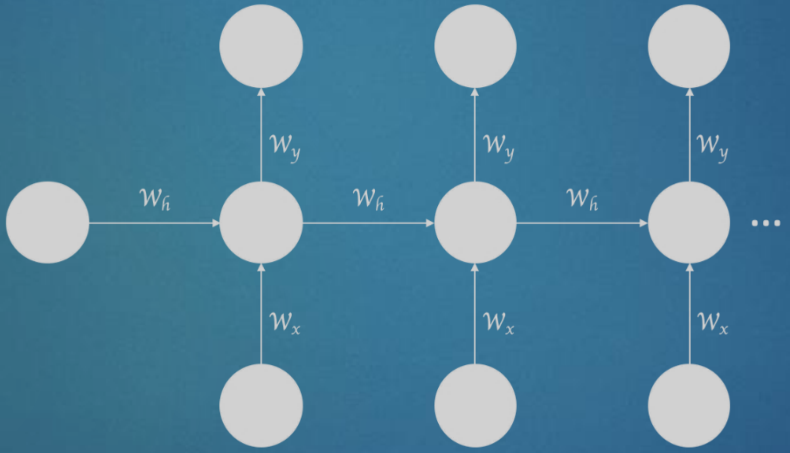
alitourani

Deep Learning from Scratch

Session #6: Recurrent Neural Networks



by: Ali Tourani – Summer 2021

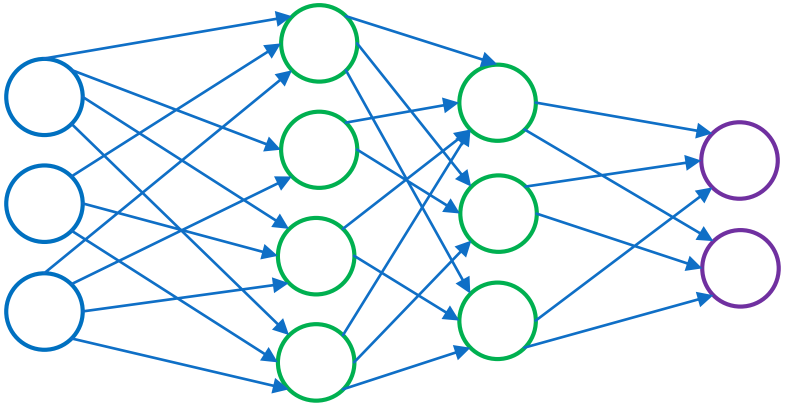
Agenda

- Sequence Models
- Recurrent Neural Networks (RNNs)
- Backpropagation Through Time (BPTT)
- Long Short-Term Memory (LSTM)
- Gated Recurrent Unit (GRU)

Sequence Models

Remember our simple ANNs?!

- Let’s call these architectures **Feedforward NNs**
  - The information is only passed in one direction
  - The connections between nodes do not form a cycle
  - **Single-layer Perceptron**
    - Only comparing the outputs with actual values
    - Gradient Descent
  - **Multi-layered Perceptron**
    - Backpropagation

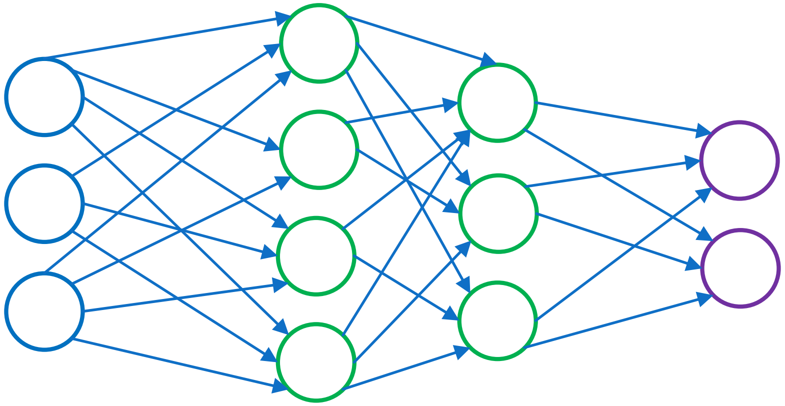


Sequence Models

Remember our simple ANNs?!

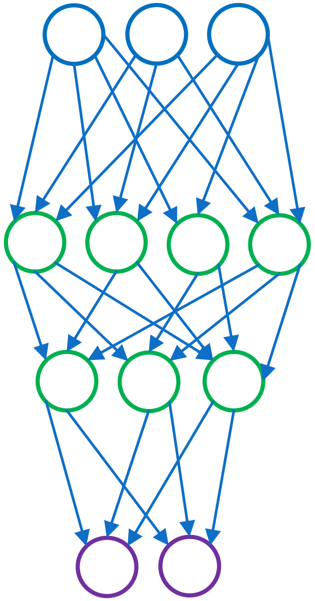
- Let’s call these architectures **Feedforward NNs**
  - The informatio … in one direction
  - T … es do not form a cycle
  - **Si**
    - … actual values
    -
  - **Mul**
    - Ba

In **FFNNs**, the current output is not dependent on the previous input.



Sequence Models

Note that …



Feed-Forward NN

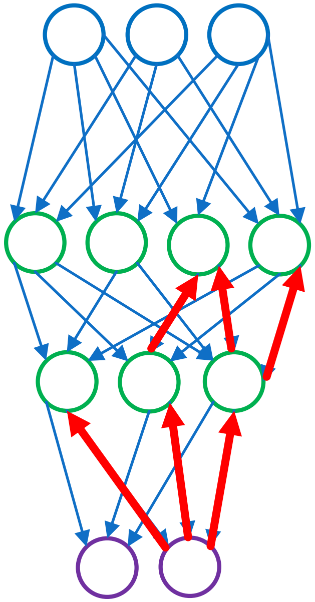
- An architecture

- Data travels from the input layer to hidden layers and finally, the output layer

Backpropagation

- A training algorithm

- Forwards the values, and only after error calculation, propagates them back to the earlier layers



Sequence Models

- **Sequence Modeling is:**
  - Predicting **what comes next**, according to the **previous information**
  - We need to know the prior state of the model to predict its next ones
- Machine learning models with sequences of data as input/output
- Samples of sequential data:
  - Text streams
  - Audio
  - Videos
  - Time-series data

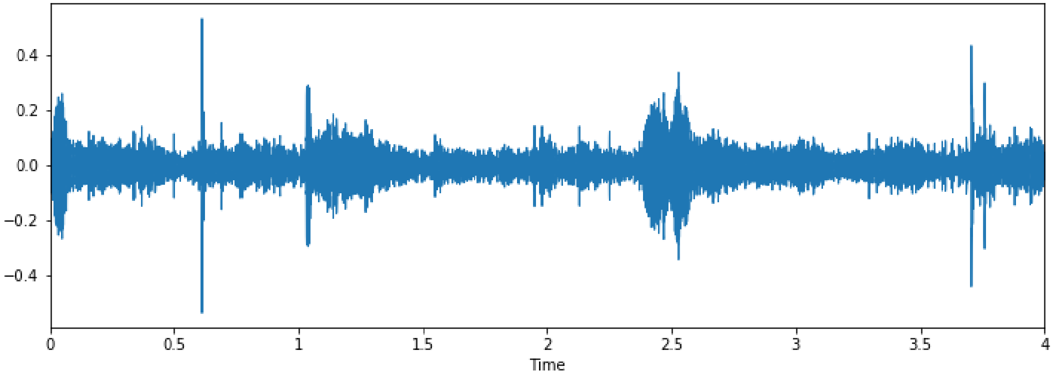
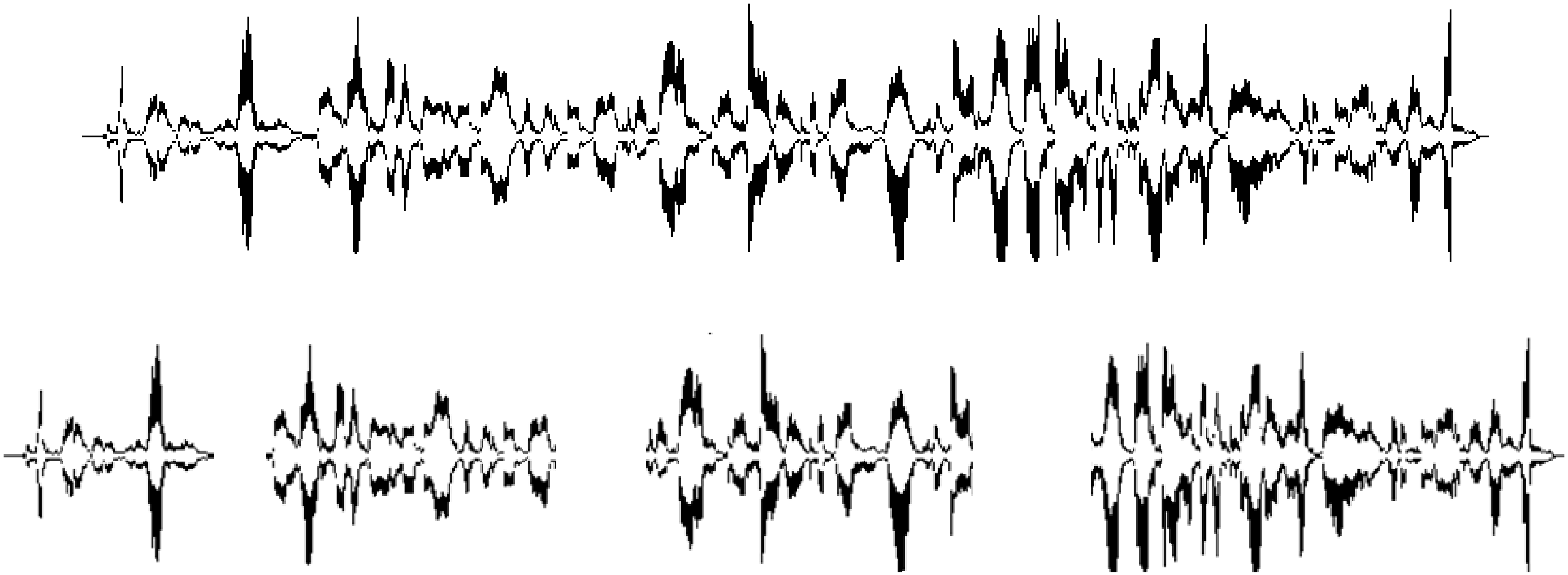


Image source: https://www.analyticsvidhya.com

Sequence Models

- Sequential data can be split into sub-samples - **Audio**



Sequence Models

- Sequential data can be split into sub-samples - **Video**

Image source: https://www.mediacolelge.com

Sequence Models

What else about Sequence Modeling?

- The current output is not independent on the previous input
  - **In contrast with Feedforward Neural Networks (FNNs)**
- The length of the input is not fixed
  - We might have a stream of data

Use case: speech tagging

- Target: marking up a word in a text based on its **definition** and **context**

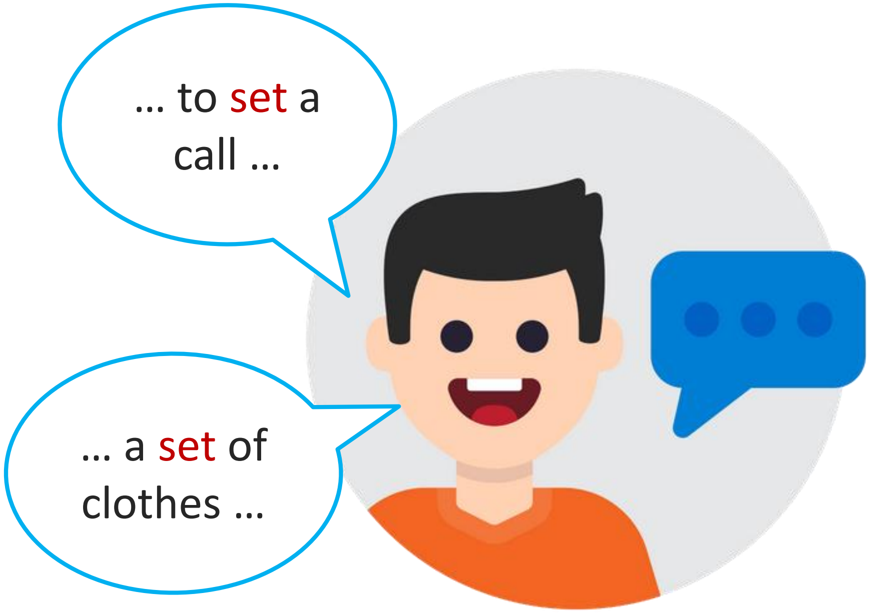
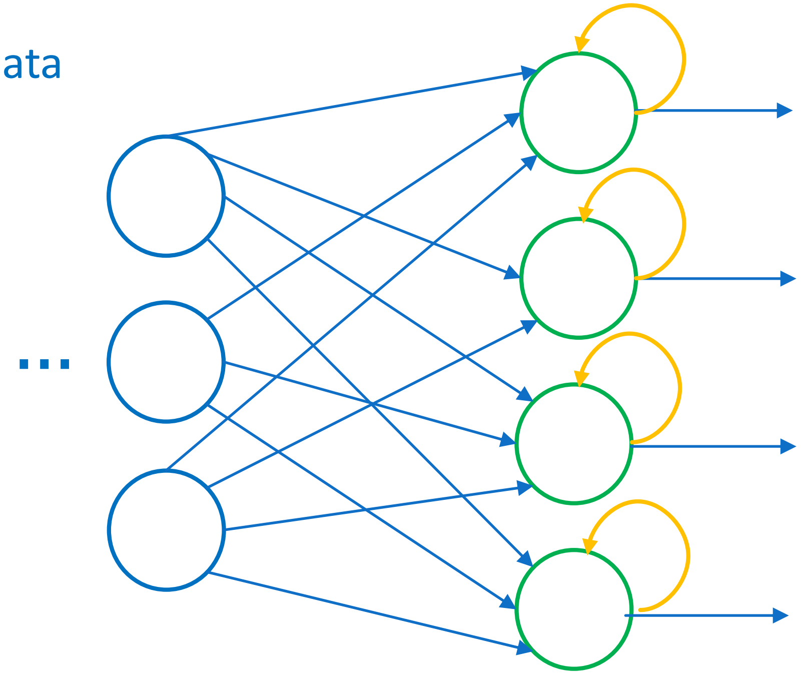


Image source: https://www.iconscout.com

Recurrent Neural Networks

- A DL algorithm + a type of ANN architecture
  - **The output of step *s* is fed to the step *s+1***
- Specialized for processing sequential data
  - They remember previous inputs
  - They can share the features
  - They use historical information
- Use cases:
  - Time series predictions
  - Natural Language Processing (NLP)



Recurrent Neural Networks

Let’s take a deeper look …

Adding the temporal dimension to our network

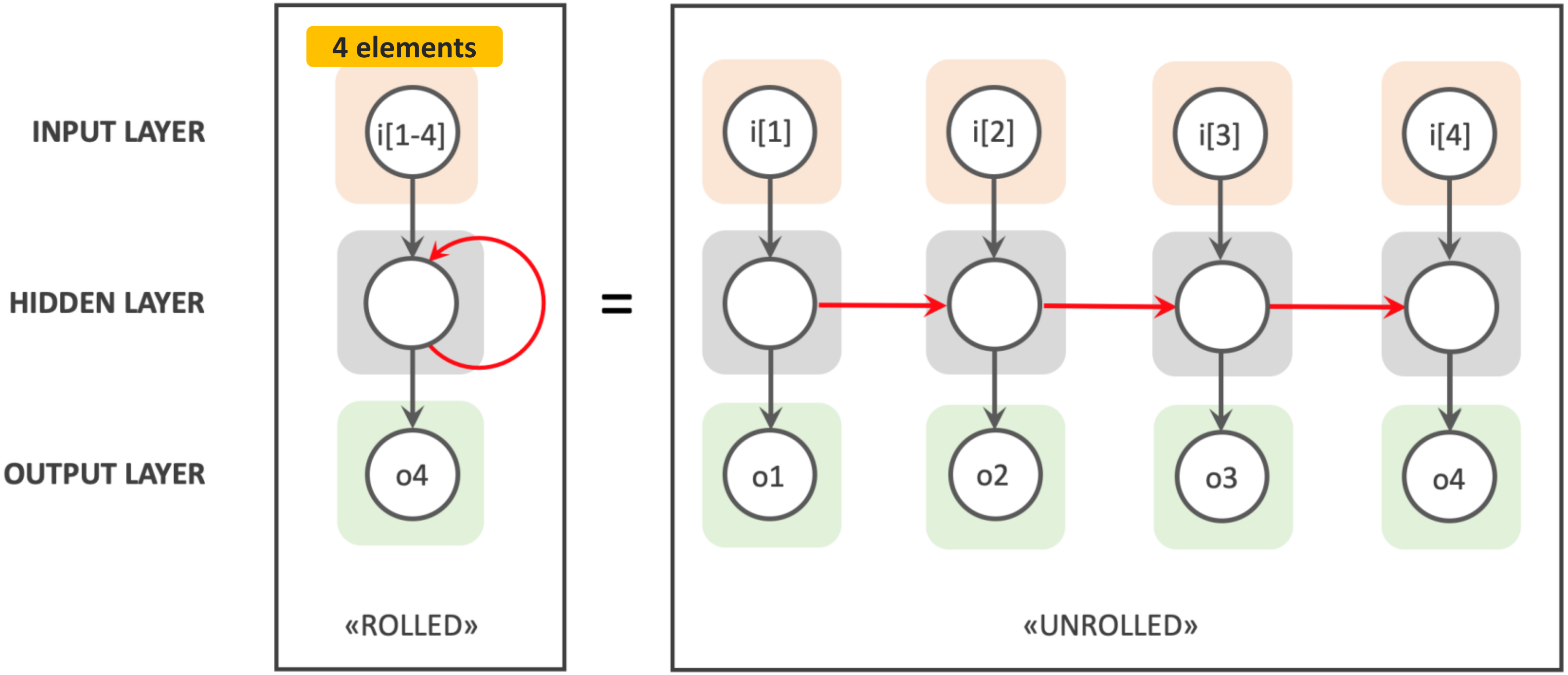
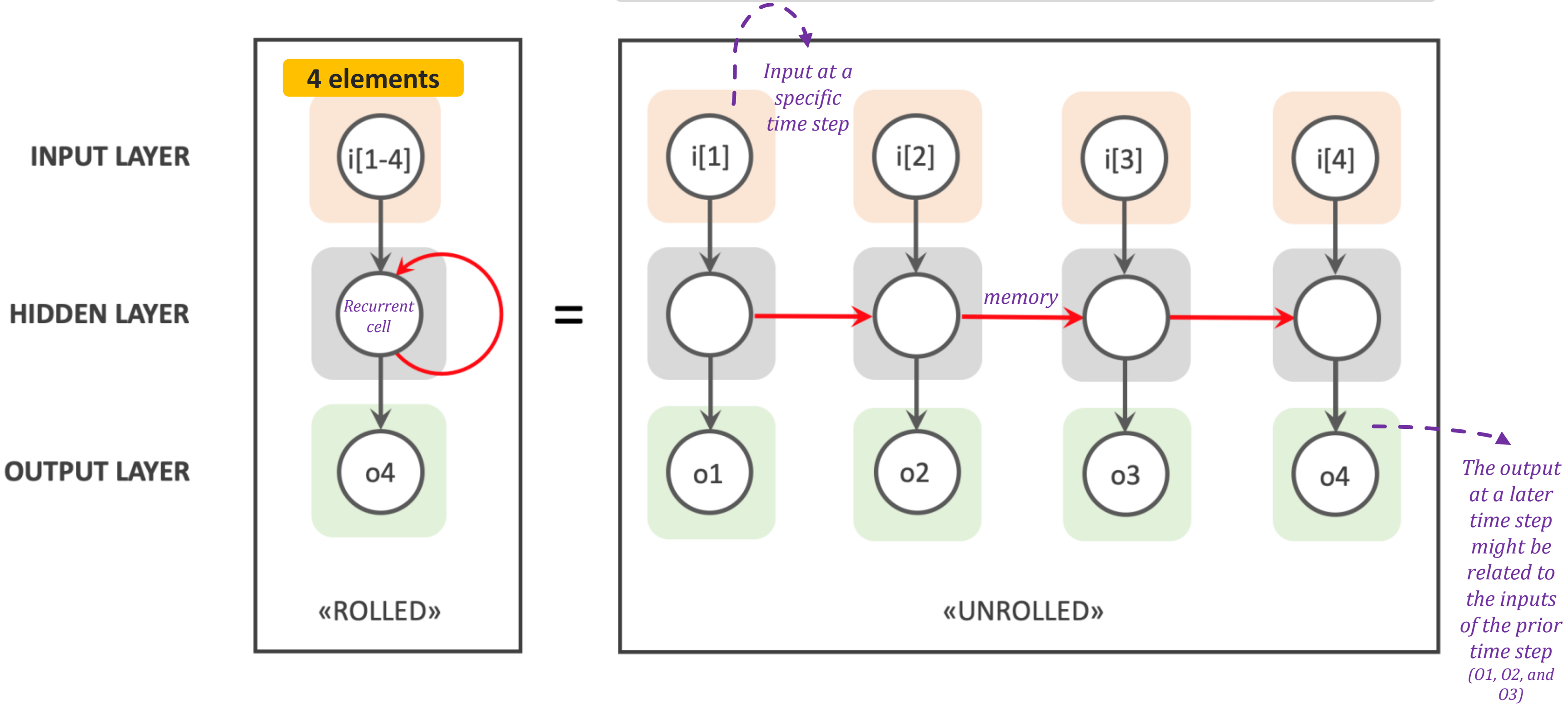


Image source: https://www.bouvet.no/

Recurrent Neural Networks

Let’s take a deeper look …

Adding the temporal dimension to our network



Input at a specific time step

The output at a later time step might be related to the inputs of the prior time step (O1, O2, and O3)

Image source: https://www.bouvet.no/

Recurrent Neural Networks

Let’s take a deeper look …

Adding the temporal dimension to our network

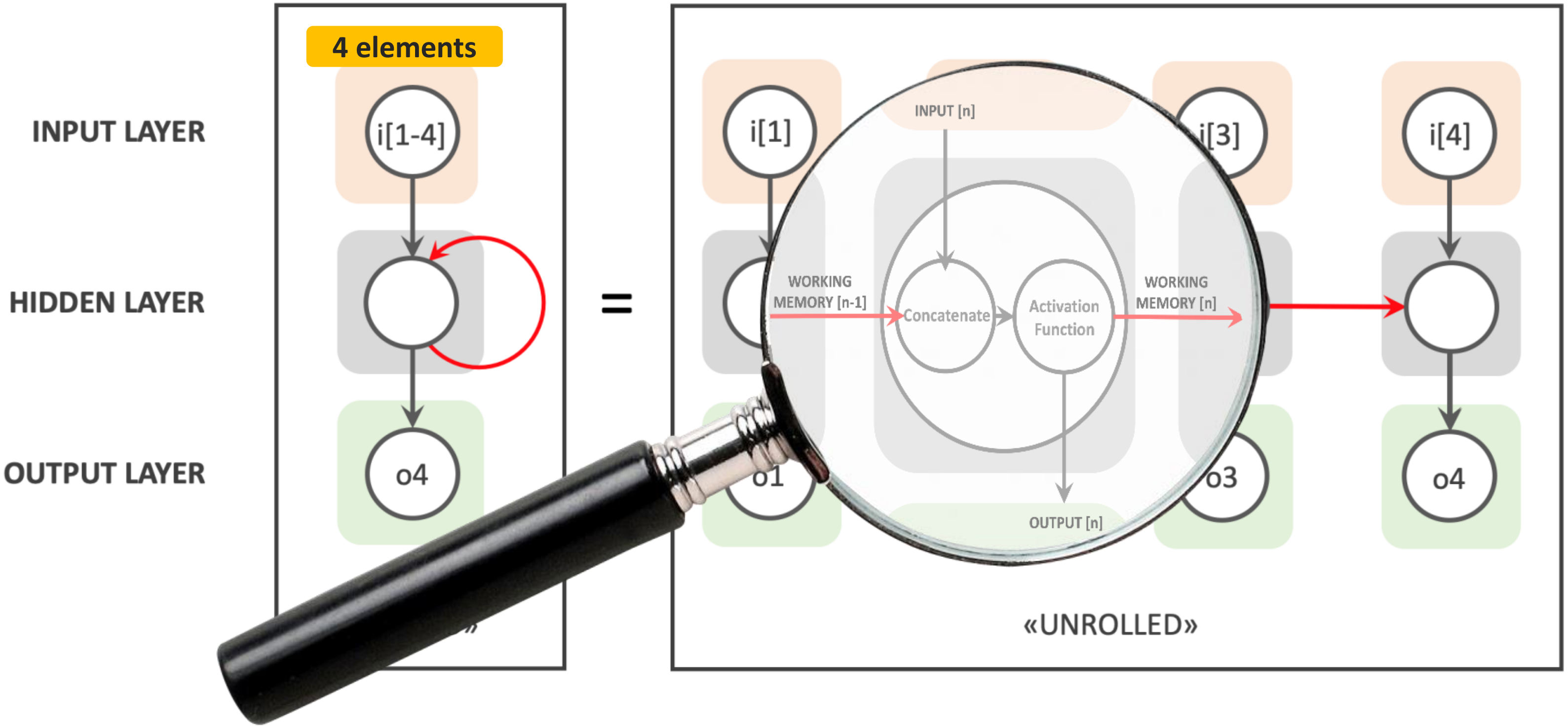


Image source: https://www.bouvet.no/

Recurrent Neural Networks

Let’s take a deeper look …

$$\hat{o}_t = f(i_t, h_{t-1})$$

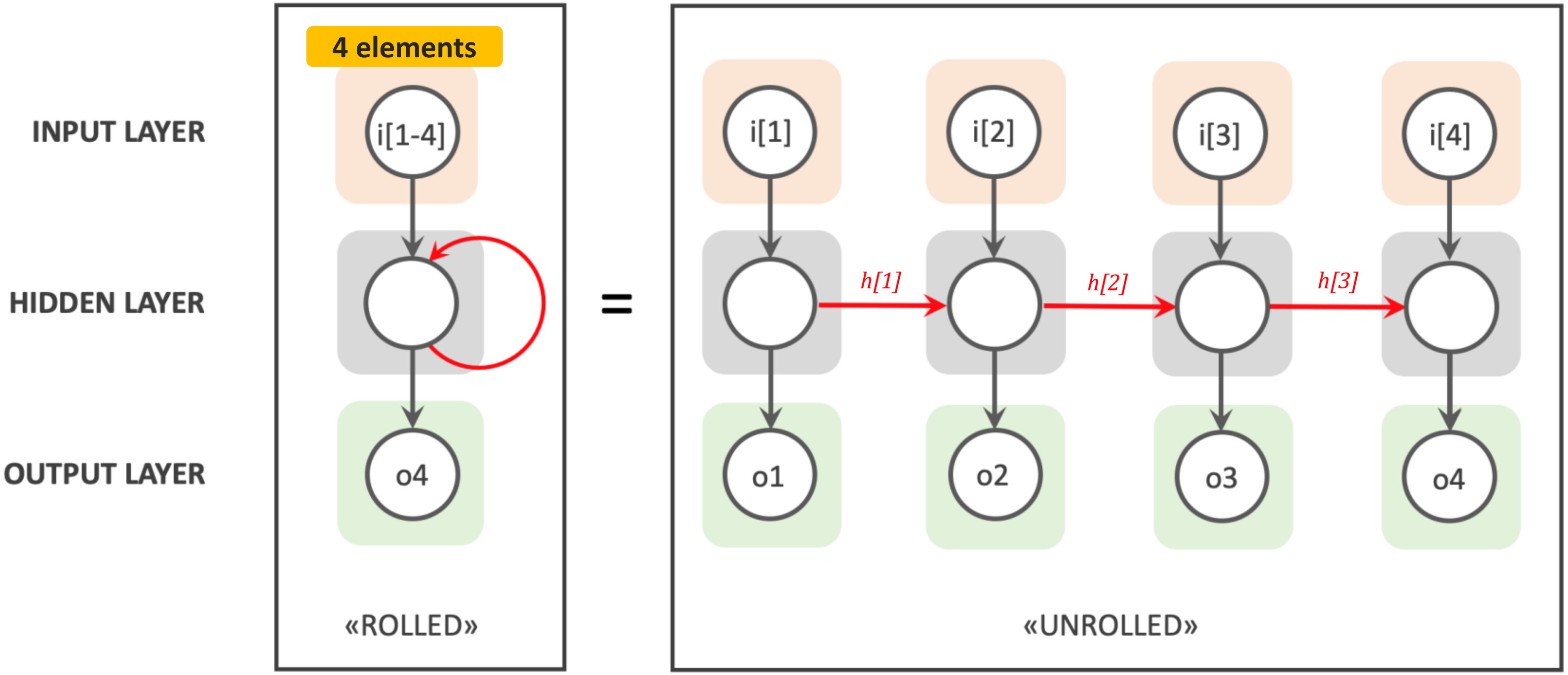
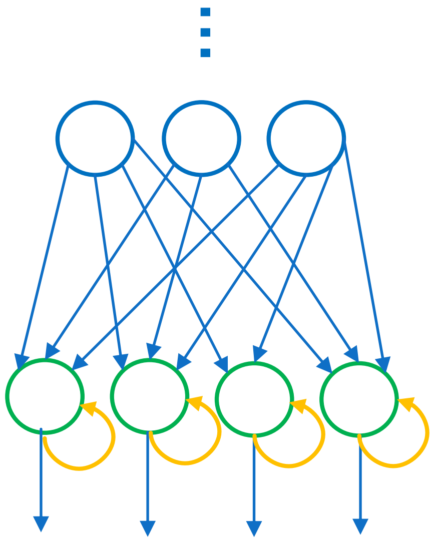


Image source: https://www.bouvet.no/

Recurrent Neural Networks

Note that …



A RNN

- An architecture

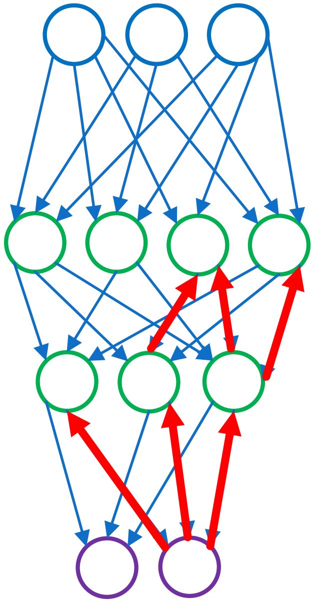
- Contains weights that are pointed into themselves

- Used for modeling temporal or sequential data

Backpropagation

- A training algorithm

- Forwards the values, and only after error calculation, propagates them back to the earlier layers



Recurrent Neural Networks

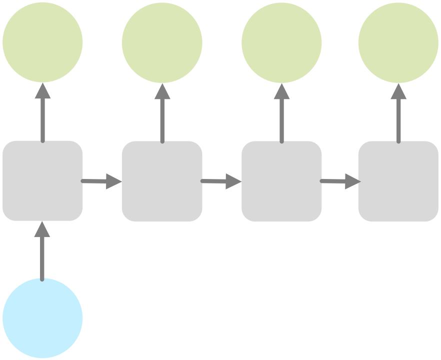
⚠ Important Notes on RNNs

- RNNs support the processing of sequential data using loops
  - A loop: a chain of identical Feed Forward ANNs
- The loss function is defined based on the loss at each time step
- Backpropagation is done at each point in time **(BPTT)**
  - Although each loop has its input-output pair, they share the same weights
- The "working memory" of standard RNNs struggles to retain longer-term dependencies
  - **Vanishing Gradient problem**

Recurrent Neural Networks

⚠ Important Notes on RNNs

- RNNs can handle variable-length sequences while keeping the order of input data and memorizing the dependencies among them
  - In contrast with the fixed dimensionality in Feedforward NNs
- RNNs can share parameters (e.g., weights) across the sequence of data



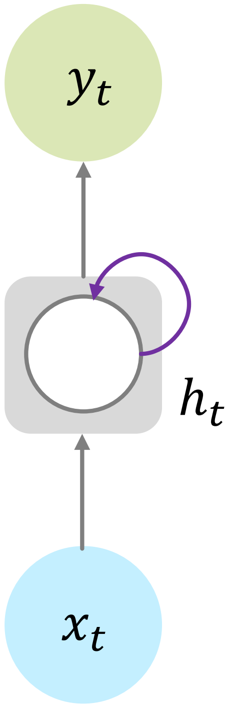
Recurrent Neural Networks

Important Notes on RNNs

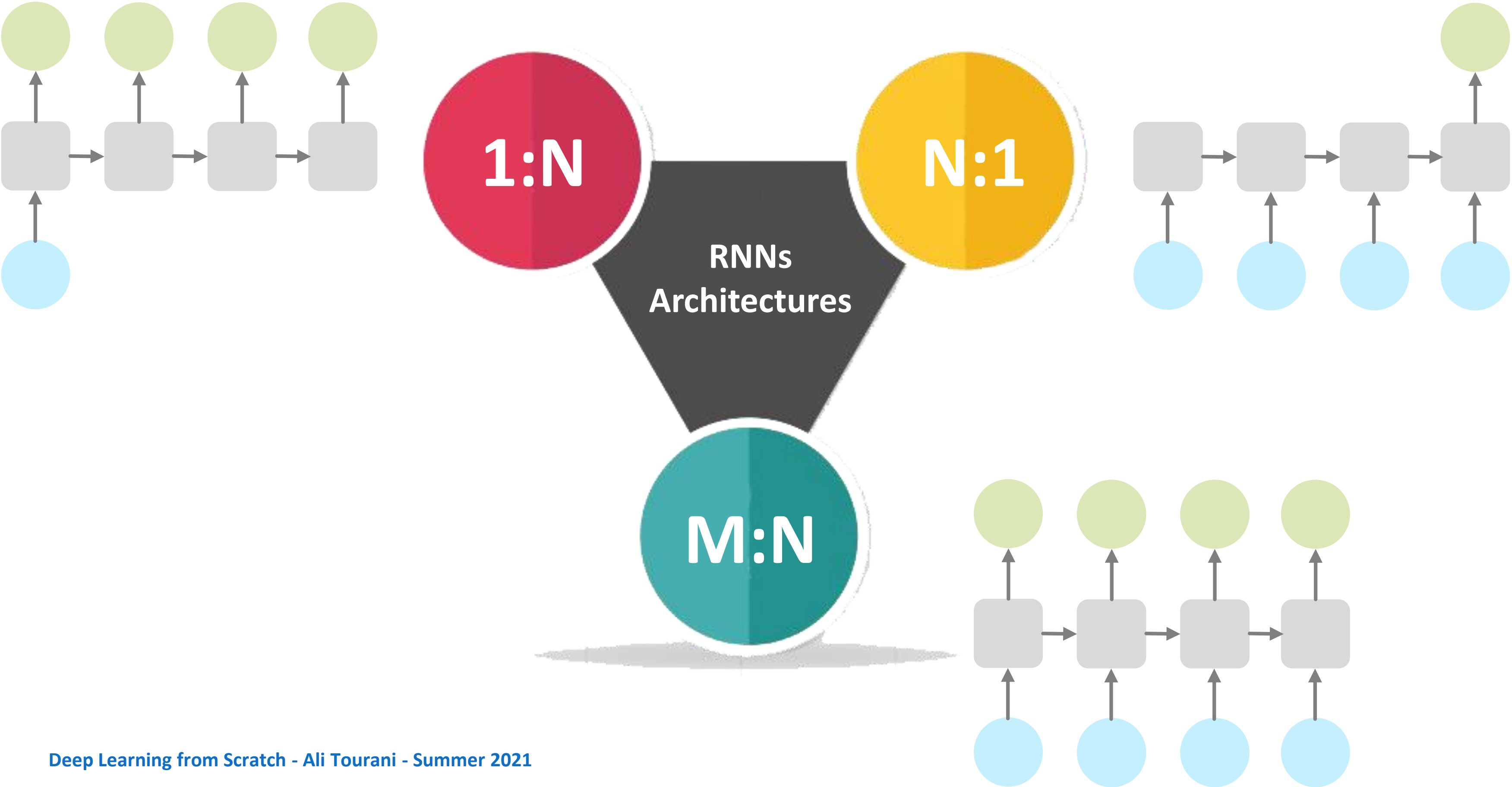
- In RNNs, we can always find a **state** that is updated at each time step

$$\underset{\text{cell state}}{h_t} = \underset{\text{weighted function}}{f_W}(x_t , \underset{\text{old state}}{h_{t-1}})$$

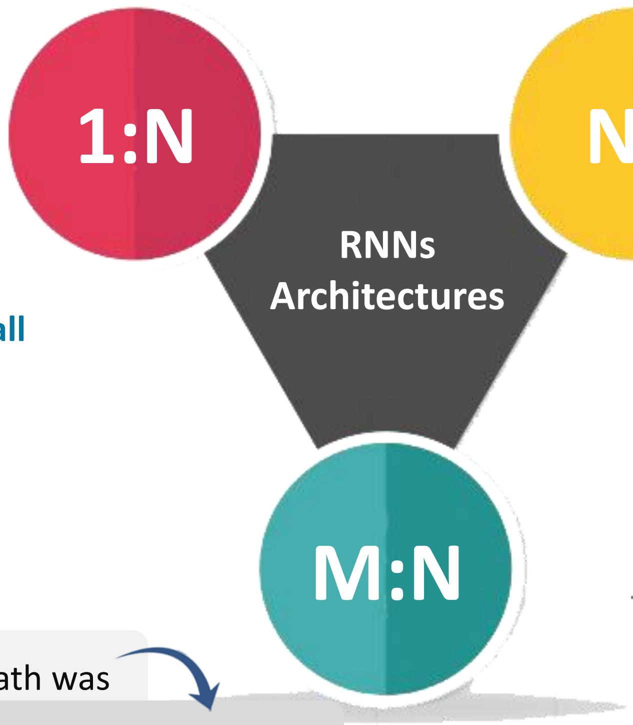
- Weights in the weighted function are used for training
- The same procedure and parameters are used at **every time step**



Recurrent Neural Networks



Recurrent Neural Networks



Use case:
- Image Captioning

A cat is playing with a green ball

Use case:
- Sentiment Classification

I love this movie. I have seen it several times

Use case for M≠N:
- Machine Translation

🇬🇧 I always knew what the right path was

🇮🇹 Ho sempre saputo quale fosse la strada giusta

Use case for M=N:
- Named Entity Recognition

My Name is Ali Person
I am from Iran GPE
I know John Person very well

Recurrent Neural Networks

Applications of Standard RNNs

Speech Recognition
(human speech to written text)

Image source: https://www.itchronicles.com

Recurrent Neural Networks

Applications of Standard RNNs

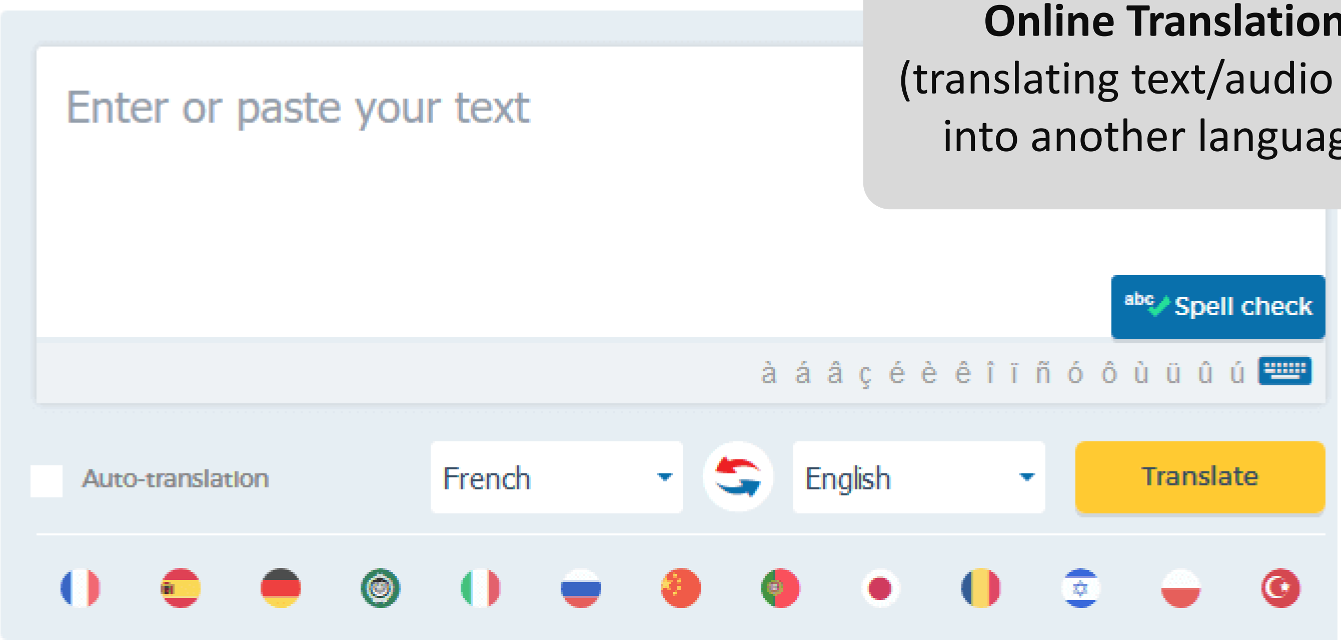
Stock Prediction
(determining the future value of a stock)

Image source: https://www.morioh.com

Recurrent Neural Networks

Applications of Standard RNNs

Online Translation
(translating text/audio data into another language)

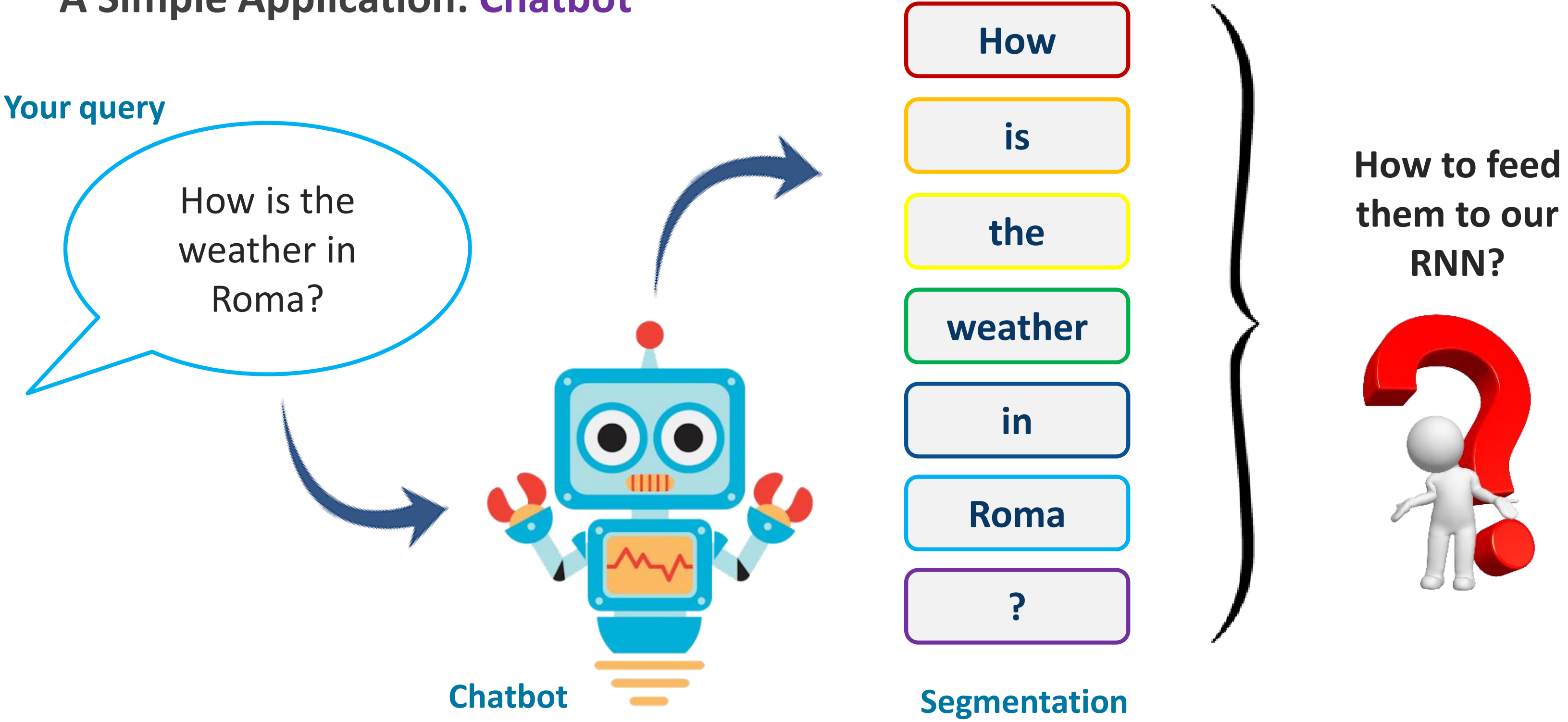


Online translator based on AI technology for French, Spanish, German, Russian and many more languages

Image source: https://www.helpdeskgeek.com

Recurrent Neural Networks

A Simple Application: Chatbot



Your query

How is the weather in Roma?

Chatbot

How

is

the

weather

in

Roma

?

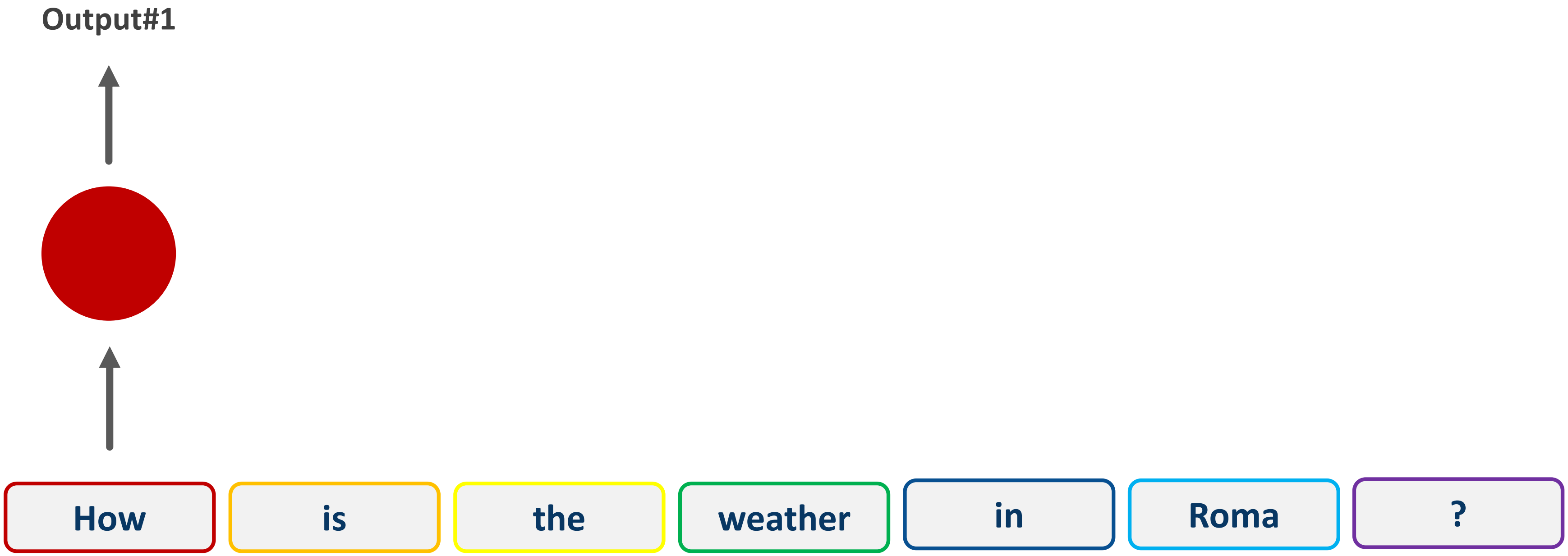
Segmentation

How to feed them to our RNN?

Image source: https://www.intelligence.marketingonline.nl/

Recurrent Neural Networks

A Simple Application: Chatbot



Output#1

How

is

the

weather

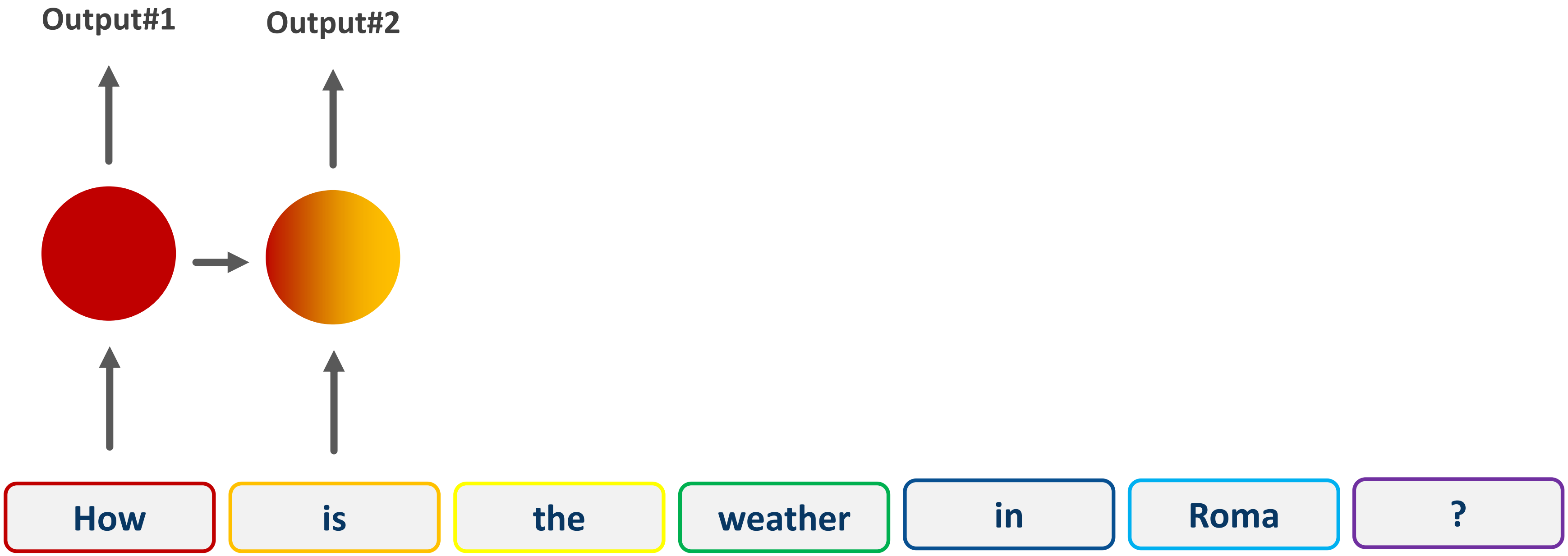
in

Roma

?

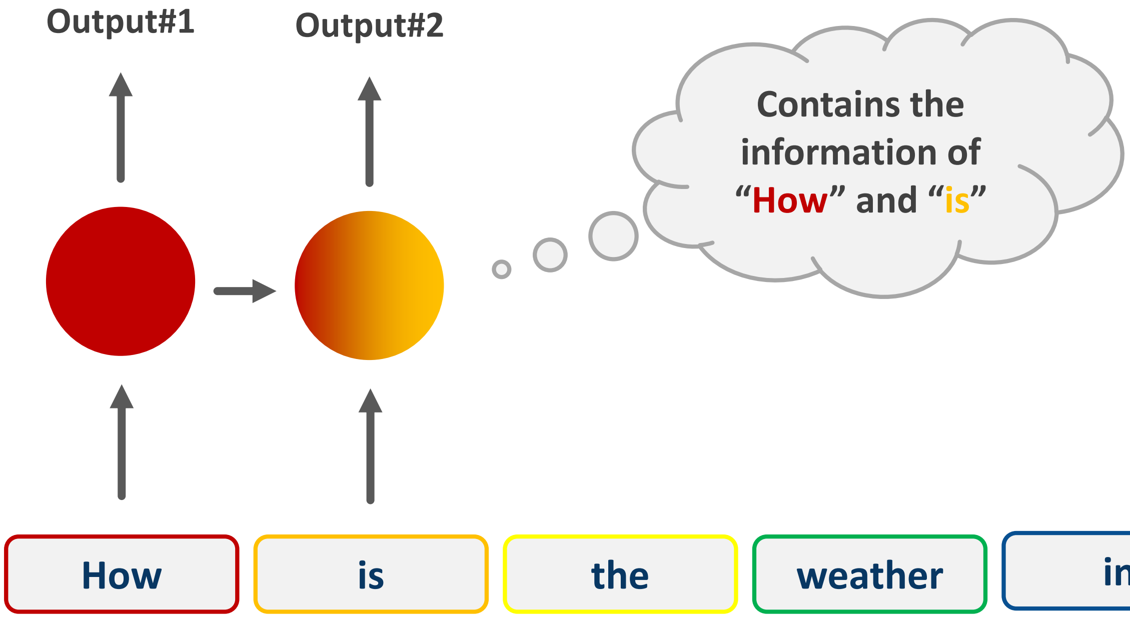
Recurrent Neural Networks

A Simple Application: Chatbot



Recurrent Neural Networks

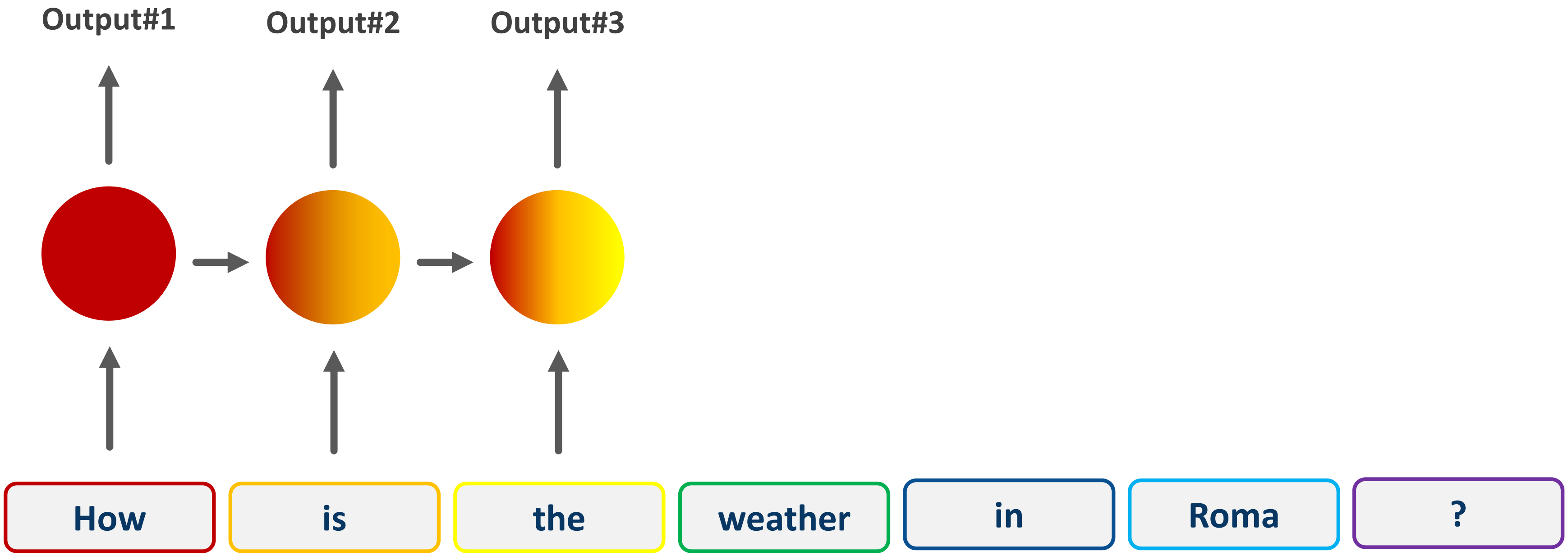
A Simple Application: Chatbot



The **hidden state** in RNNs represent information of the previous steps

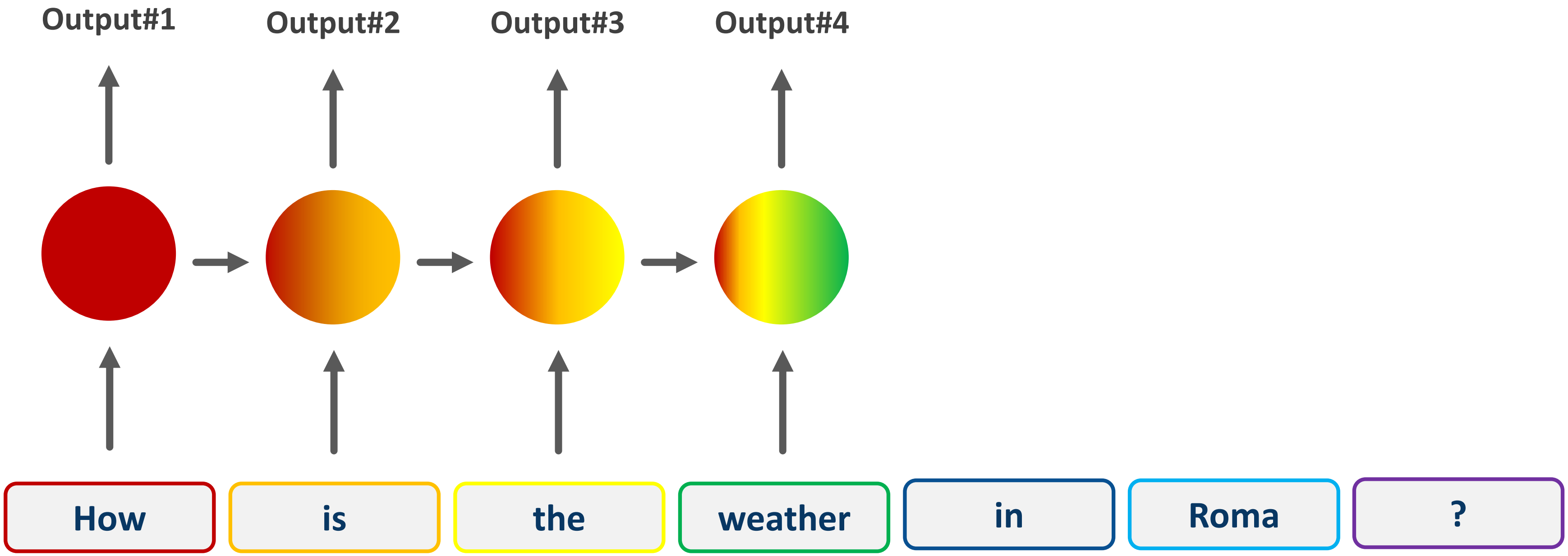
Recurrent Neural Networks

A Simple Application: Chatbot



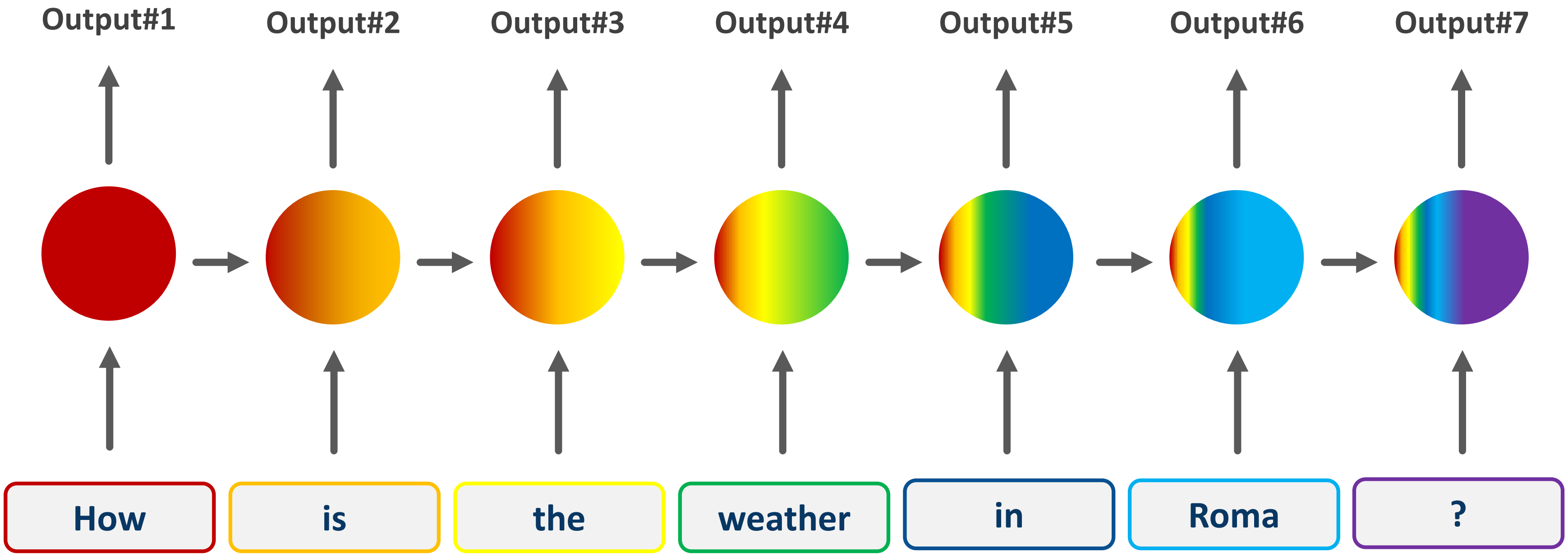
Recurrent Neural Networks

A Simple Application: Chatbot



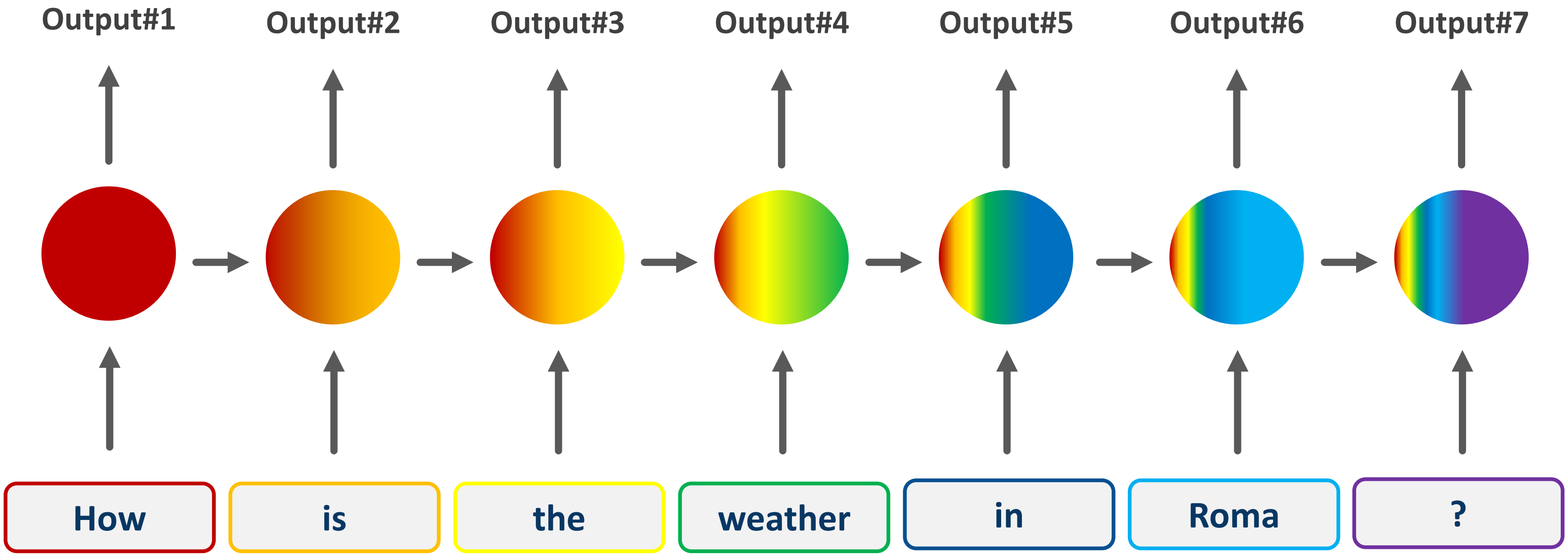
Recurrent Neural Networks

A Simple Application: **Chatbot**



Recurrent Neural Networks

A Simple Application: Chatbot



Now the RNN holds the information of all the previous steps

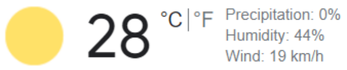
Recurrent Neural Networks

A Simple Application: Chatbot

How is the weather in Roma?

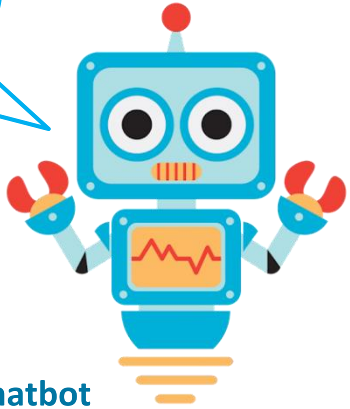
Your query

Output#7



28 °C | °F

Precipitation: 0%
Humidity: 44%
Wind: 19 km/h



Chatbot

Image source: https://www.intelligence.marketingonline.nl/

Recurrent Neural Networks

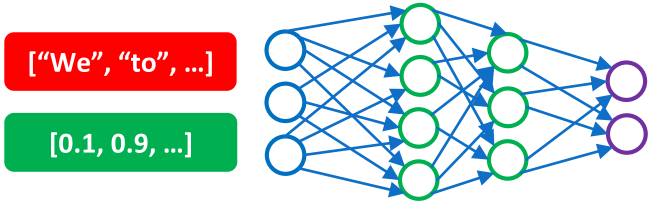
A Simple Application: Word Prediction

We decided to go on a short trip this ________.

Given

Want to know

- How to represent the concepts behind words to our ANN?
  - Surely, ANNs cannot understand the meaning of the words!
  - We have to find a way to make them numerical
    - Goal: applying mathematical operations



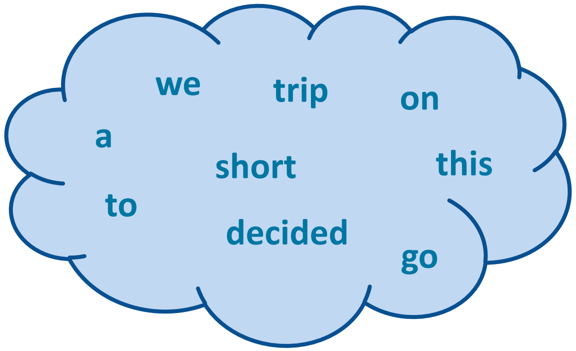
["We", "to", …]

[0.1, 0.9, …]

Recurrent Neural Networks

A Simple Application: Word Prediction

We decided to go on a short trip this _________.



Vocabulary

Word	Index
a	00001
decided	00002
go	00003
…	…

Indexing

"a" = [1, 0, 0, 0, 0, …]

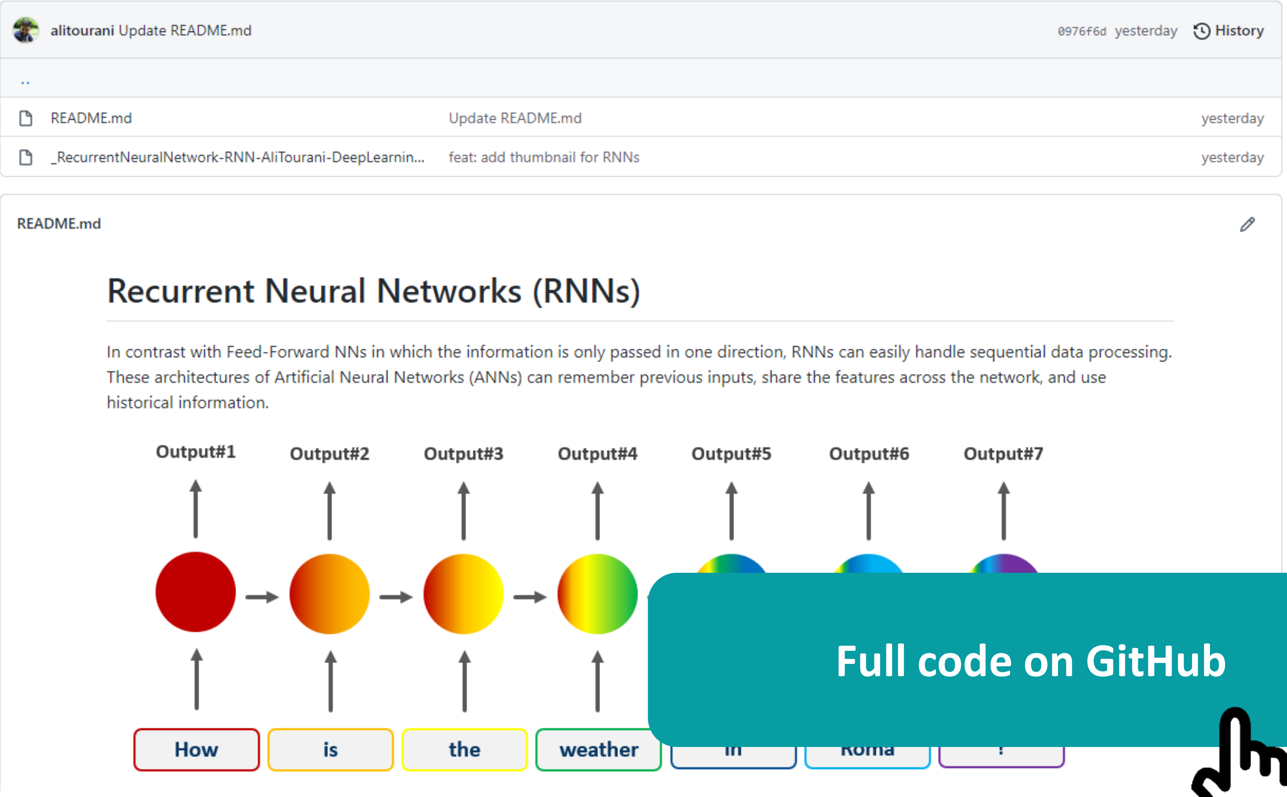
"decided" = [0, 1, 0, 0, 0, …]

"go" = [0, 0, 1, 0, 0, …]

....

Embedding

Recurrent Neural Networks



alitourani Update README.md 0976f6d yesterday History

..

README.md	Update README.md	yesterday
_RecurrentNeuralNetwork-RNN-AliTourani-DeepLearnin...	feat: add thumbnail for RNNs	yesterday

README.md

Recurrent Neural Networks (RNNs)

In contrast with Feed-Forward NNs in which the information is only passed in one direction, RNNs can easily handle sequential data processing. These architectures of Artificial Neural Networks (ANNs) can remember previous inputs, share the features across the network, and use historical information.

Full code on GitHub

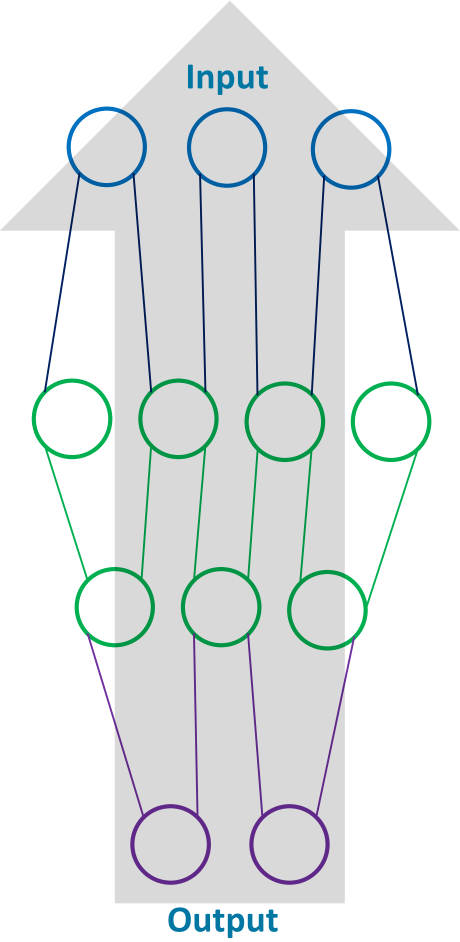
Backpropagation Through Time

Backpropagation in Feedforward ANNs

- Recall: Session#1
- GDA algorithm
  - The gradient of the loss with respect to each weight parameter
  - Shifting parameters to minimize final loss

$$\Delta w_i(t) = \mu(-\frac{\delta J(W)}{\delta w})$$

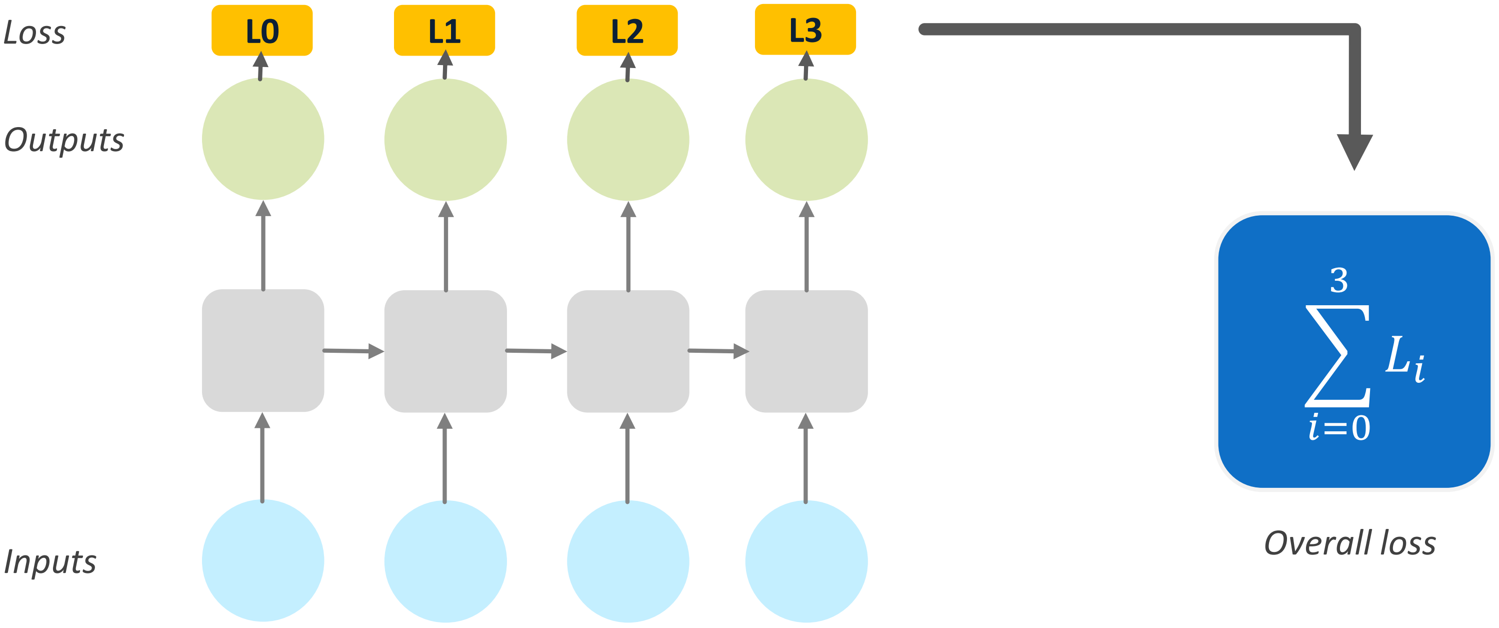
$$\frac{\delta J(W)}{\delta w_1} = \frac{\delta J(W)}{\delta y} * \frac{\delta y}{\delta b} * \frac{\delta b}{\delta a} * \frac{\delta a}{\delta w_1}$$



Backpropagation Through Time

Backpropagation in RNNs

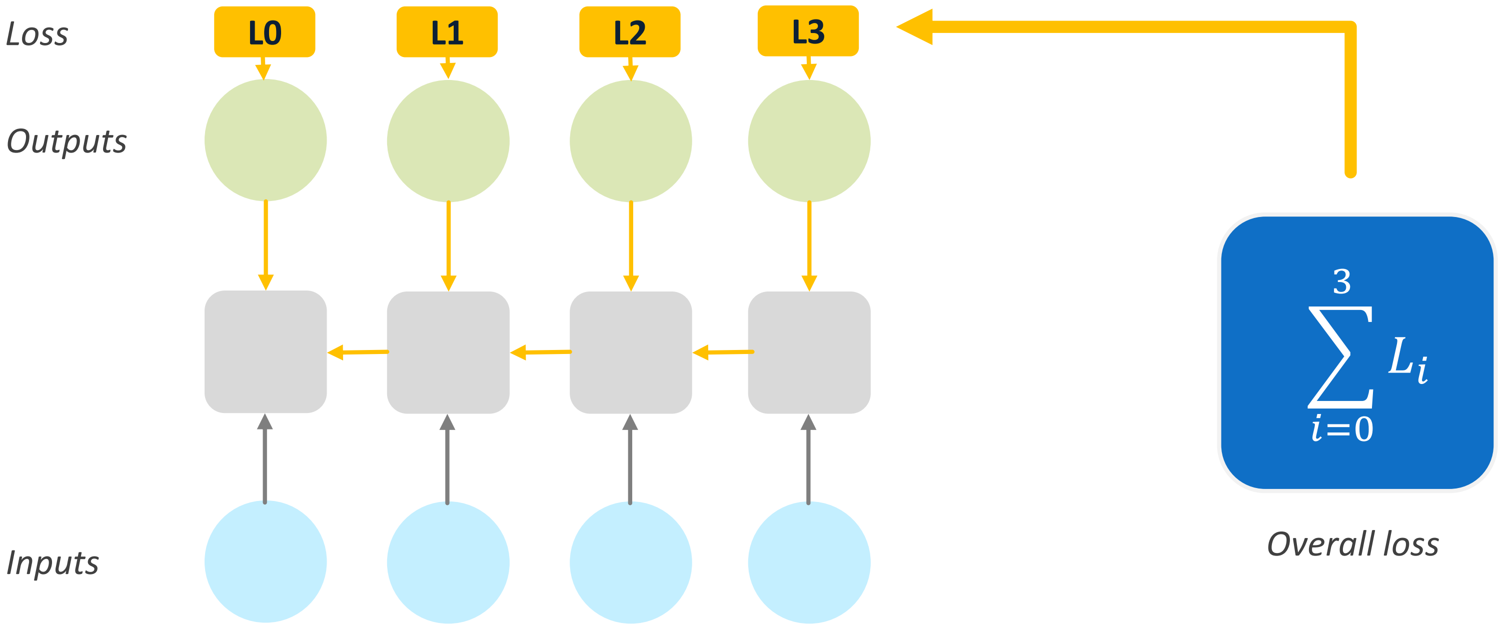
- In the forward pass, the loss values are calculated at each time step.



Backpropagation Through Time

Backpropagation in RNNs

- In the backward pass, the overall loss spreads errors in each time step.



Backpropagation Through Time

Backpropagation in RNNs

- Flowing gradients in a repetitive manner cause some problems:
  - Many **matrix multiplications** for updating the weights
  - Adding **many computation factors** while moving towards the initial state
  - We may face one of these problems:

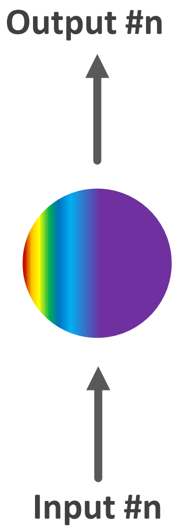
1. **Exploding gradient:** accumulation of large error gradients → *huge updates* ⚠
2. **Vanishing gradient:** minimal gradients → *preventing the update process* ⚠

Backpropagation Through Time

Vanishing Gradient Problem

- Recall the GDA and backpropagation
- If the number of hidden layers is huge, the gradient diminishes dramatically as it is propagated backward
- The error is so tiny when it reaches the layers close to the input
  - A smaller error means less impact on the learning process → **Vanishing Gradients**
- The network captures only short-term dependencies

Vanishing gradients challenges the network, so it is unclear which direction the parameters should move to improve the cost function.

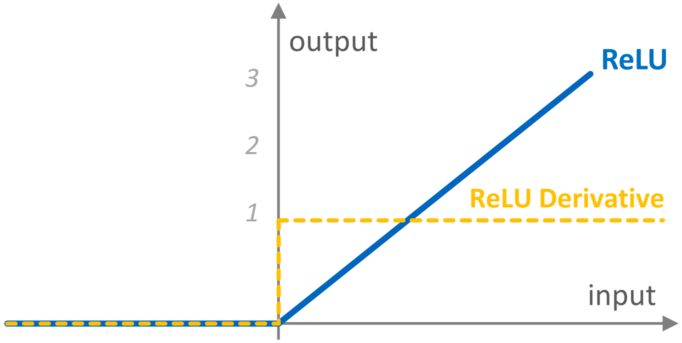


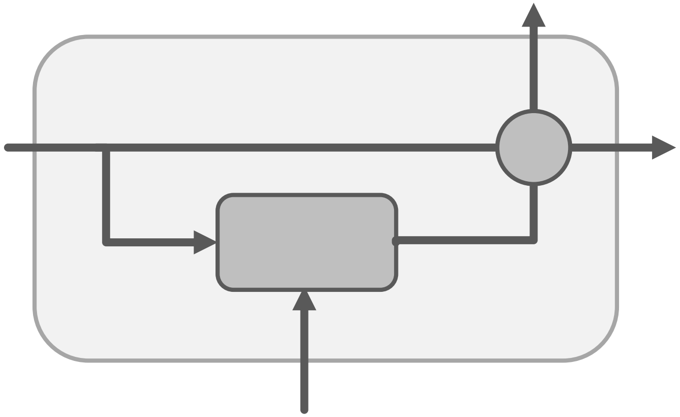
Backpropagation Through Time

Vanishing Gradient Problem

- We can resolve this issue by:
  - Smartly selecting the AFs of the network
    - **ReLU** is a great choice!
  - Smartly initializing the parameters
    - Trying to prevent the weights from shrinking to zero
  - Using **Gated Cells**
    - Using a complex recurrent unit with gates

      to enable controlling the data



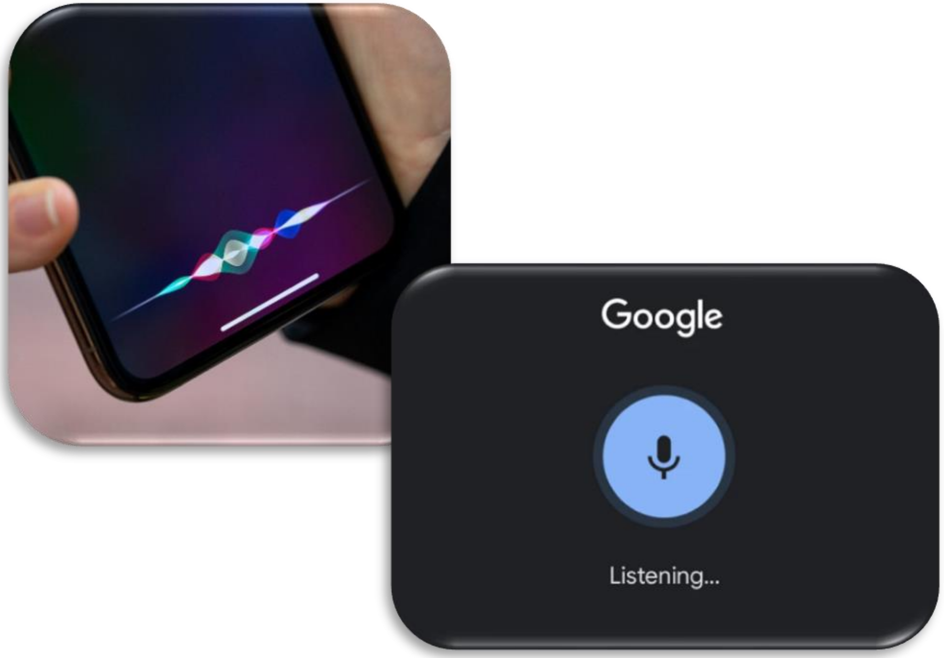


Long Short-Term Memory

- A popular deep learning algorithm for Sequence Models
- It uses a **Gated Cell** to track information that flows in the network throughout many time steps

Use cases:

- Apple’s Siri
- Google’s voice search
- Time-series predictions
- Text classification



Long Short-Term Memory

What is LSTM?

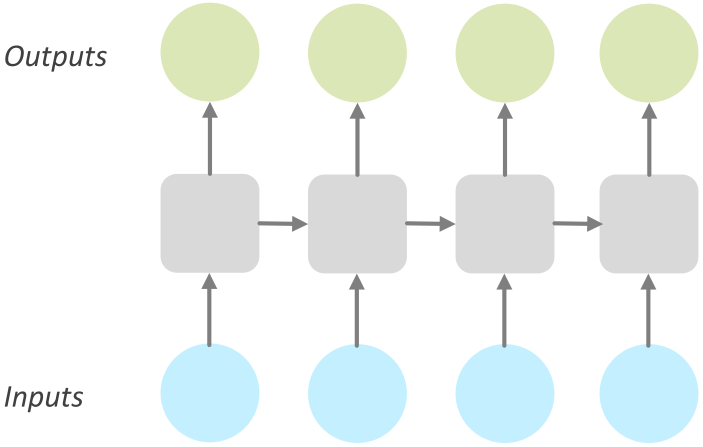
- Traditional RNNs are not good at capturing long-range dependencies
  - The main reason is the **Vanishing Gradient problem**
  - They may stop the ANN from training
- **LSTM** enables them to remember the input over a longer period
- How does it reflect the inputs?
  - Through passing the **cell state** + **hidden state** to the

    next time step

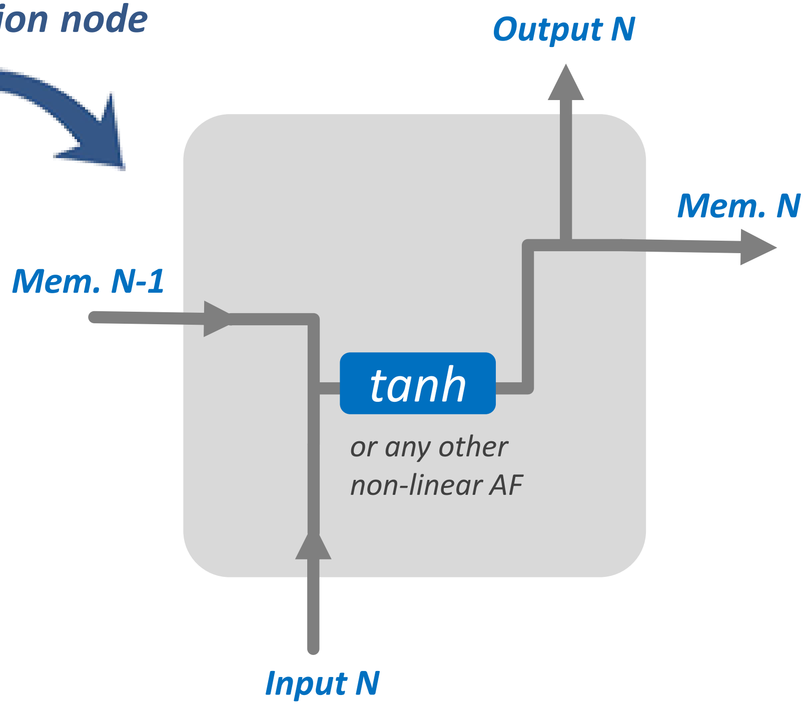
Image source: https://www.dlpng.com

Long Short-Term Memory

Standard RNNs

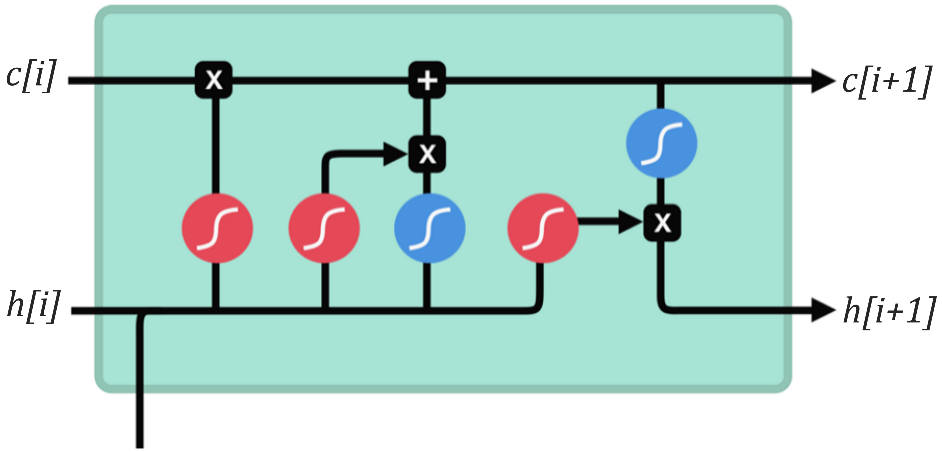


A closer look at the Computation node



Long Short-Term Memory

LSTMs' repeating components contain Computational Blocks to control the flow of information.



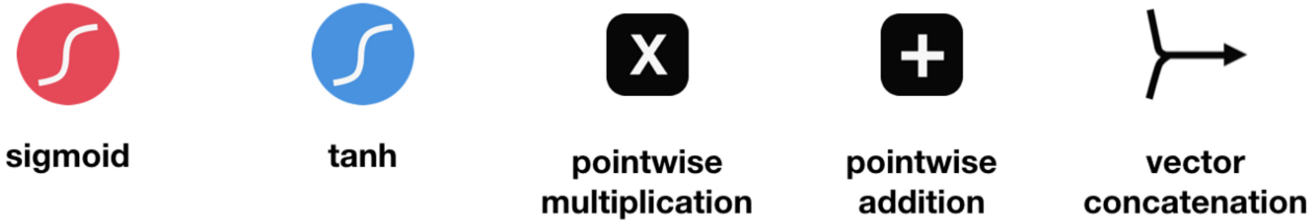


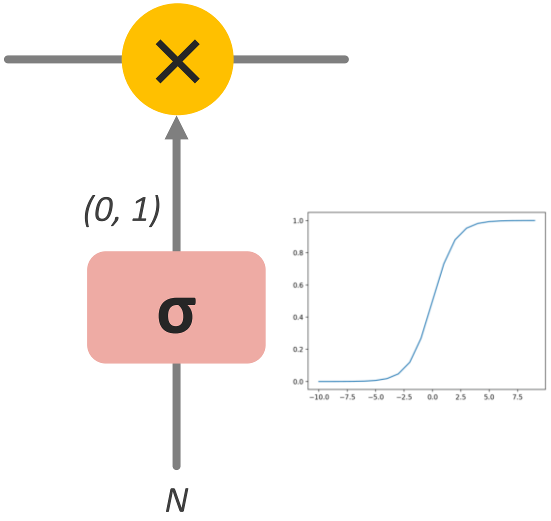
Image source: https://www.towardsdatascience.com

Long Short-Term Memory

Memory Management

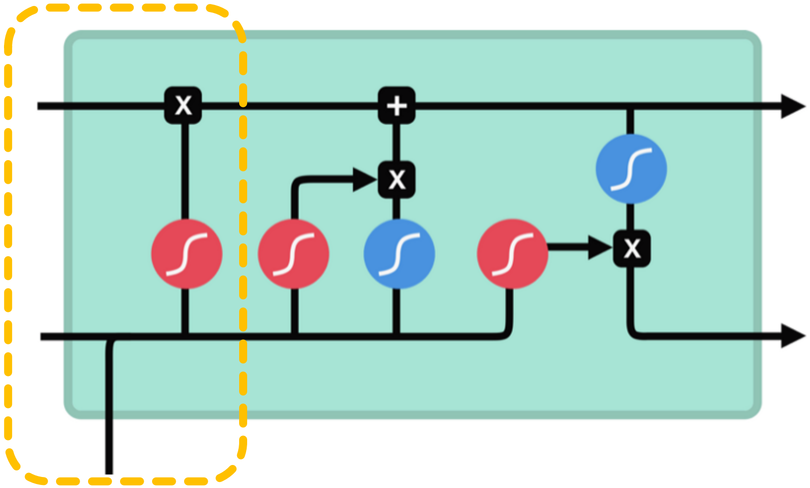
- LSTMs use a Gating Mechanism for:
  - Controlling the gradient propagation
  - Keeping, updating, ignoring, or forgetting information in the memory cell

A Gate lets the information pass via an ANN layer (e.g., Sigmoid) and a pointwise multiplication



Long Short-Term Memory

Using this block, LSTM can ***forget*** *irrelevant parts of the Previous State (PS) through passing the PS through a Sigmoid AF for filtration*



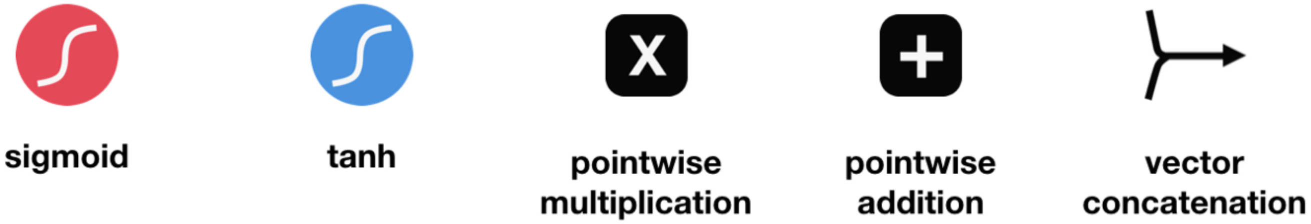
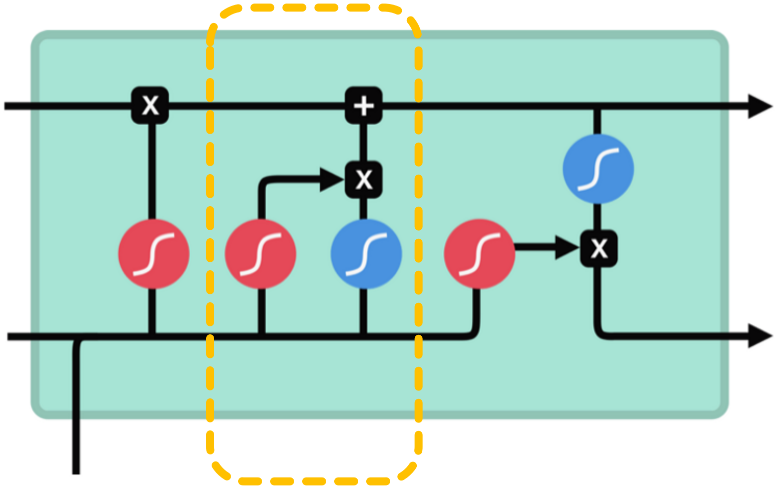


Image source: https://www.towardsdatascience.com

Long Short-Term Memory

Using this block, LSTM can ***store*** *relevant information into the cell state and keeping them in the memory*



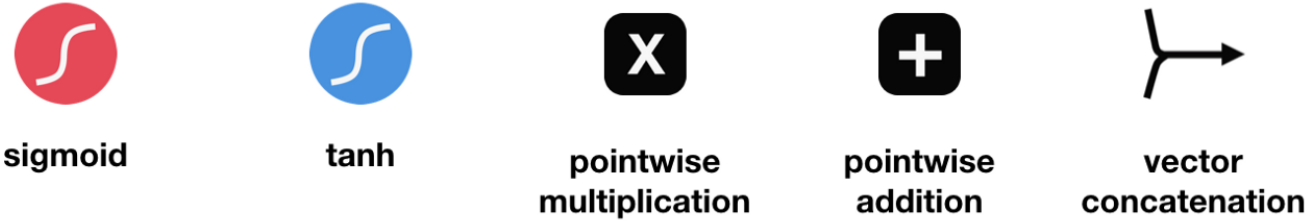
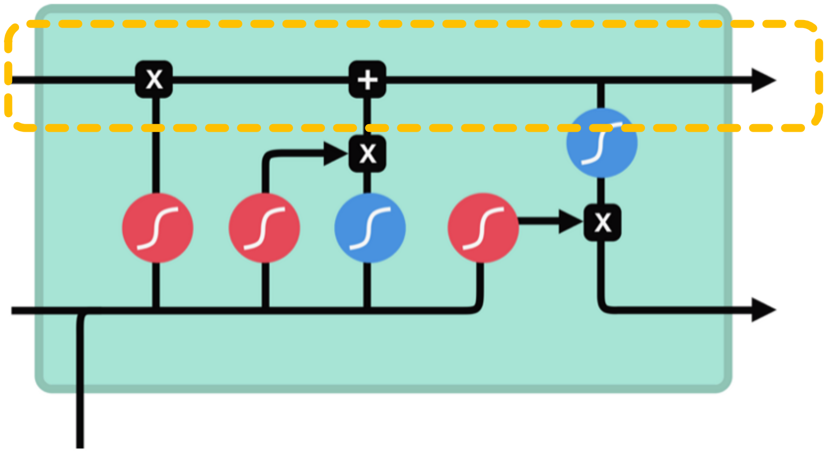


Image source: https://www.towardsdatascience.com

Long Short-Term Memory

Using this block, LSTM can ***update*** *the selected values of information in the cell state*



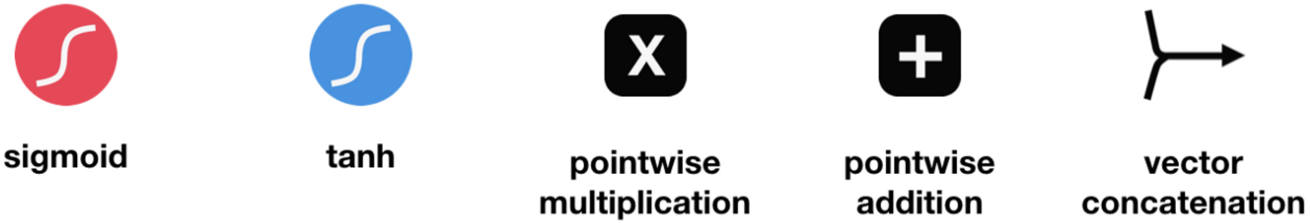
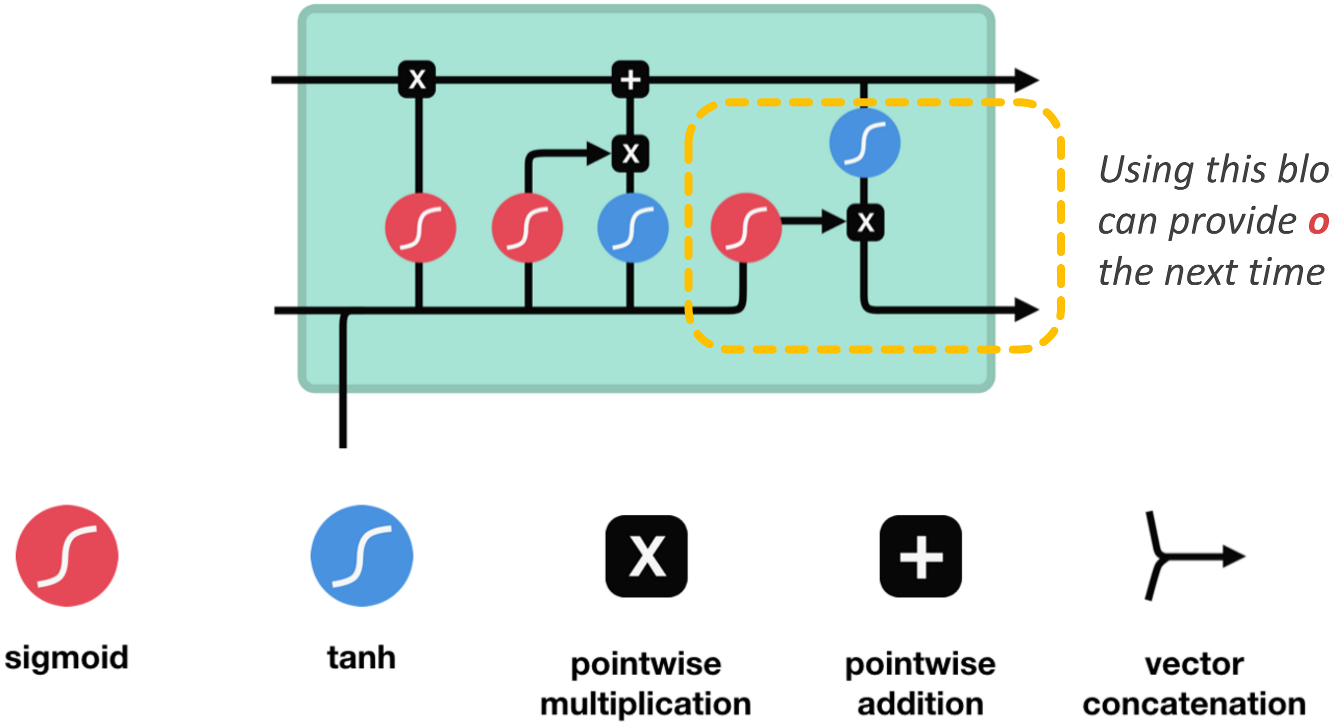


Image source: https://www.towardsdatascience.com

Long Short-Term Memory



Using this block, LSTM can provide ***output*** *for the next time step*

Image source: https://www.towardsdatascience.com

Long Short-Term Memory

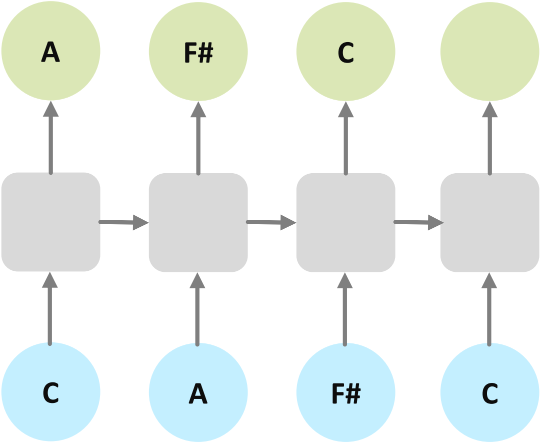
⚠ Important Notes on LSTMs

- They have a **separate Cell State** along with the output in each step
- They use **gates** for information flow management and control
- They can provide a Backpropagation through Time (BPTT) process with uninterrupted gradient flow
  - An improved training process and efficient updating of weights
- They can easily capture long-range dependencies
- In contrast with simple RNNs, LSTMs can grab information from the distant past to predict the current/future state:

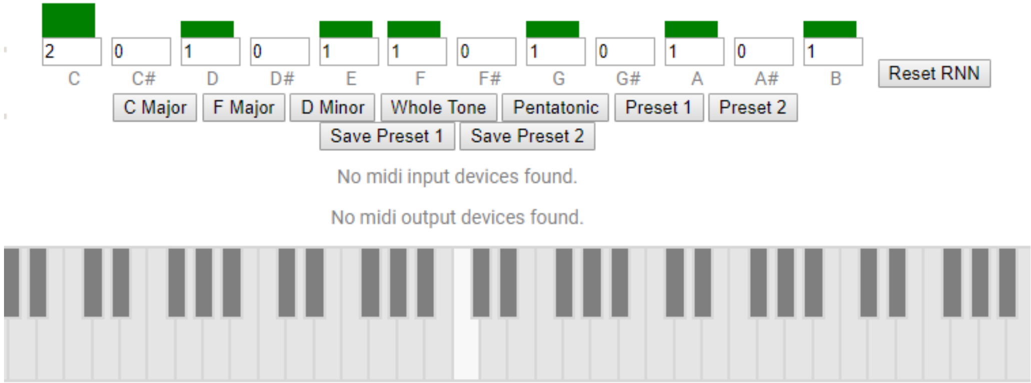
> Football is my favorite sport. I have seen many matches so far, and that is why I always dream to be a _________.

Long Short-Term Memory

Applications of LSTMs



Music Generation
(generating the next character in a given sheet)



No midi input devices found.

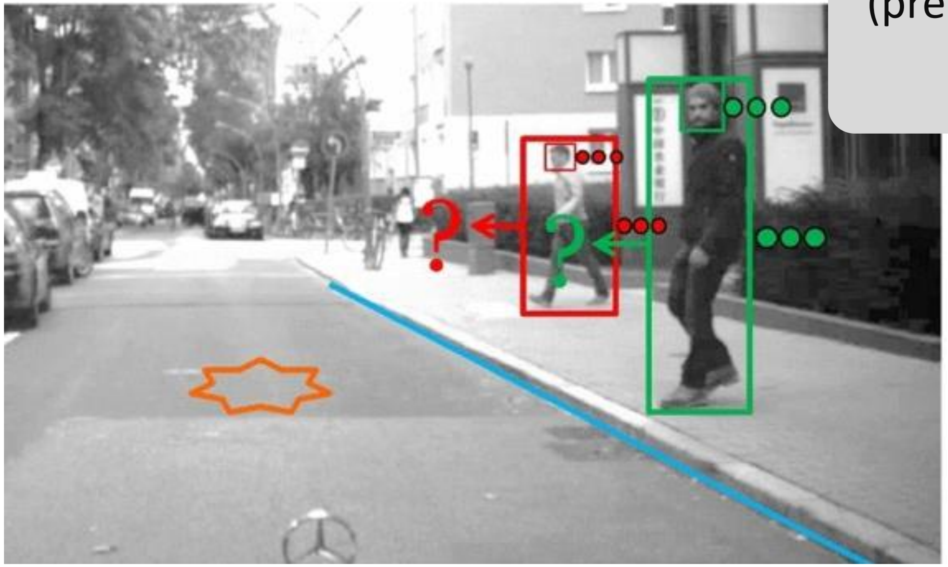
No midi output devices found.

Image source: https://www.deeplearn.js

Long Short-Term Memory

Applications of LSTMs

Trajectory Prediction
(predicting the next location of an object in the scene)



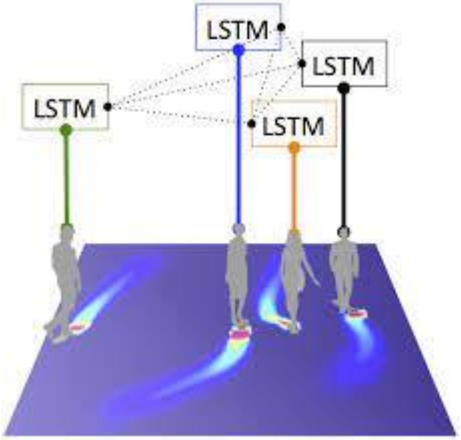
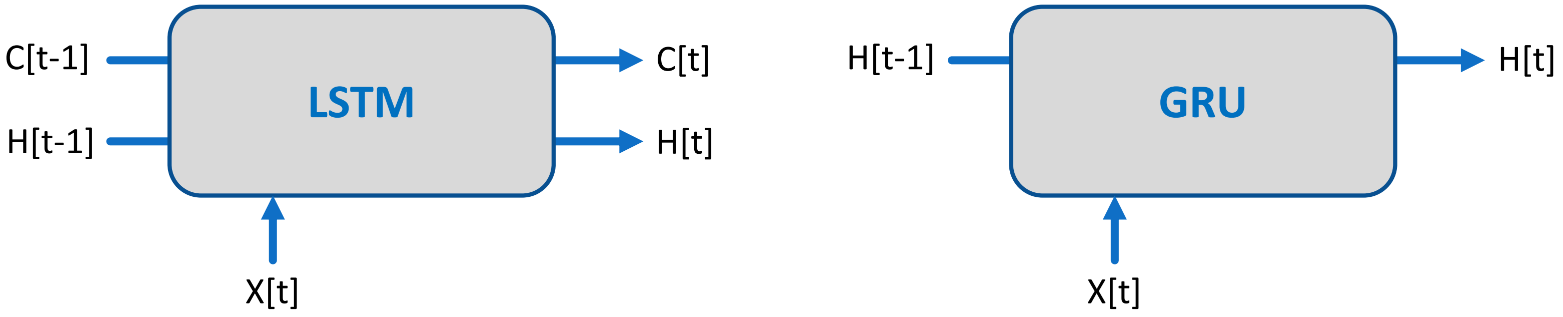


Image source: https://www.paperswithcode.com

Gated Recurrent Unit

- One of the most recent RNN approaches, *AKA* ***GRU***
- Very similar to LSTM, but with a much simpler architecture
  - GRUs do not contain a **Cell State**
- Equipped with two fundamental gates
  - A Reset Gate (short-term memory) and an Update Gate (long-term memory)



References

- http://www.IntroToDeepLearning.com
- https://towardsdatascience.com/sequence-models-and-recurrent-neural-networks-rnns-62cadeb4f1e1
- https://towardsdatascience.com/introduction-to-sequence-modeling-problems-665817b7e583
- https://www.bouvet.no/bouvet-deler/explaining-recurrent-neural-networks
- https://machinelearningmastery.com/how-to-fix-vanishing-gradients-using-the-rectified-linear-activation-function/

Questions?