MDEC™ t- academy | Future Skills Academy for Emerging Technologies

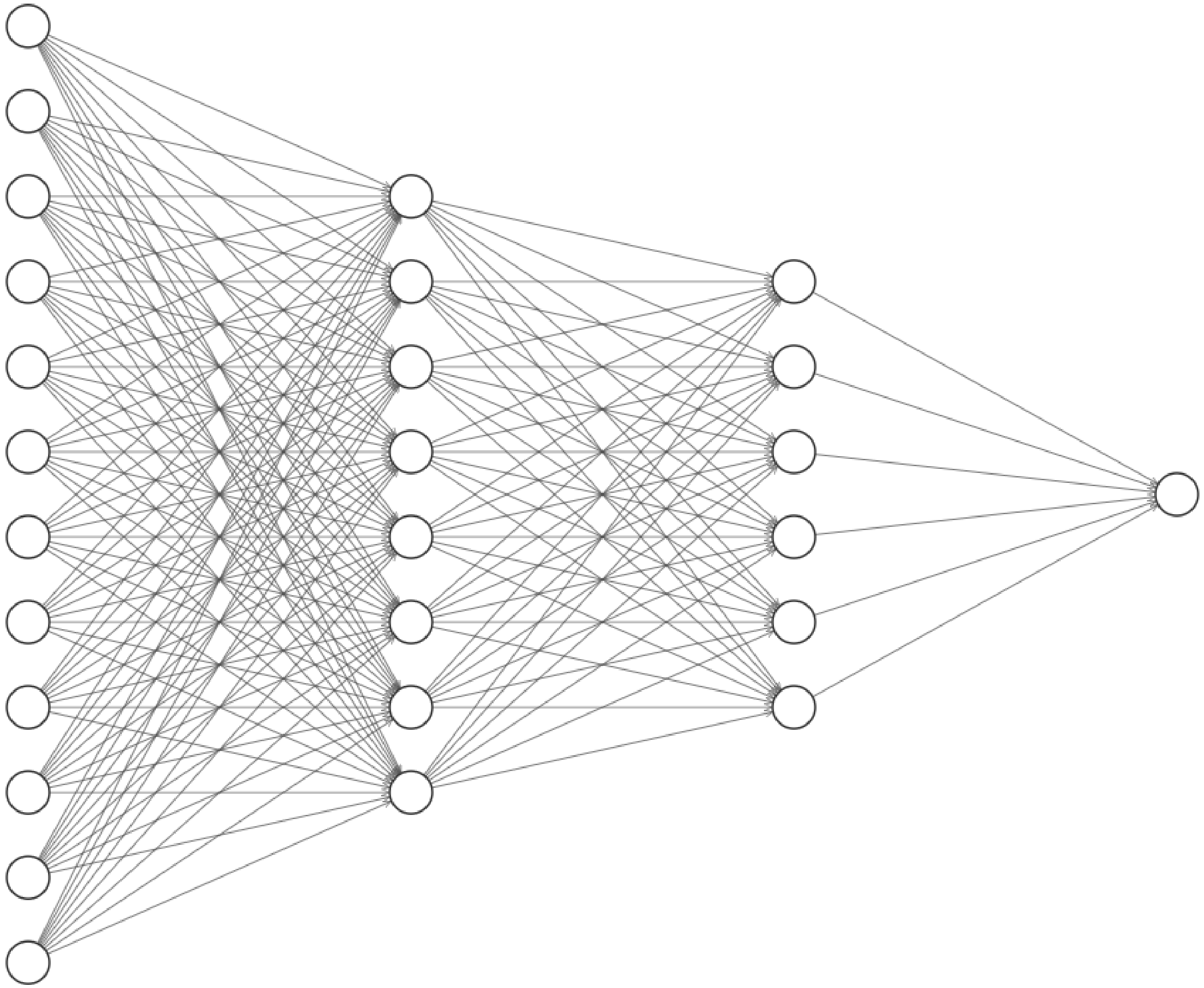
CERTIFIED DEEP LEARNING PROFESSIONAL (CDLP)

ICTC International Council for Technology Certifications

Digital Tech Faculty Expert (DTeX)

ICTC International Council for Technology Certifications

Recurrent Neural Networks



Deep Learning

Deep Learning

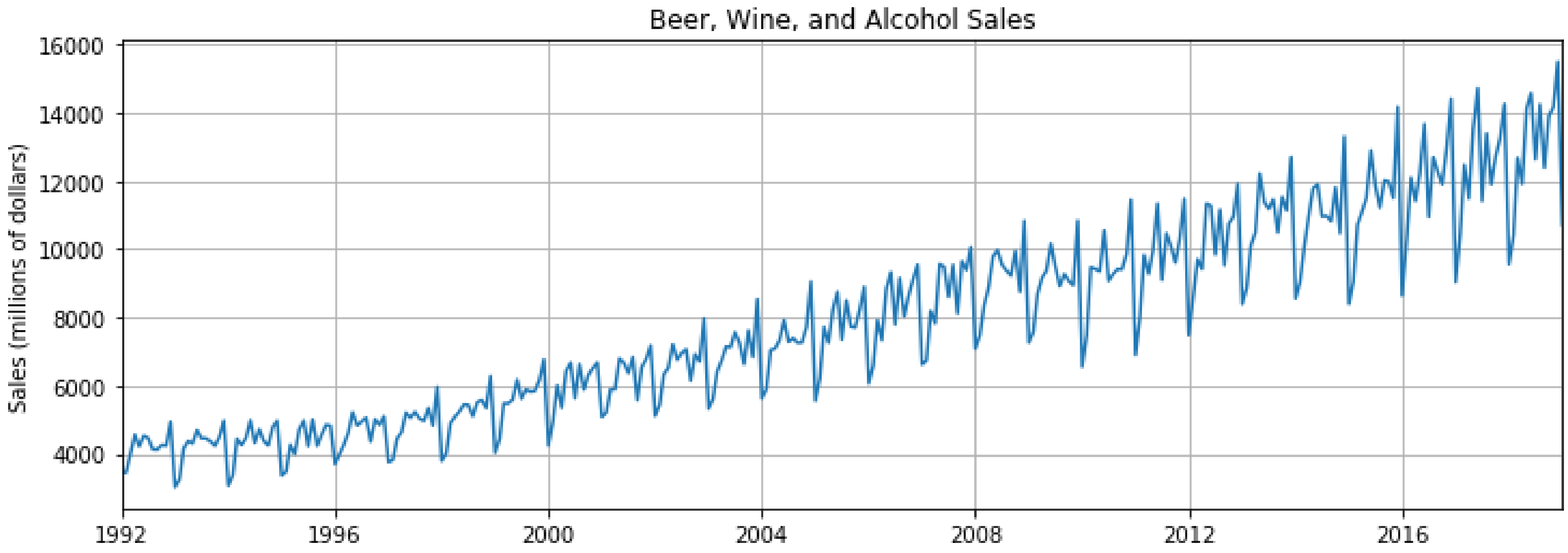
- We’ve used Neural Networks to solve Classification and Regression problems, but we still haven’t seen how Neural Networks can deal with sequence information.

Deep Learning

- Just as CNNs were more effective for use with 2D image data, RNNs are more effective for sequence data (e.g. time-stamped sales data, sequence of text, heart beat data, etc…)

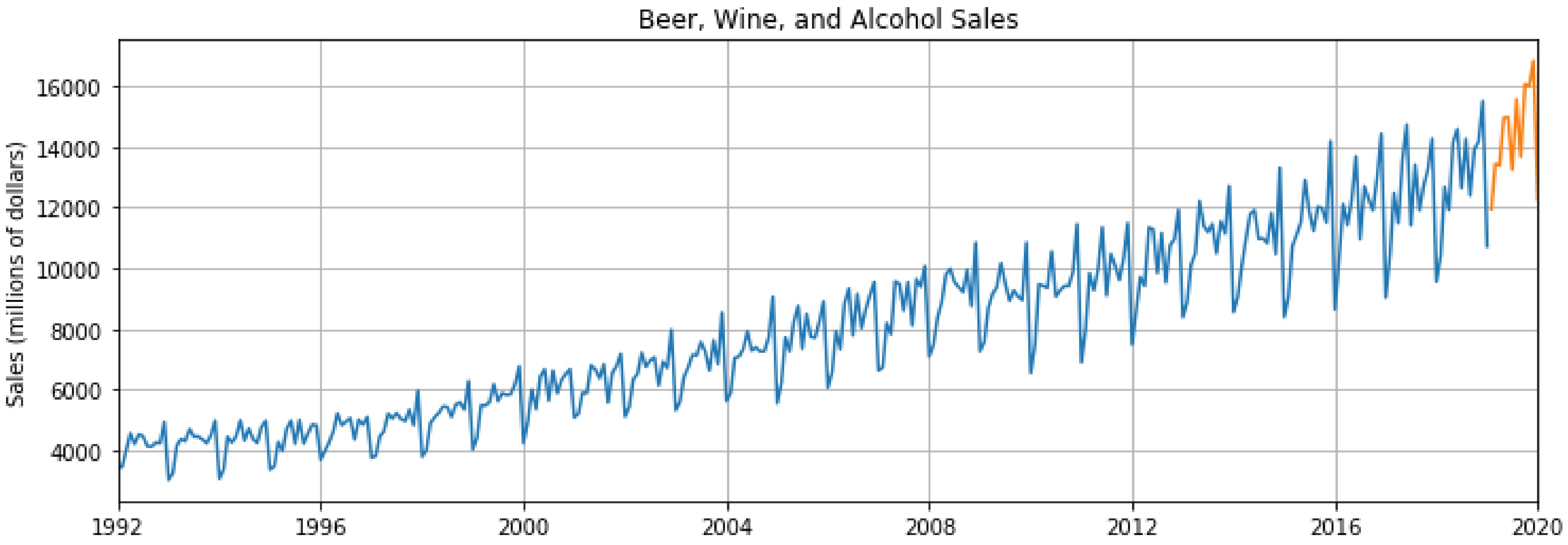
Deep Learning

- Time Series



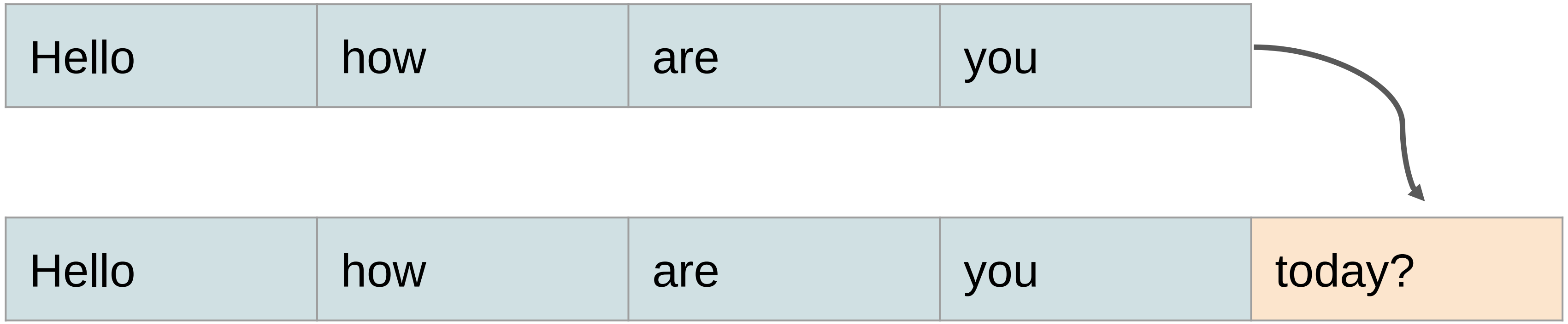
Deep Learning

- Time Series



Deep Learning

- Sequences



Deep Learning

Section Overview

- RNN Theory
- LSTMs and GRU Theory
- Basic Implementation of RNN
- Time Series with an RNN
- Exercise and Solution

ICTC International Council for Technology Certifications

Recurrent Neural Networks Theory

Deep Learning

Deep Learning

Examples of Sequences

- Time Series Data (Sales)
- Sentences
- Audio
- Car Trajectories
- Music

Deep Learning

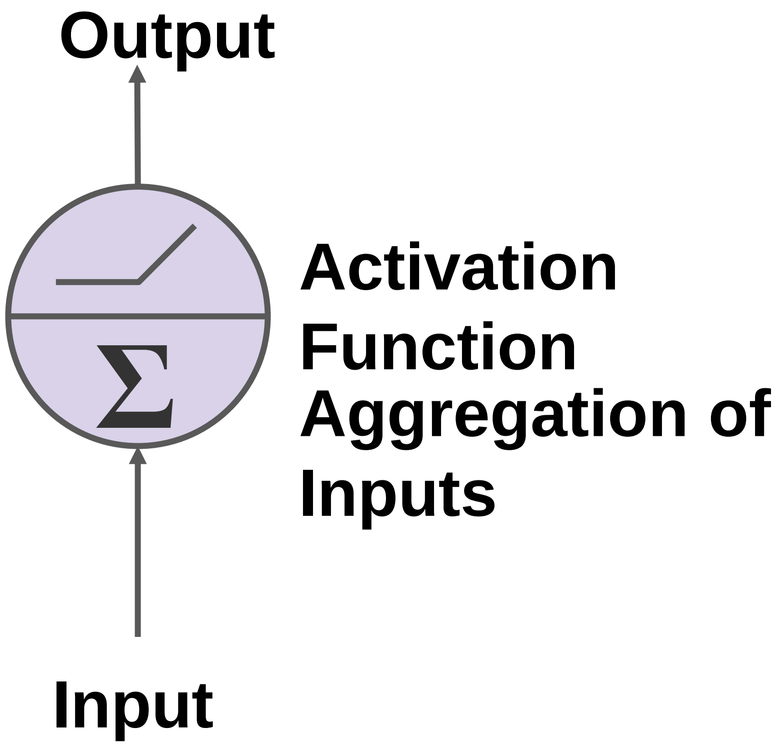
- Let’s imagine a sequence:
    - [1,2,3,4,5,6]
  - Would you be able to predict a similar sequence shifted one time step into the future?
    - [2,3,4,5,6,7]

Deep Learning

- To do this properly, we need to somehow let the neuron "know" about its previous history of outputs.
- One easy way to do this is to simply feed its output back into itself as an input!

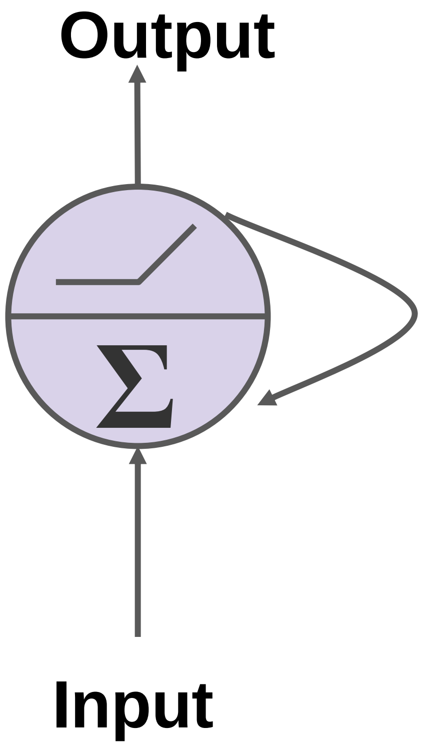
Deep Learning

- Normal Neuron in Feed Forward Network



Deep Learning

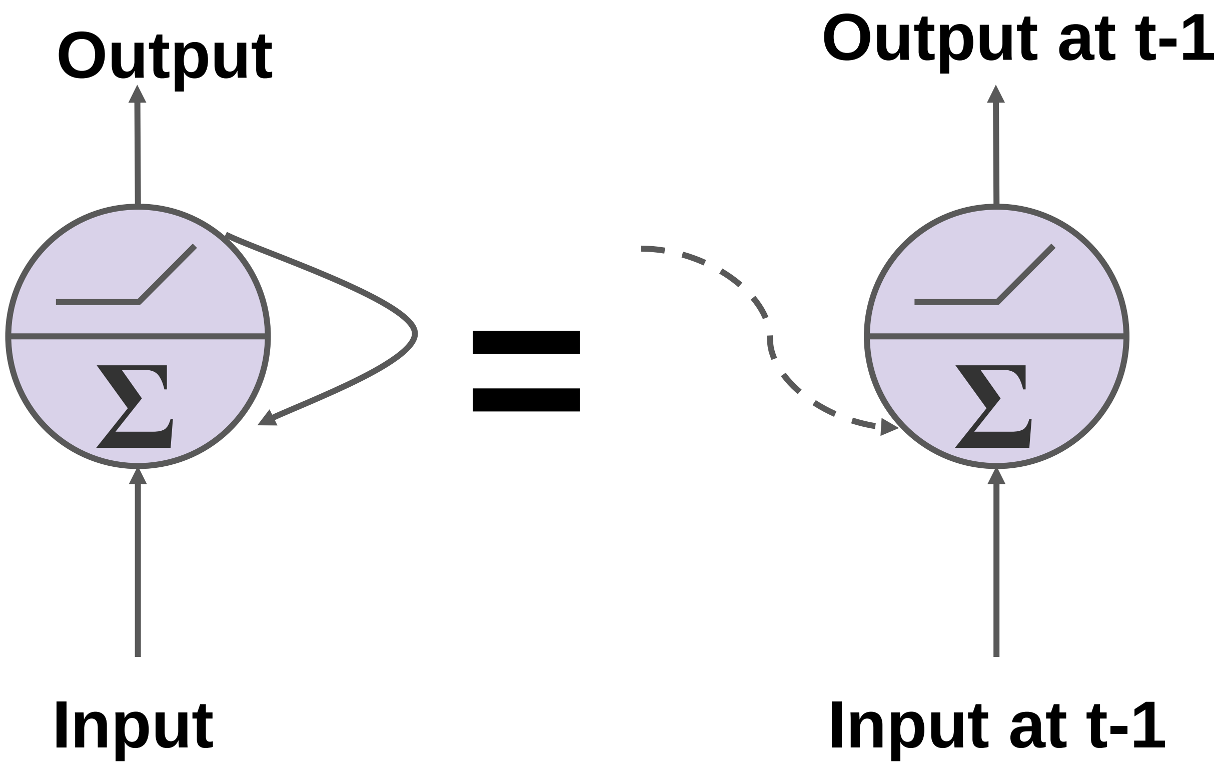
- Recurrent Neuron



- Sends output back to itself!
- Let’s see what this looks like over time!

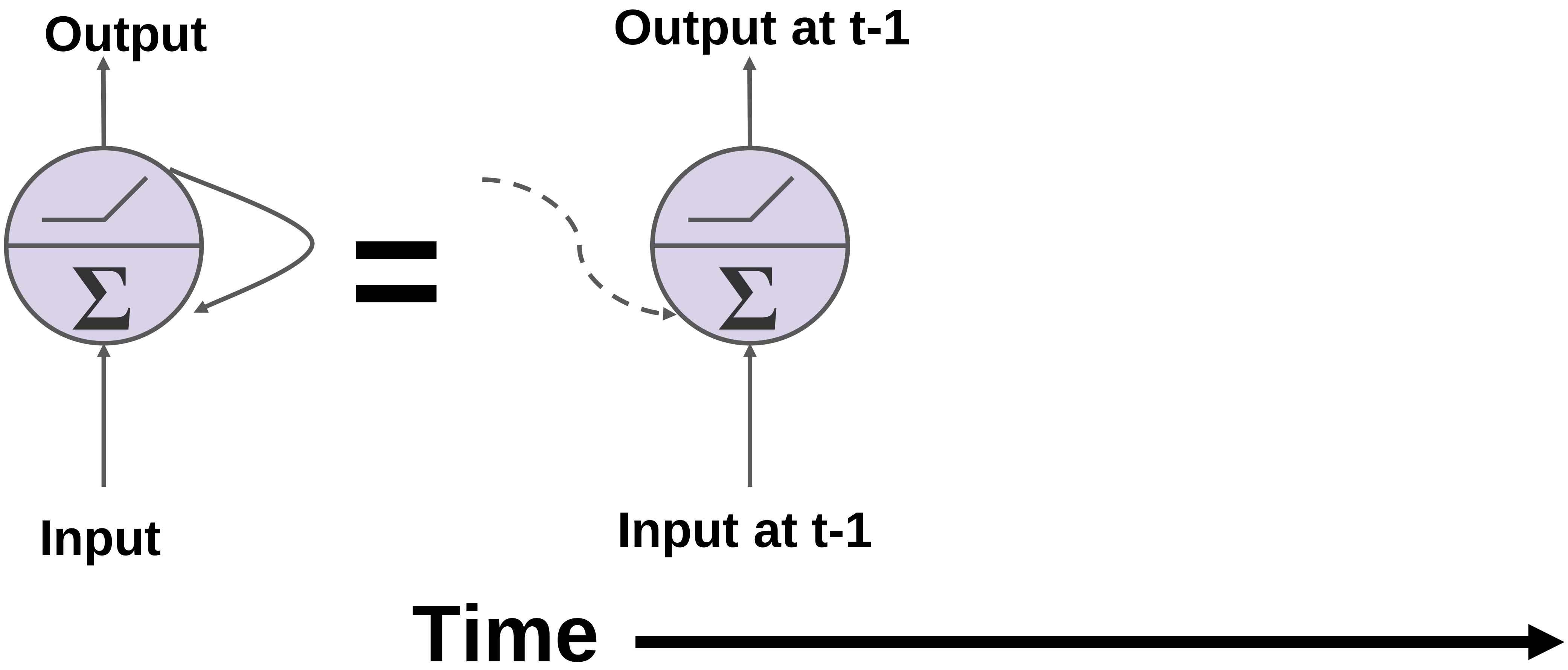
Deep Learning

- Recurrent Neuron



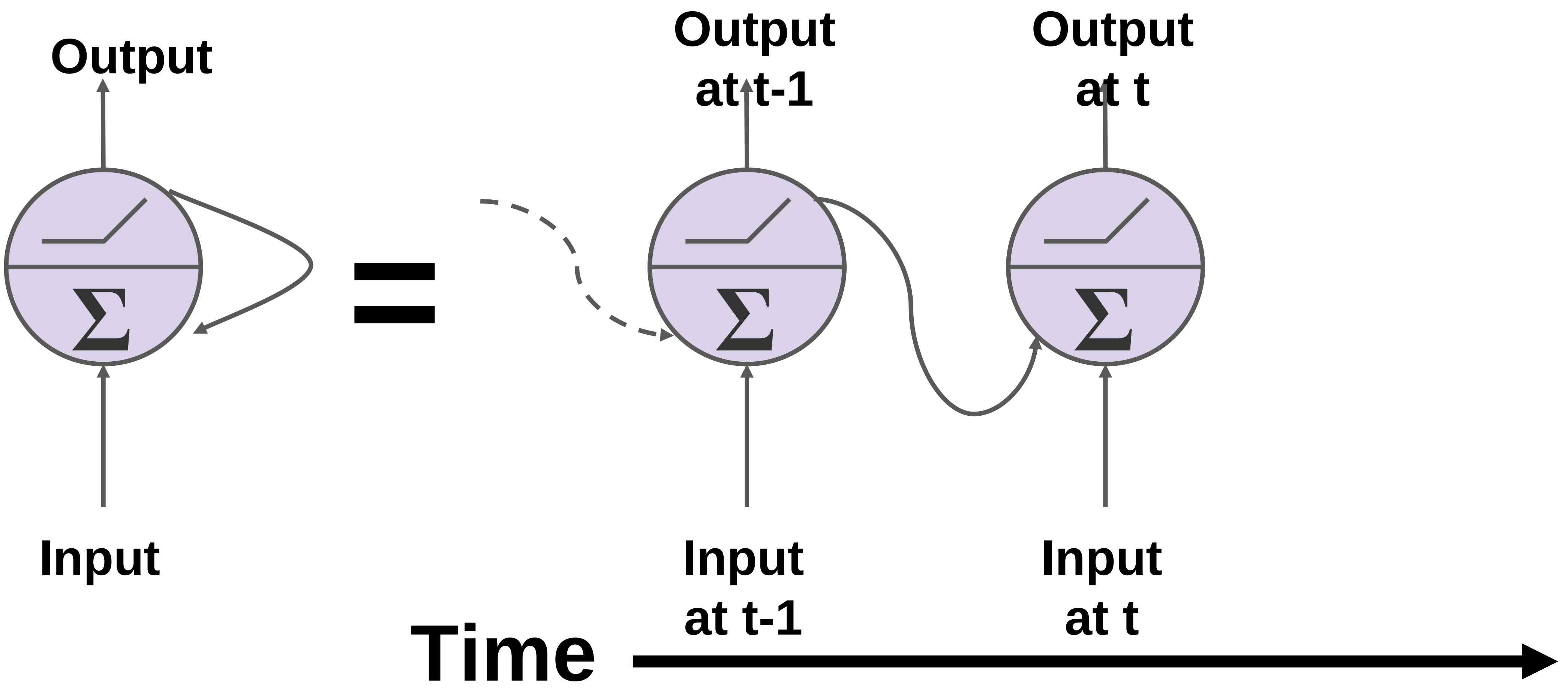
Deep Learning

- Recurrent Neuron



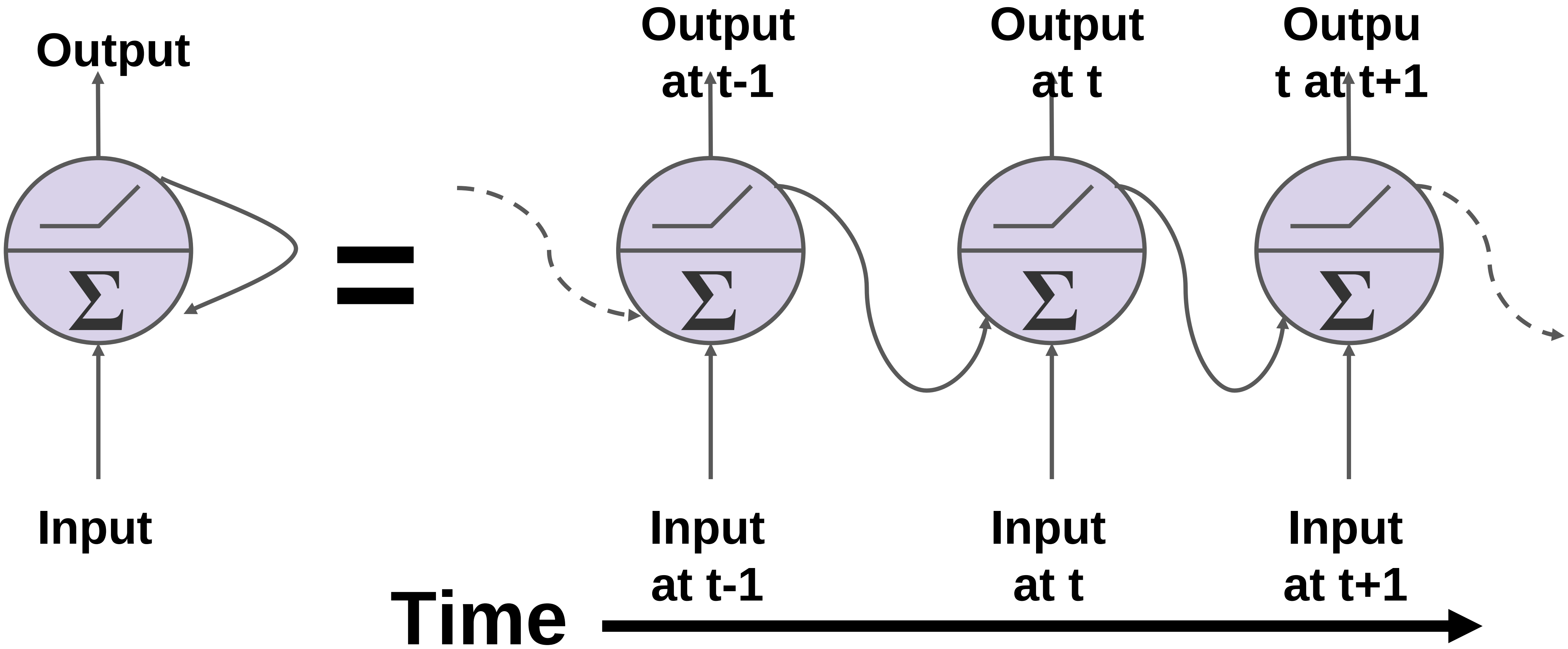
Deep Learning

- Recurrent Neuron



Deep Learning

- Recurrent Neuron



Deep Learning

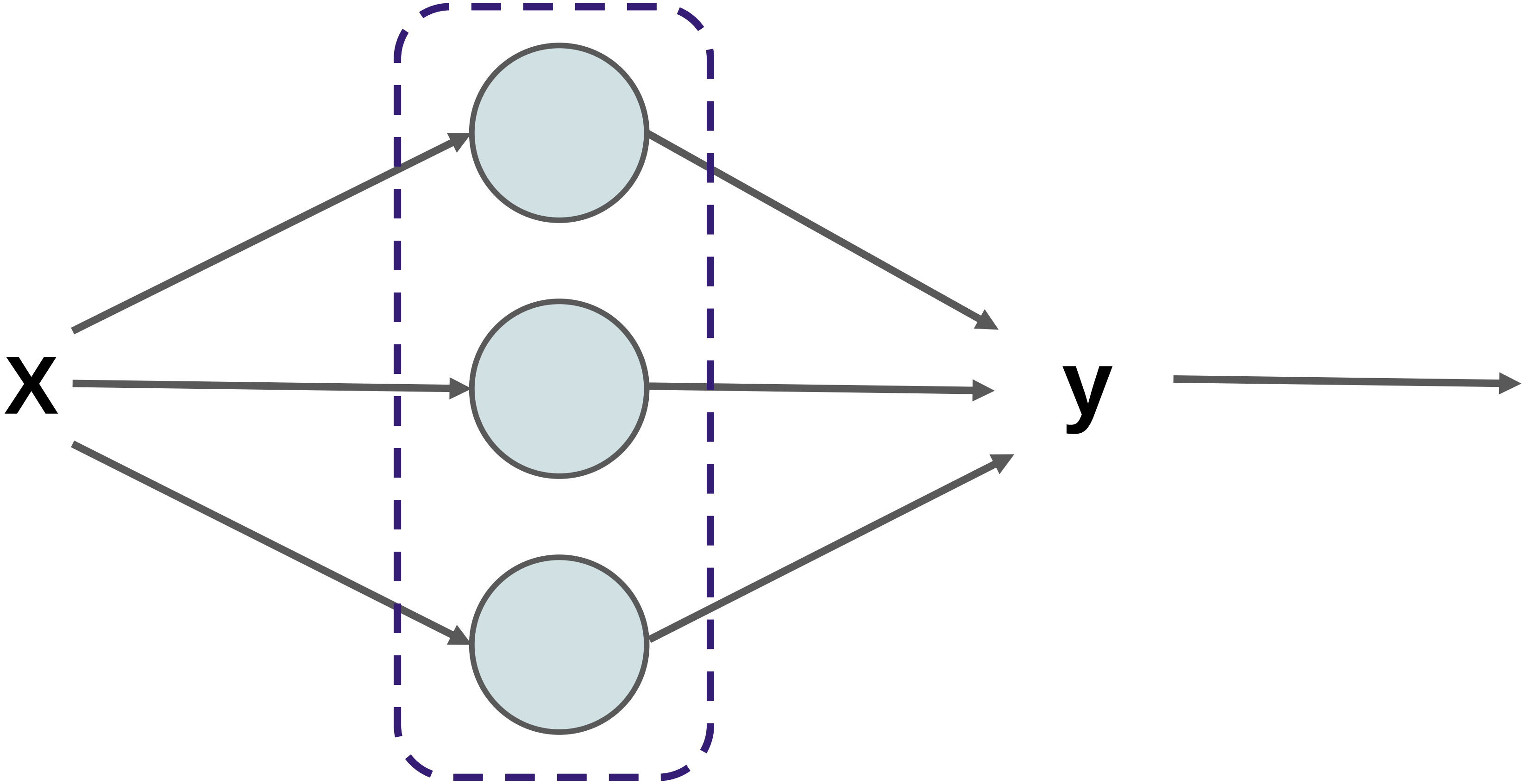
- Cells that are a function of inputs from previous time steps are also known as *memory cells.*
- RNN are also flexible in their inputs and outputs, for both sequences and single vector values.

Deep Learning

- We can also create entire layers of Recurrent Neurons...

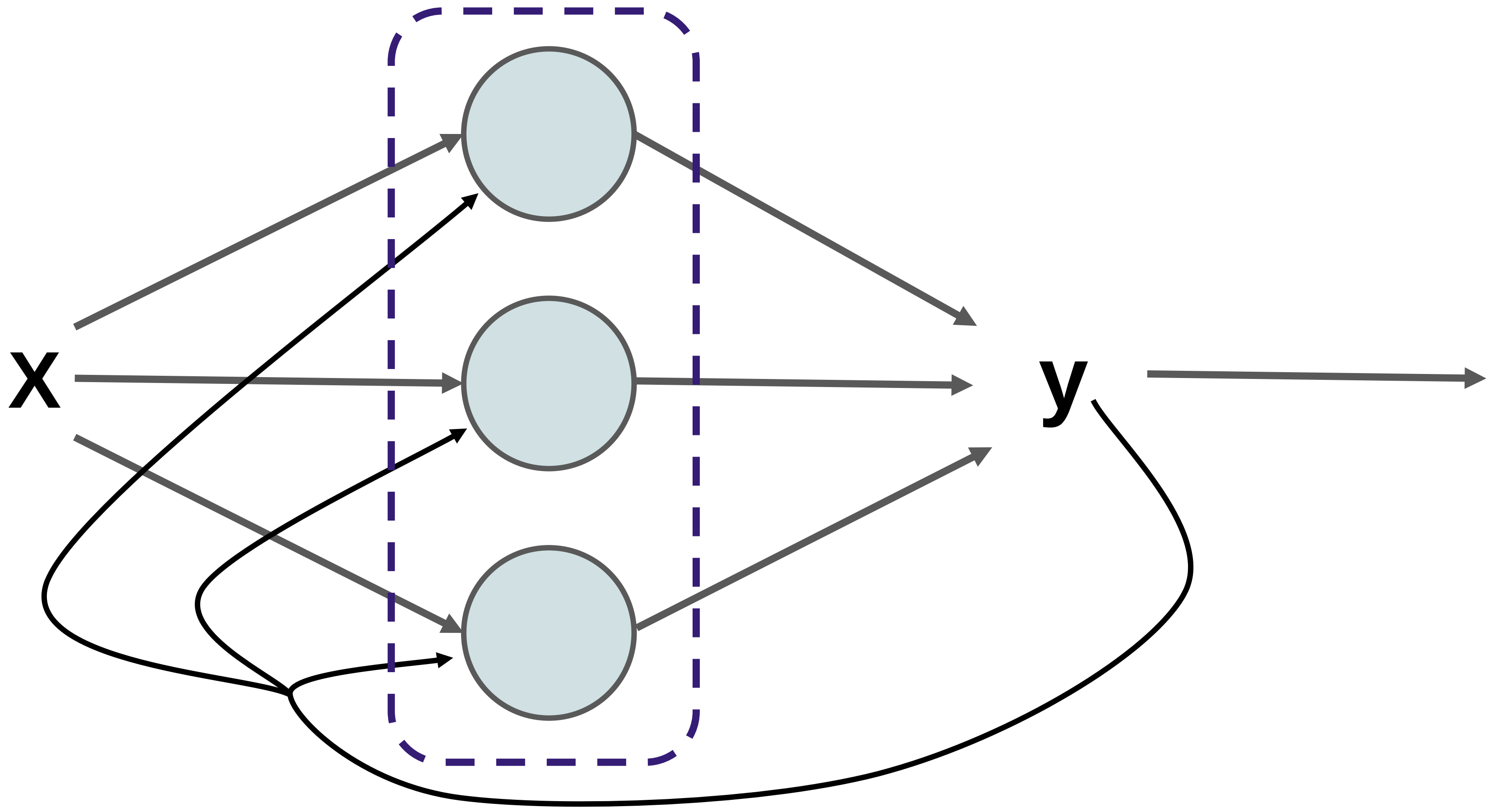
Deep Learning

- ANN Layer with 3 Neurons:



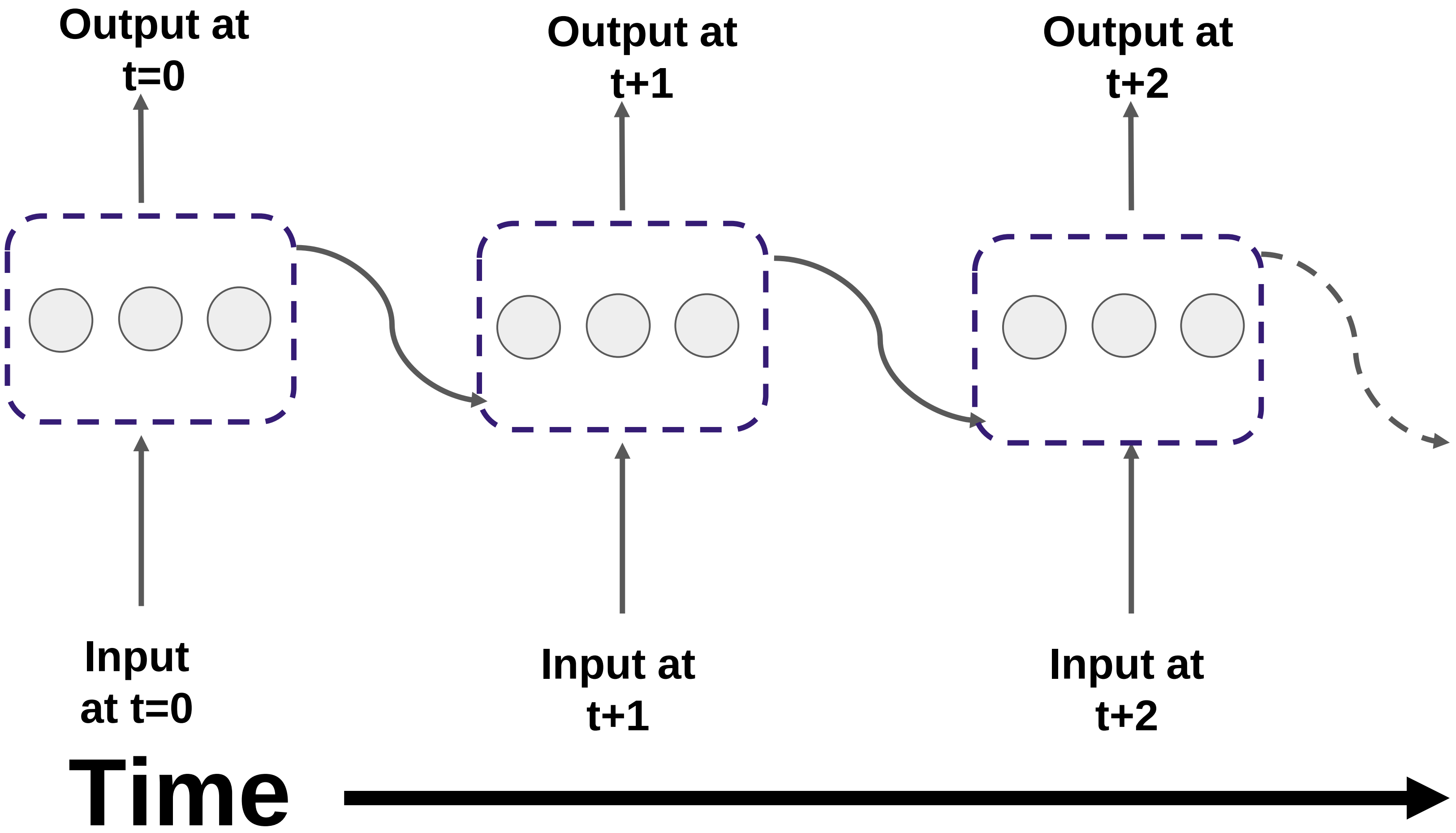
Deep Learning

- RNN Layer with 3 Neurons:



Deep Learning

- "Unrolled" layer.

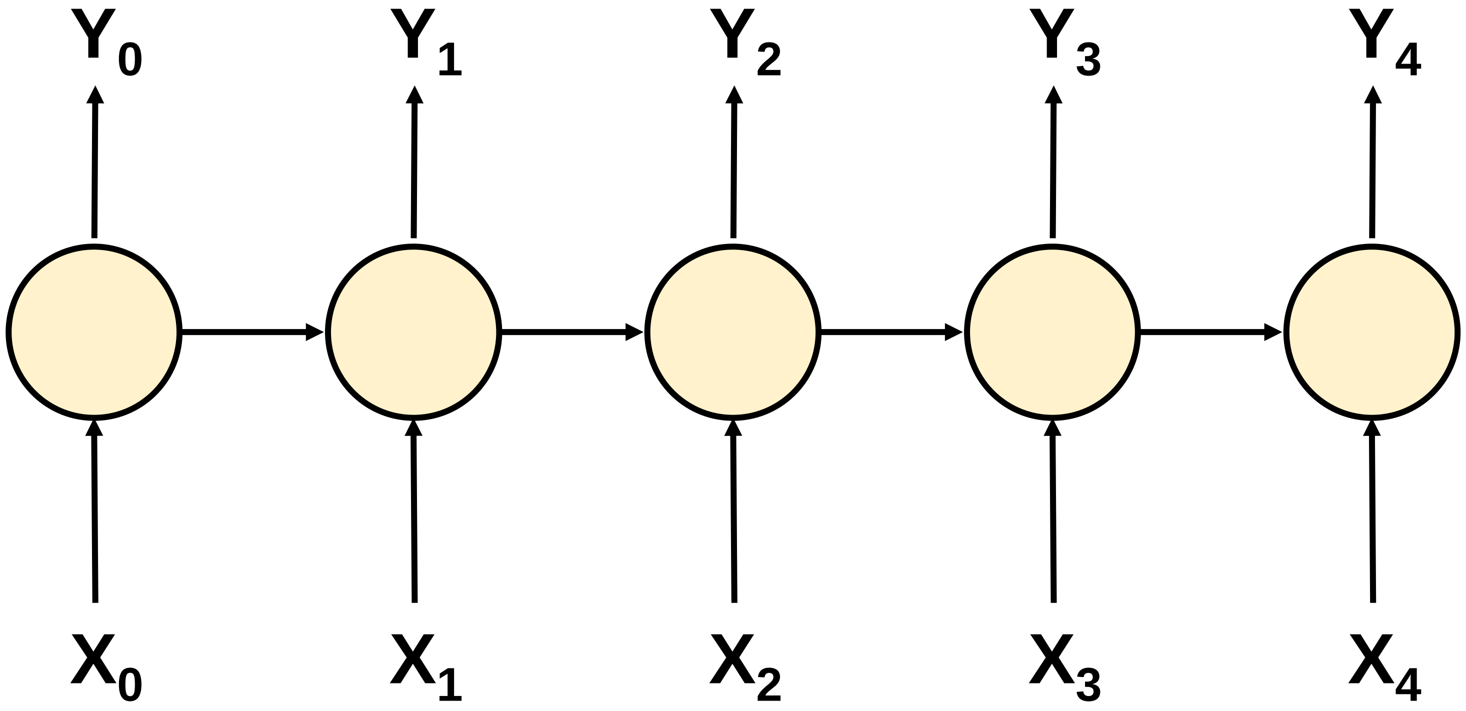


Deep Learning

- RNN are also very flexible in their inputs and outputs.
- Let’s see a few examples.

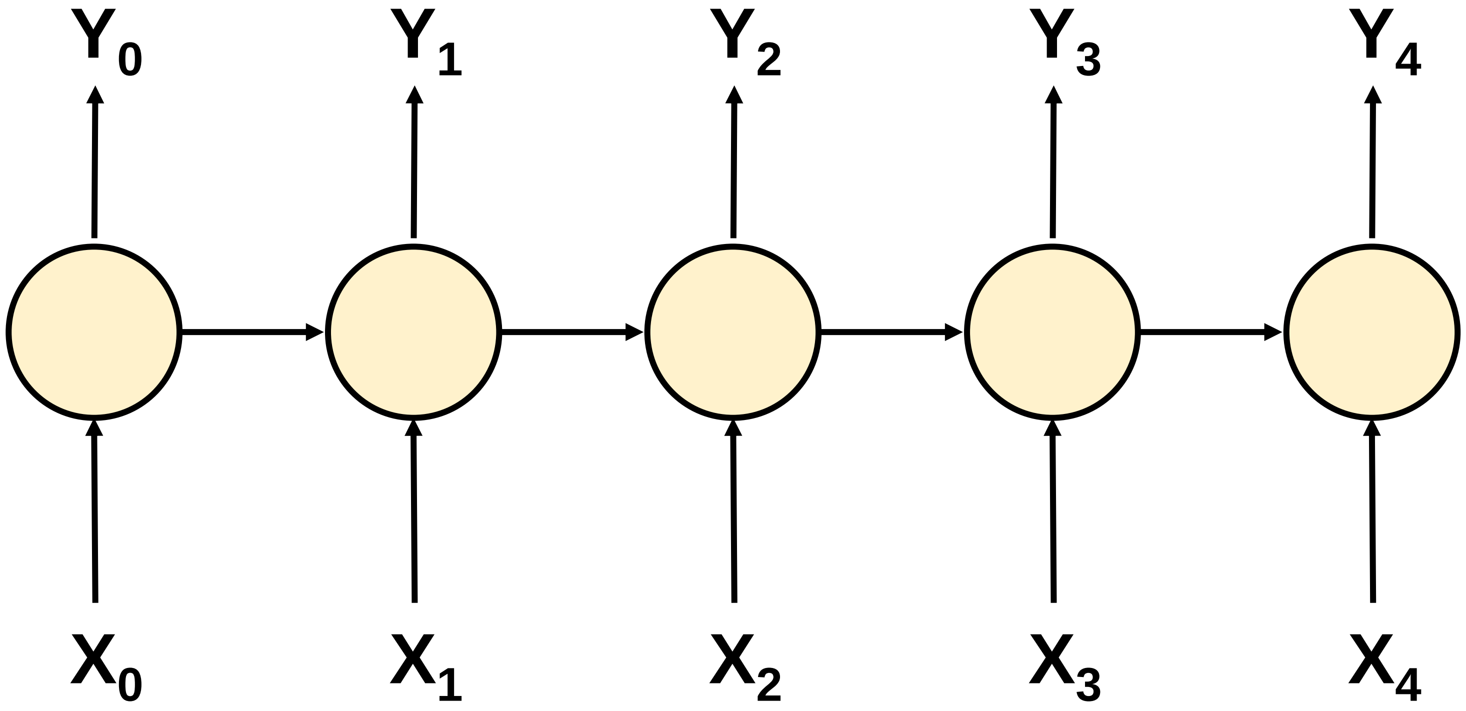
Deep Learning

- Sequence to Sequence (Many to Many)



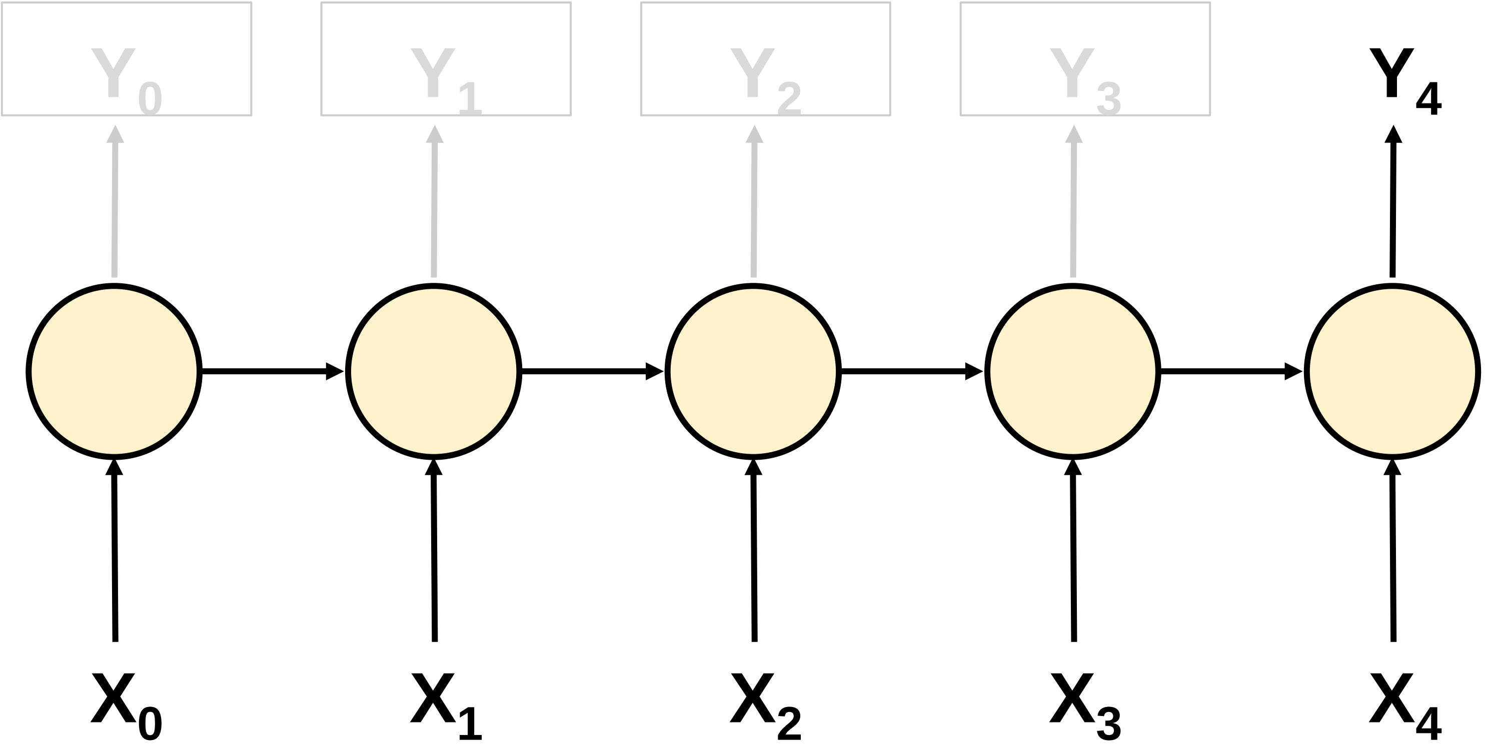
Deep Learning

- Given 5 previous words, predict the next 5



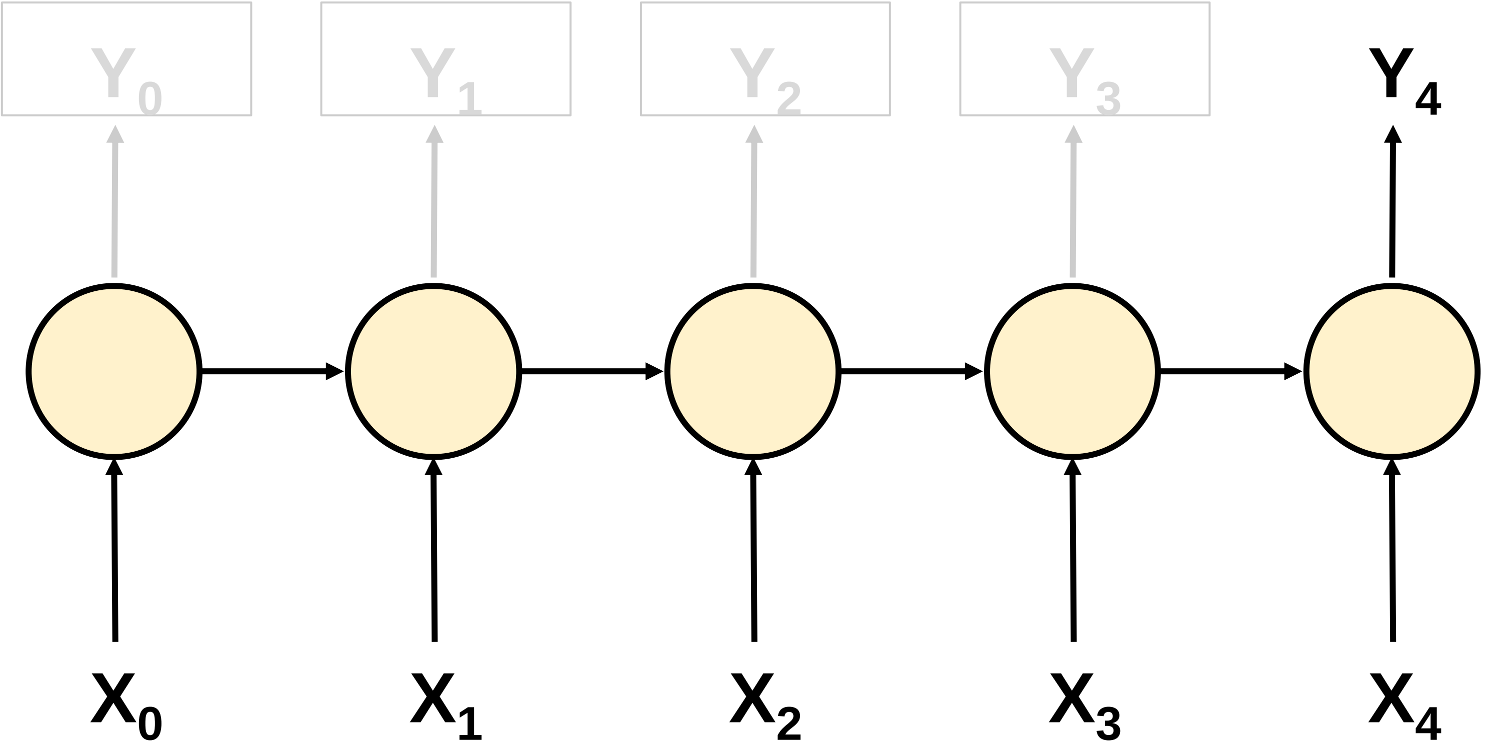
Deep Learning

- Sequence to Vector (Many to One)



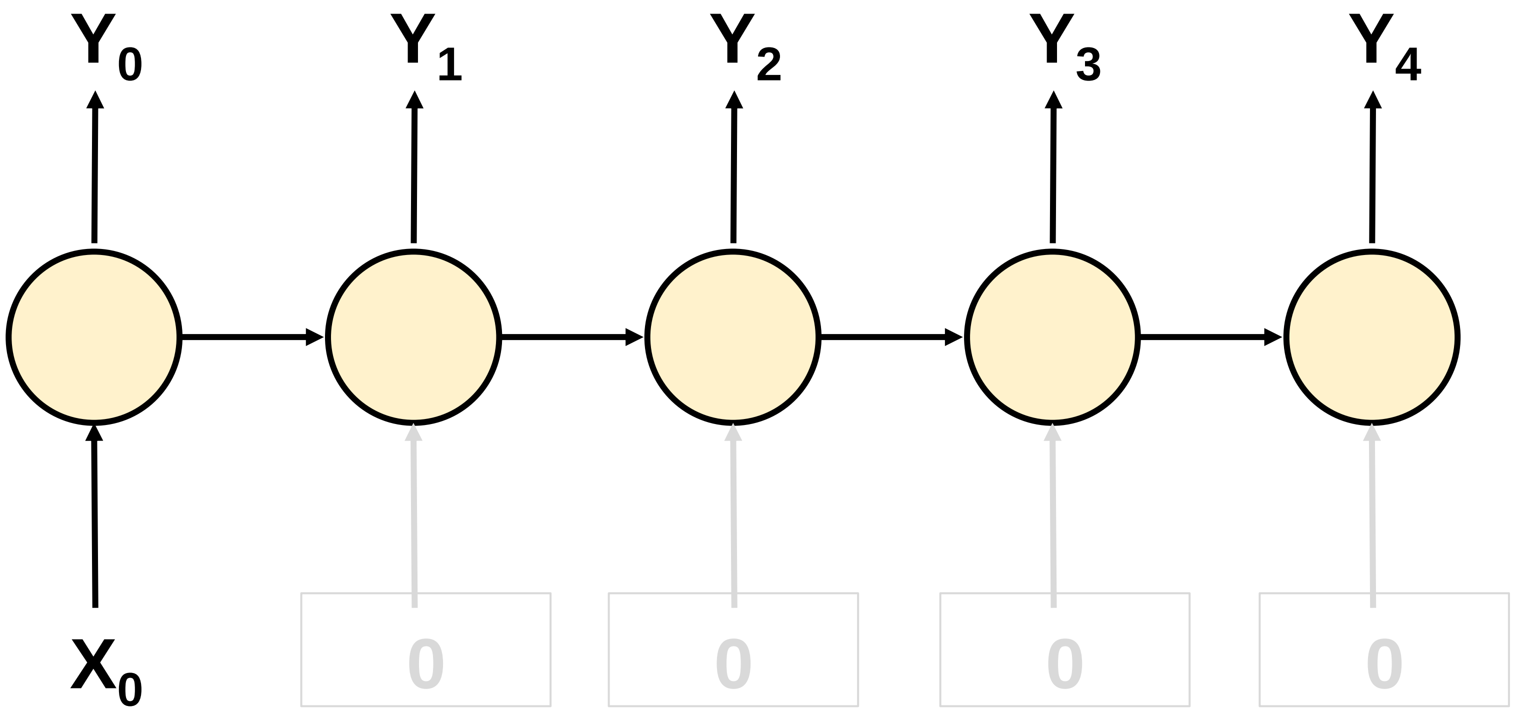
Deep Learning

- Given 5 previous words, predict next word



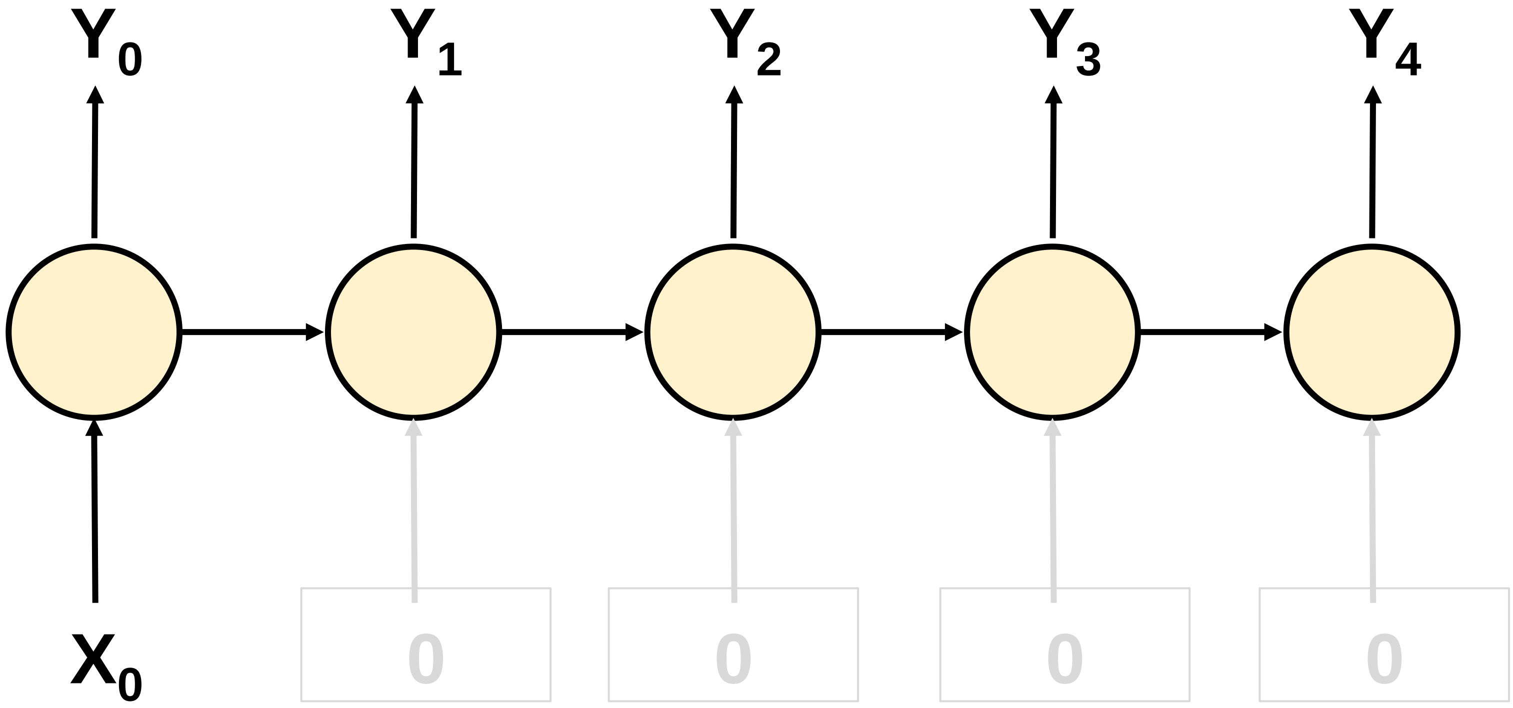
Deep Learning

- Vector to Sequence (One to Many)



Deep Learning

- Given 1 word predict the next 5 words



Deep Learning

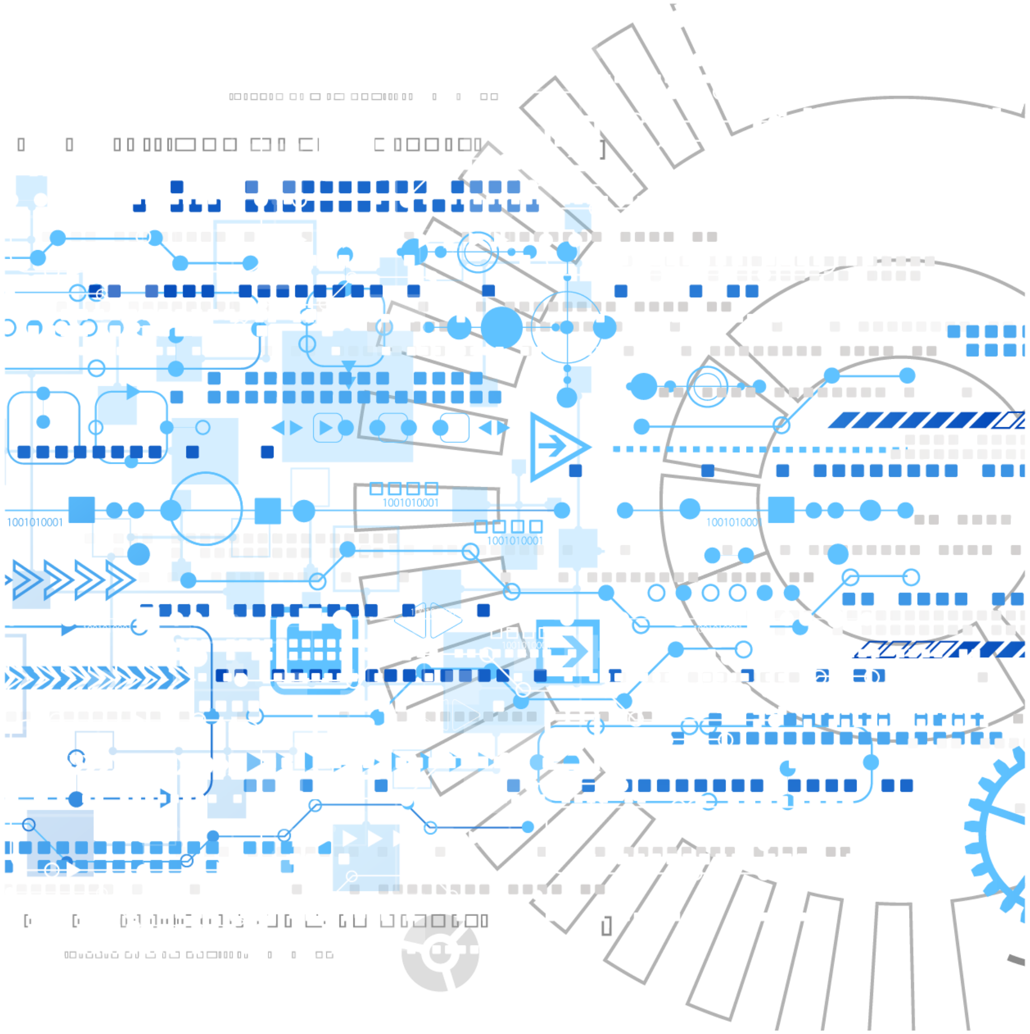
- A basic RNN has a major disadvantage, we only really "remember" the previous output.
- It would be great it we could keep track of longer history, not just short term history.

Deep Learning

- Another issue that arises during training is the “vanishing gradient”.
- Let’s explore vanishing gradients in more detail before moving on to discussing LSTM (Long Short Term Memory Units).

ICTC International Council for Technology Certifications

Exploding and Vanishing Gradients



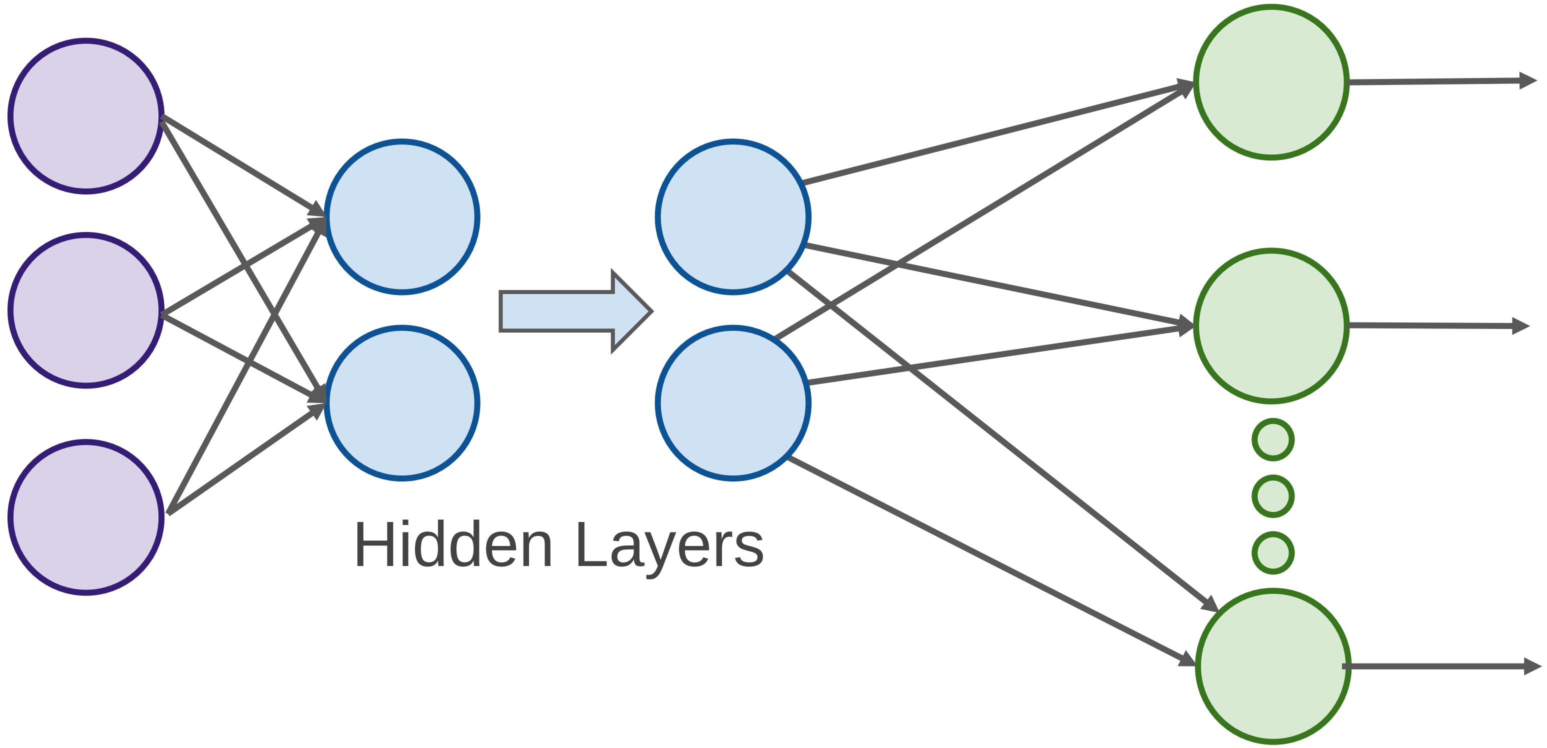
Deep Learning

Deep Learning

- As our networks grow deeper and more complex, we have 2 issues arise:
    - Exploding Gradients
    - Vanishing Gradients
  - Recall that the gradient is used in our calculation to adjust weights and biases in our network.

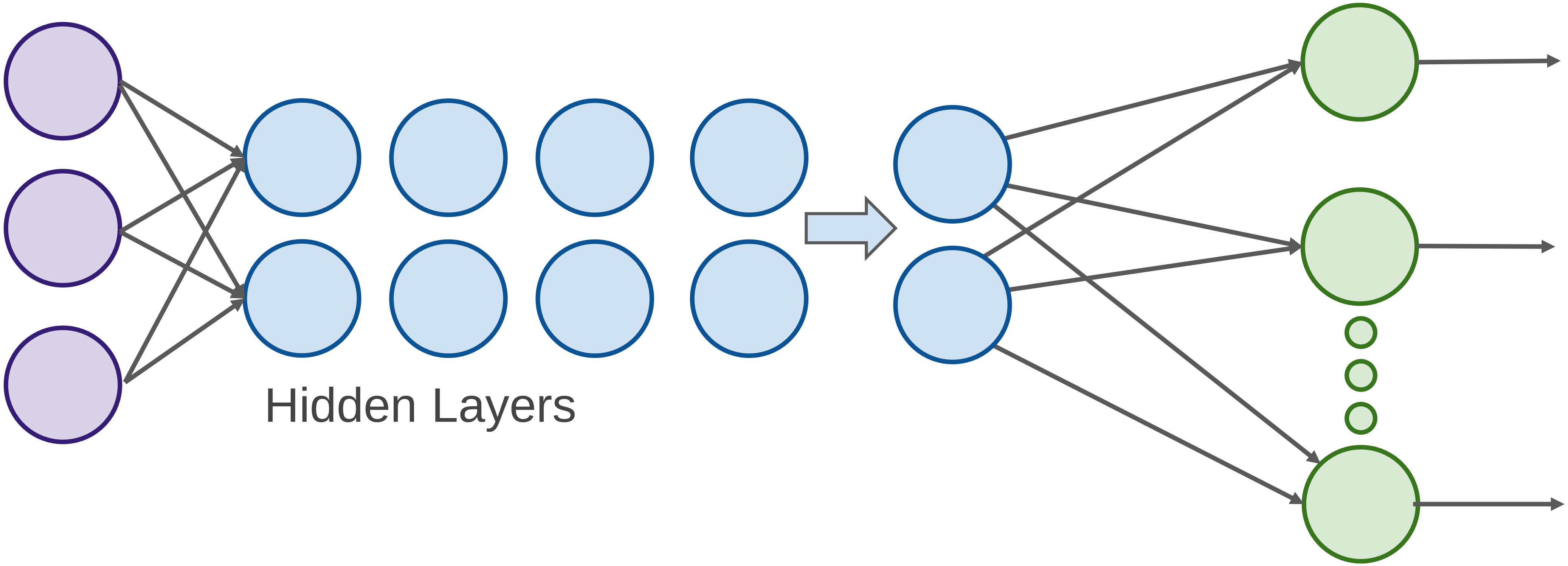
Deep Learning

Let’s think about a network.



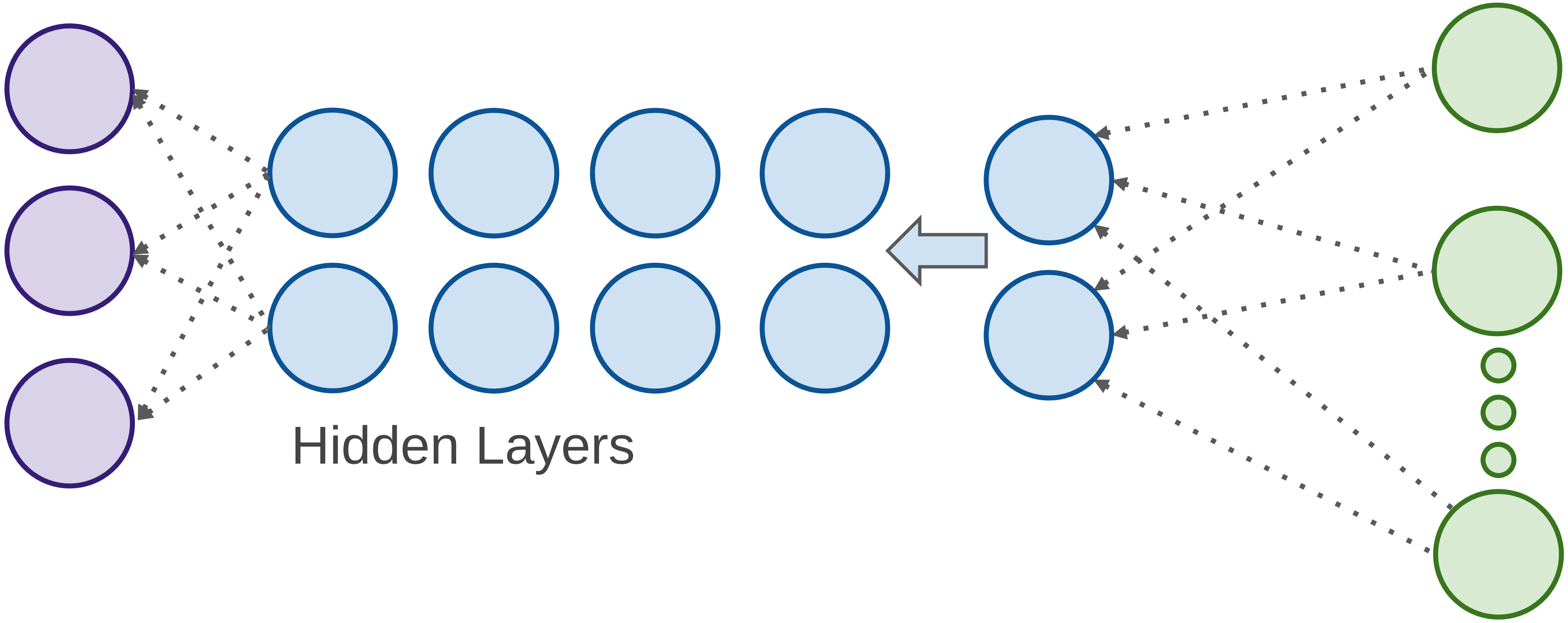
Deep Learning

For complex data we need deep networks



Deep Learning

Issues can arise during backpropagation



Deep Learning

- Backpropagation goes backwards from the output to the input layer, propagating the error gradient.
- For deeper networks issues can arise from backpropagation, vanishing and exploding gradients!

Deep Learning

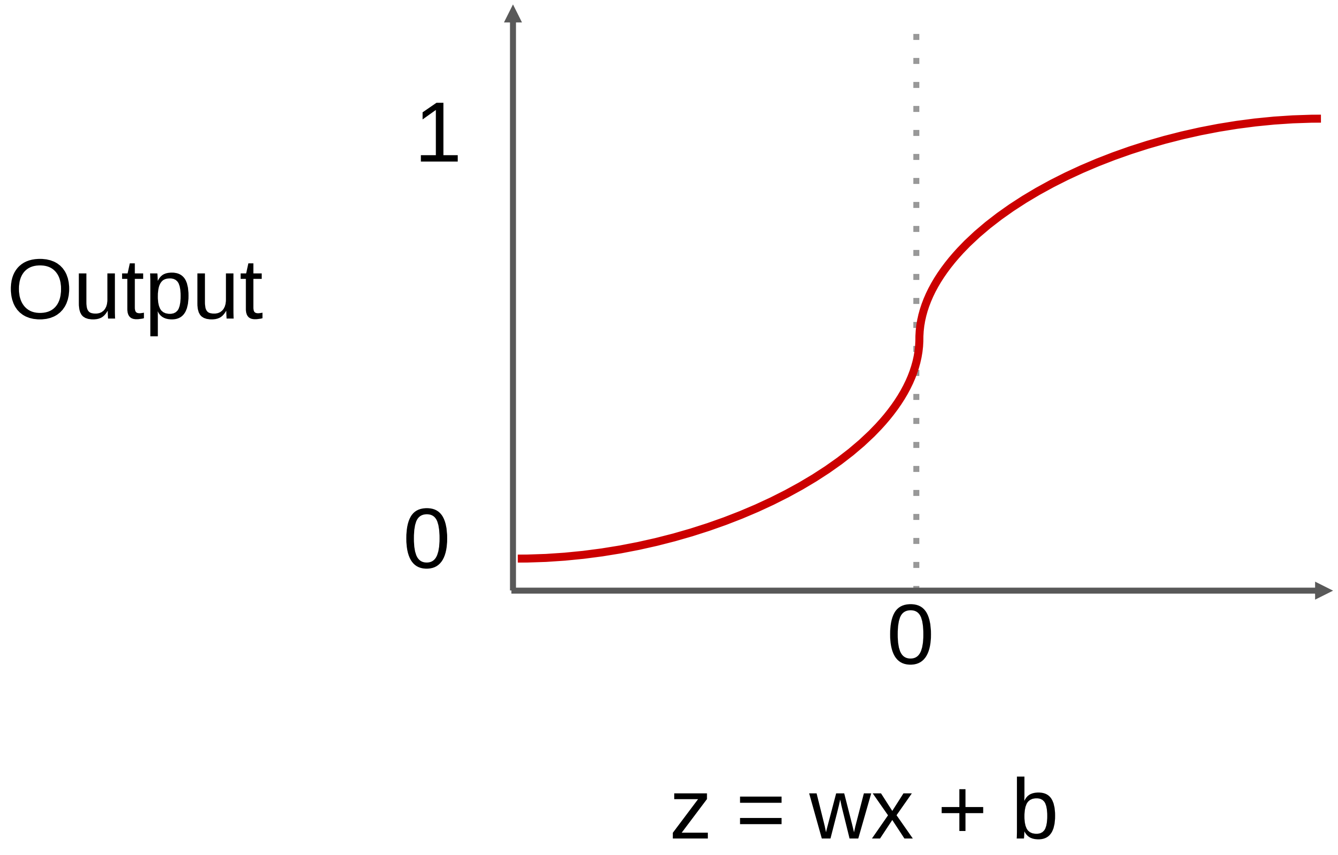
- As you go back to the “lower” layers, gradients often get smaller, eventually causing weights to never change at lower levels.
- The opposite can also occur, gradients explode on the way back, causing issues.

Deep Learning

- Let’s discuss why this might occur and how we can fix it.
- Then in the next lecture we’ll discuss how these issues specifically affect RNN and how to use LSTM and GRU to fix them.

Deep Learning

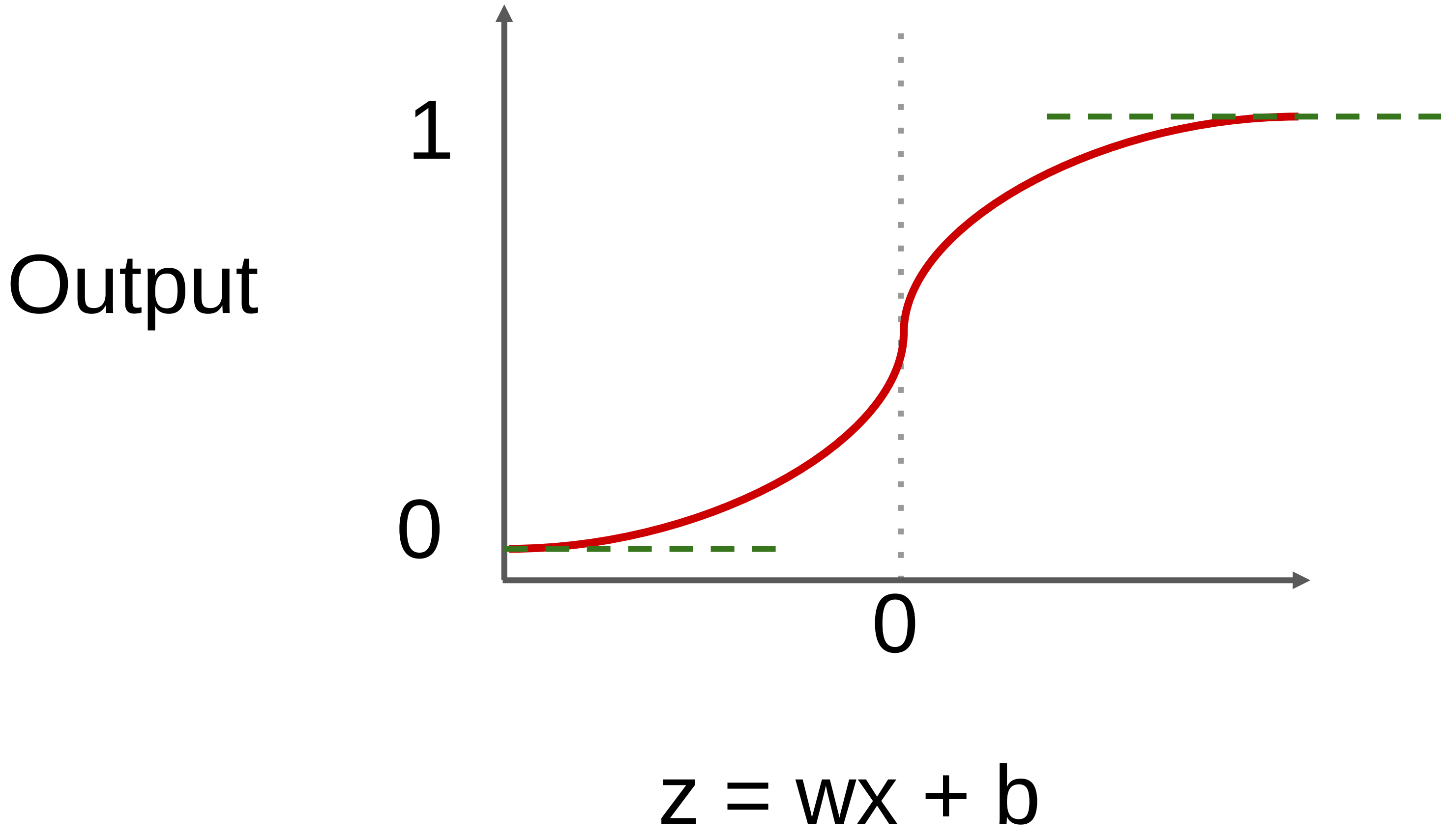
Why does this happen?



$$f(x) = \frac{1}{1 + e^{-(x)}}$$

Deep Learning

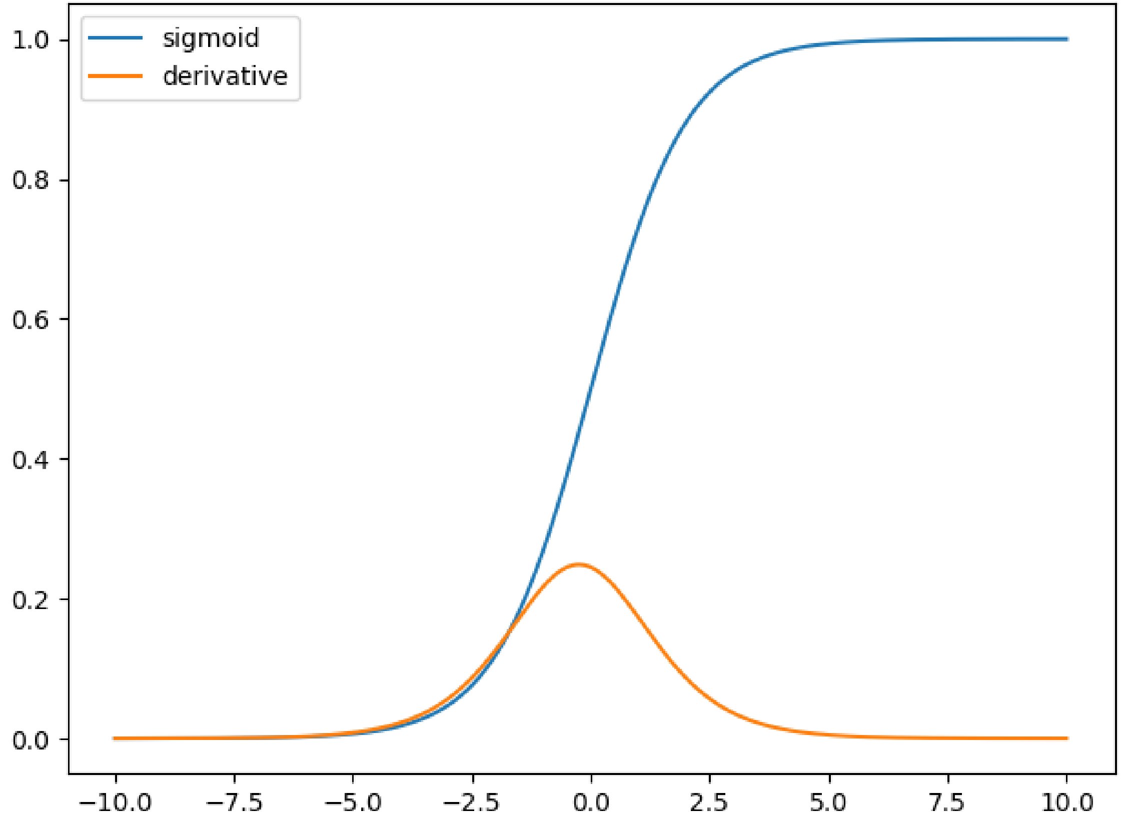
Why does this happen?



$$f(x) = \frac{1}{1+e^{-(x)}}$$

Deep Learning

The derivative can be much smaller!



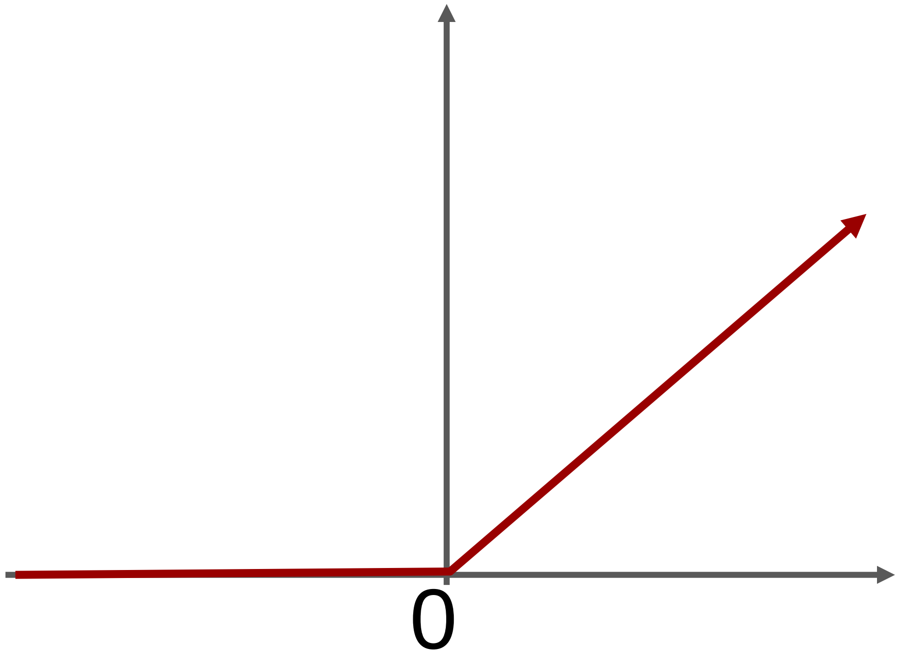
Deep Learning

- When **n** hidden layers use an activation like the sigmoid function, **n** small derivatives are multiplied together.
- The gradient could decrease exponentially as we propagate down to the initial layers.

Deep Learning

Using Different Activation Functions

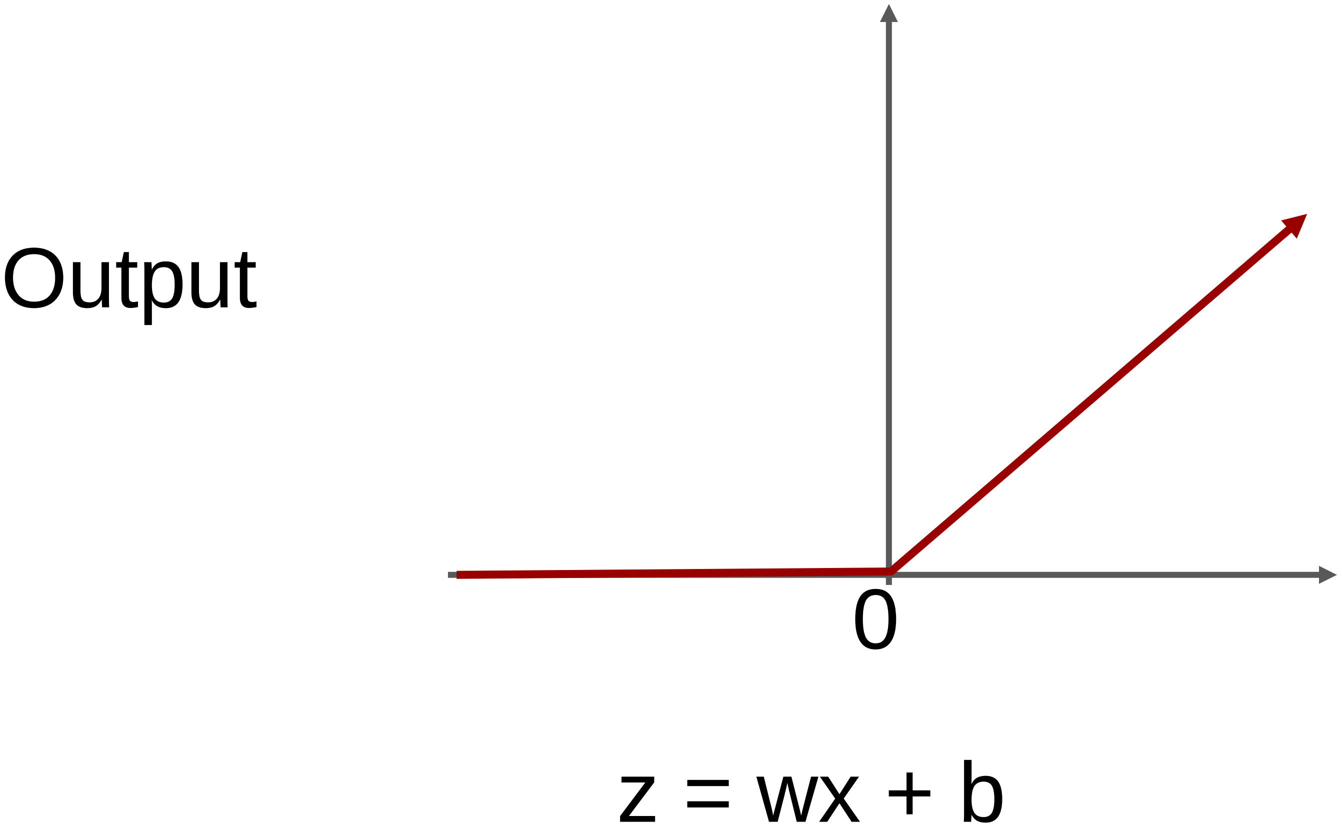
Output



z = wx + b

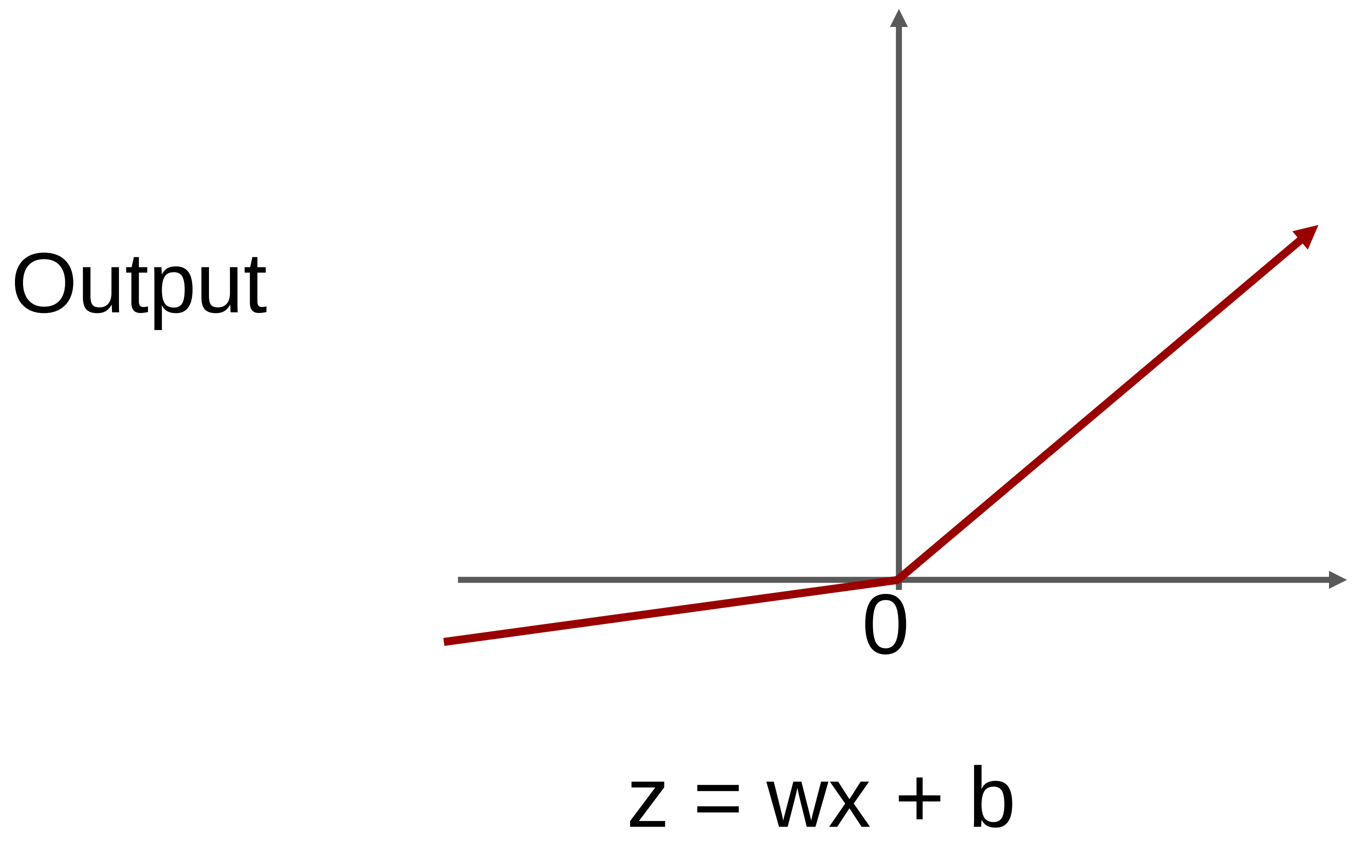
Deep Learning

The ReLu doesn't saturate positive values.



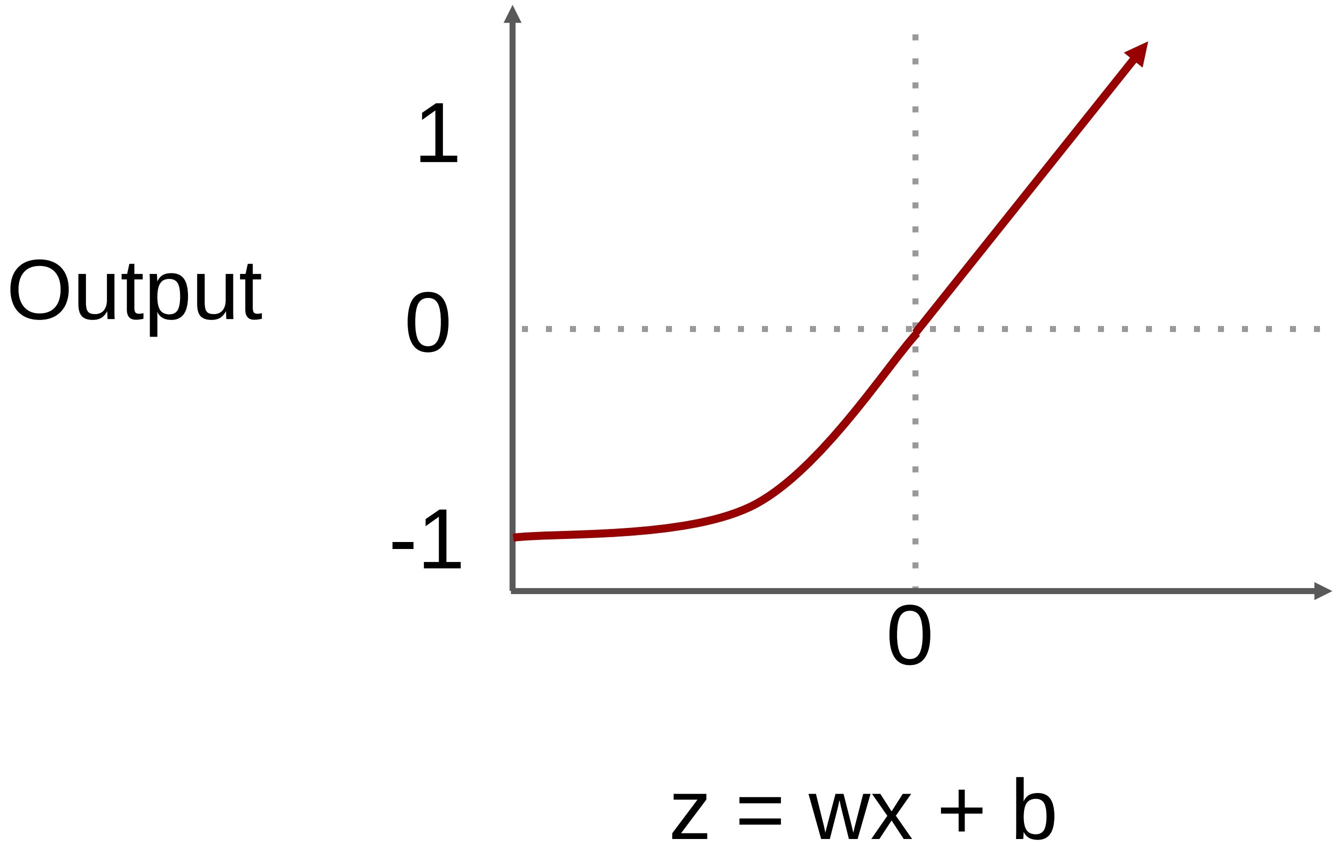
Deep Learning

"Leaky" ReLU



Deep Learning

Exponential Linear Unit (ELU)



Deep Learning

- Another solution is to perform batch normalization, where your model will normalize each batch using the batch mean and standard deviation.

Deep Learning

- Choosing different initialization of weights can also help alleviate these issues (Xavier Initialization).

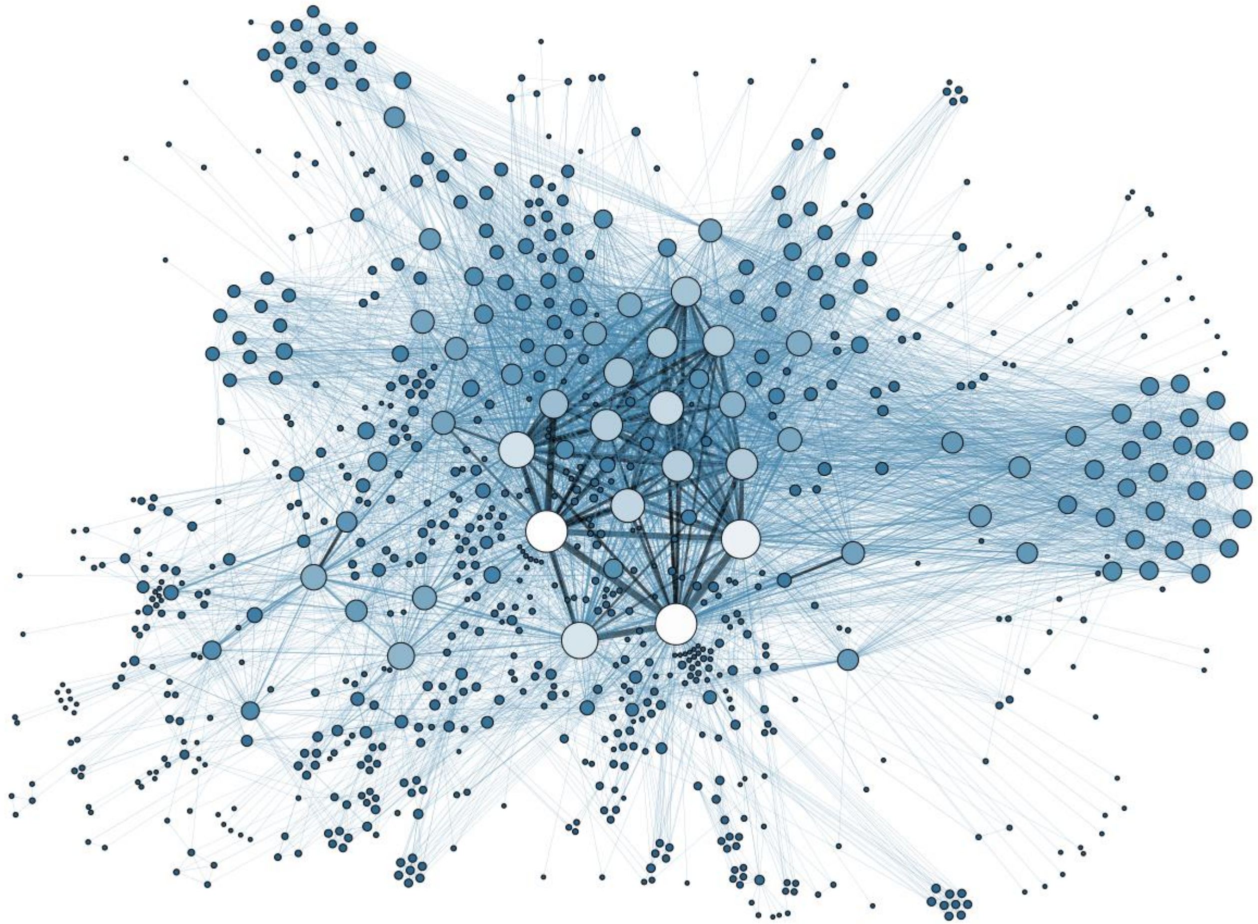
Deep Learning

- Apart from batch normalization, researchers have also used “gradient clipping”, where gradients are cut off before reaching a predetermined limit (e.g. cut off gradients to be between -1 and 1)

Deep Learning

- RNN for Time Series present their own gradient challenges, let’s explore special LSTM (Long Short Term Memory) neuron units that help fix these issues!

LTSM and GRU Units



Deep Learning

Deep Learning

- Many of the solutions previously presented for vanishing gradients can also apply to RNN: different activation functions, batch normalizations, etc…
- However because of the length of time series input, these could slow down training

Deep Learning

- A possible solution would be to just shorten the time steps used for prediction, but this makes the model worse at predicting longer trends.

Deep Learning

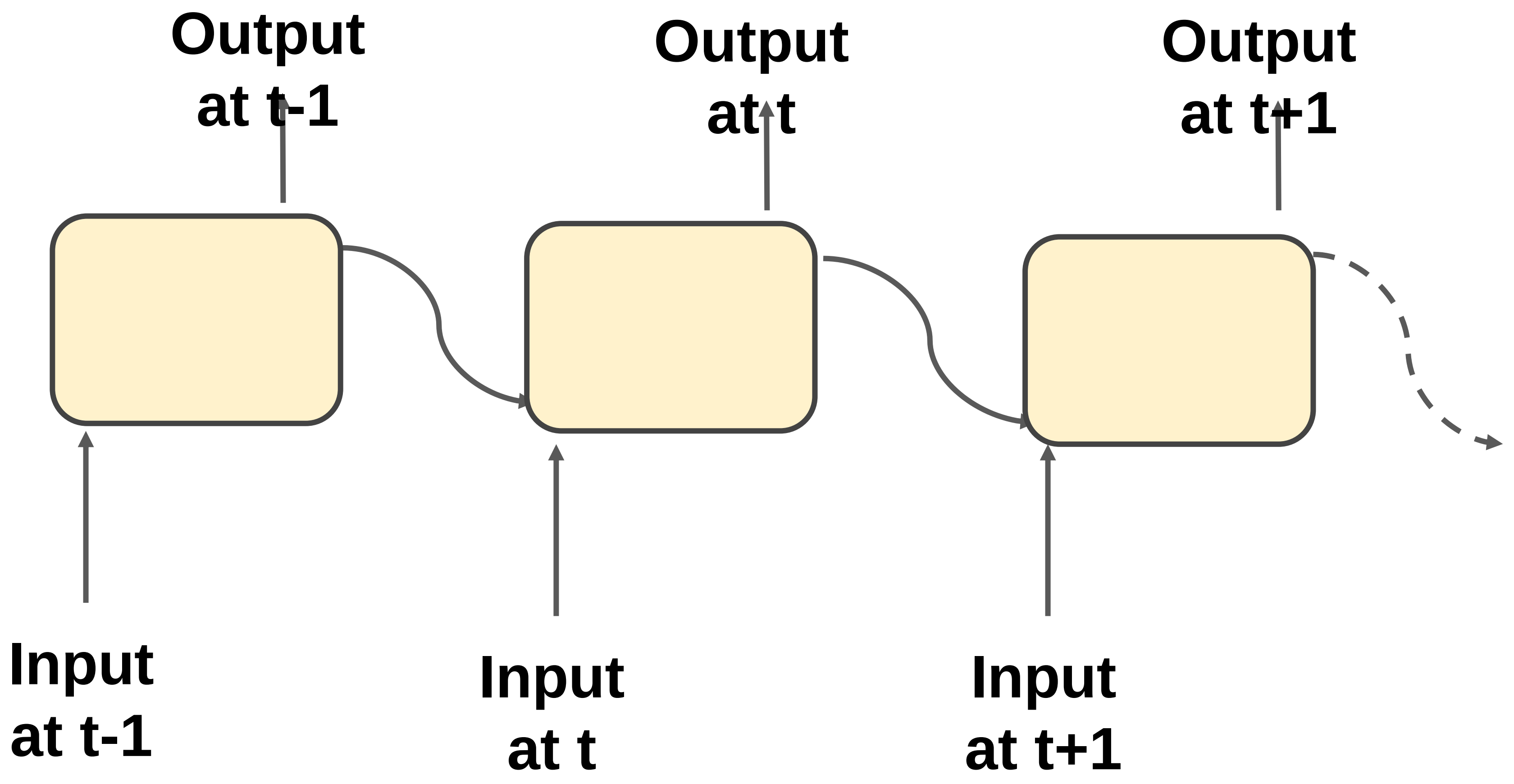
- Another issue RNN face is that after awhile the network will begin to “forget” the first inputs, as information is lost at each step going through the RNN.
- We need some sort of “long-term memory” for our networks.

Deep Learning

- The LSTM (Long Short-Term Memory) cell was created to help address these RNN issues.
- Let’s go through how an LSTM cell works!

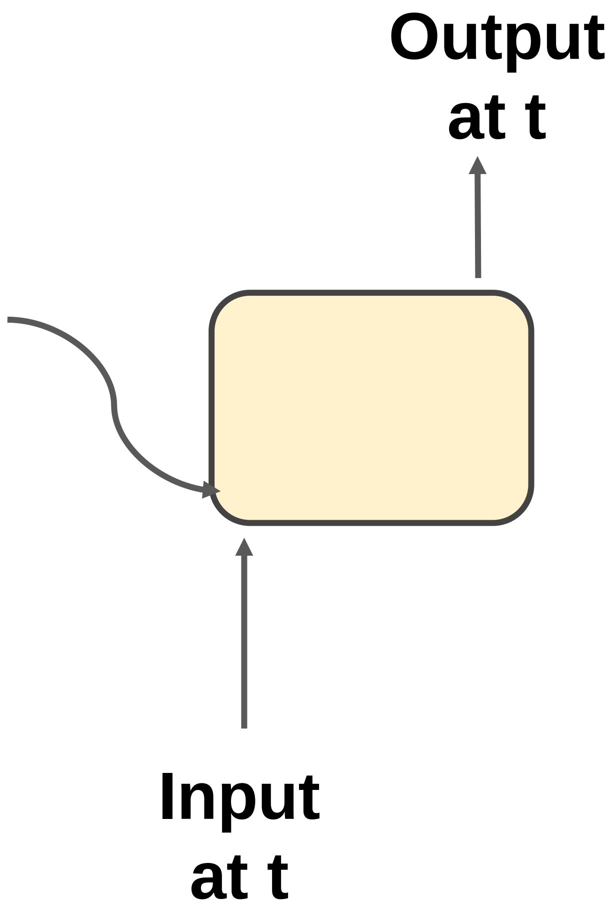
Deep Learning

A typical RNN



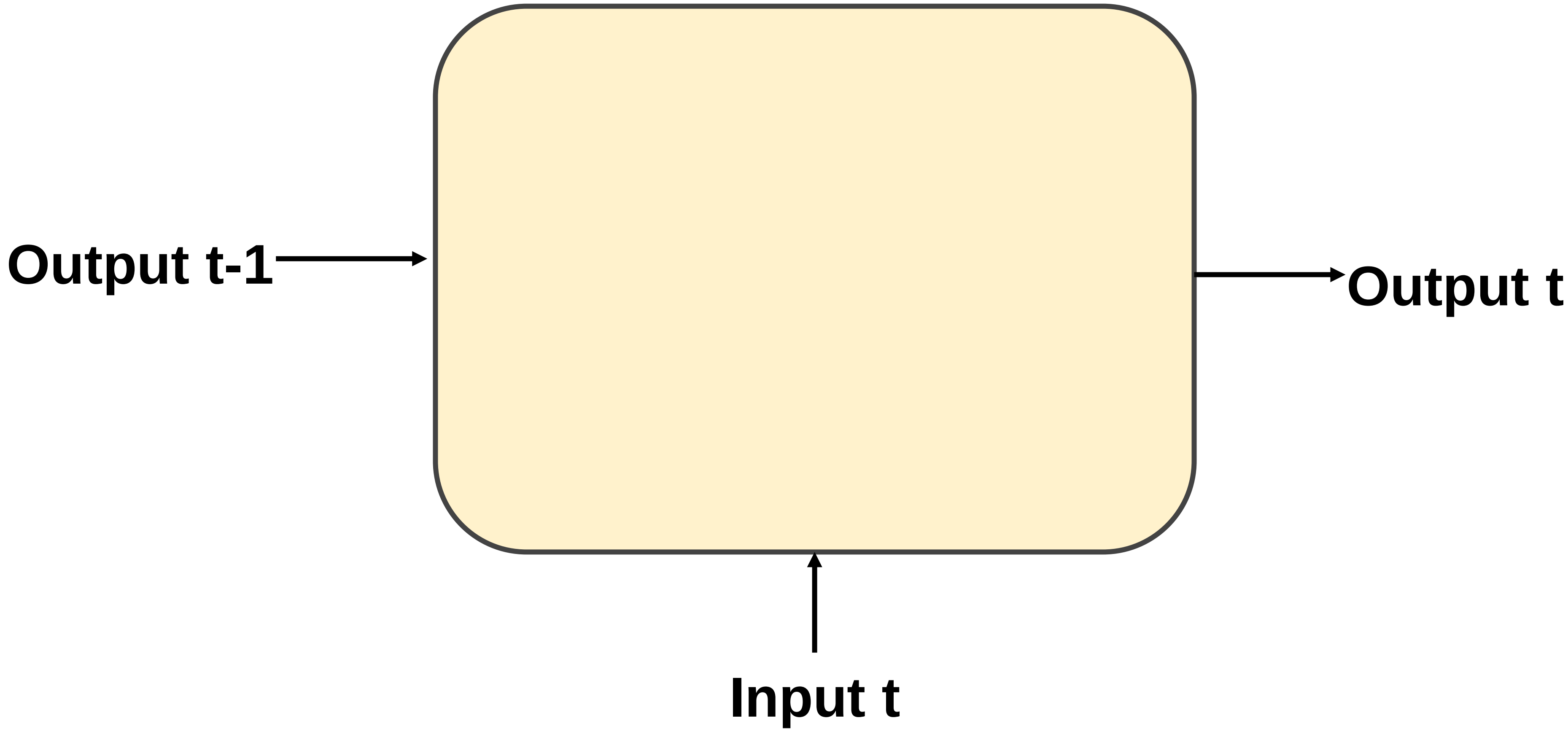
Deep Learning

RNN



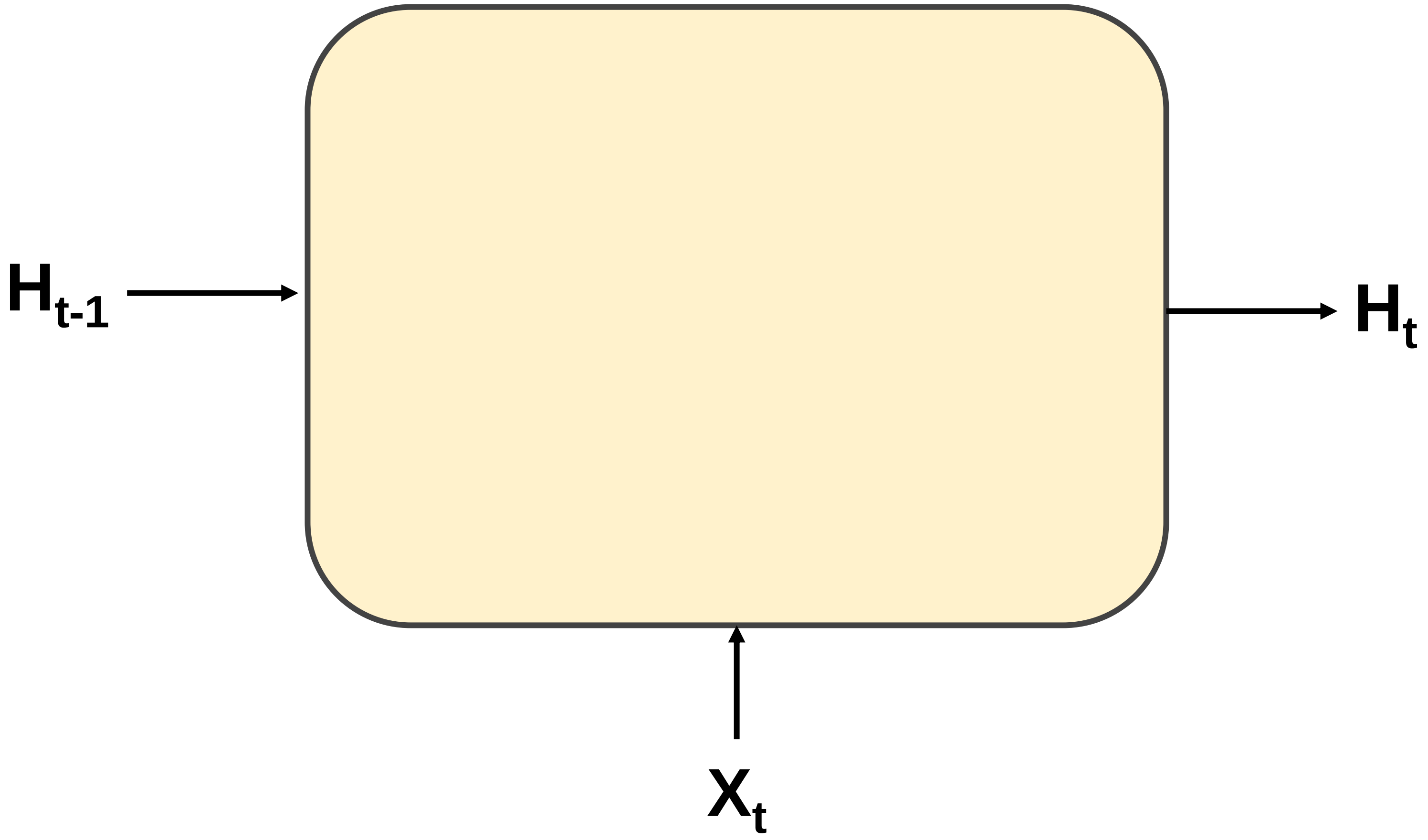
Deep Learning

RNN



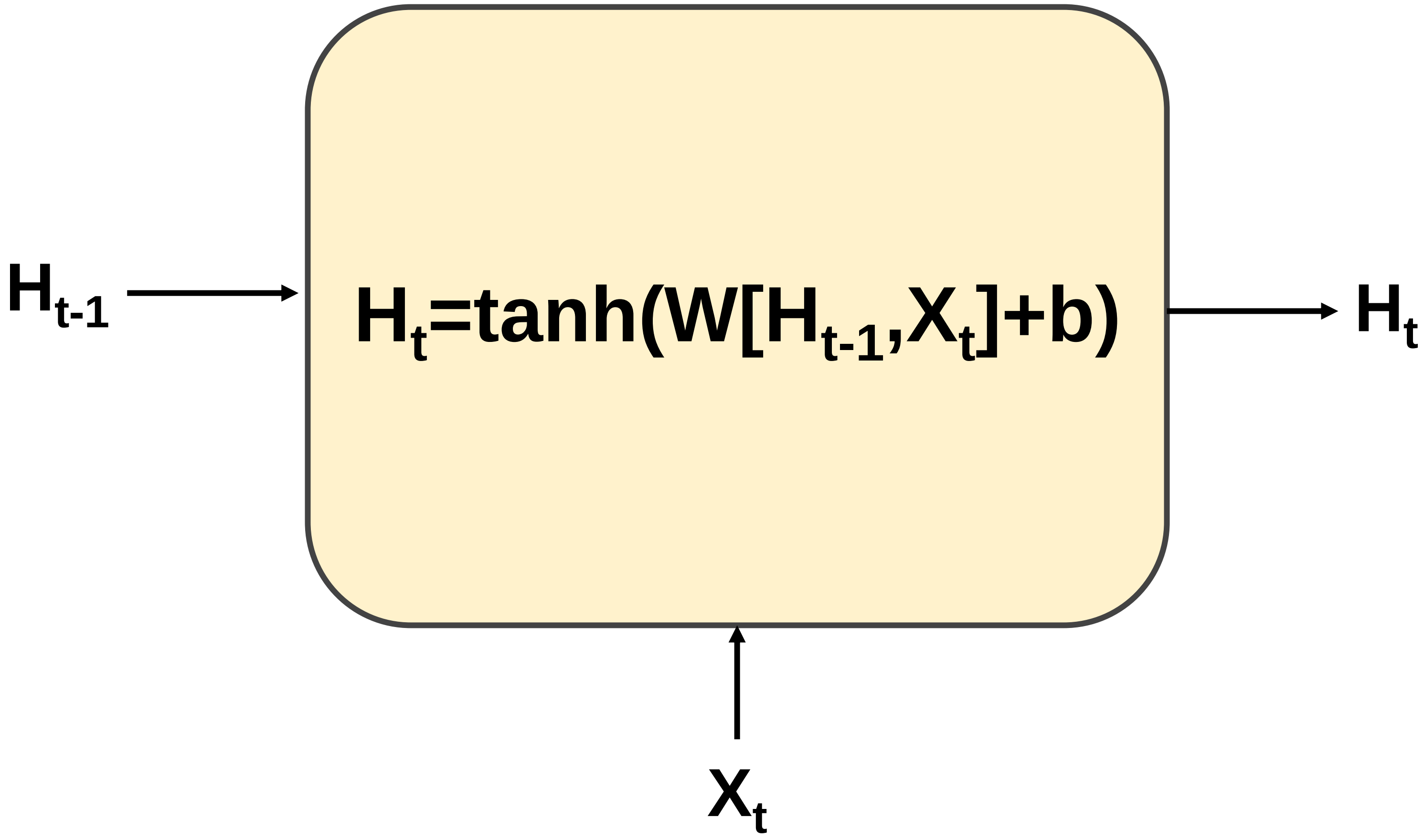
Deep Learning

RNN



Deep Learning

RNN

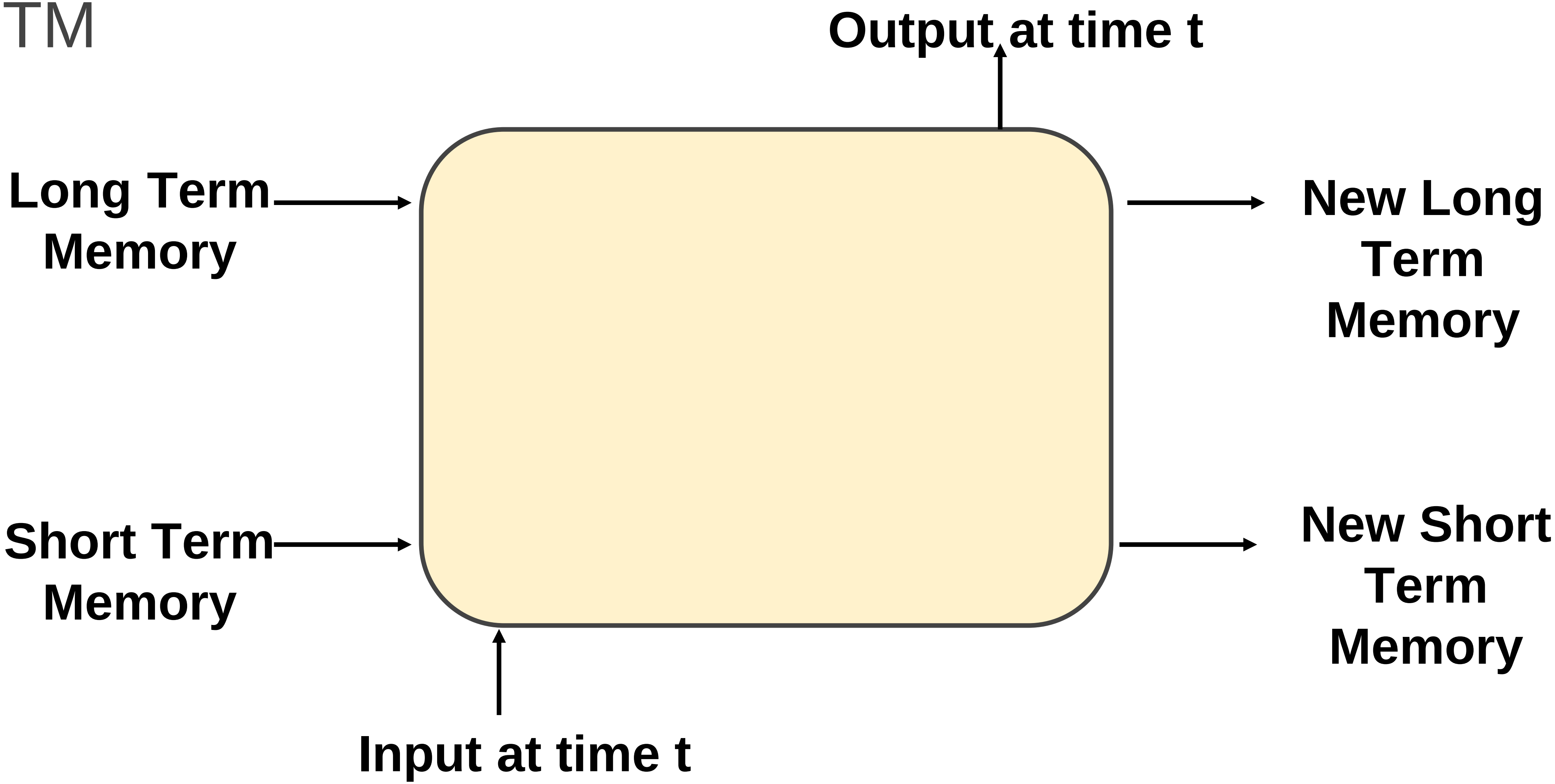


Deep Learning

LSTM

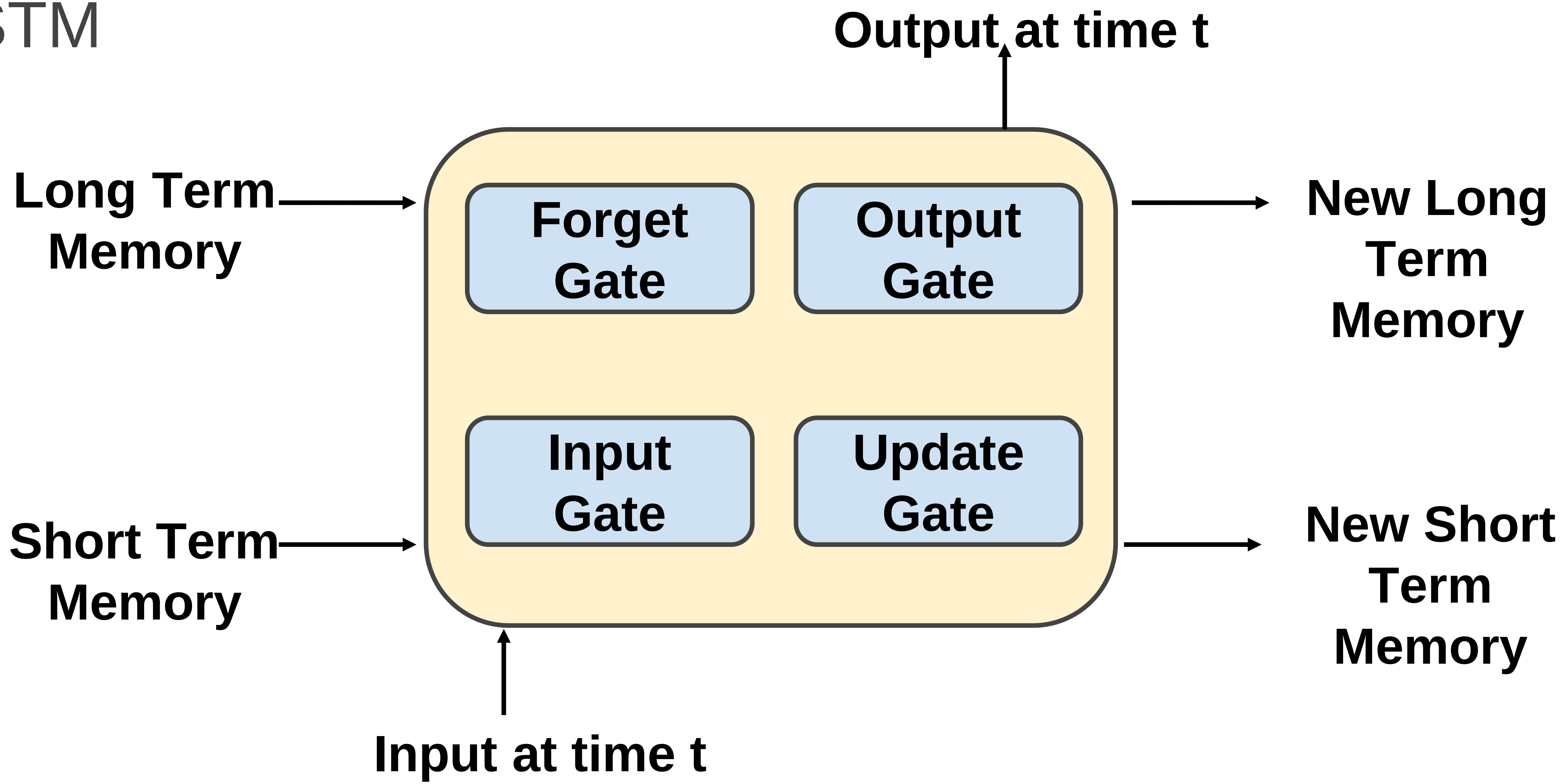
Deep Learning

LSTM



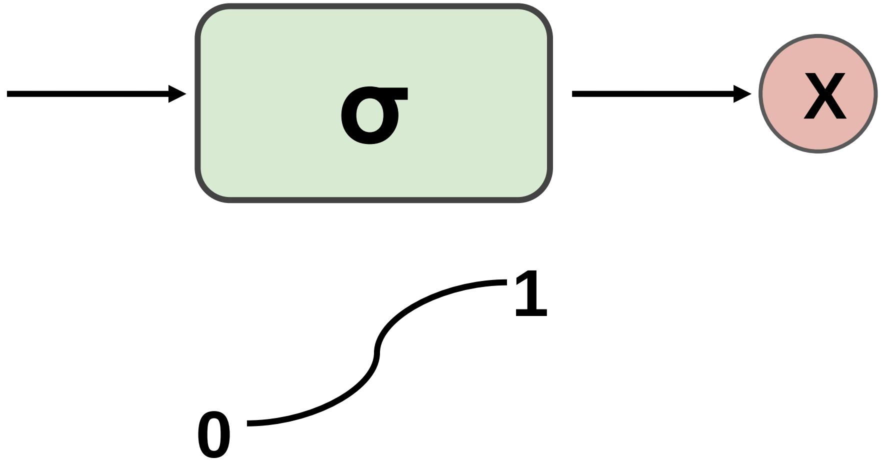
Deep Learning

LSTM



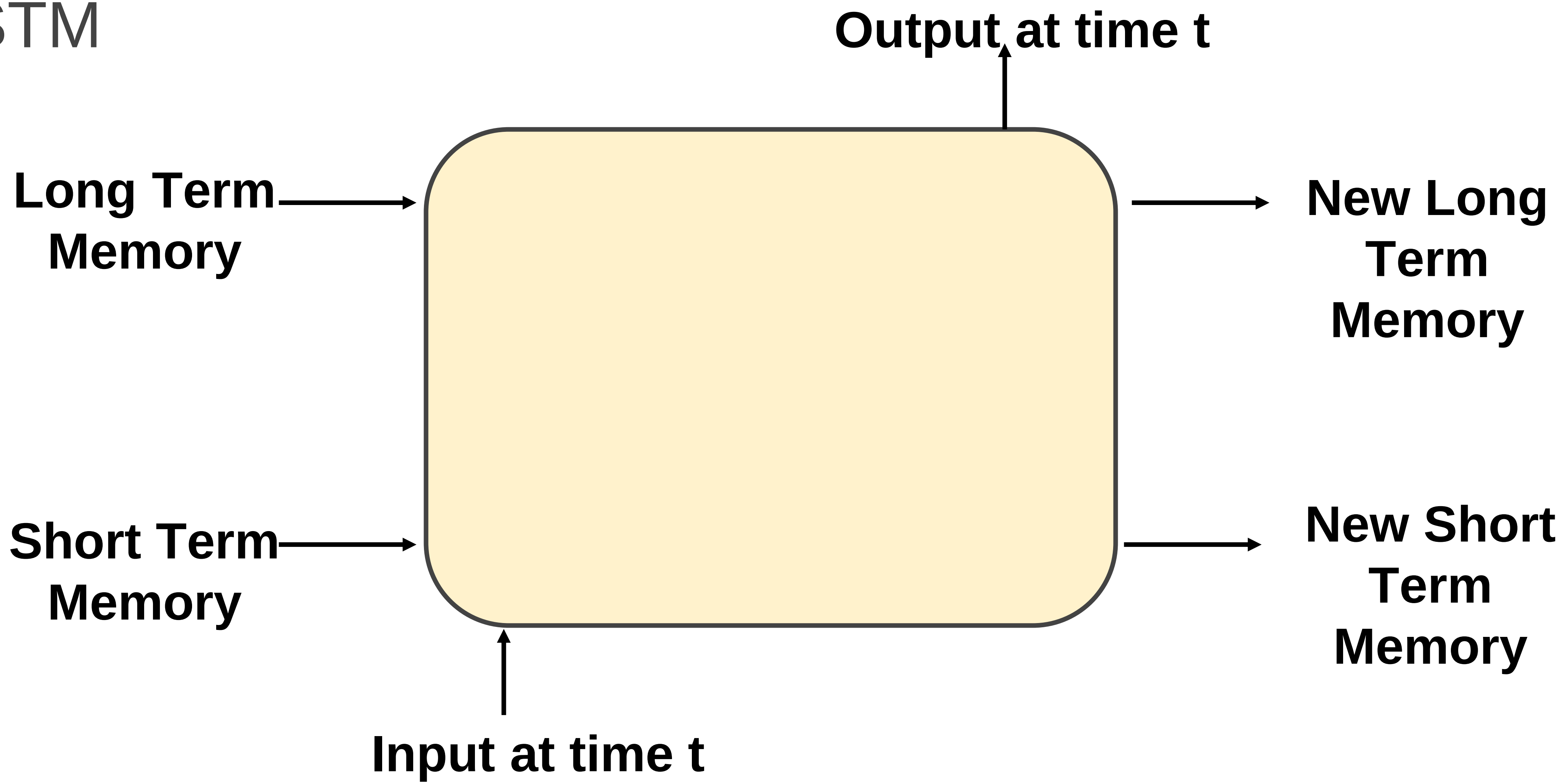
Deep Learning

Gates optionally let information through



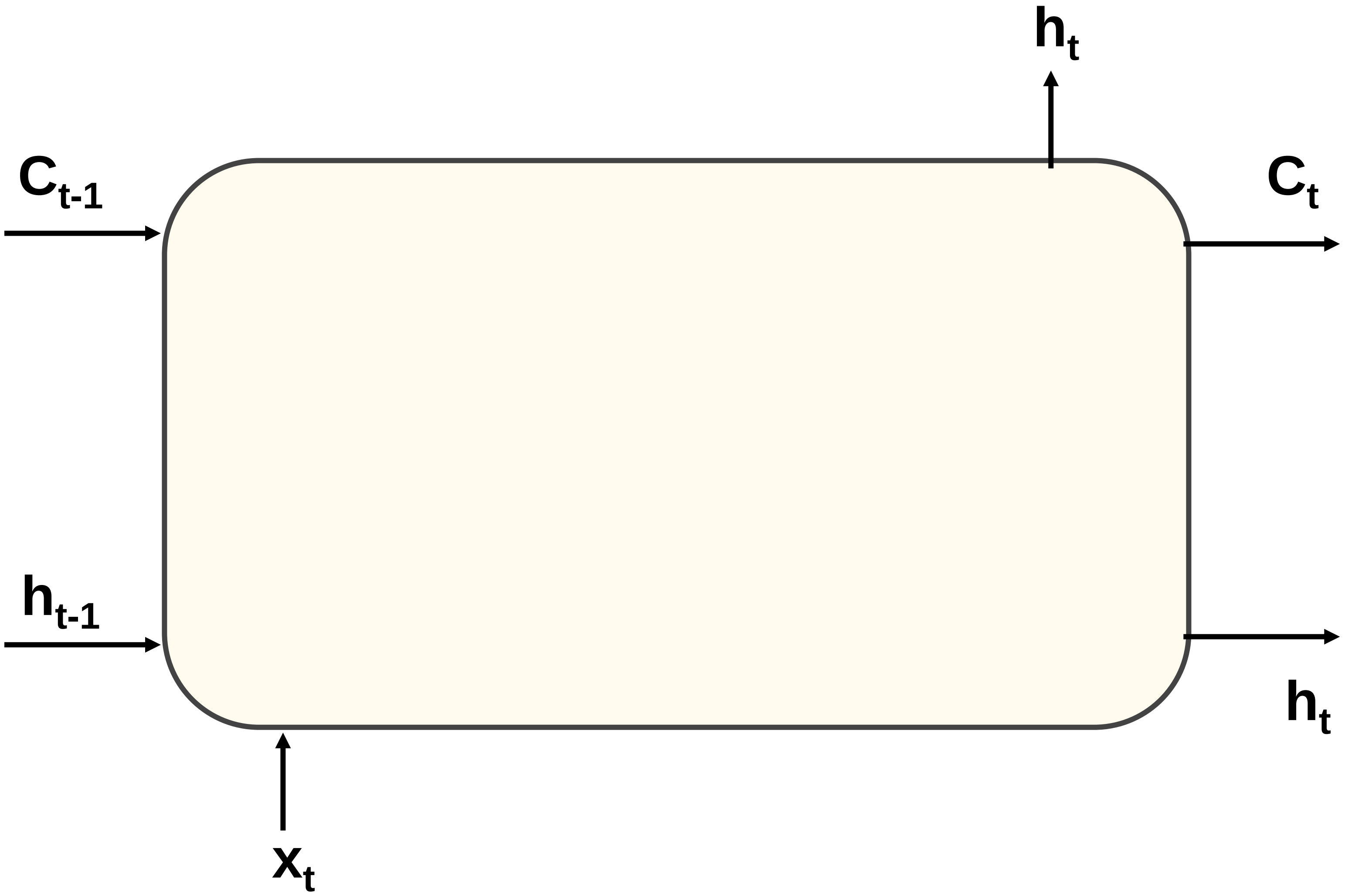
Deep Learning

LSTM

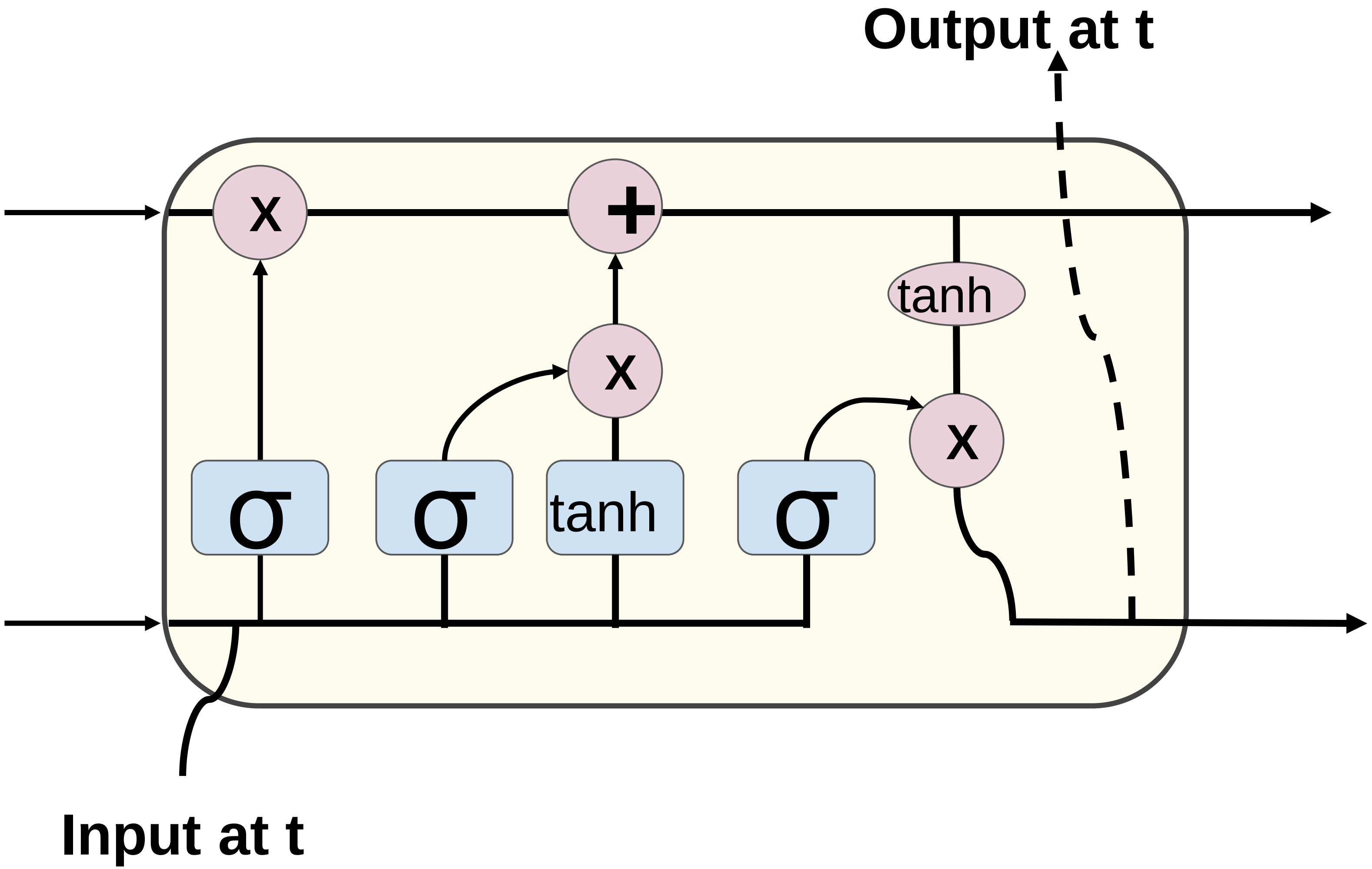


Deep Learning

LSTM



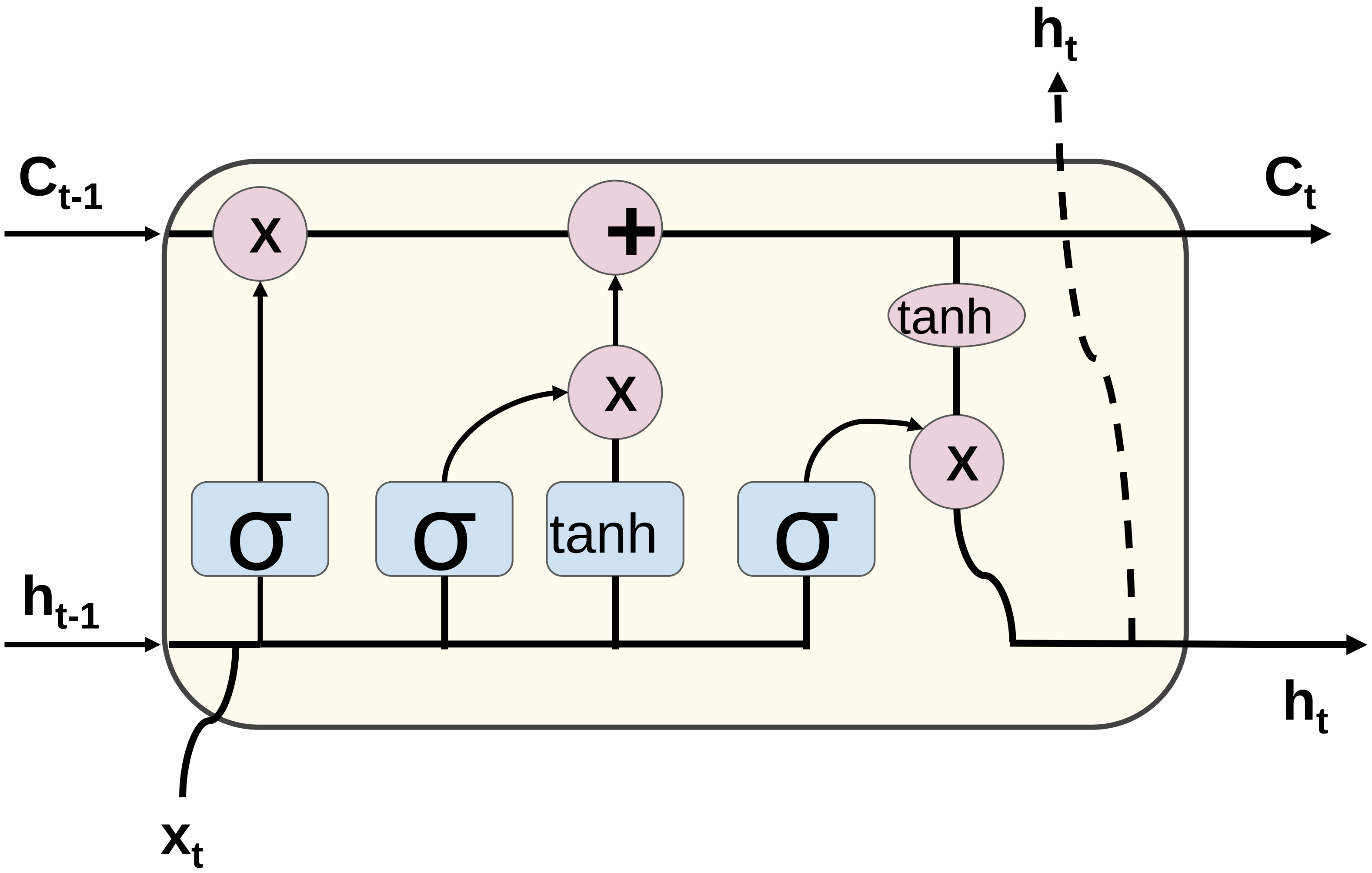
An LSTM Cell



Here we can see the entire LSTM cell.

Let’s go through the process!

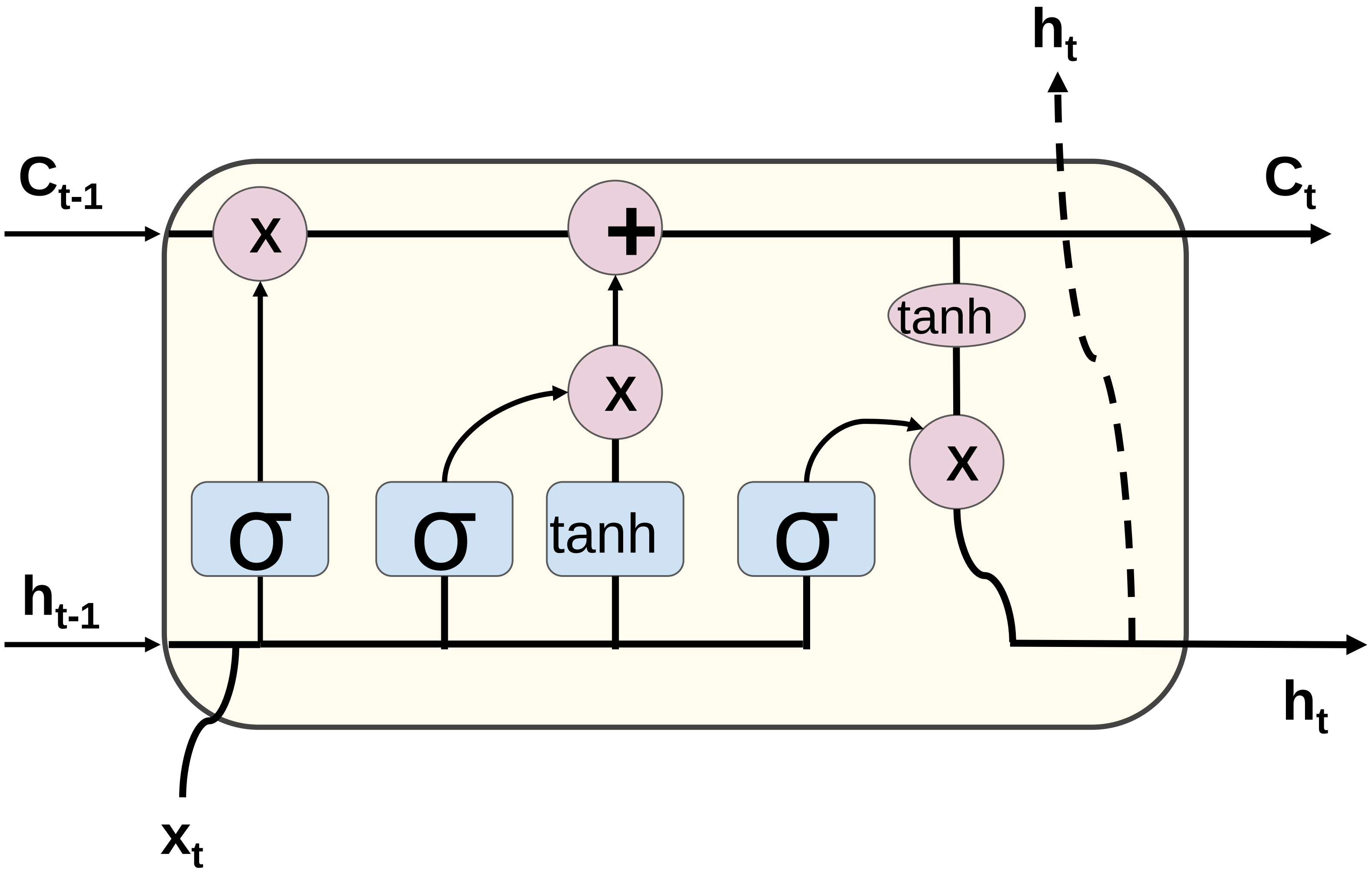
An LSTM Cell



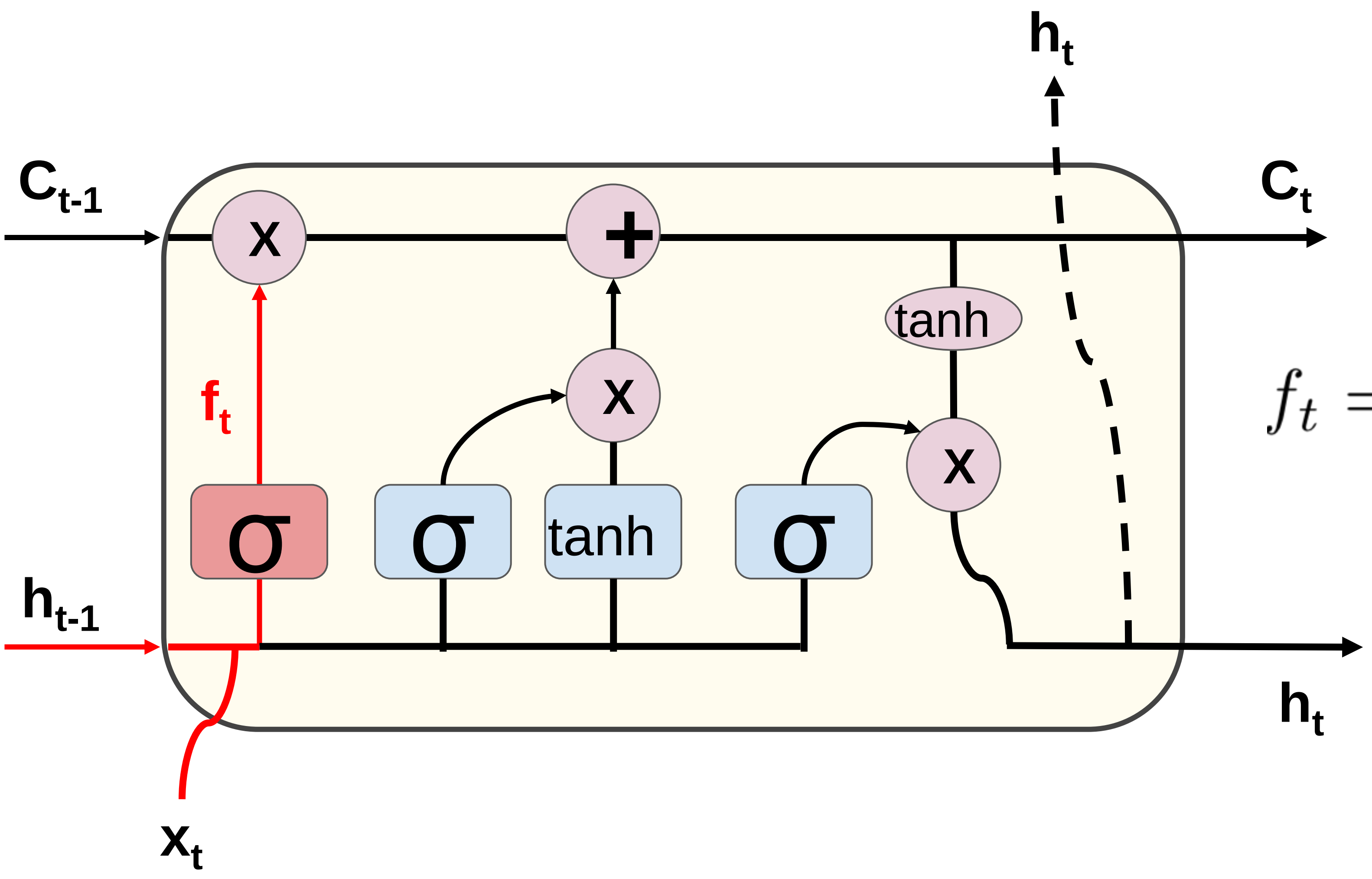
Here we can see the entire LSTM cell.

Let’s go through the process!

An LSTM Cell

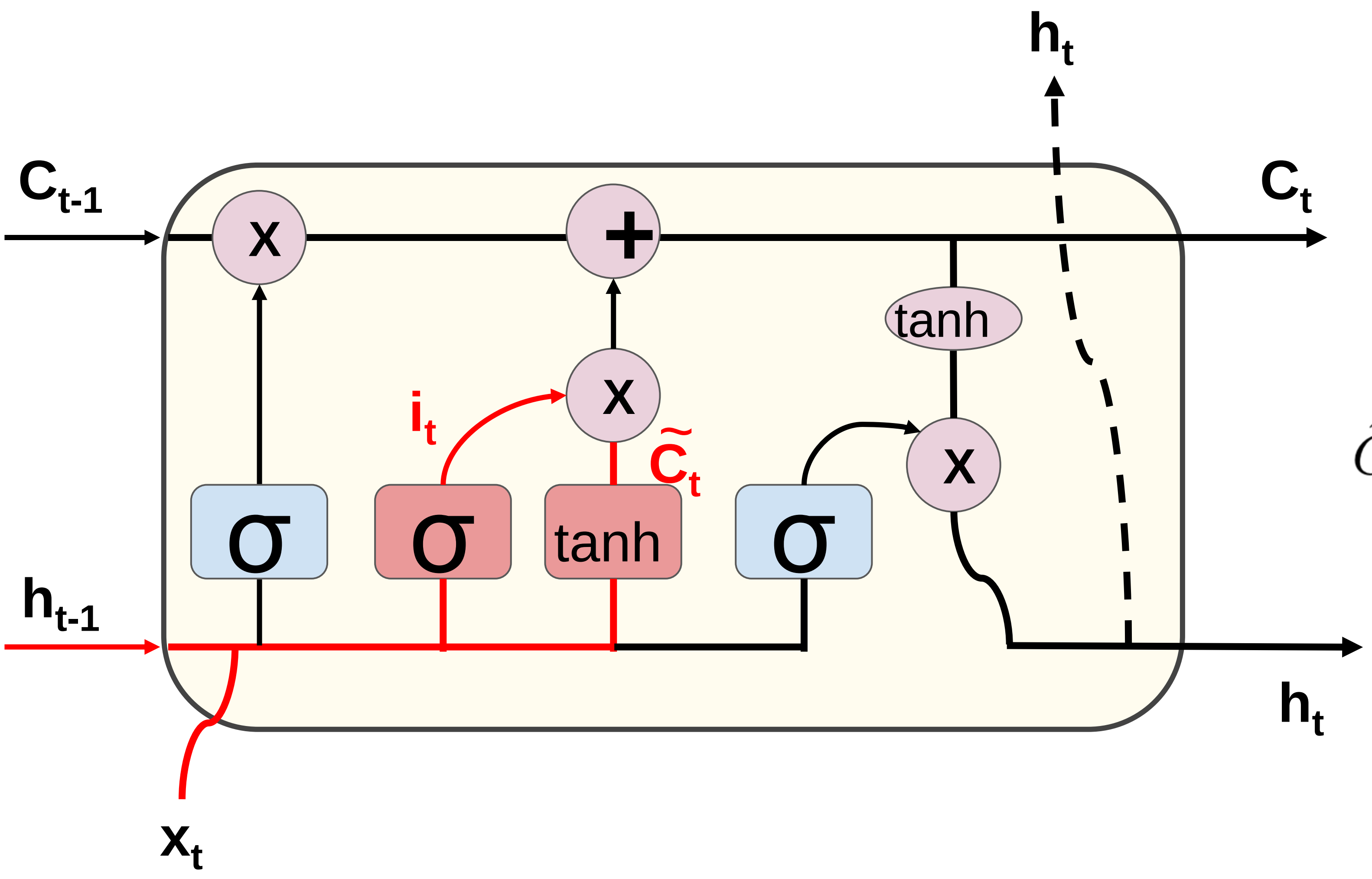


An LSTM Cell



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

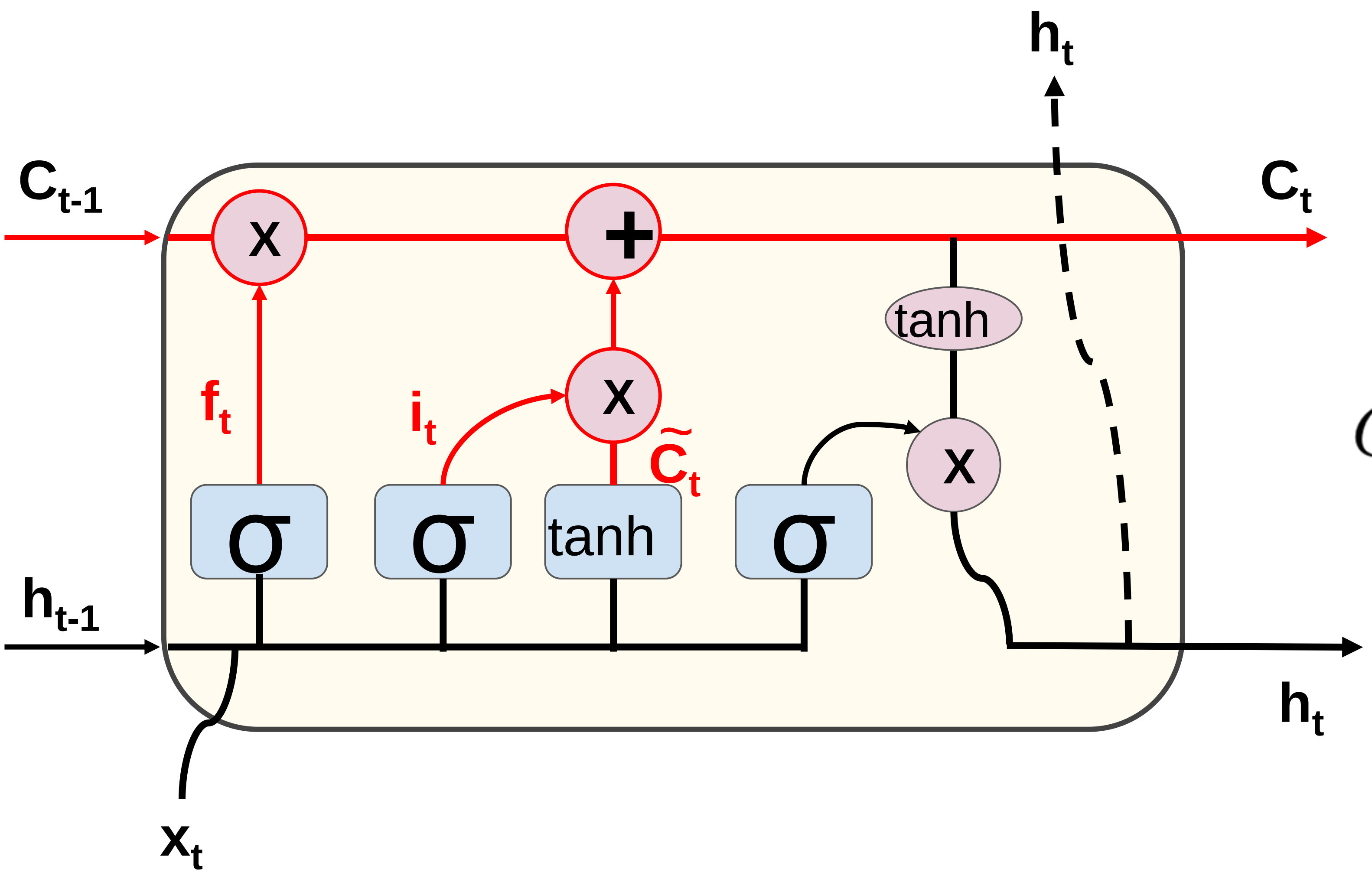
An LSTM Cell



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

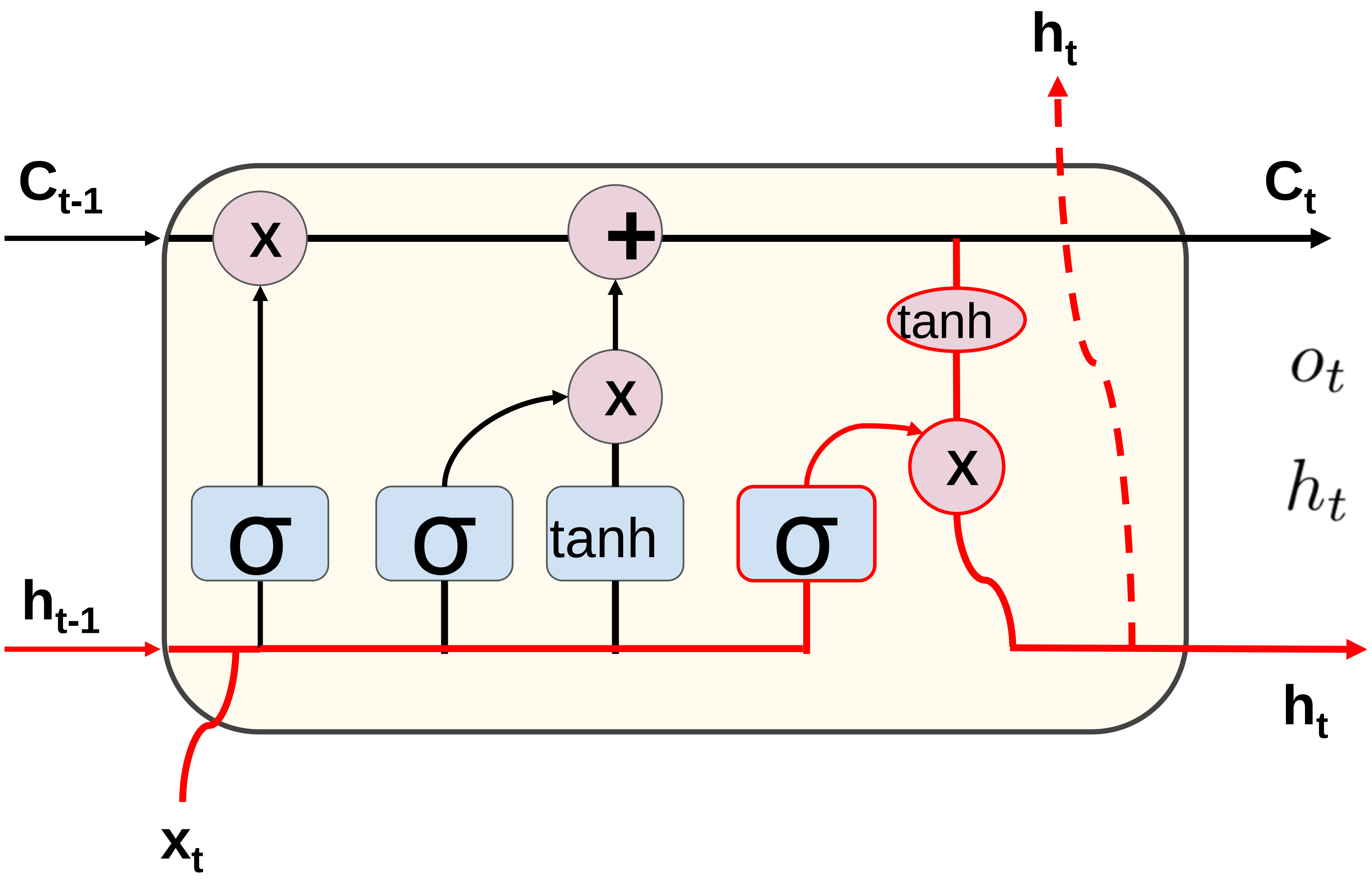
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

An LSTM Cell



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

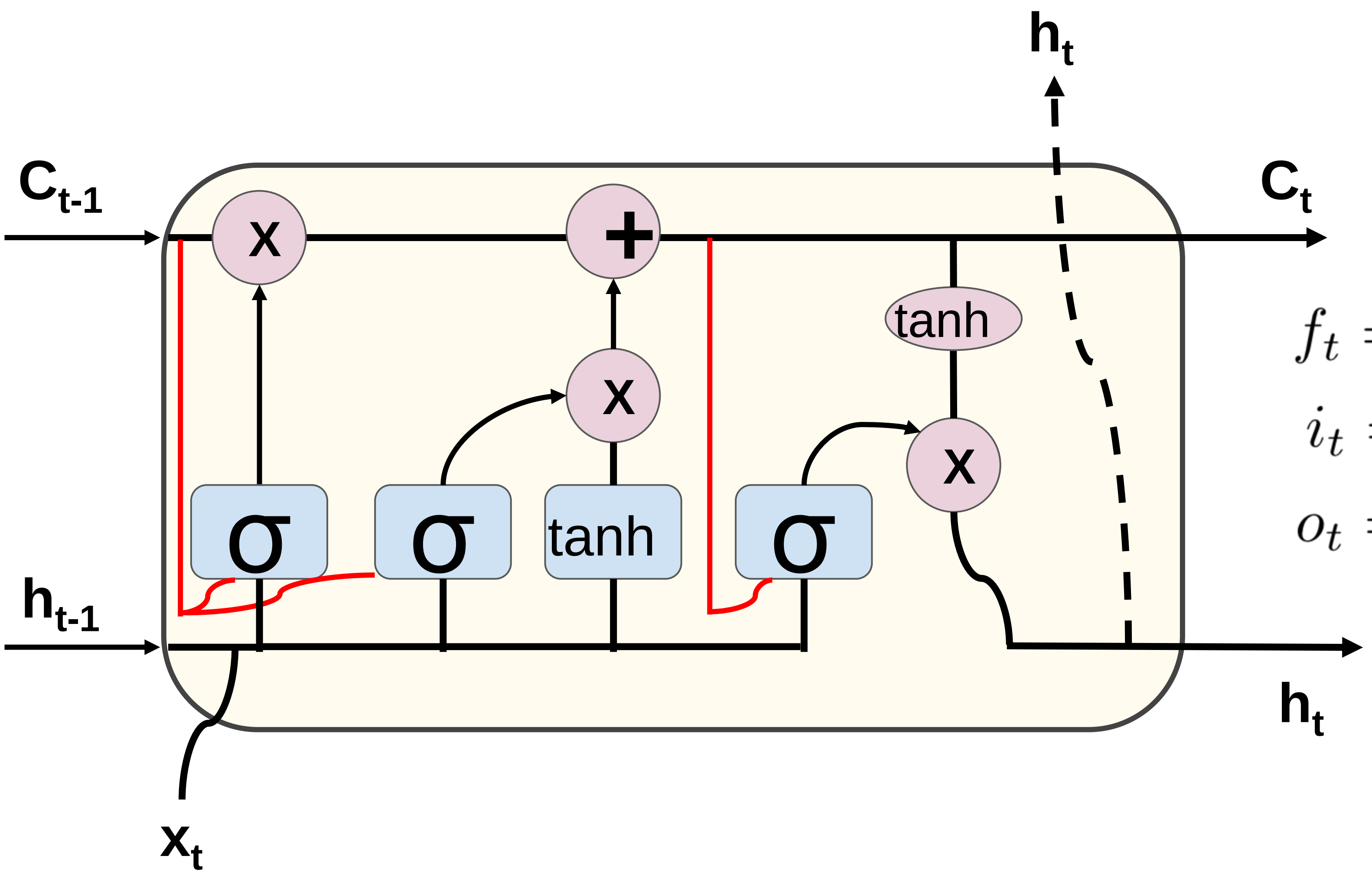
An LSTM Cell



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

An LSTM Cell with “peepholes”

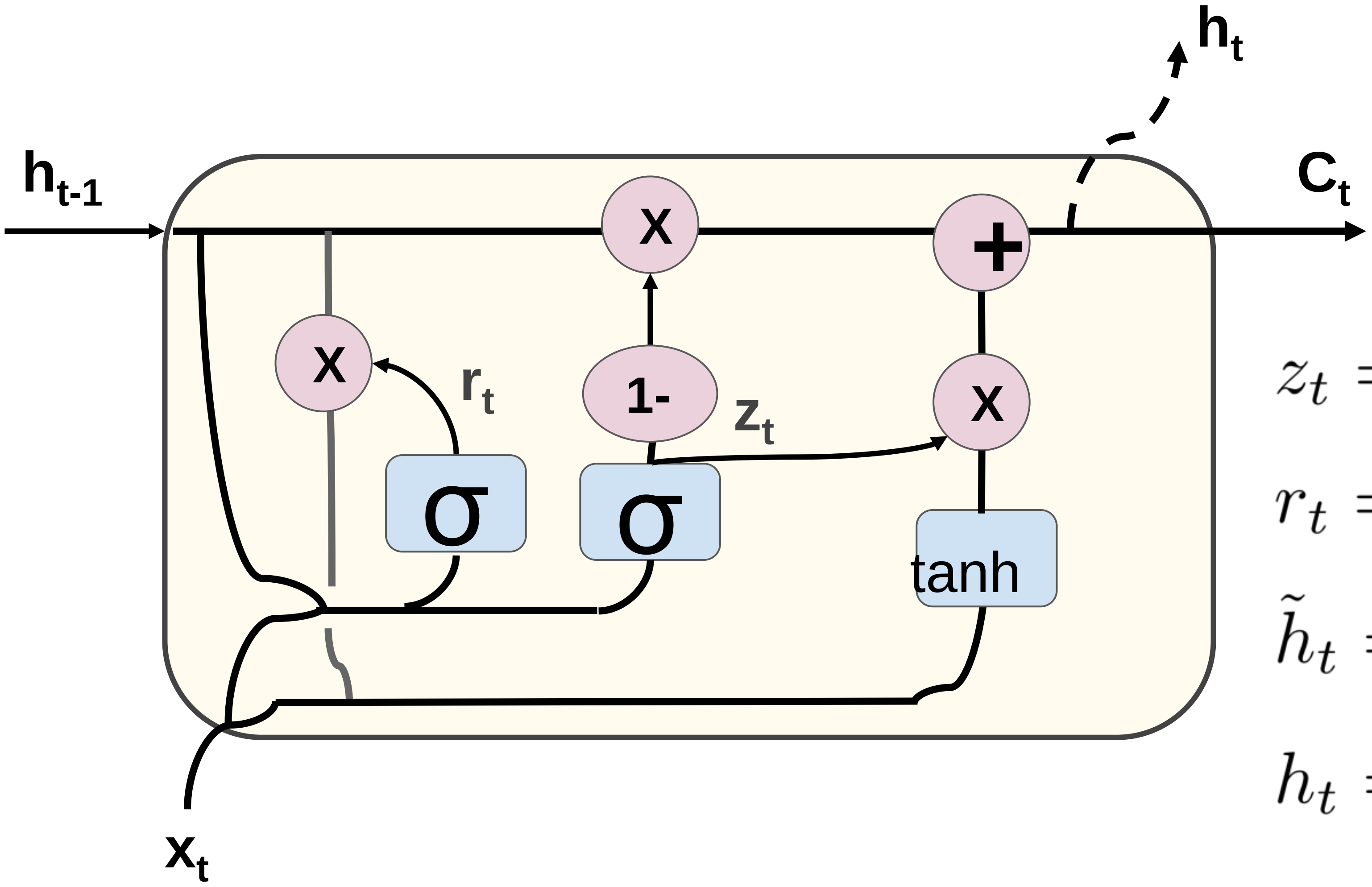


$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$

$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$

$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

Gated Recurrent Unit (GRU)



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

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

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

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

Deep Learning

- Fortunately with our deep learning python library , we simply need to call the import for RNN or LSTM instead of needing to code all of this ourselves!
- Let’s explore how to use LSTMs with Python code!

ICTC International Council for Technology Certifications

Basic RNN

Deep Learning

Deep Learning

- Let’s now explore how to use RNN on a basic time series, such as a sine wave.
- Before we jump to the notebook, let’s quickly discuss what RNN sequence batches look like.

Deep Learning

- Let’s imagine a simple time series:
    - **[0,1,2,3,4,5,6,7,8,9]**
  - We separate this into 2 parts:
    - **[0,1,2,3,4,5,6,7,8]** **[9]**
  - Given **training sequence**, predict the **next sequence value**.

Deep Learning

- Keep in mind we can usually decide how long the training sequence and predicted label should be:
    - **[0,1,2,3,4]** ⇨ **[5,6,7,8,9]**

Deep Learning

- We can also edit the size of the training point, as well as how many sequences to feed per batch:
    - [0,1,2,3] ⇨ [4]
    - [1,2,3,4] ⇨ [5]
    - [2,3,4,5] ⇨ [6]

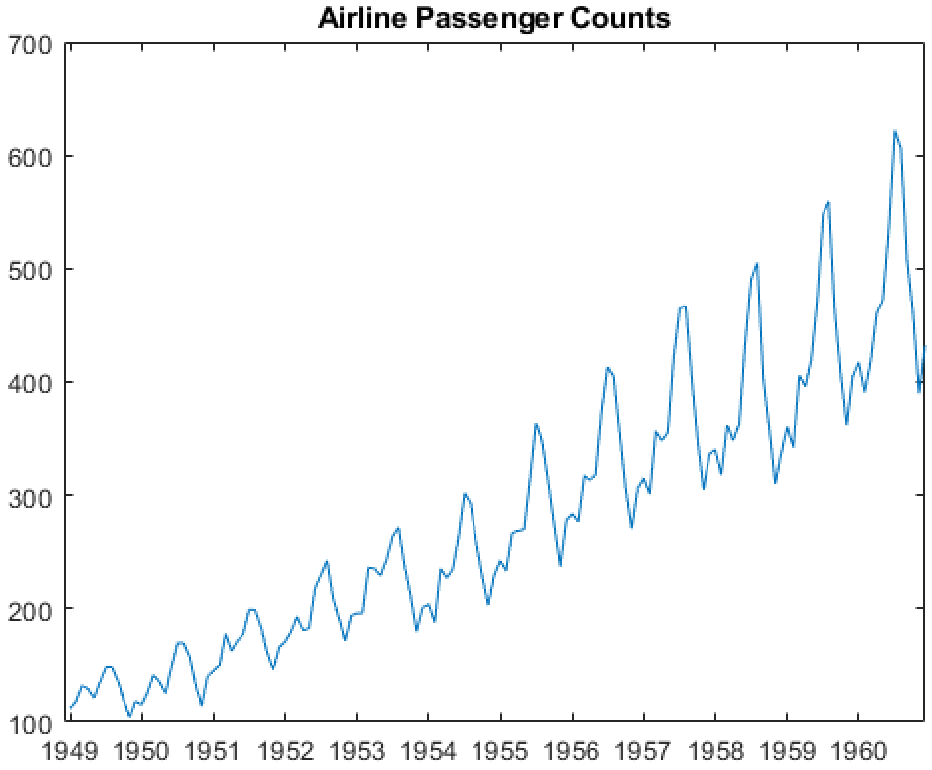
Deep Learning

- So how do we decide how long the training sequence should be?

- There is no definitive answer, but it should be at least long enough to capture any useful trend information.

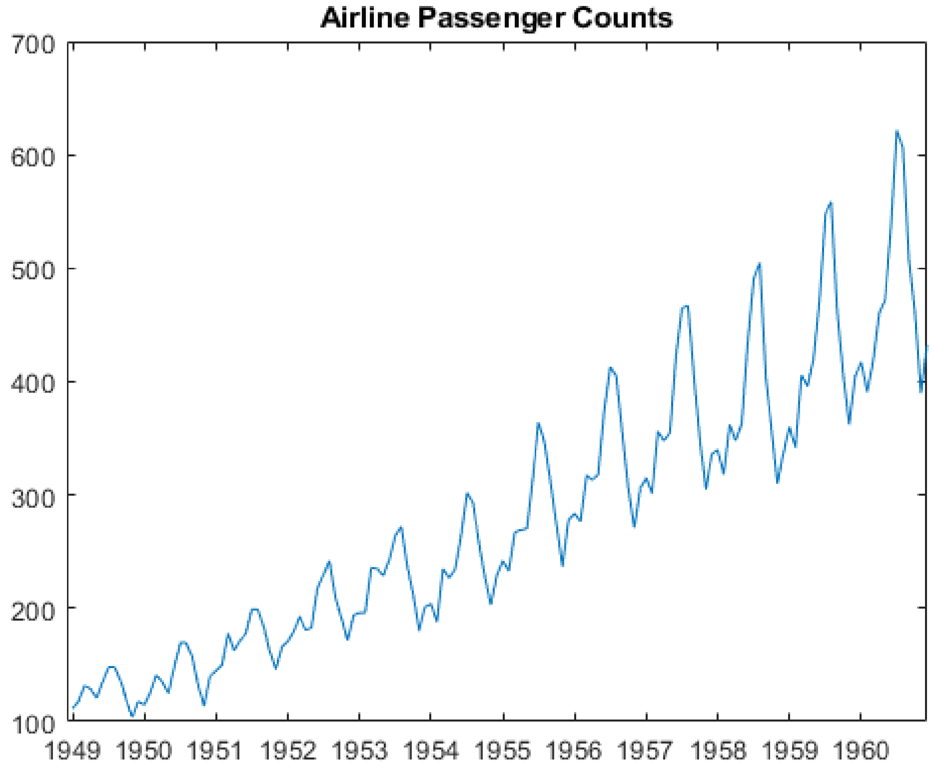
Deep Learning

- For example, if dealing with seasonal data:



Deep Learning

- If this is monthly, we should include at least 12 months in the training sequence



Deep Learning

- This often takes domain knowledge and experience, as well as simply experimenting and using RMSE to measure error of forecasted predictions.
- Typically a good starting choice for the label is just one data point into the future.

Deep Learning

- How do we forecast with RNNs?
- Let's imagine all our data is:
    - **[0,1,2,3,4,5,6,7,8,9]**
    - And we trained on sequences such as:
        - [0,1,2,3] [4]
        - [1,2,3,4] [5]
        - [2,3,4,5] [6]

Deep Learning

- Then our forecasting technique is to predict a time step ahead, and then incorporate our prediction into the next sequence we predict off of.
- Let’s walk through a quick example!

Deep Learning

- How do we forecast with RNNs?
- Let’s imagine all our data is:
    - **[0,1,2,3,4,5,6,7,8,9]**
  - And we trained on sequences such as:
    - [0,1,2,3] ⇨ [4]
    - [1,2,3,4] ⇨ [5]
    - [2,3,4,5] ⇨ [6]

Deep Learning

- **[6,7,8,9]** ⇨ **[10]**   Forecast prediction!

Deep Learning

- [6,7,8,9] ⇨ [10] Forecast prediction!
- Then to keep predicting further:

Deep Learning

- **[6,7,8,9]** ⇨ **[10]** Forecast prediction!

Then to keep predicting further:

  - **[7,8,9,10]**⇨**[11.2]**

Deep Learning

- [6,7,8,9] ⇨ [10]   Forecast prediction!

Then to keep predicting further:

- [7,8,9,10] ⇨ [11.2]
- [8,9,10,11.2] ⇨ [12.4]

Deep Learning

- [6,7,8,9] ⇨ [10] Forecast prediction!

Then to keep predicting further:

  - [7,8,9,10]⇨[11.2]
  - [8,9,10,11.2]⇨[12.4]
  - [9,10,11.2,12.4]⇨[14]

Deep Learning

- [6,7,8,9] ⇨ [10] Forecast prediction!

Then to keep predicting further:

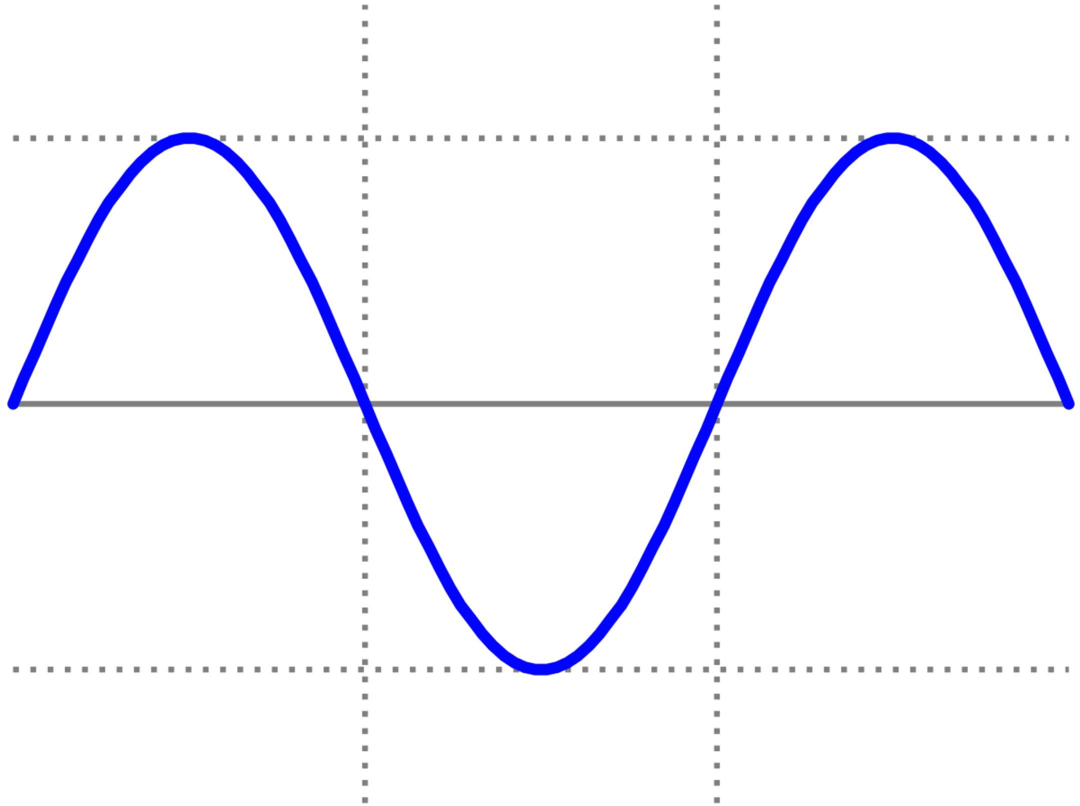
  - [7,8,9,10] ⇨ [11.2]
  - [8,9,10,11.2] ⇨ [12.4]
  - [9,10,11.2,12.4] ⇨ [14]
  - [10,11.2,12.4,14] ⇨ Completed Forecast

Deep Learning

- Let’s explore this further with Python!

ICTC International Council for Technology Certifications

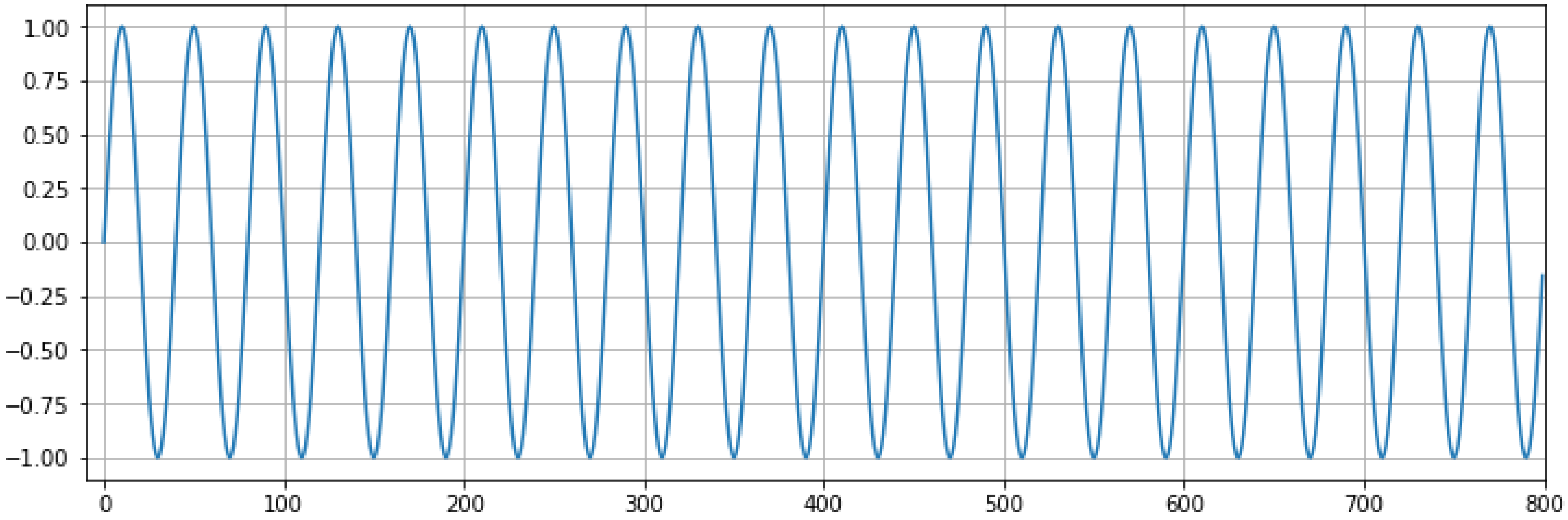
Basic RNN on a Sine Wave



Deep Learning

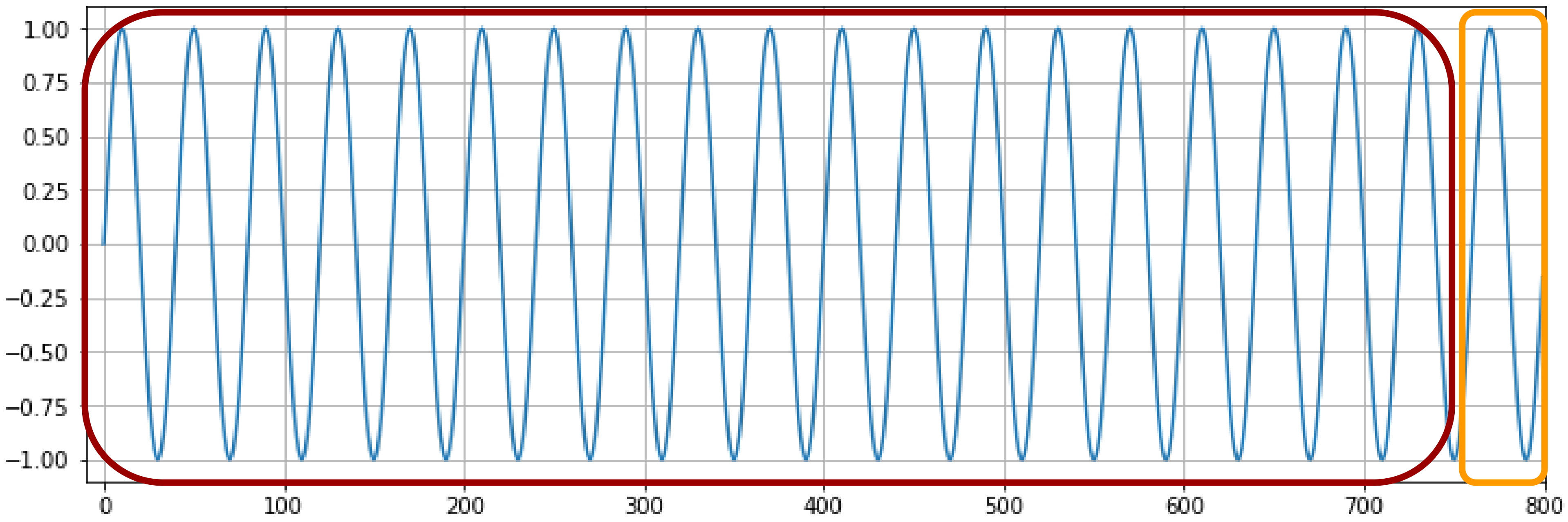
Deep Learning

- Let's now train and evaluate our RNN
- Recall our original data:



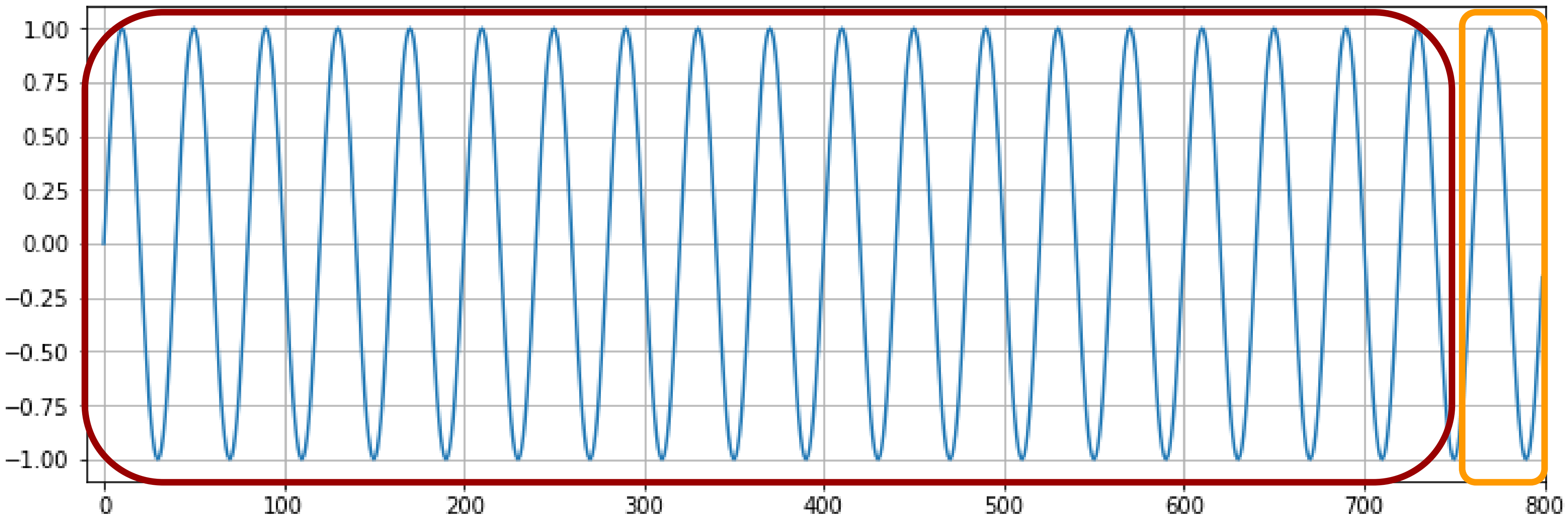
Deep Learning

- We split this into **train_set** and **test_set**:



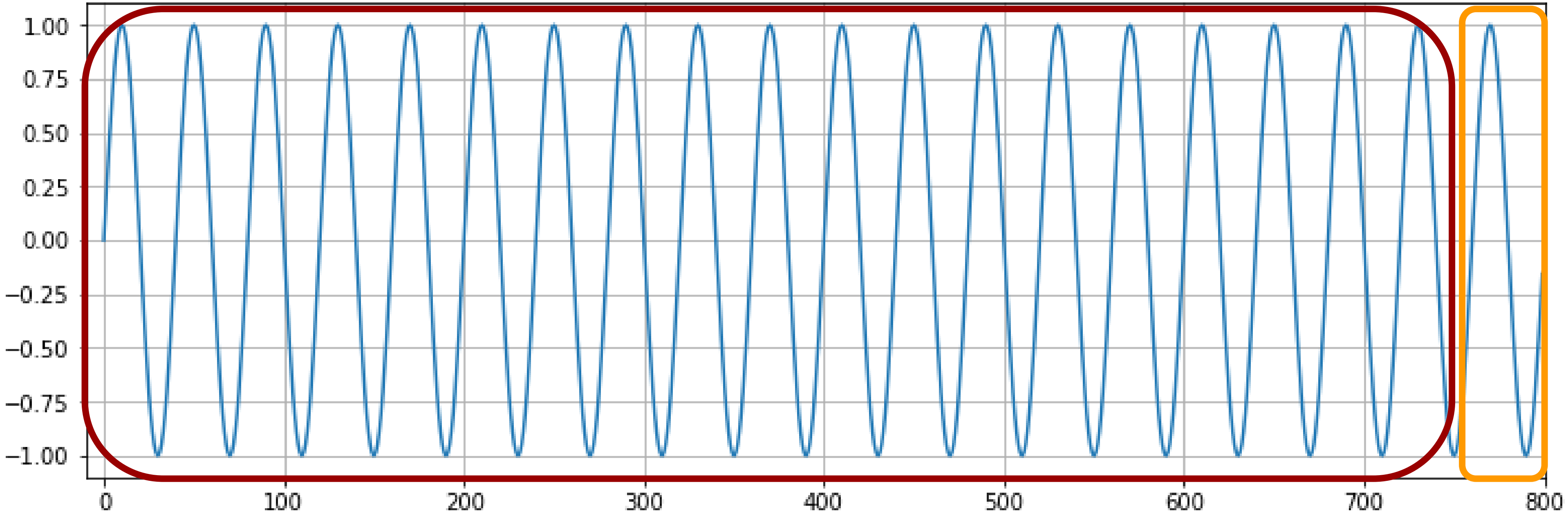
Deep Learning

- During training, we could evaluate performance on the test data.



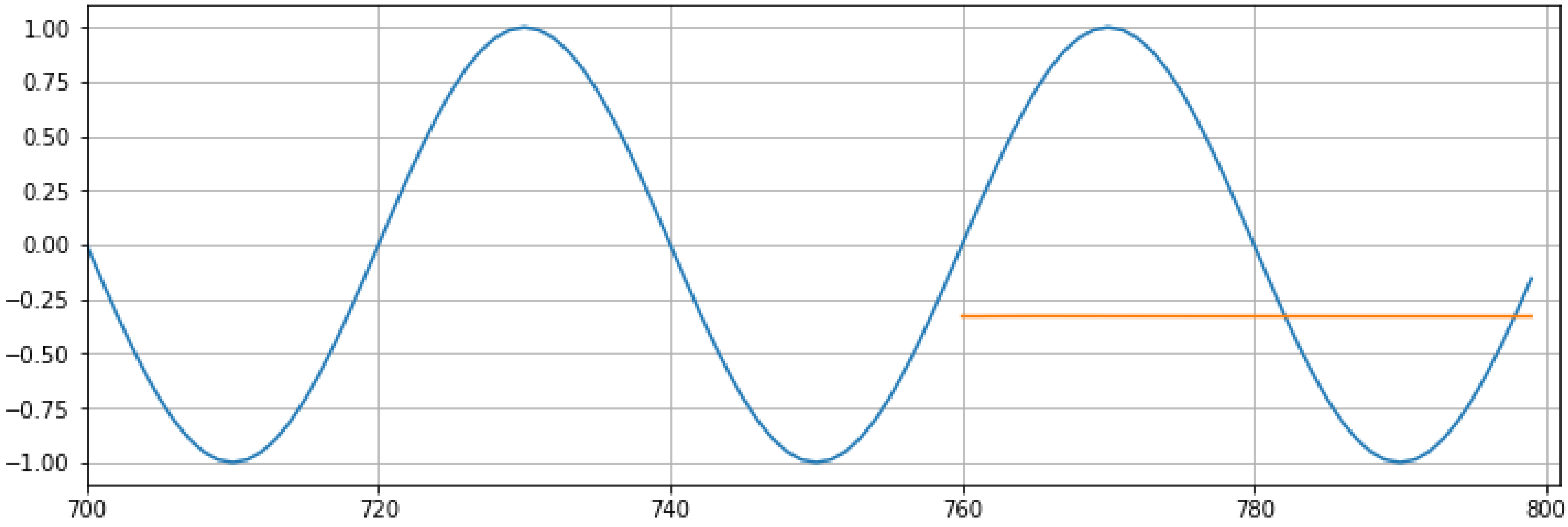
Deep Learning

- Let’s zoom in on that range!



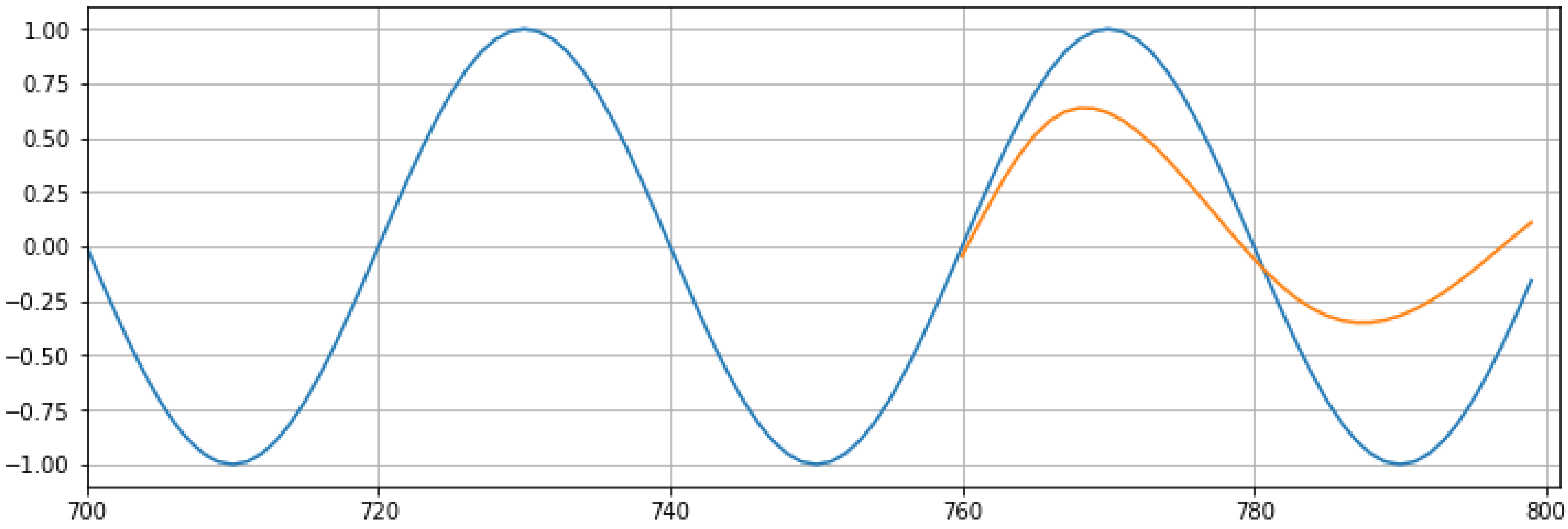
Deep Learning

- As we train, we will forecast on this range to visually see the training.



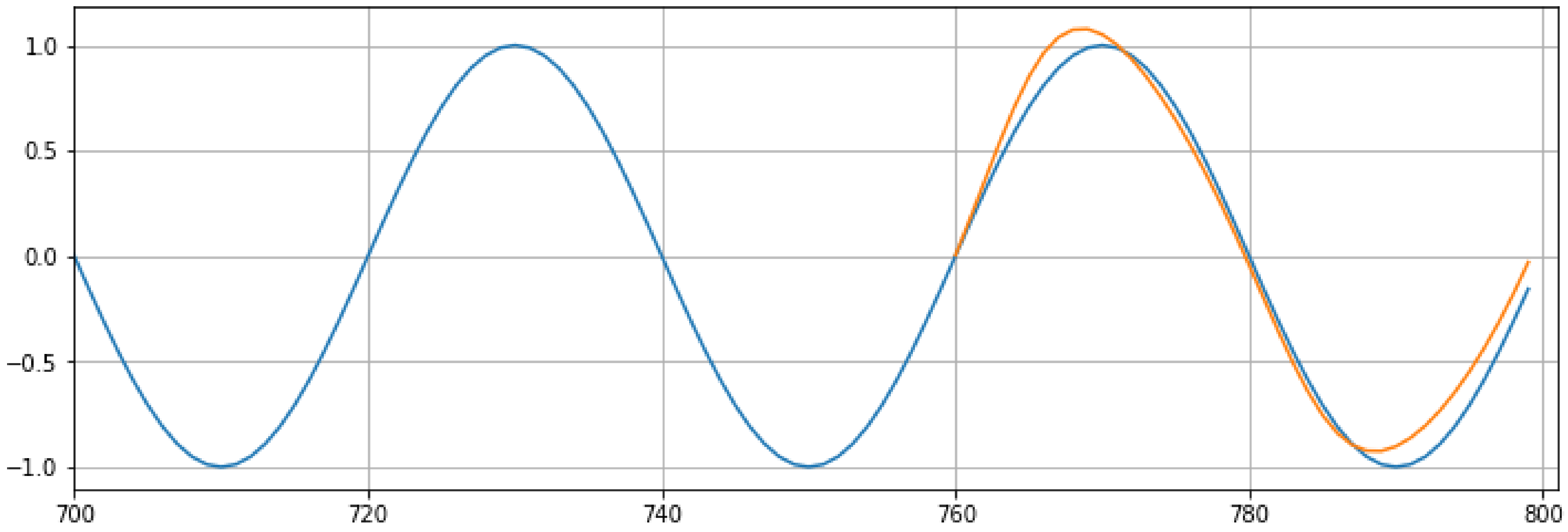
Deep Learning

- As we train, we will forecast on this range to visually see the training.



Deep Learning

- As we train, we will forecast on this range to visually see the training.

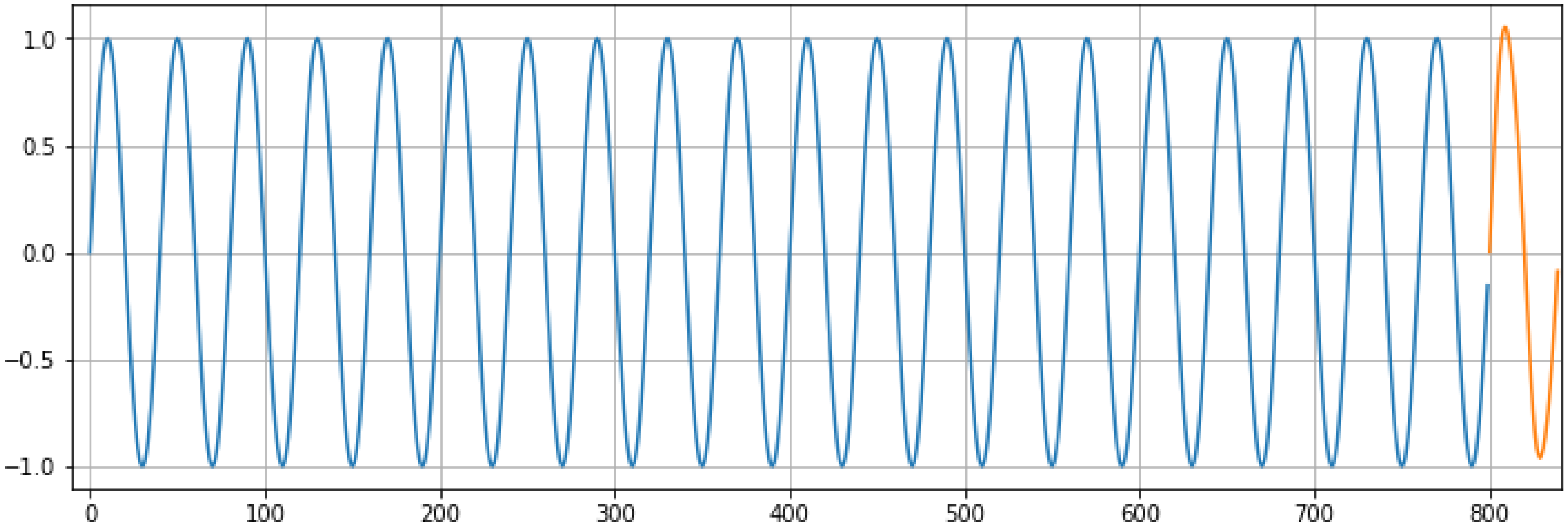


Deep Learning

- Eventually once we are satisfied with the results on the test portion, it will be time to forecast into the unknown future.
- This means **retraining** on **all** available data (both train and test) and forecasting beyond the scope of the original data.

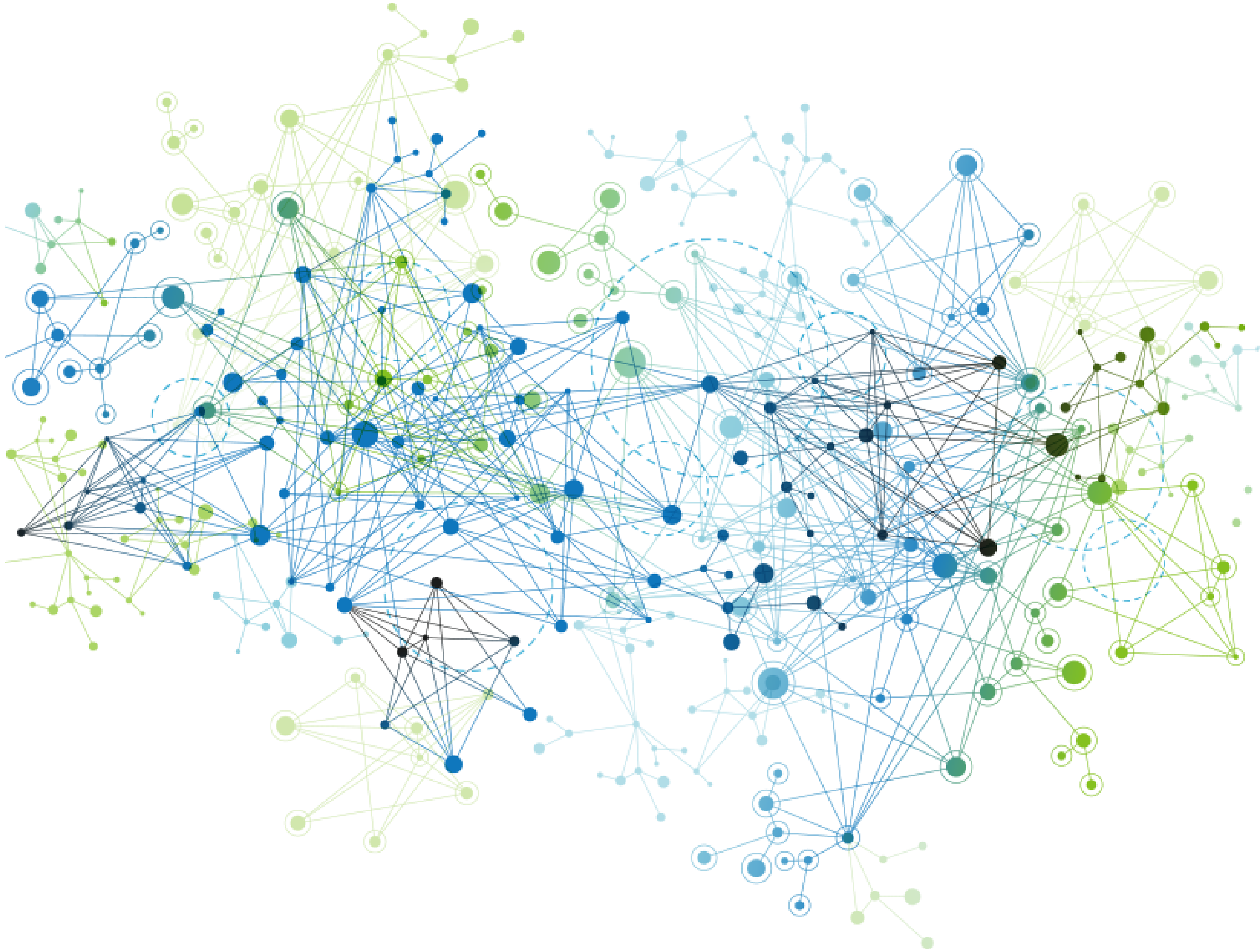
Deep Learning

- Keep in mind, for real data, we would no longer have values to compare these predictions to!



International Council for Technology Certifications

RNN on a Time Series



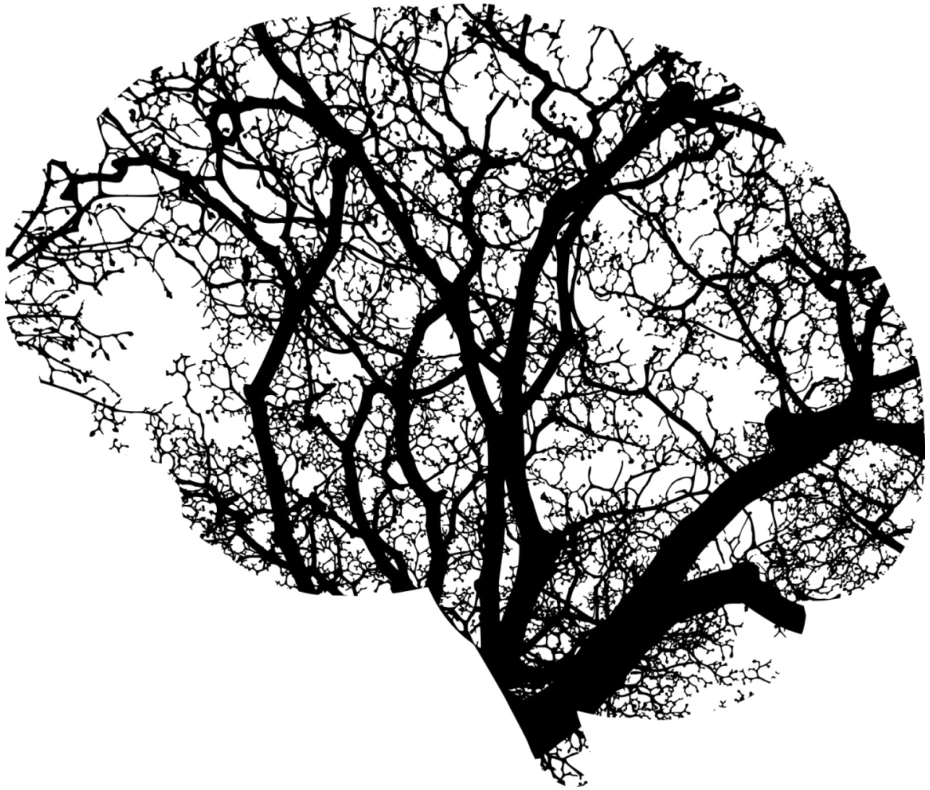
Deep Learning

RNN Exercises

RNN Exercises Solutions

ICTC International Council for Technology Certifications

Multivariate Time Series with LSTM RNNs



Deep Learning

Deep Learning

- As a quick bonus, we'll discuss how to use LSTMs and RNNs to predict multivariate time series.
- Keep in mind, there are a few cons to using LSTMs for this approach!

Deep Learning

- As with all neural networks, the model is essentially a black box, difficult to interpret.
- Also there are many well-studied alternatives that are simpler, such as SARIMAX and VARMAX models.

Deep Learning

- We highly recommend you try those more conventional approaches before settling on LSTMs or RNNs for multivariate time series data.
- Fortunately, setting up for multivariate data only requires 2 main changes.

Deep Learning

- Multivariate Time Series:
    - Change input shape in LSTM layer to reflect 2-D structure
    - Final dense layer should have a neuron per feature/variable.
  - Let’s explore these changes!