Long short-term memory

Neural Networks that remember

Victor ADASCALITEI

Neural Networks recap

Neural Networks recap

What we know so far?

Neural Networks recap

What we know so far?

Convolutional Neural Networks (**CNN**s)

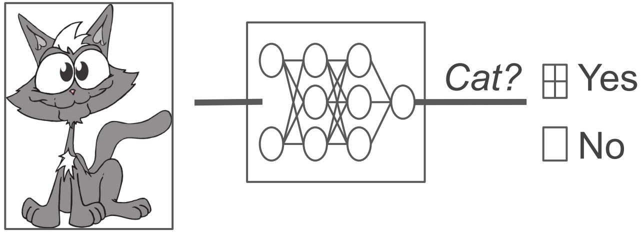
Generative Adversarial Networks (**GAN**s)

Neural Networks recap

What we know so far?

Convolutional Neural Networks (**CNN**s)

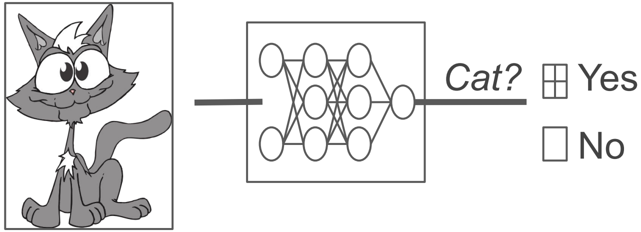
Generative Adversarial Networks (**GAN**s)



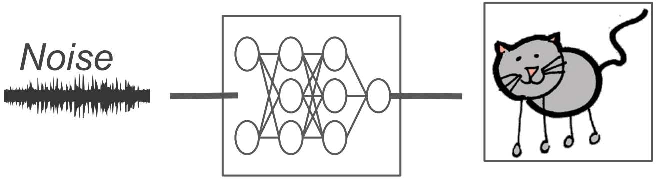
Neural Networks recap

What we know so far?

Convolutional Neural Networks (**CNN**s)



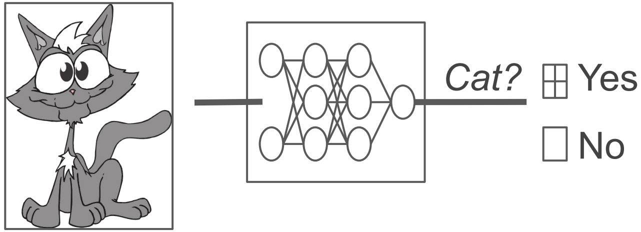
Generative Adversarial Networks (**GAN**s)



Neural Networks recap

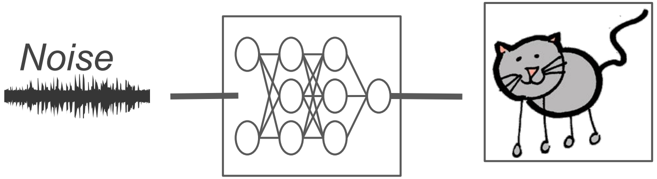
What we know so far?

Convolutional Neural Networks (**CNN**s)



More generally

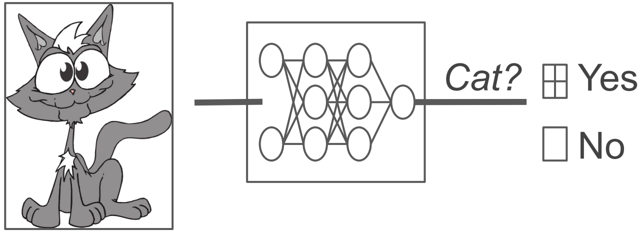
Generative Adversarial Networks (**GAN**s)



Neural Networks recap

What we know so far?

Convolutional Neural Networks (**CNN**s)

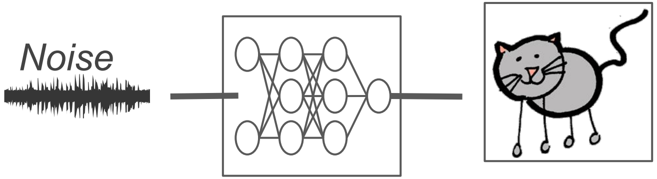


Cat?

⊞ Yes

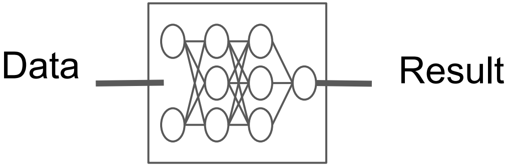
□ No

Generative Adversarial Networks (**GAN**s)



Noise

More generally



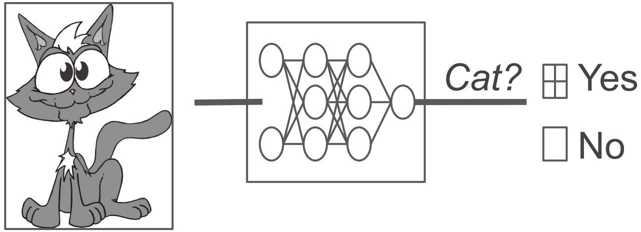
Data

Result

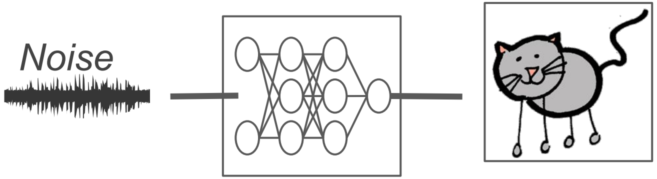
Neural Networks recap

What we know so far?

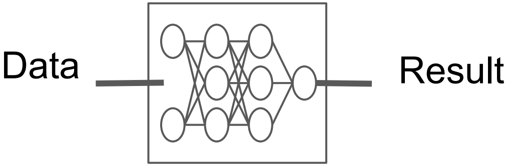
Convolutional Neural Networks (**CNN**s)



Generative Adversarial Networks (**GAN**s)



More generally

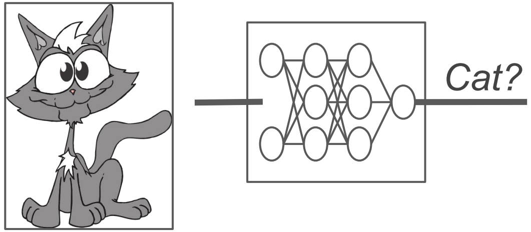


OR

Neural Networks recap

What we know so far?

Convolutional Neural Networks (**CNN**s)

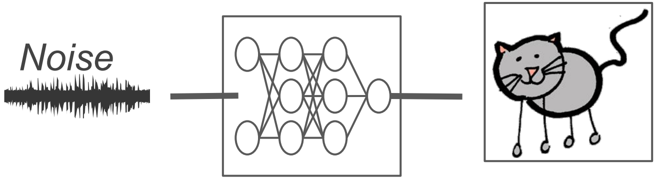


Cat? ⊞ Yes

□ No

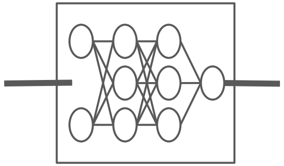
Generative Adversarial Networks (**GAN**s)

Noise



More generally

Data



Result

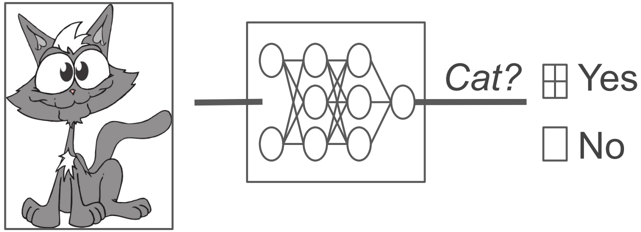
OR

Neural Network(Data) = Result

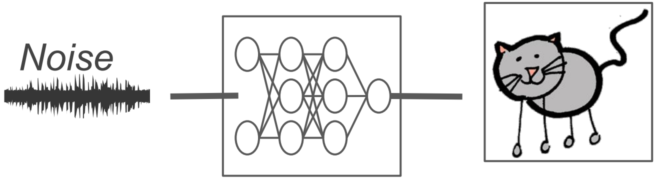
Neural Networks recap

What we know so far?

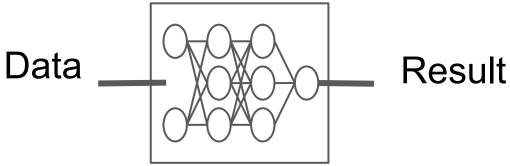
Convolutional Neural Networks (**CNN**s)



Generative Adversarial Networks (**GAN**s)



More generally



OR

$$\underbrace{\text{Neural Network}}_{f}(\underbrace{\text{Data}}_{x}) = \underbrace{\text{Result}}_{y}$$

Neural Networks recap

They are universal function approximators

Neural Networks recap

They are universal function approximators

(Given **enough** data, they can approximate any function)

Neural Networks recap

They are universal function approximators

(Given **enough** data, they can approximate any function)

True for images

Neural Networks recap

They are universal function approximators

(Given **enough** data, they can approximate any function)

True for images and classifiers

Neural Networks recap

They are universal function approximators

(Given **enough** data, they can approximate <u>any</u> function)

True for images and classifiers

but

Neural Networks recap

They are universal function approximators

(Given **enough** data, they can approximate any function)

True for images and classifiers

but

What about text ?

Neural Networks recap

They are universal function approximators

(Given **enough** data, they can approximate any function)

True for images and classifiers

but

What about text ?

… and speech ?

… and music ?

Neural Networks recap

They are universal function approximators

(Given **enough** data, they can approximate any function)

True for images and classifiers

but

What about text ?

… and speech ?

… and music ?

What about time/context dependent data?

Neural Networks

Input in relation with the output

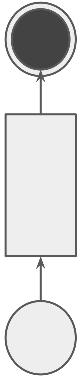
Neural Networks

Input in relation with the output

○ - Input ▭ - Network ● - Output

Neural Networks

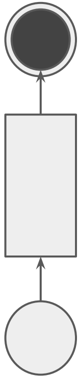
Input in relation with the output



○ - Input ▭ - Network ◉ - Output

Neural Networks

Input in relation with the output

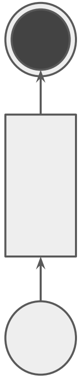


One to One

○ - Input ▭ - Network ● - Output

Neural Networks

Input in relation with the output



One to One

One to Many

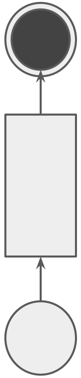
Many to One

Many to Many

○ - Input ▭ - Network ● - Output

Neural Networks

Input in relation with the output



One to One

???

One to Many

???

Many to One

???

Many to Many

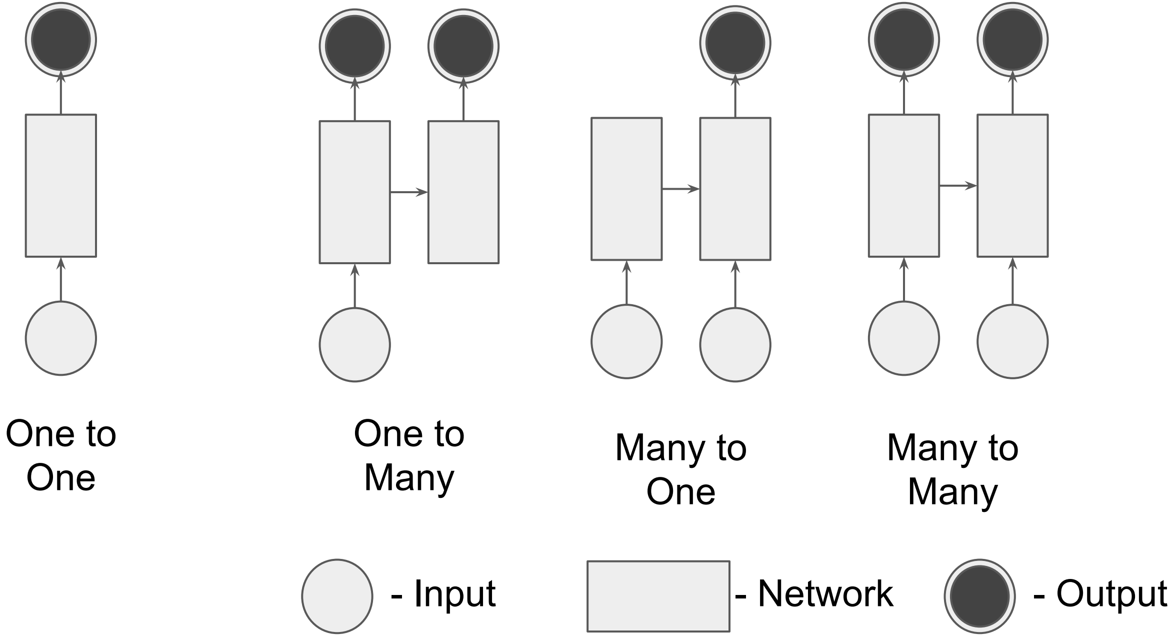
○ - Input

▭ - Network

● - Output

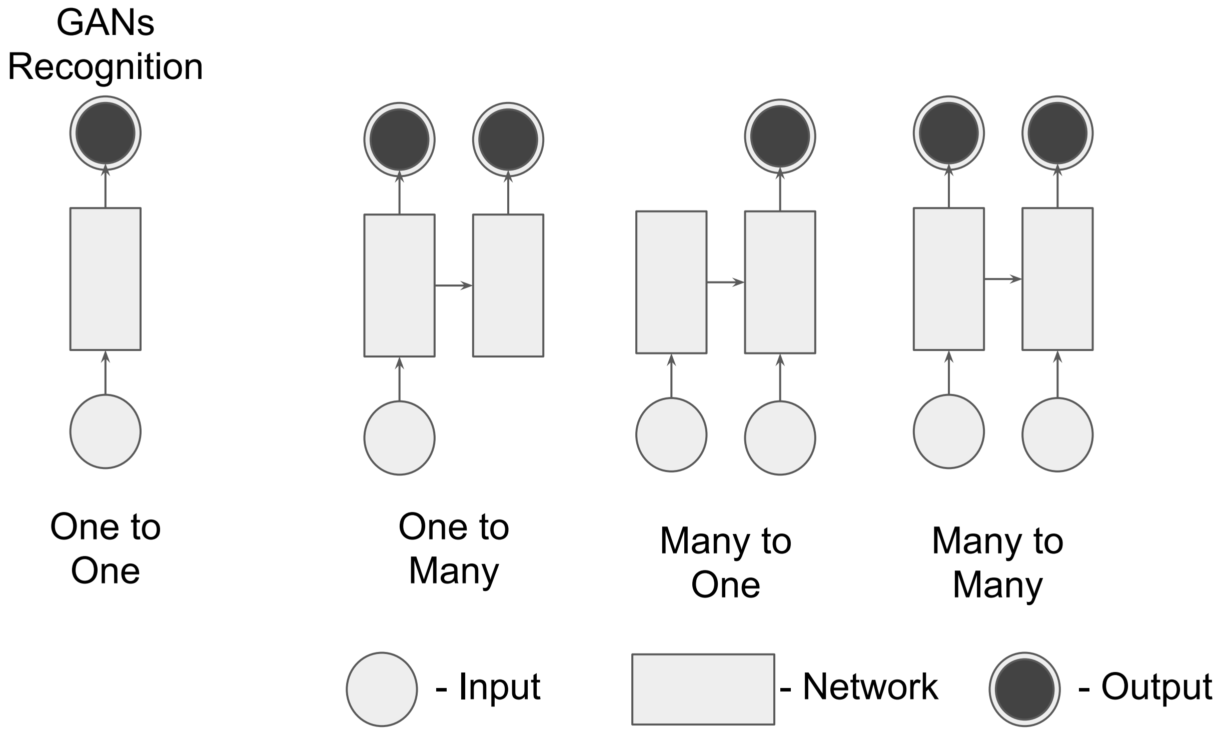
Neural Networks

Input in relation with the output



Neural Networks

Input in relation with the output



GANs
Recognition

One to
One

One to
Many

Many to
One

Many to
Many

○ - Input

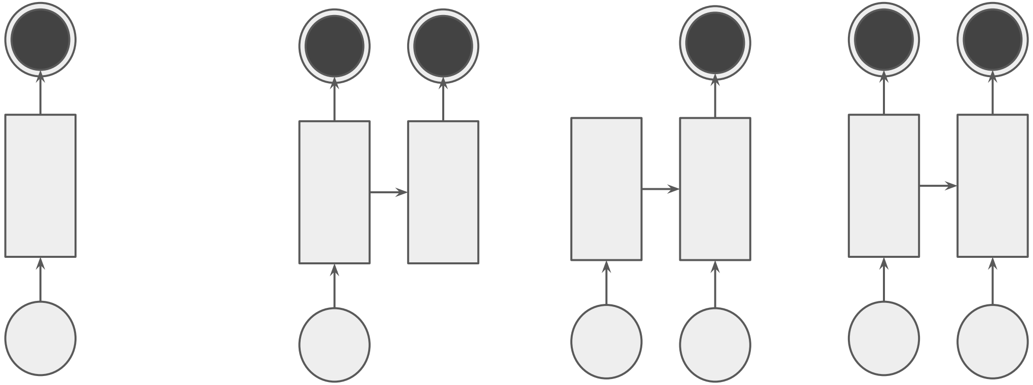
▭ - Network

● - Output

Neural Networks

Input in relation with the output

Image Captioning



One to One

One to Many

Many to One

Many to Many

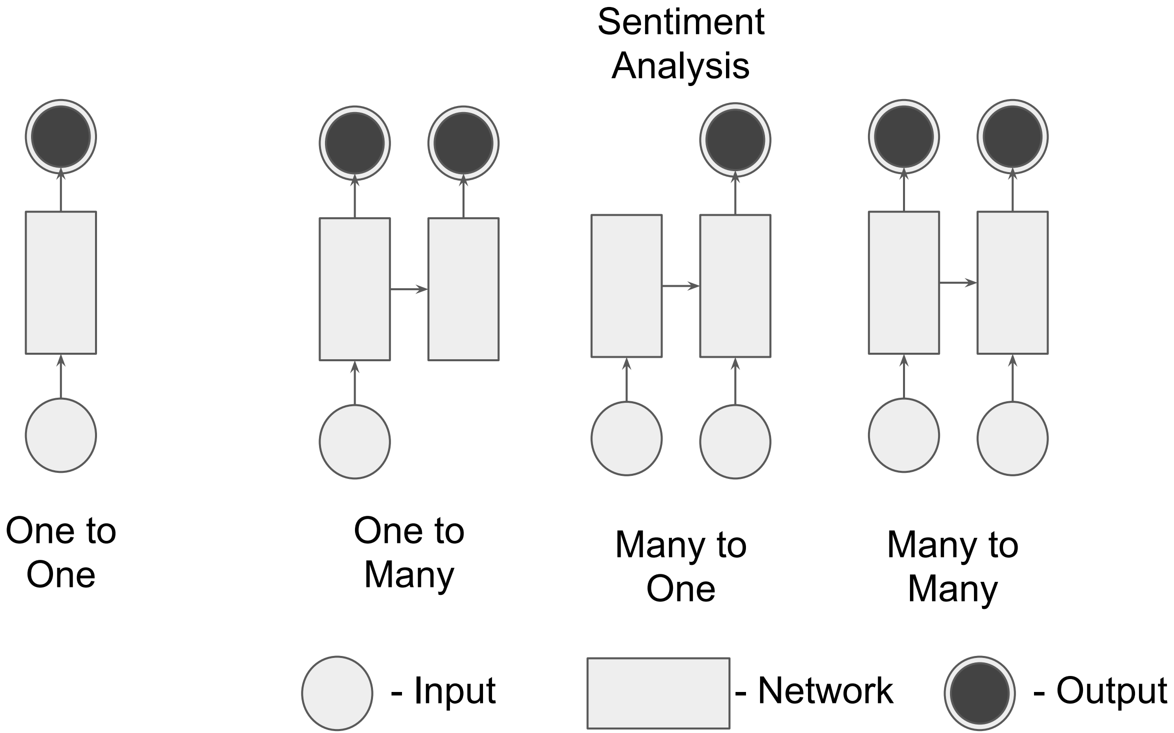
○ - Input

▭ - Network

● - Output

Neural Networks

Input in relation with the output



Sentiment Analysis

One to One

One to Many

Many to One

Many to Many

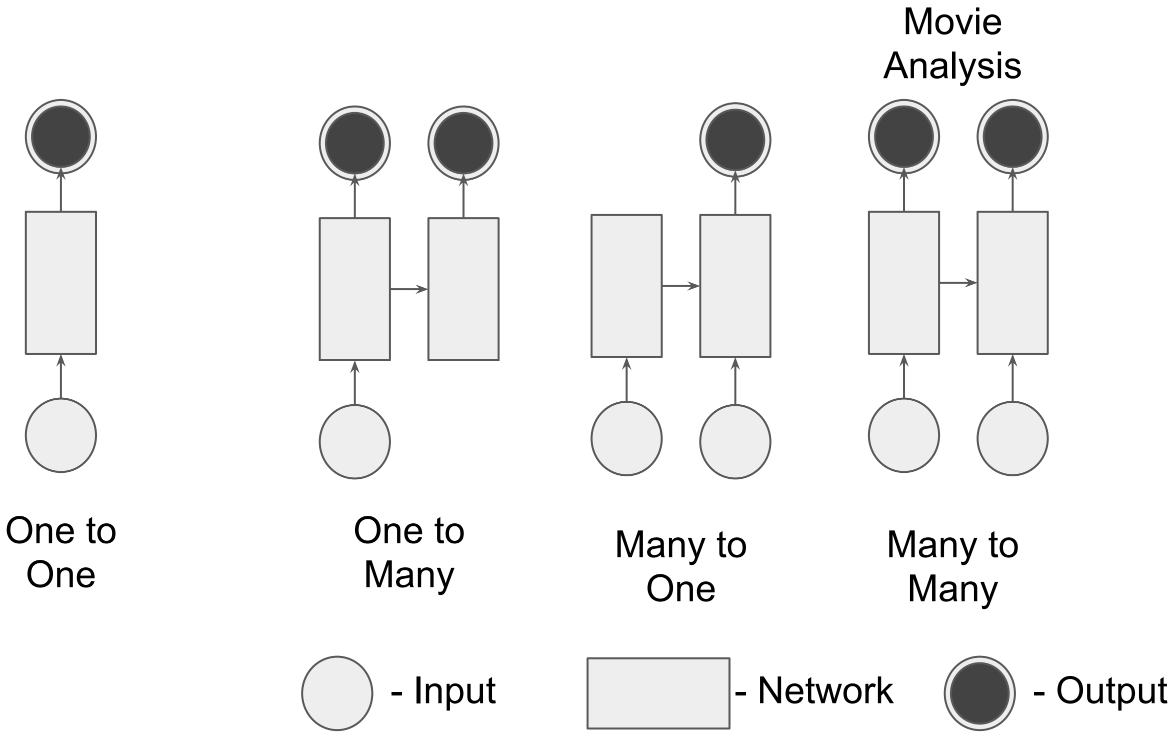
○ - Input

▭ - Network

● - Output

Neural Networks

Input in relation with the output



Movie Analysis

One to One

One to Many

Many to One

Many to Many

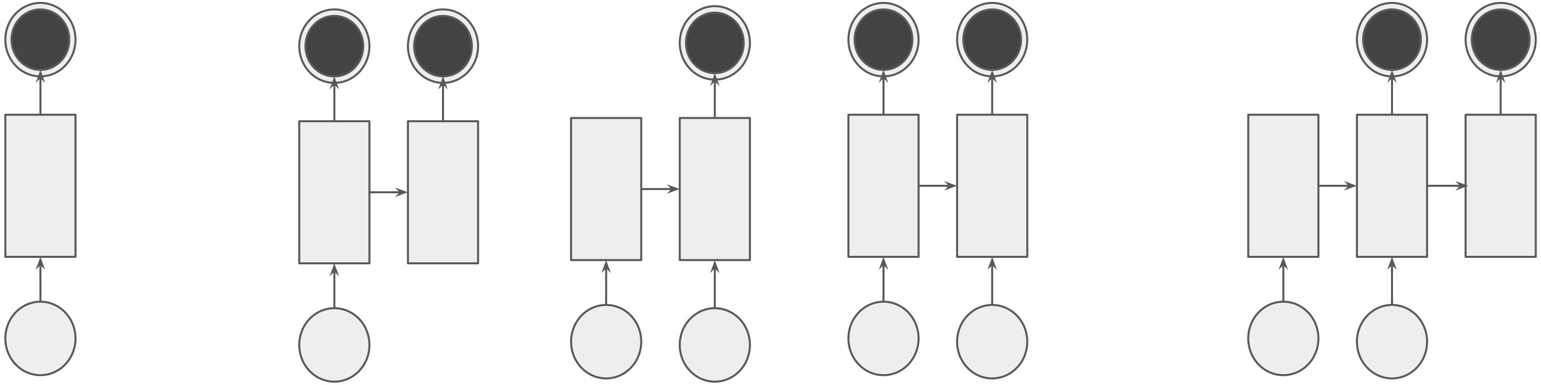
- Input

- Network

- Output

Neural Networks

Input in relation with the output



One to One

One to Many

Many to One

Many to Many

Many to Many*

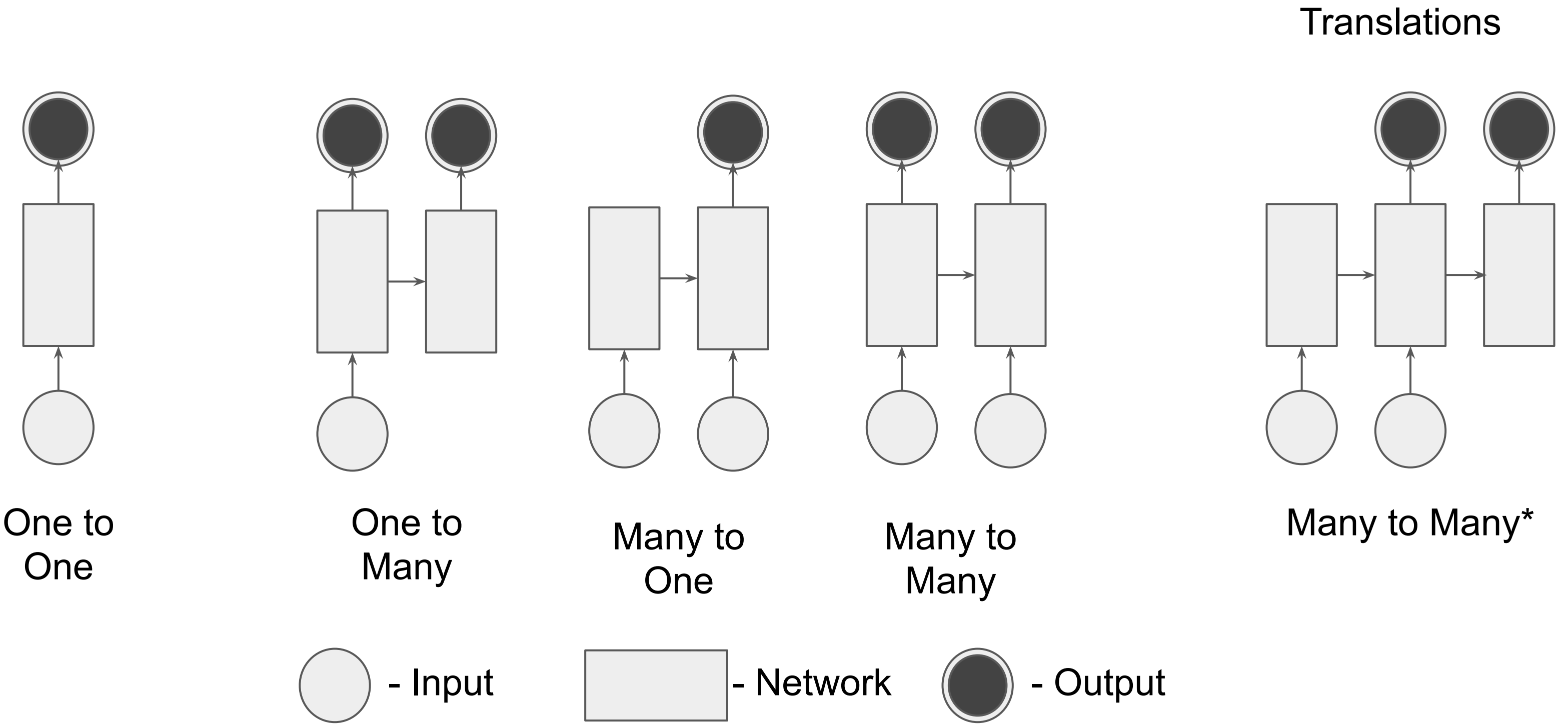
○ - Input

▭ - Network

● - Output

Neural Networks

Input in relation with the output



Translations

One to One

One to Many

Many to One

Many to Many

Many to Many*

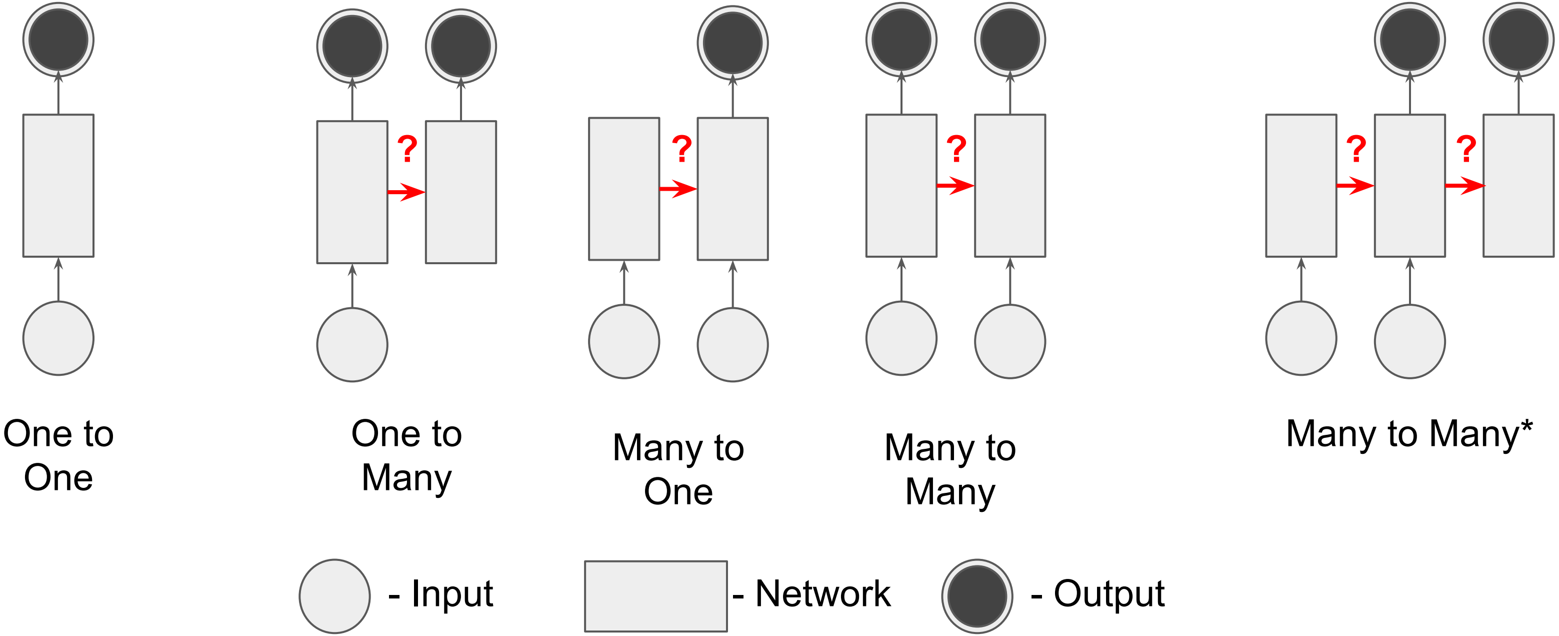
○ - Input

▭ - Network

● - Output

Neural Networks

Input in relation with the output



One to One

One to Many

Many to One

Many to Many

Many to Many*

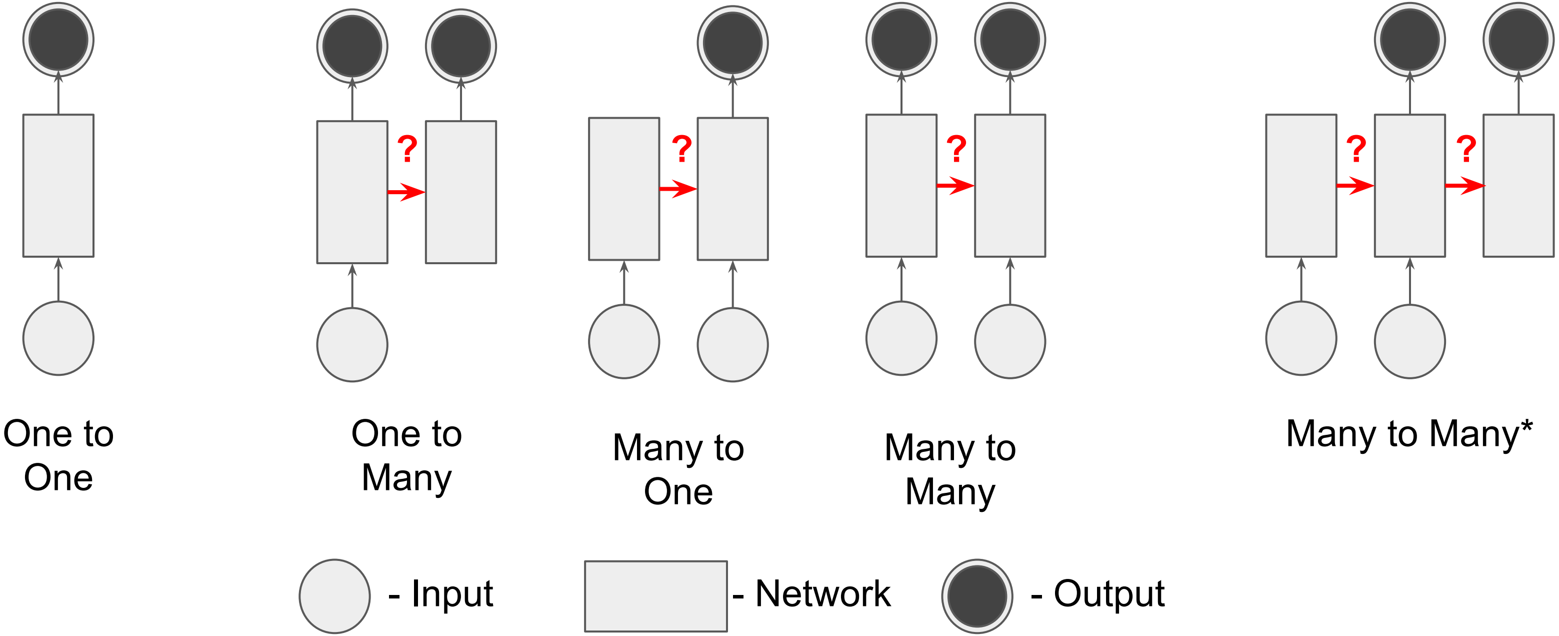
○ - Input

▭ - Network

● - Output

Recurrent Neural Networks

Input in relation with the output



One to One

One to Many

Many to One

Many to Many

Many to Many*

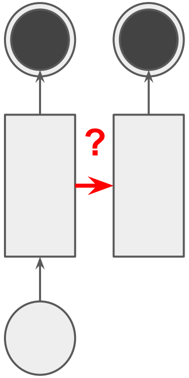
○ - Input

▭ - Network

● - Output

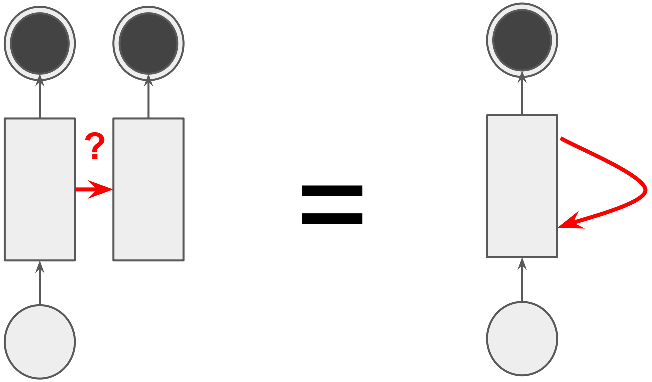
Recurrent Neural Networks

Input in relation with the output



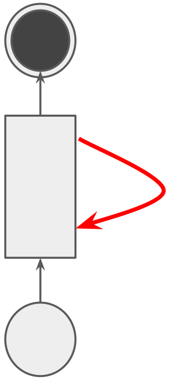
Recurrent Neural Networks

Input in relation with the output



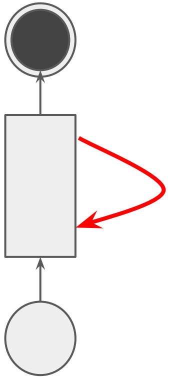
Recurrent Neural Networks

Input in relation with the output



Recurrent Neural Networks

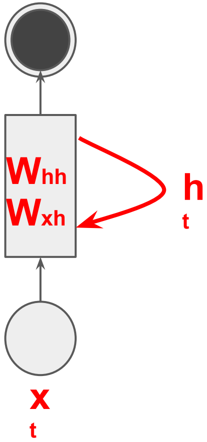
Input in relation with the output



$$h_{t+1} = \tanh(W_{hh}h_t + W_{xh}x_t)$$

Recurrent Neural Networks

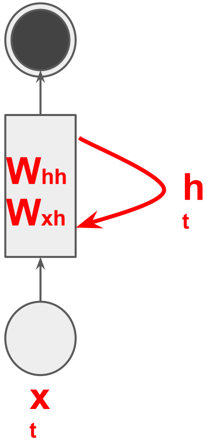
Explained



$$h_{t+1} = \tanh(W_{hh}h_t + W_{xh}x_t)$$

Recurrent Neural Networks

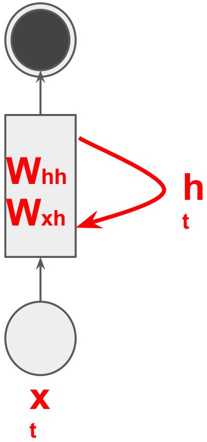
Explained



$$h_{t+1} = \tanh(W_{hh}h_t + W_{xh}x_t)$$

Recurrent Neural Networks

Explained



$$h_{t+1} = \tanh(W_{hh}h_t + W_{xh}x_t)$$

Recurrent Neural Networks

The problem

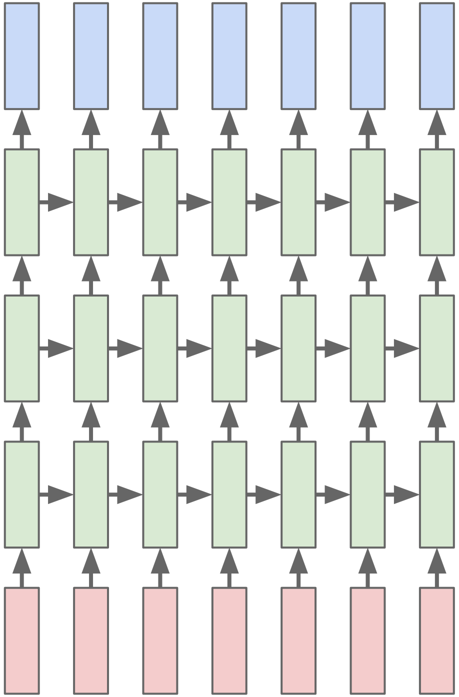
Recurrent Neural Networks

The problem

$$h_{t+1} = \tanh(W_{hh} h_t + W_{xh} x_t)$$

Recurrent Neural Networks

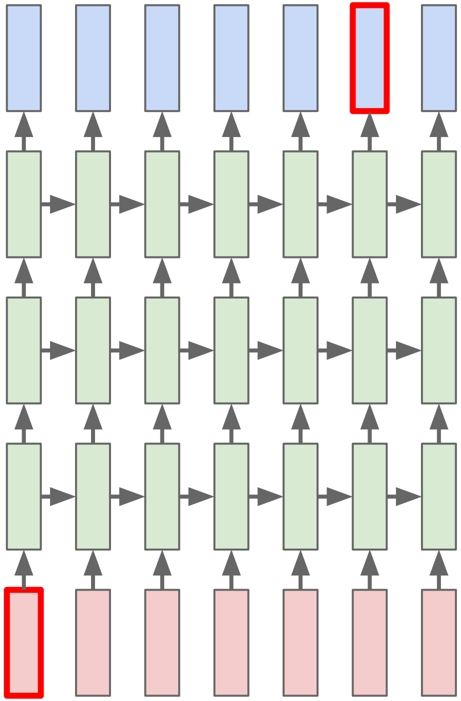
The problem



$$h_{t+1} = \tanh(W_{hh}h_t + W_{xh}x_t)$$

Recurrent Neural Networks

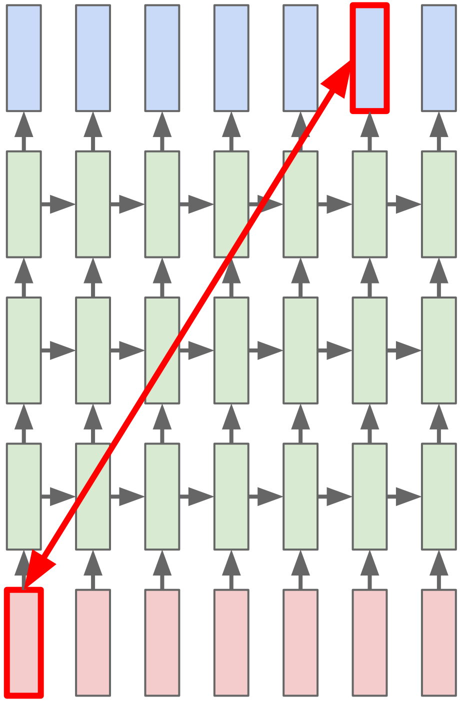
The problem



$$h_{t+1} = \tanh(W_{hh}h_t + W_{xh}x_t)$$

Recurrent Neural Networks

The problem

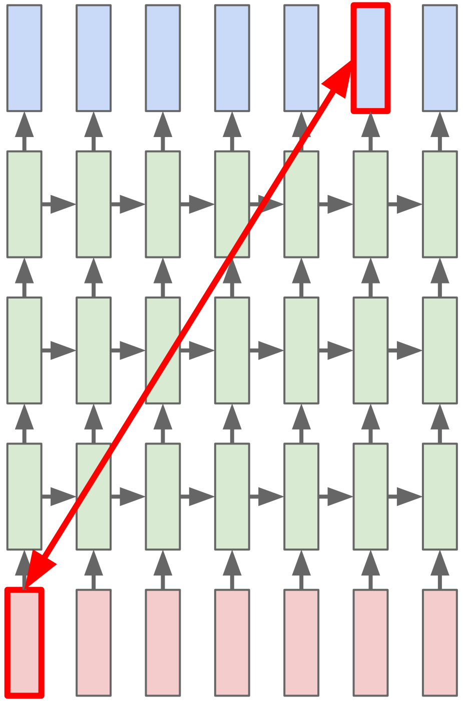


$$h_{t+1} = \tanh(W_{hh}h_t + W_{xh}x_t)$$

Recurrent Neural Networks

The problem

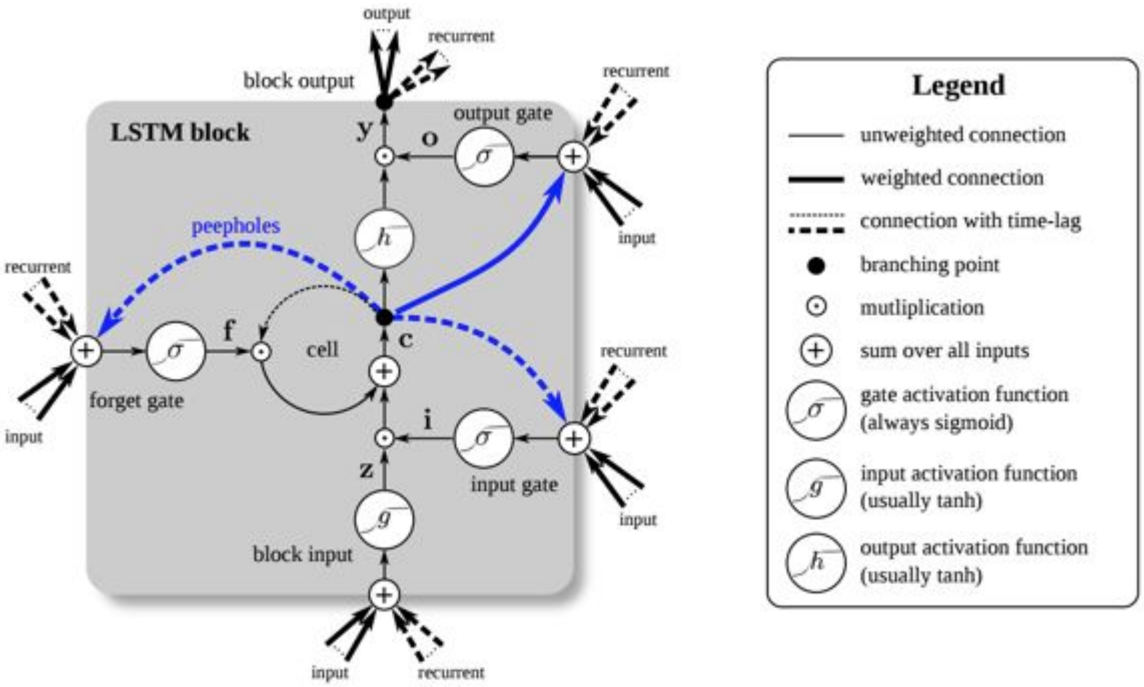
Vanishing gradient



$$h_{t+1} = \tanh(W_{hh}h_t + W_{xh}x_t)$$

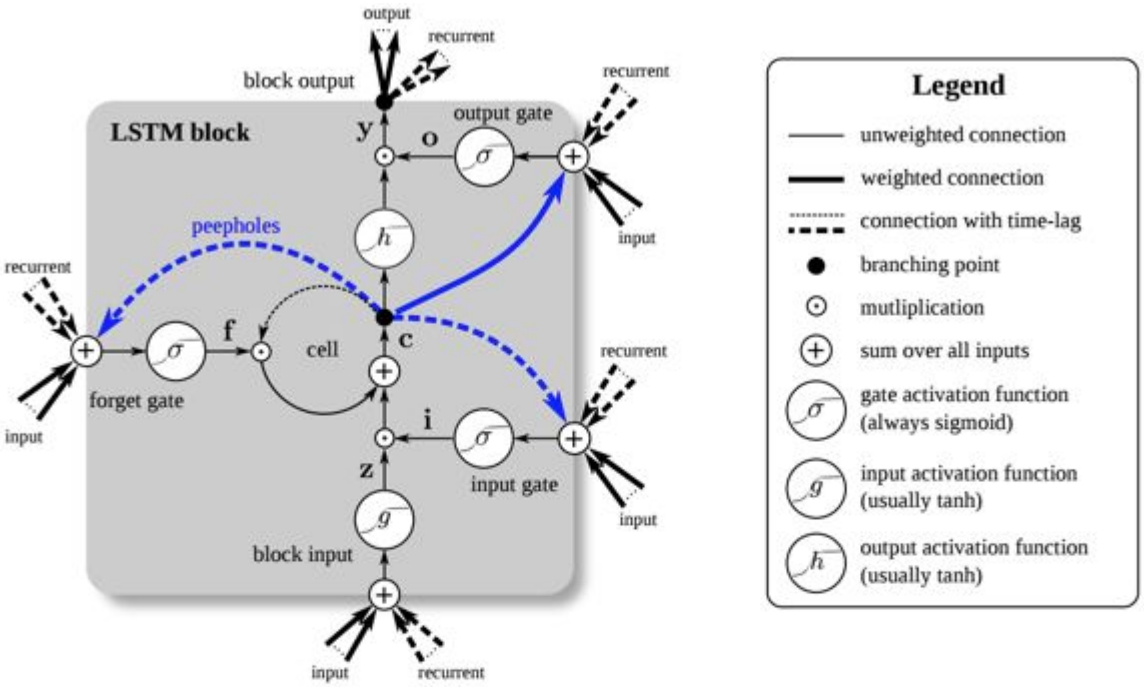
Long short-term memory cell

Long short-term memory cell



[1]

Long short-term memory cell



$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \text{sigm} \\ \text{sigm} \\ \text{sigm} \\ \tanh \end{pmatrix} W^l \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}$$

$$c_t^l = f \odot c_{t-1}^l + i \odot g$$

$$h_t^l = o \odot \tanh(c_t^l)$$

[1]

Code example

[Github]

Sources

[1] “LSTM: A Search Space Odyssey” -- https://arxiv.org/pdf/1503.04069.pdf

[2] “RNN Escapades” -- London ML meetup 09/2015 Andrej Karpathy https://docs.google.com/presentation/d/1qs2IuSdZvbNfzw217kH5-1Z9DjG0Ng6fJiabaLNQVaY