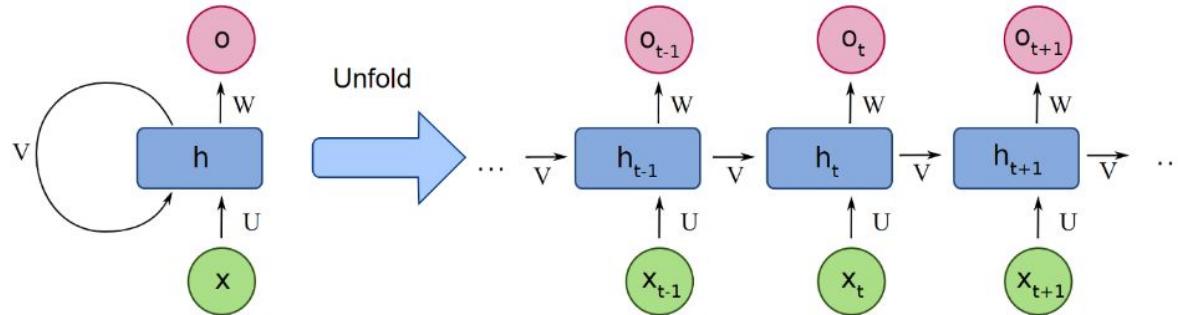
The context is propagated to the next time step via the **hidden state**.

The next prediction is then modelled with both the hidden state and the next word

The hidden state allows the network to have **memory**

Recurrent Neural Network

In RNNs, weights are reused across time steps



RNN

$$h_t = \sigma(U x_t + V h_{t-1} + b_h)$$

$$O_t = \sigma(W h_t + b_o)$$

Back propagation through **time**

Non-RNN

$$O_t = \sigma(W x_t + b)$$

Back propagation through **layers**

Long-term Dependency

Sometimes, the contexts are far away

The clouds are in the **sky**

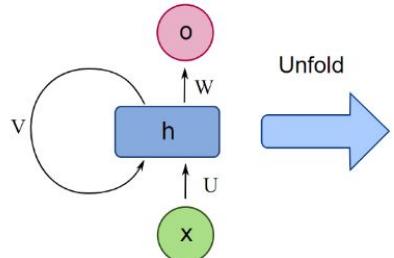
I grew up in The Netherlands ... I speak fluent **Dutch**

Long-term Dependency

Sometimes, the contexts are far away

The clouds are in the **sky**

I grew up in The Netherlands ... I speak fluent **Dutch**

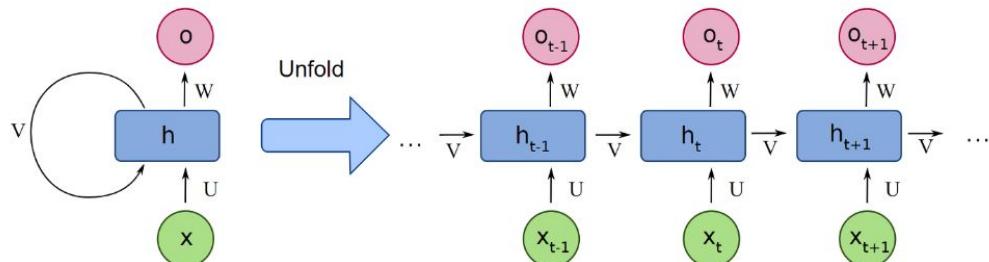


Long-term Dependency

Sometimes, the contexts are far away

The clouds are in the **sky**

I grew up in The Netherlands ... I speak fluent **Dutch**

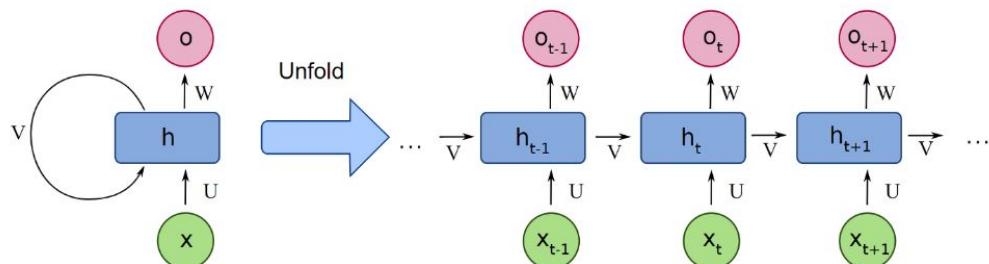


Long-term Dependency

Sometimes, the contexts are far away

The clouds are in the **sky**

I grew up in The Netherlands ... I speak fluent **Dutch**



$$\mathbf{x}_t = F(\mathbf{x}_{t-1}, \mathbf{u}_t, \theta) \quad (1)$$

$$\mathbf{x}_t = \mathbf{W}_{rec}\sigma(\mathbf{x}_{t-1}) + \mathbf{W}_{in}\mathbf{u}_t + \mathbf{b} \quad (2)$$

$$\frac{\partial \mathcal{E}}{\partial \theta} = \sum_{1 \leq t \leq T} \frac{\partial \mathcal{E}_t}{\partial \theta} \quad (3)$$

$$\frac{\partial \mathcal{E}_t}{\partial \theta} = \sum_{1 \leq k \leq t} \left(\frac{\partial \mathcal{E}_t}{\partial \mathbf{x}_t} \frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} \frac{\partial^+ \mathbf{x}_k}{\partial \theta} \right) \quad (4)$$

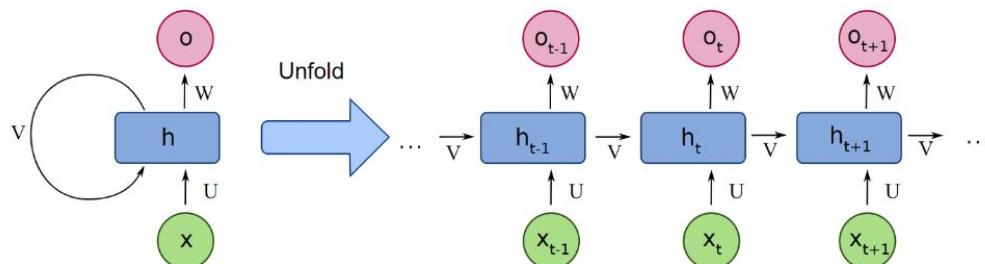
$$\frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} = \prod_{t \geq i > k} \frac{\partial \mathbf{x}_i}{\partial \mathbf{x}_{i-1}} = \prod_{t \geq i > k} \mathbf{W}_{rec}^T diag(\sigma'(\mathbf{x}_{i-1})) \quad (5)$$

Long-term Dependency

Sometimes, the contexts are far away

The clouds are in the **sky**

I grew up in The Netherlands ... I speak fluent **Dutch**



If $\mathbf{W}_{rec}^T diag(\sigma'(\mathbf{x}_{i-1}))$ is < 1 , in a long product, gradients

start to vanish

$$\mathbf{x}_t = F(\mathbf{x}_{t-1}, \mathbf{u}_t, \theta) \quad (1)$$

$$\mathbf{x}_t = \mathbf{W}_{rec}\sigma(\mathbf{x}_{t-1}) + \mathbf{W}_{in}\mathbf{u}_t + \mathbf{b} \quad (2)$$

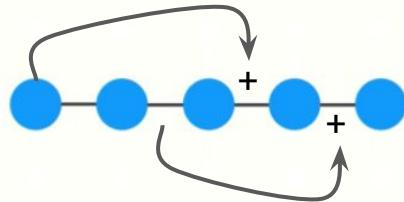
$$\frac{\partial \mathcal{E}}{\partial \theta} = \sum_{1 \leq t \leq T} \frac{\partial \mathcal{E}_t}{\partial \theta} \quad (3)$$

$$\frac{\partial \mathcal{E}_t}{\partial \theta} = \sum_{1 \leq k \leq t} \left(\frac{\partial \mathcal{E}_t}{\partial \mathbf{x}_t} \frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} \frac{\partial \mathbf{x}_k}{\partial \theta} \right) \quad (4)$$

$$\frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} = \prod_{t \geq i > k} \frac{\partial \mathbf{x}_i}{\partial \mathbf{x}_{i-1}} = \prod_{t \geq i > k} \mathbf{W}_{rec}^T diag(\sigma'(\mathbf{x}_{i-1})) \quad (5)$$

Long Short-term Memory

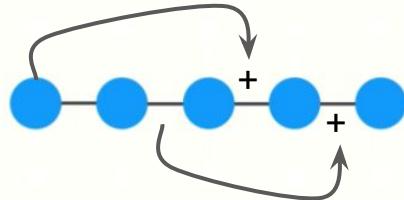
Residual connection is a general solution to the vanishing gradient problem.



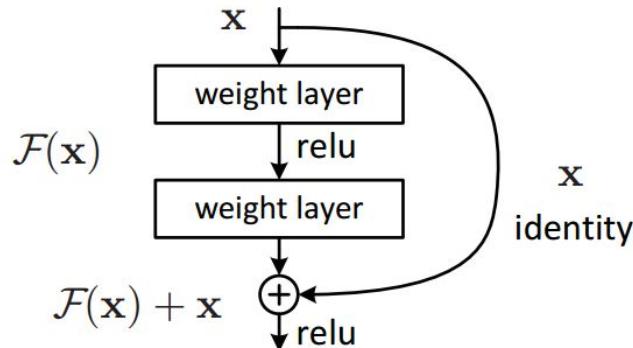
However, for very long dependency (which is common in RNNs), this solution is not sufficient. We should instead regulate which information to **keep** and which to **forget**.

Long Short-term Memory

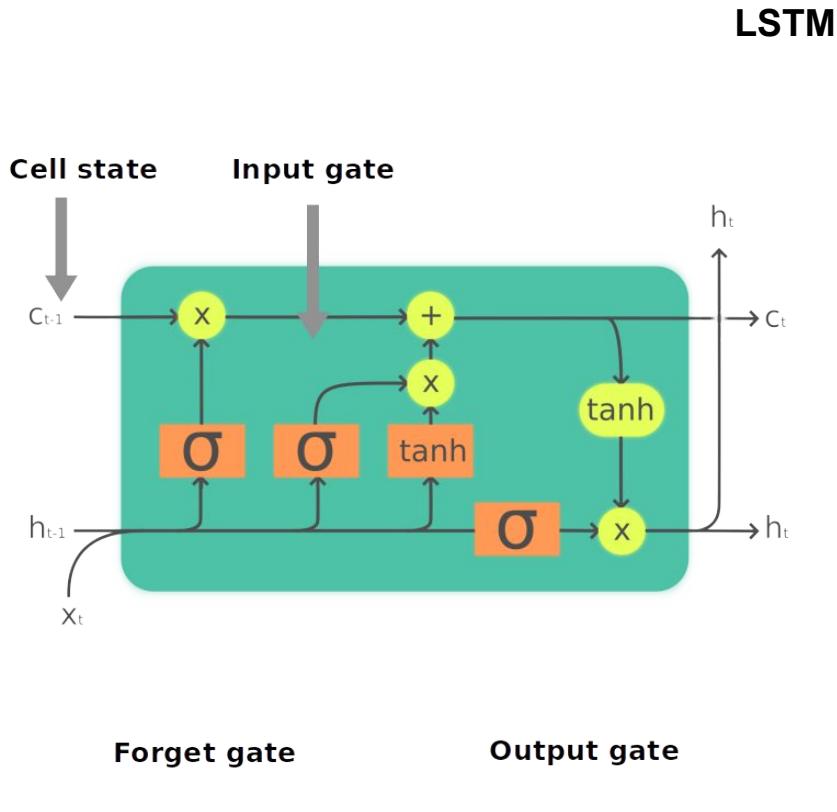
Residual connection is a general solution to the vanishing gradient problem.



However, for very long dependency (which is common in RNNs), this solution is not sufficient. We should instead regulate which information to **keep** and which to **forget**.



Long Short-term Memory



LSTM regulates the flow of information through gates.

- **Input gate**: controls the extent to which a new value flows into the cell;
- **Forget gate**: controls the extent a value remains in the cell;
- **Output gate**: controls the extent to which the value in the cell is used to calculate the output activation.

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$$

$$c_t = \sigma_h(W_c x_t + U_c h_{t-1} + b_c)$$

$$h_t = f_t \circ c_{t-1} + i_t \circ c_t$$
$$h_t = o_t \circ \sigma_h(c_t)$$

Sequence Modeling

LSTM

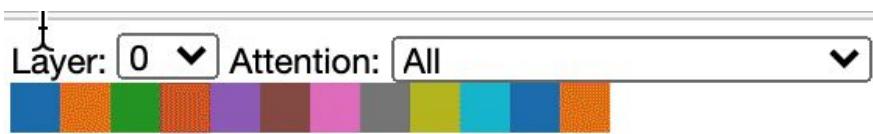
BISHOP OF ELY:

Marry, and will, my lord, to weep in such a one were prettiest;
Yet now I was adopted heir
Of the world's lamentable day,
To watch the next way with his father with his face?

ESCALUS:

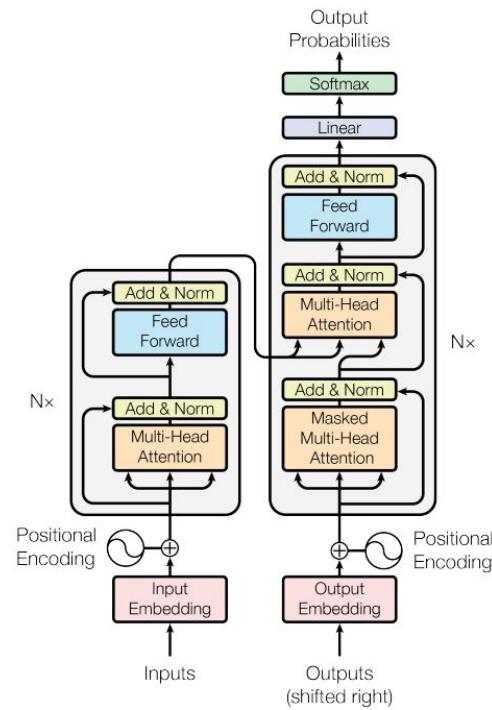
The cause why then we are all resolved more sons.

Self-Attention and Transformers



[CLS] the rabbit quickly hopped [SEP] the turtle slowly crawled [SEP]

[CLS] the rabbit quickly hopped [SEP] the turtle slowly crawled [SEP]



Sequence Modeling

Transformer: GPT-3

The unicorns told the scientists that they had left the valley to explore the world, but that after years of traveling the world, they found humans to be incredibly stupid and decided to return to the valley to live in peace.

“We were shocked”, said Dr. St. Maurice. “We had no idea that the unicorns had been waiting for us. We also had no idea that the unicorns were capable of speech. It was a truly magical moment.”

<https://www.buildgpt3.com/post/88/>

<https://openai.com/api/>

Final Notes

CNNs allow us to use images in machine learning in an efficient way

Classifiers want to extract feature for estimation

Generative models want to learn the distribution of the data

CNNs can be used for sequence modelling

RNNs are better than CNNs for NLP

LSTMs solves an inherent problem to RNNs

There is so much more...

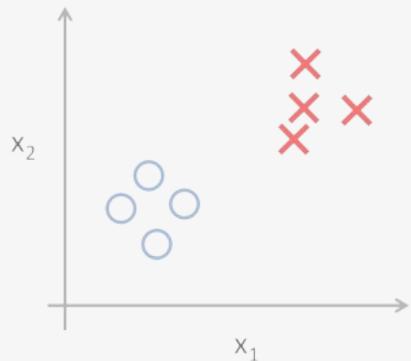
The END

bryan.cardenasguevara@surf.nl
bryca@pm.me

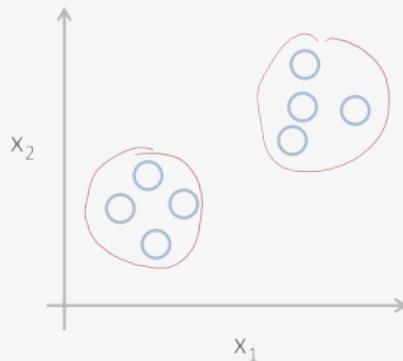


Reinforcement Learning

Supervised



Unsupervised

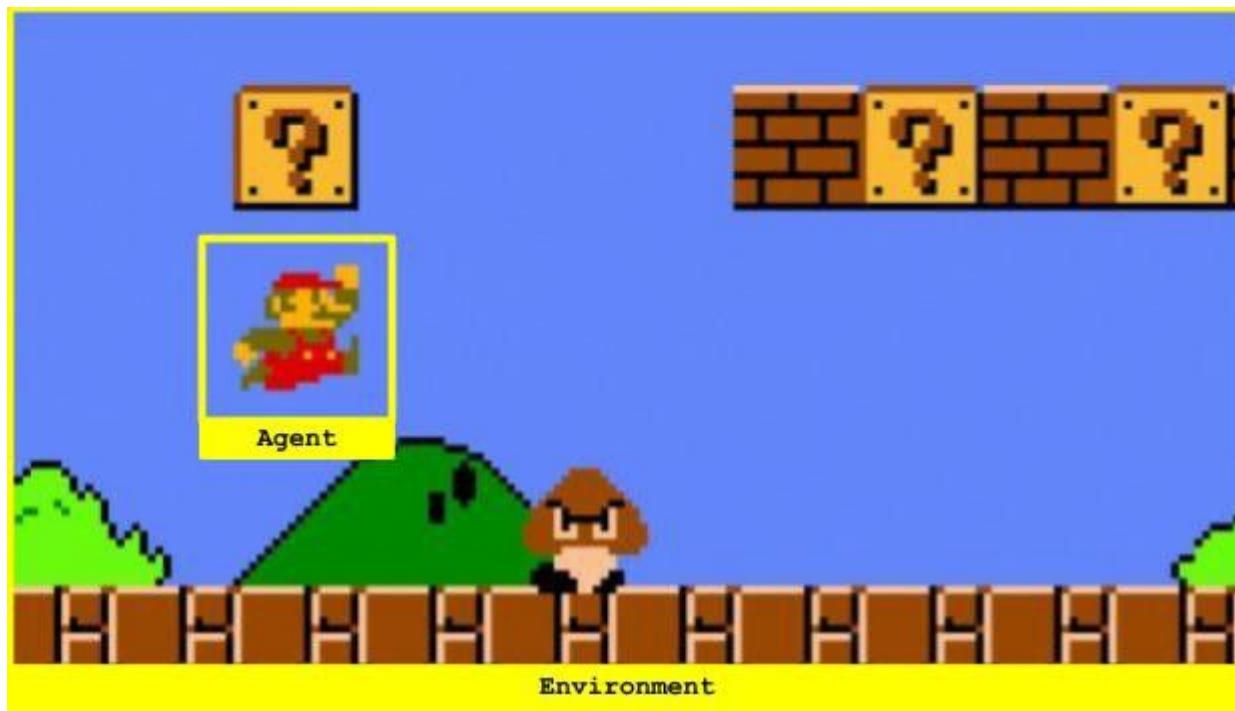


Reinforcement

Learn from mistakes



Agent & Environment

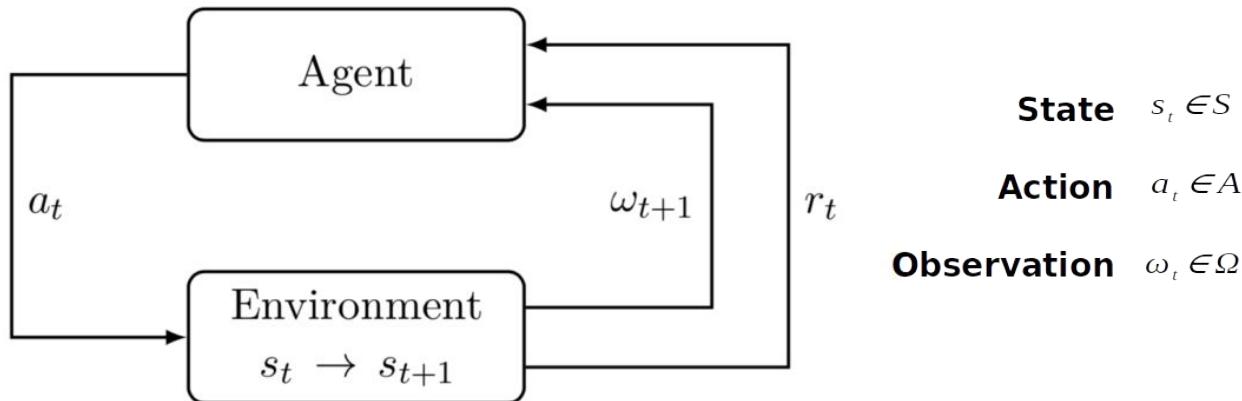


Agent & Environment

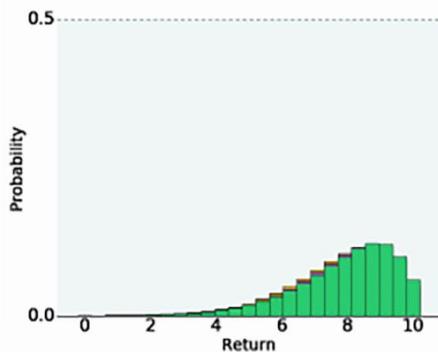
Subset of ML that deals with **sequential decision-making**

Maximize cumulative reward given sequences of actions in an environment

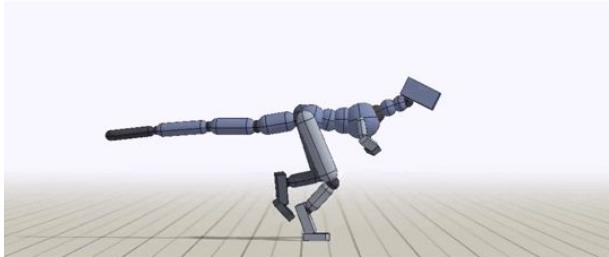
Learn good behaviour through experience



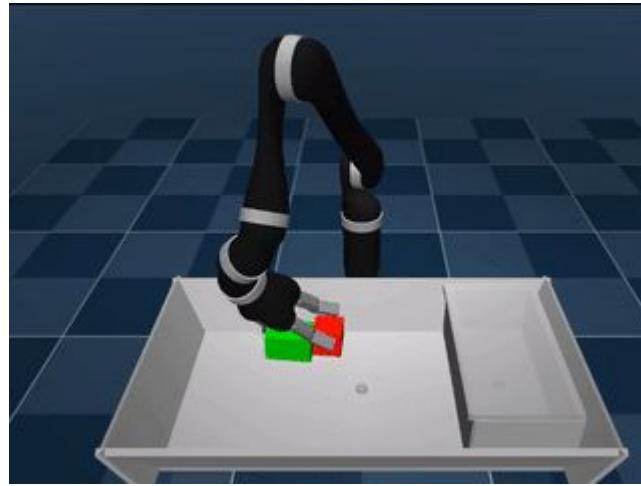
Agent & Environment



T-Rex: Walk



Simulated Character



Agent & Environment



Our World is way too complicated

Reinforcement learning is very promising but has a long way to go