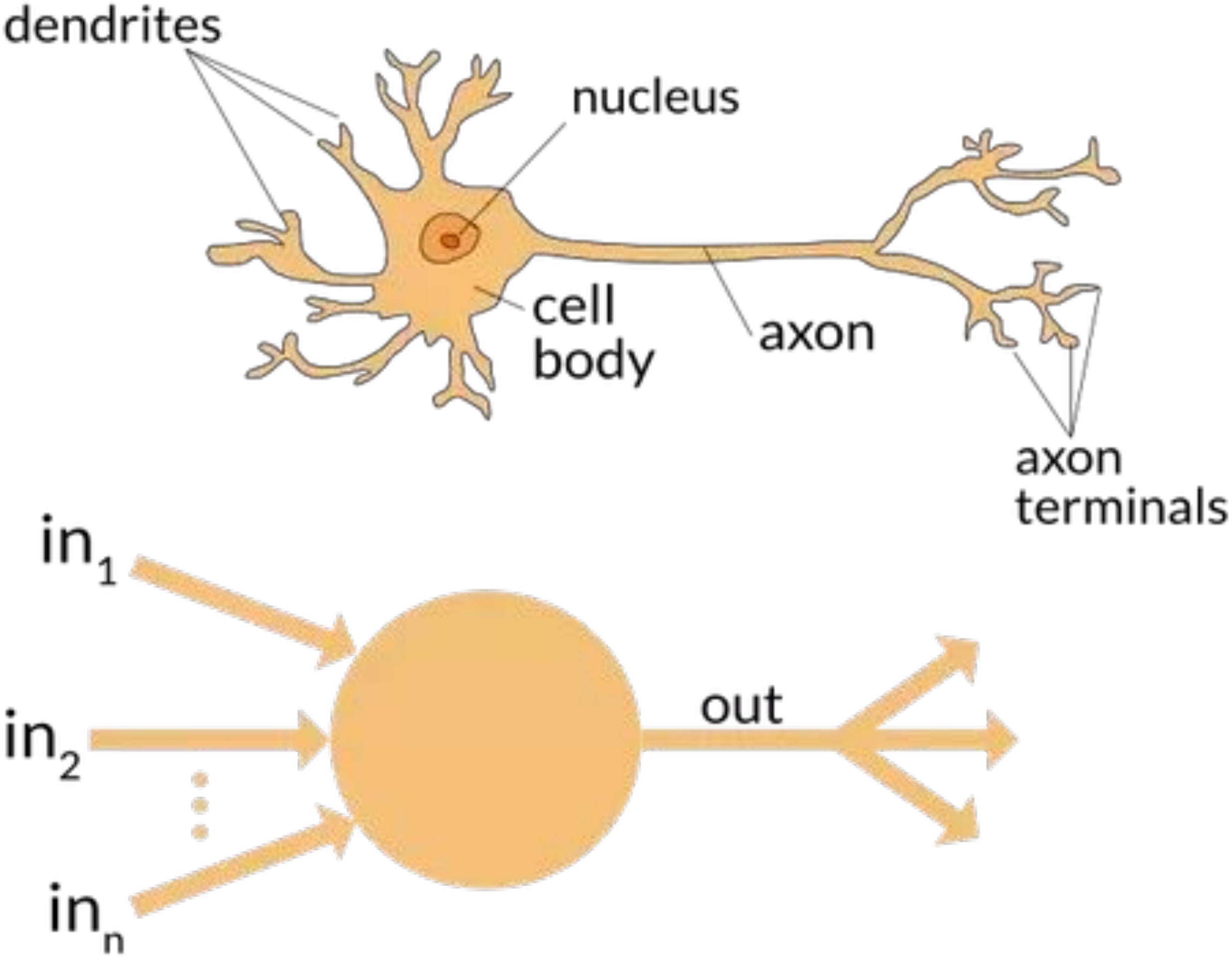
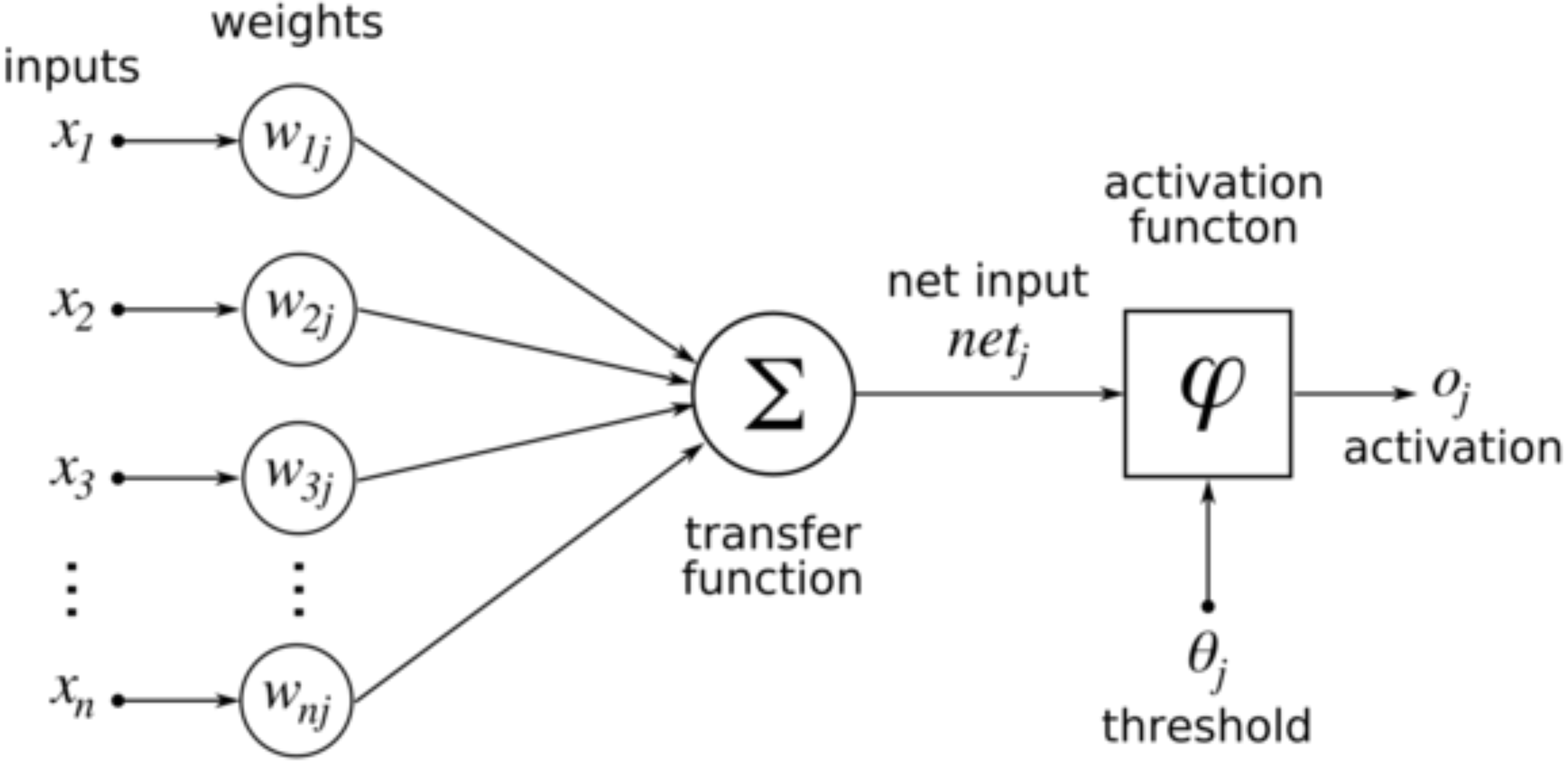
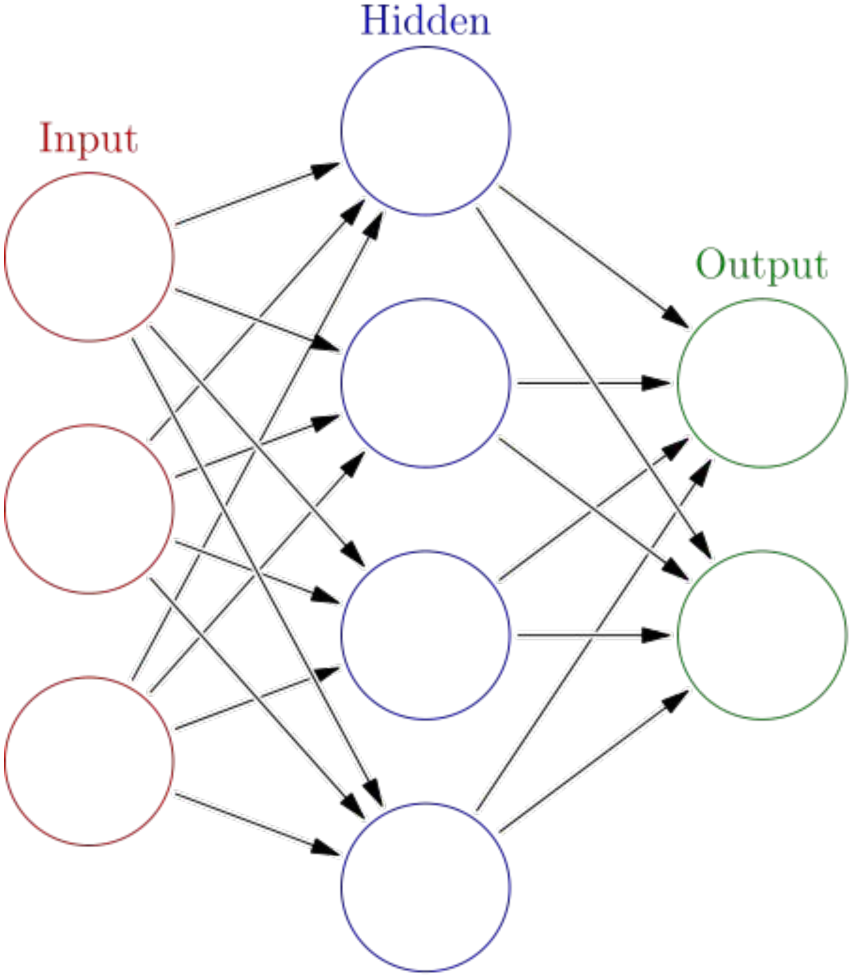
Recurrent Neural Networks

Week 12 - Day 02

Recap


dendrites
nucleus
cell body
axon
axon terminals
in_1
in_2
in_n
out

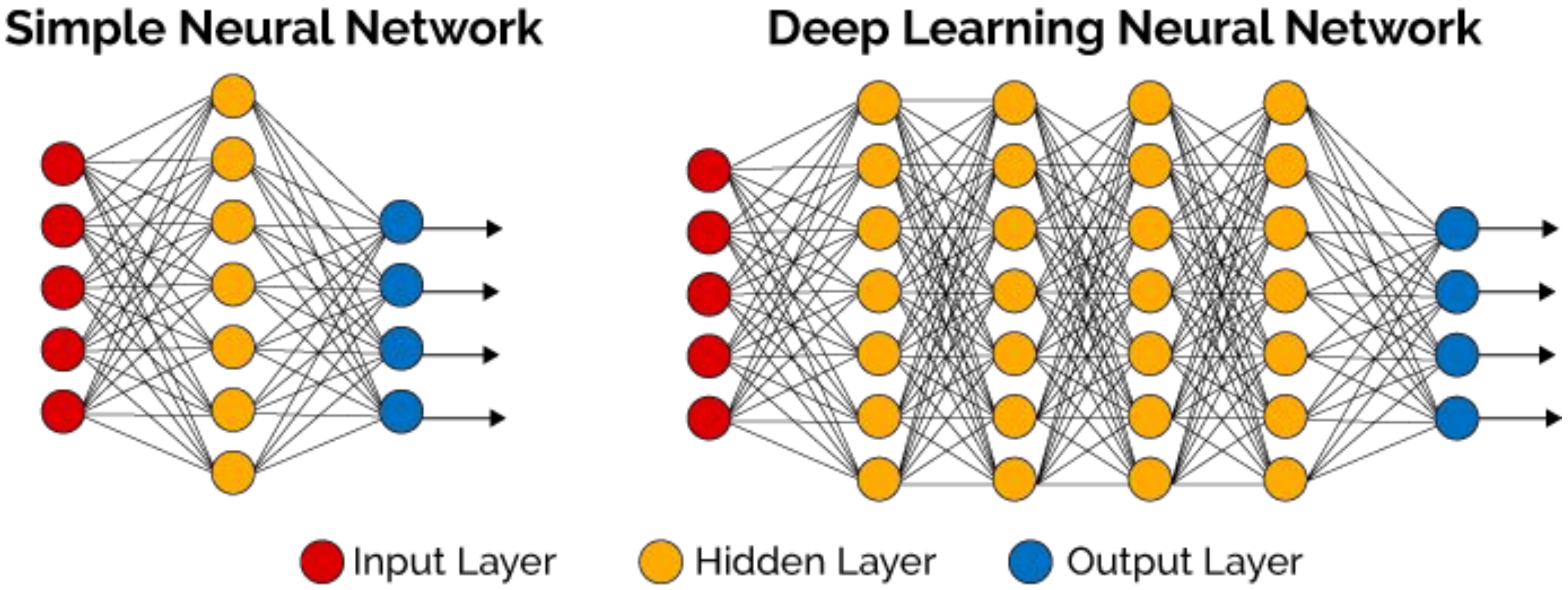

weights
inputs
x_1
w_{1j}
x_2
w_{2j}
x_3
w_{3j}
x_n
w_{nj}
Σ
transfer function
net input
net_j
activation functon
φ
o_j
activation
θ_j
threshold


Hidden
Input
Output

```
MLPClassifier(hidden_layer_sizes=(100, 10))
```

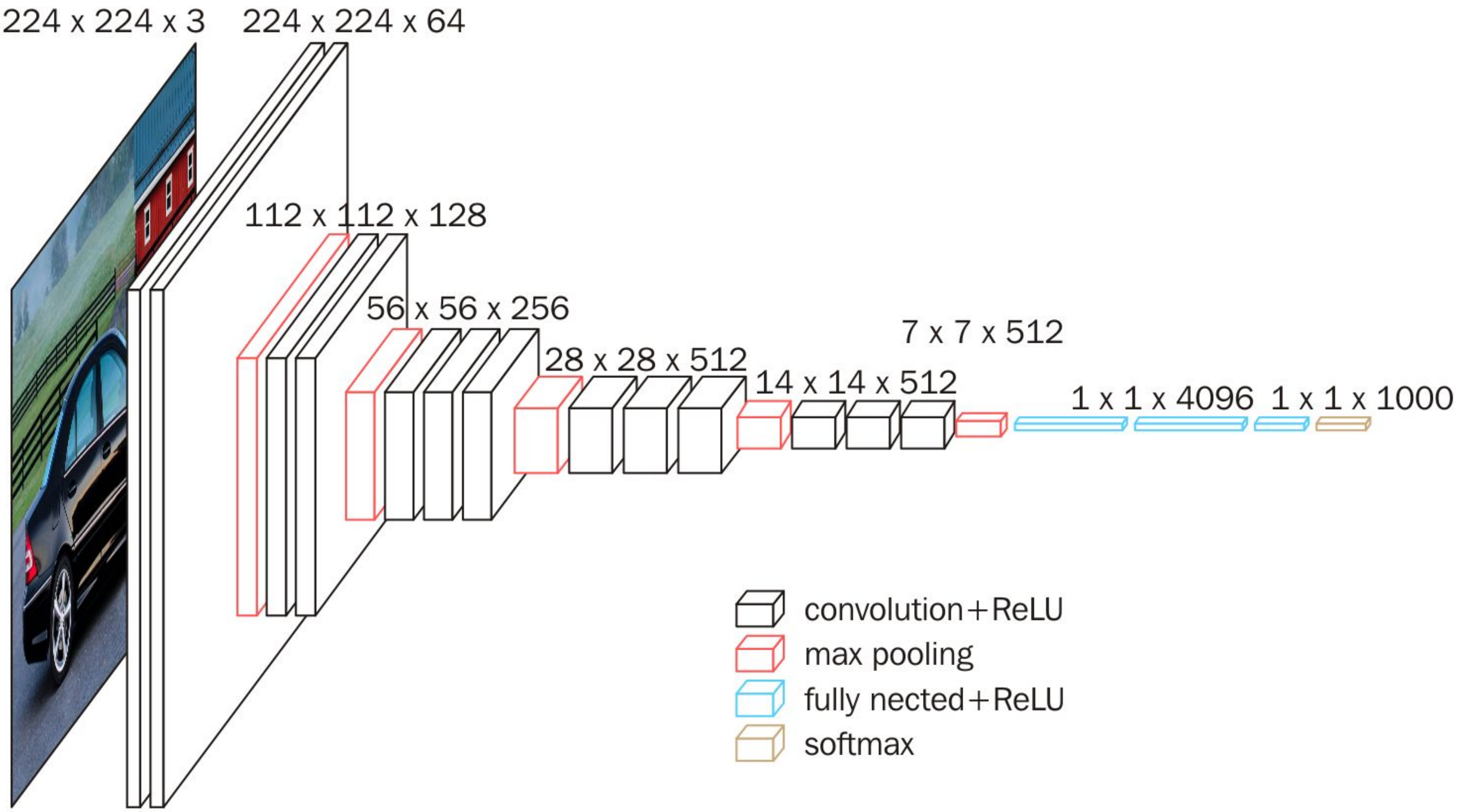
Backpropagation

to train ANN



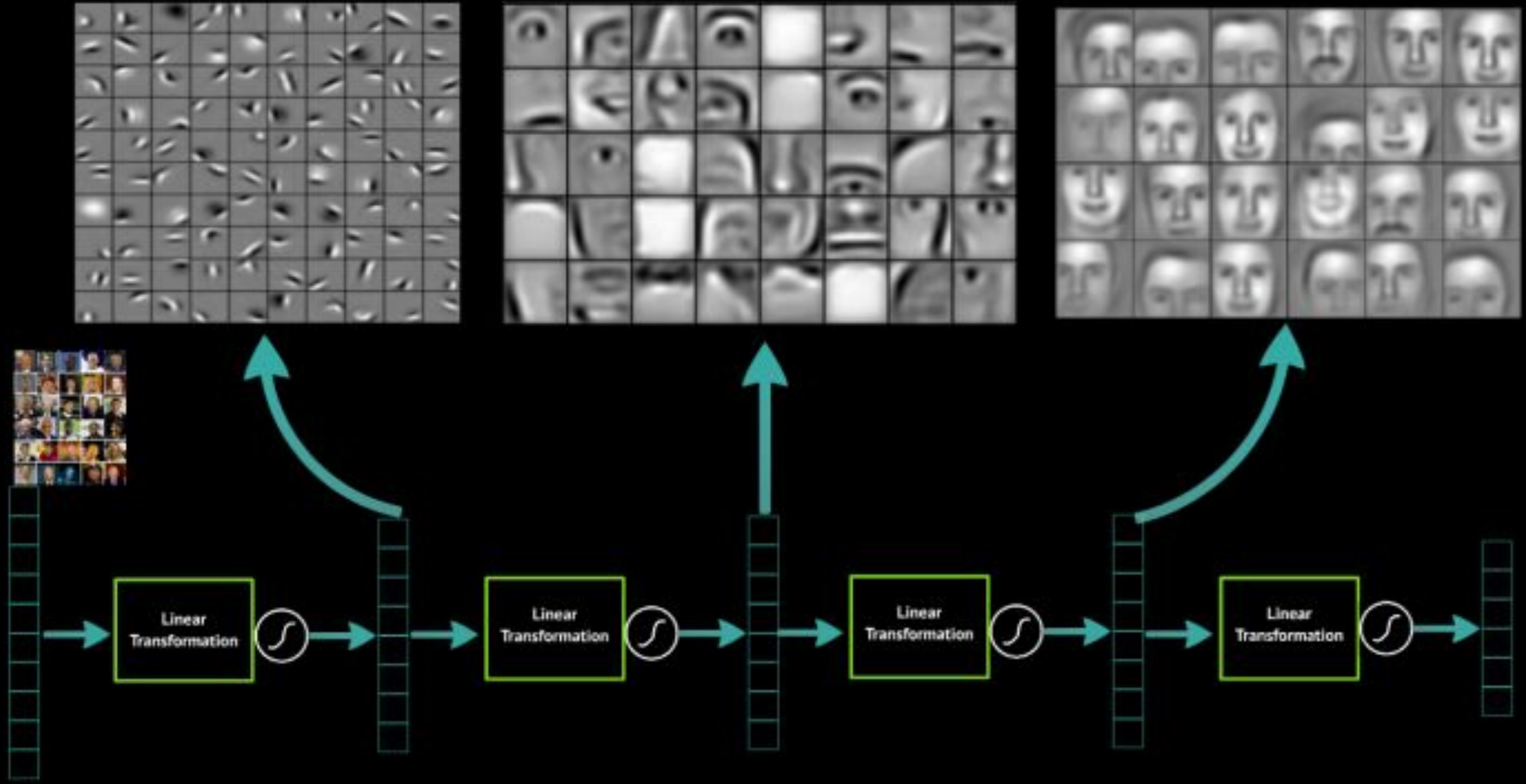
Simple Neural Network

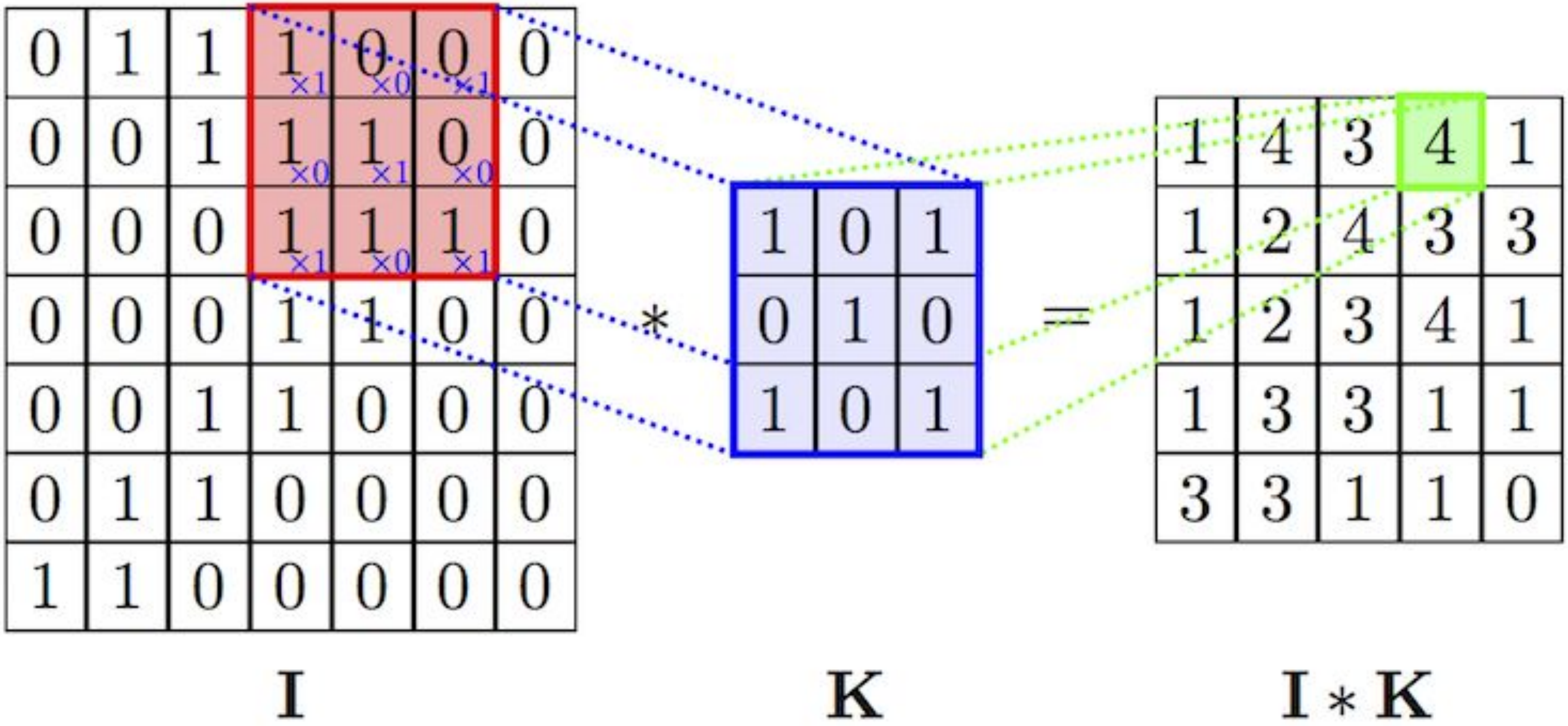
Deep Learning Neural Network


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

Convolutional Layers

Deep Learning learns layers of features





0	1	1	1	0	0	0
0	0	1	1	1	0	0
0	0	0	1	1	1	0
0	0	0	1	1	0	0
0	0	1	1	0	0	0
0	1	1	0	0	0	0
1	1	0	0	0	0	0

$\mathbf{I}$

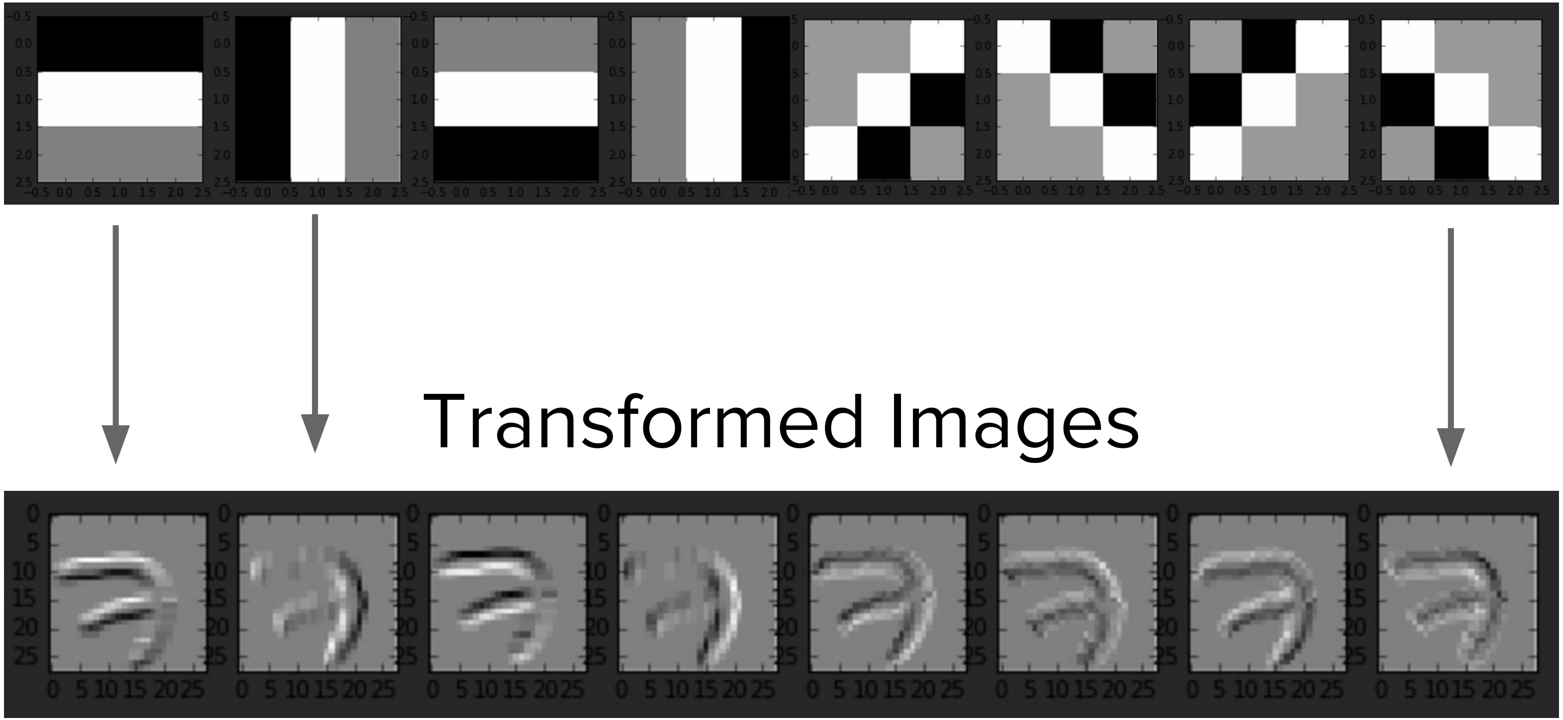
1	0	1
0	1	0
1	0	1

$\mathbf{K}$

1	4	3	4	1
1	2	4	3	3
1	2	3	4	1
1	3	3	1	1
3	3	1	1	0

$\mathbf{I} * \mathbf{K}$

Filters



Transformed Images

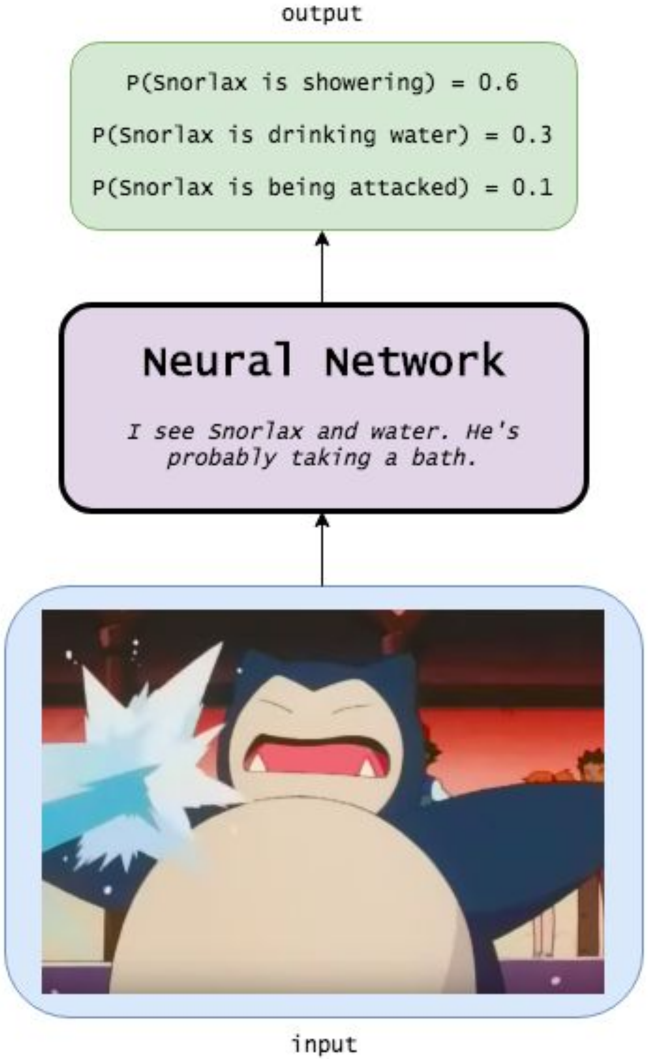
No need to build the features!

Recurrent Neural Networks

Let’s talk about

Pokémon™

CNN = images



output

P(Snorlax is showering) = 0.6

P(Snorlax is drinking water) = 0.3

P(Snorlax is being attacked) = 0.1

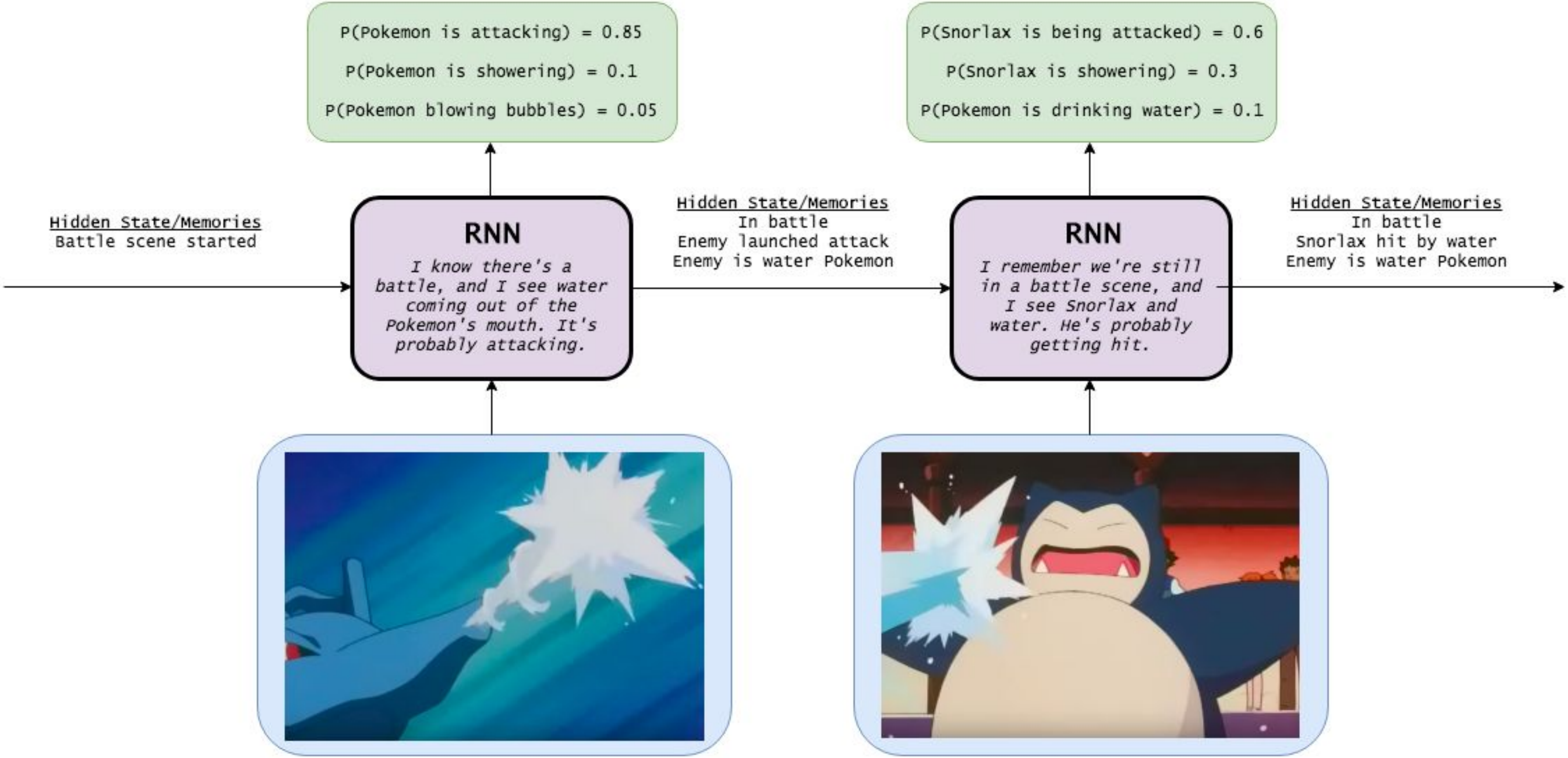
Neural Network

I see Snorlax and water. He's probably taking a bath.

input

CNN = what I see

RNN = what I see + what happened before



P(Pokemon is attacking) = 0.85

P(Pokemon is showering) = 0.1

P(Pokemon blowing bubbles) = 0.05

P(Snorlax is being attacked) = 0.6

P(Snorlax is showering) = 0.3

P(Pokemon is drinking water) = 0.1

<u>Hidden State/Memories</u>
Battle scene started

RNN

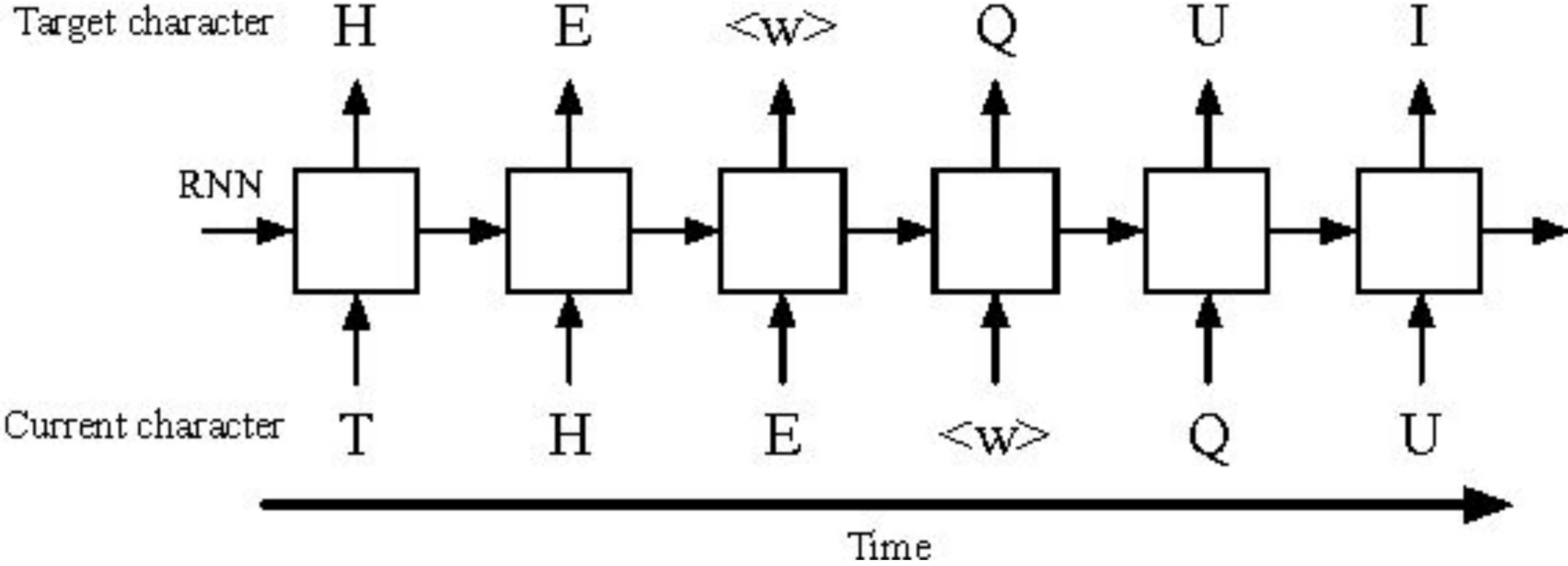
I know there's a battle, and I see water coming out of the Pokemon's mouth. It's probably attacking.

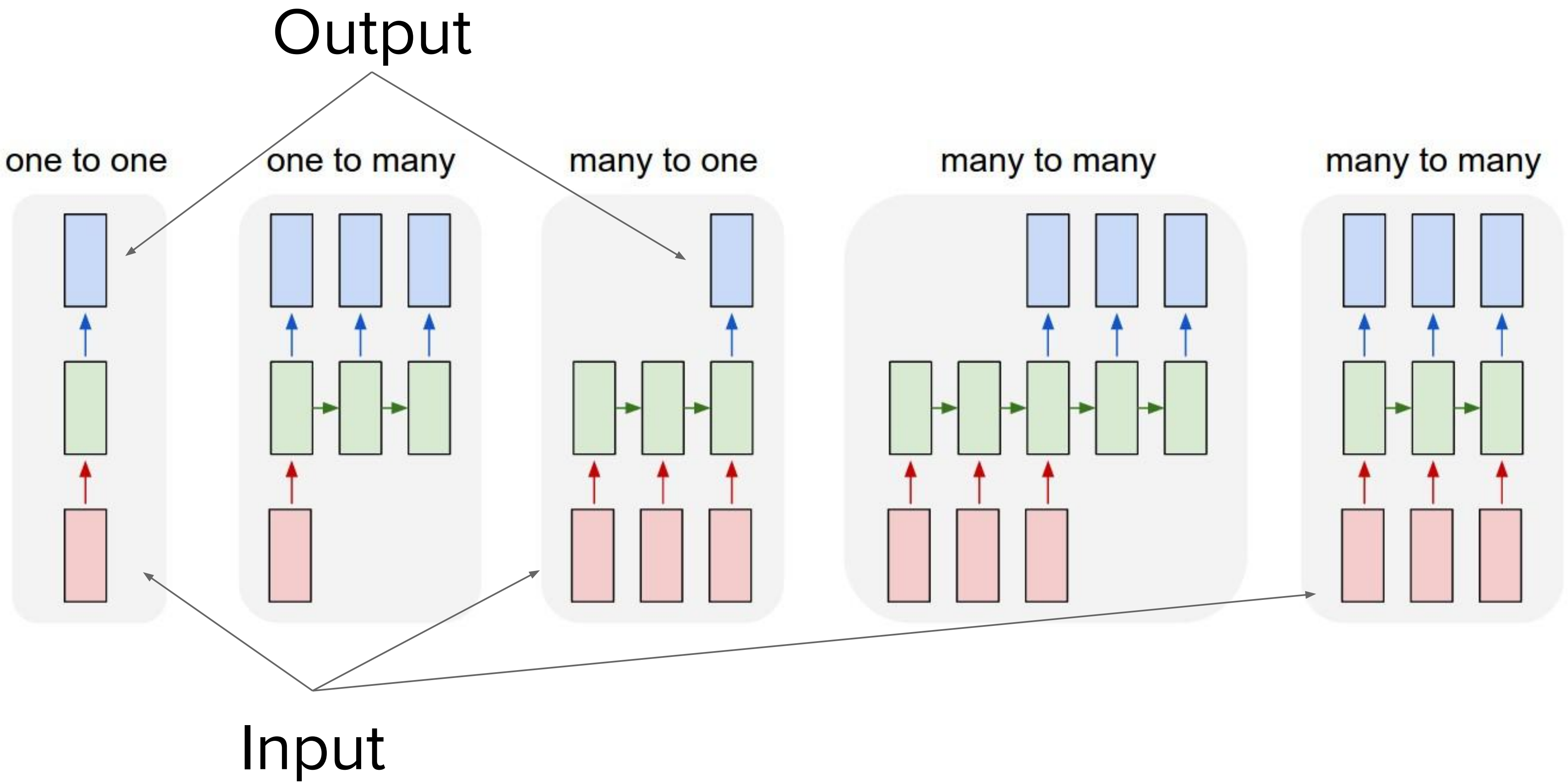
<u>Hidden State/Memories</u>
In battle
Enemy launched attack
Enemy is water Pokemon

RNN

I remember we're still in a battle scene, and I see Snorlax and water. He's probably getting hit.

<u>Hidden State/Memories</u>
In battle
Snorlax hit by water
Enemy is water Pokemon


Target character
H
E
<w>
Q
U
I
RNN
Current character
T
H
E
<w>
Q
U
Time


Output
one to one
one to many
many to one
many to many
many to many
Input

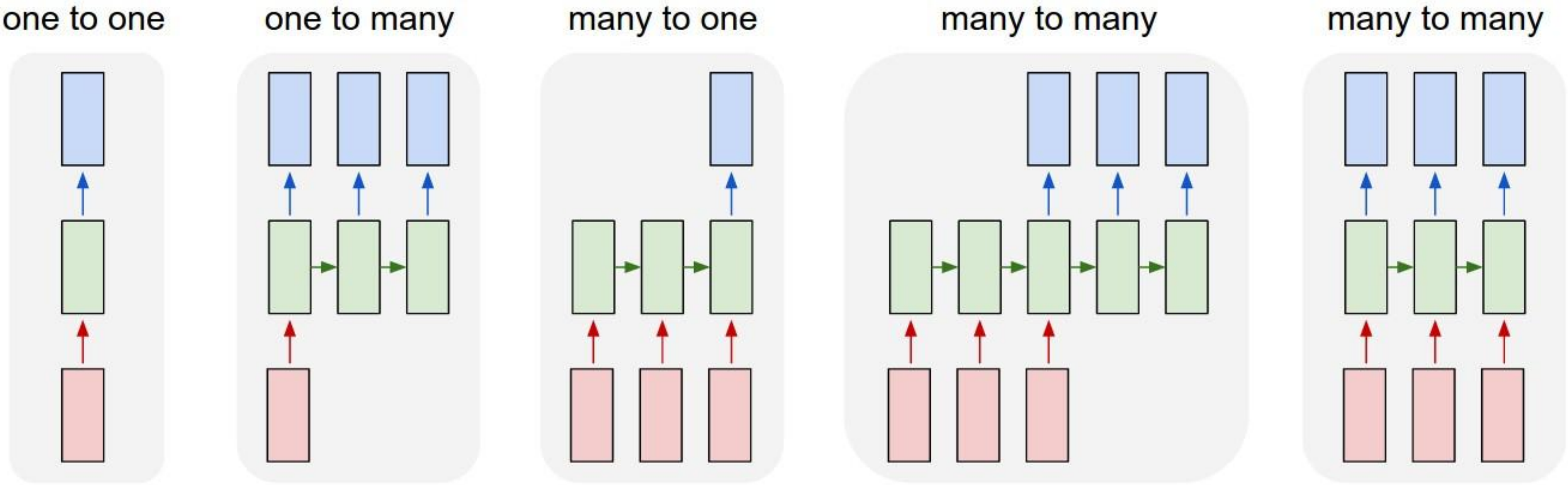
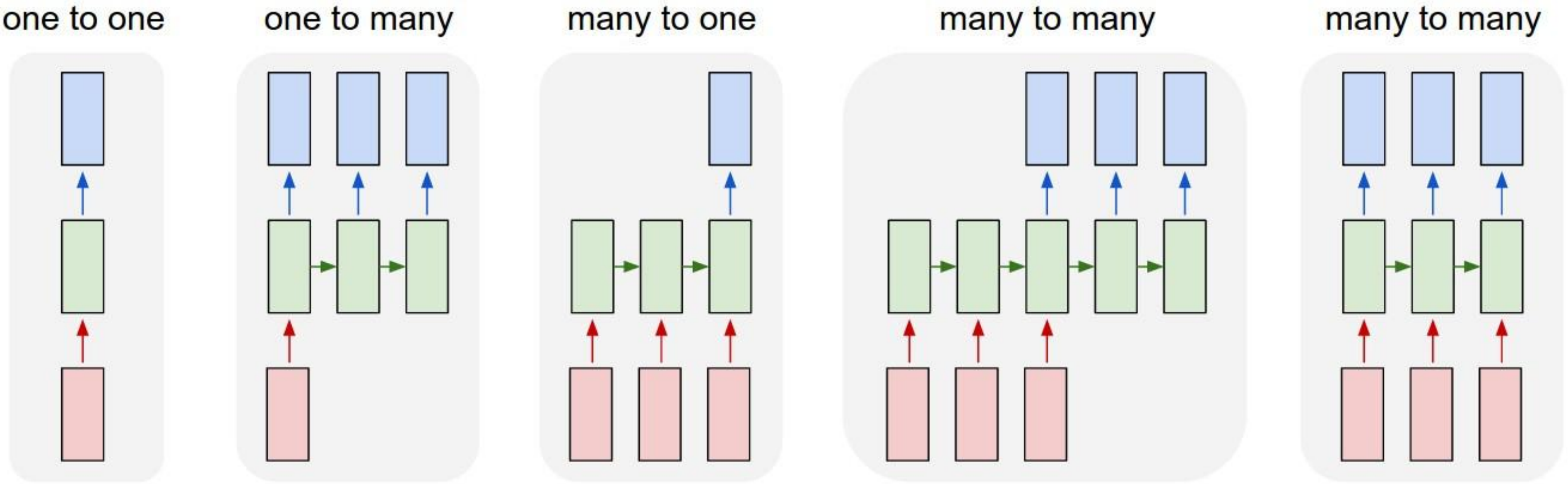
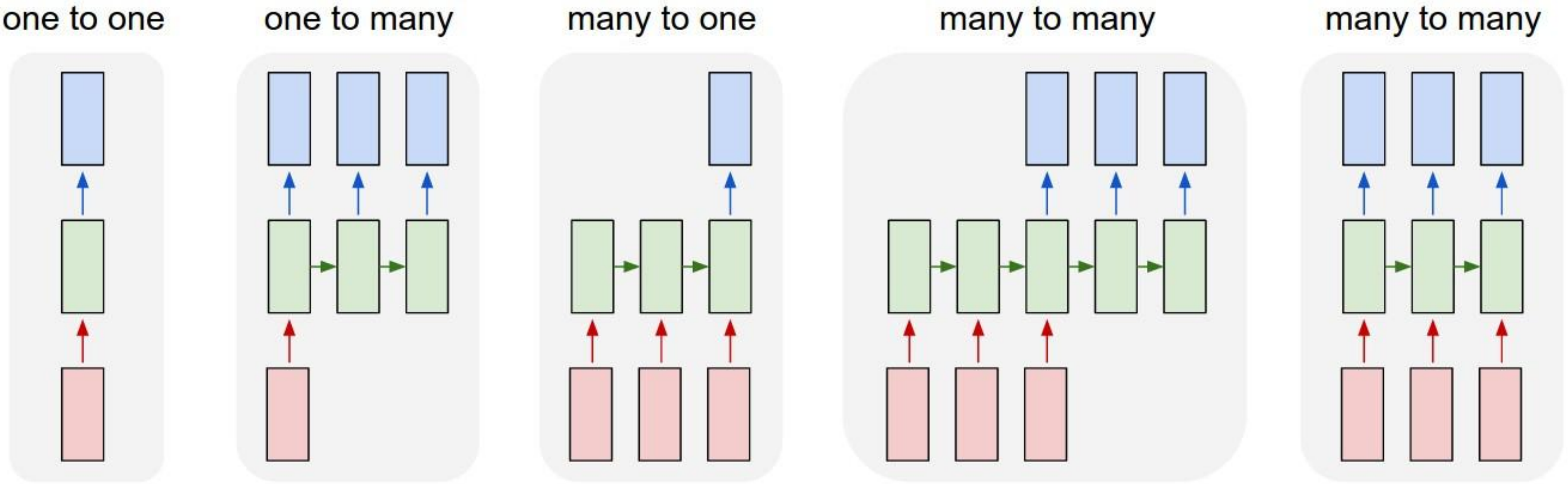


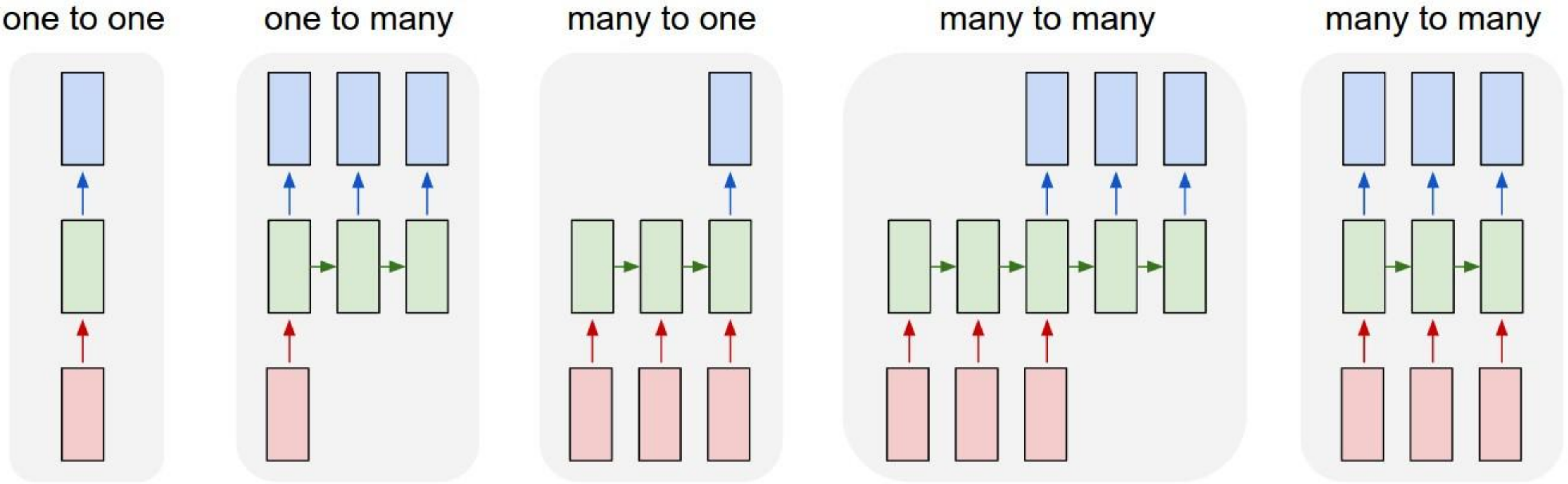
Image caption



Time series forecasting



Language translation

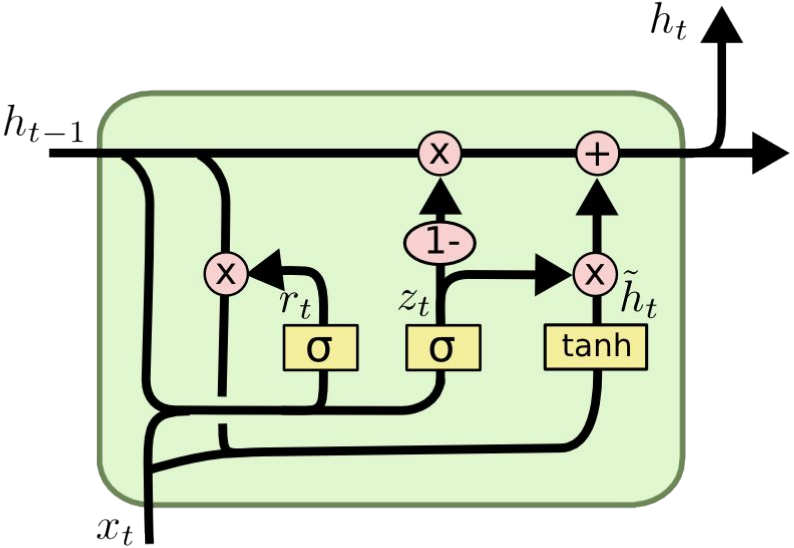


Frame-by-frame classification

The Unreasonable Effectiveness of Recurrent Neural Networks

Long Short Term Memory

LSTM = A complex RNN



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

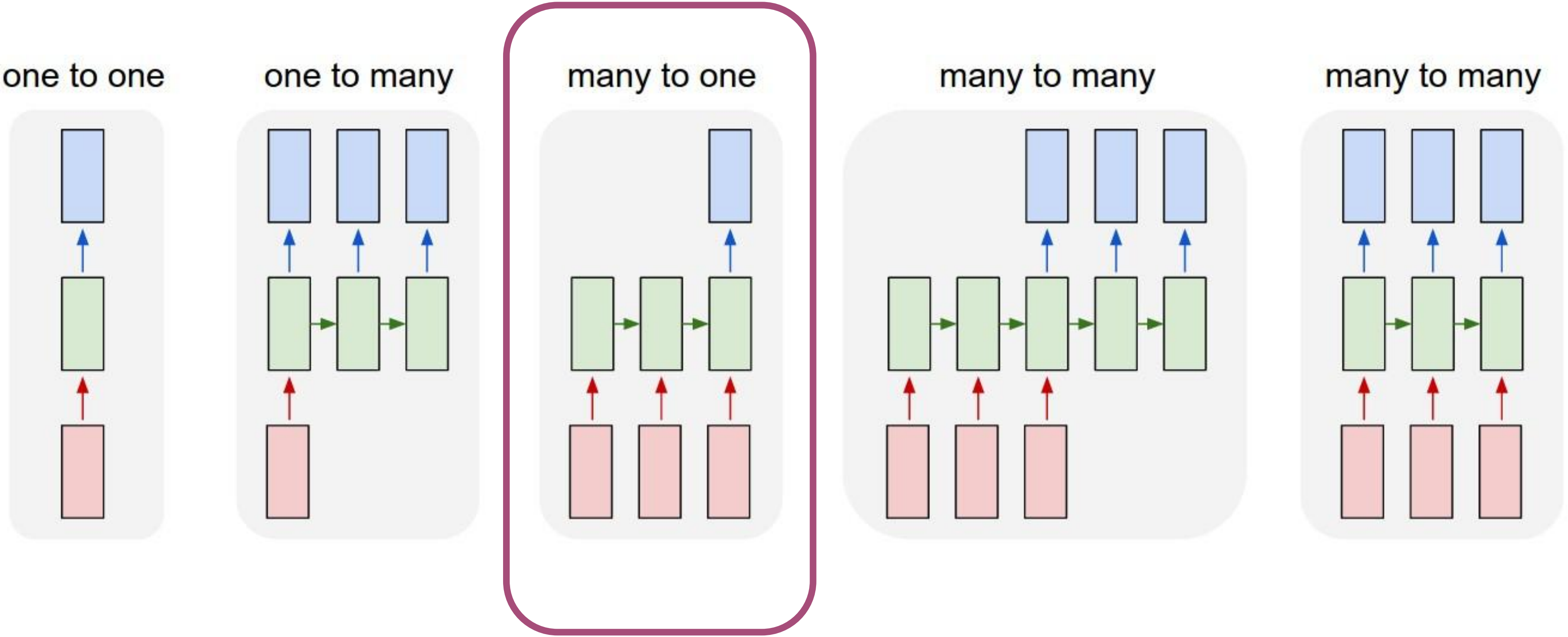
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Suggested Read - LSTM

LSTM + TS


one to one
one to many
many to one
many to many
many to many

Pros/cons

- no assumptions (stationarity...)
- good results
- non-linear models
- complex models (SARIMA are simple)
- require a lot of data (it depends...)

Time series shootout: ARIMA vs. LSTM

Conclusion

- LSTM works fine for tasks where ARIMA is known to work well, even
  - given very little data
  - without much hyperparameter tuning
  - without having more than one level of LSTM
  - for multi-step forecasts
- LSTM works great for multiple seasonality, again, without any tuning!

The Great Time Series Classification Bake Off: An Experimental Evaluation of Recently Proposed Algorithms. Extended Version