Deep Learning

Recurrent Neural Networks (RNN)

Learning Objectives

By the end of this lesson, you will be able to:

- Analyze sequential data and use it in RNNs
- Examine different architectures of RNN
- Evaluate how to classify text using RNN
- Analyze the architecture of a CNN and an RNN hybrid model
- Classify videos using a hybrid RNN model

Business Scenario

ABC Corporation is a financial company that requires a predictive model to analyze financial market trends, analyze time series data, and forecast future outcomes.

To achieve this, the company employs a sequence modeling program that can handle and predict sequential data. They use a recurrent neural network to analyze financial data from the past and use it to forecast future market trends.

The company implements a hybrid model by combining the one-to-many architecture of the RNN with a convolutional neural network to predict a series of sequential events. The hybrid model enables ABC Corporation to make more accurate predictions and gain a competitive edge in the financial market.

Sequential Modeling

Discussion

Discussion: Sequential Modeling

- What is sequential modeling?
- What are some real-world applications of sequential modeling?

Sequential Modeling

Sequential modeling is a task or problem domain that involves predicting or modeling patterns in sequential data.

Different kinds of sequential data can be modeled, interpreted, and predicted by a sequential modeling tool, including:

Text

Audio

Video

Sequential Modeling: Advantages

The advantages of sequential modeling are listed below:

- It captures temporal dependencies in data.
- It can handle variable-length input.
- It is advantageous in time series prediction tasks.
- It is widely used in natural language processing.
- It is essential in speech and audio processing.
- It enables sequential decision-making in various domains.

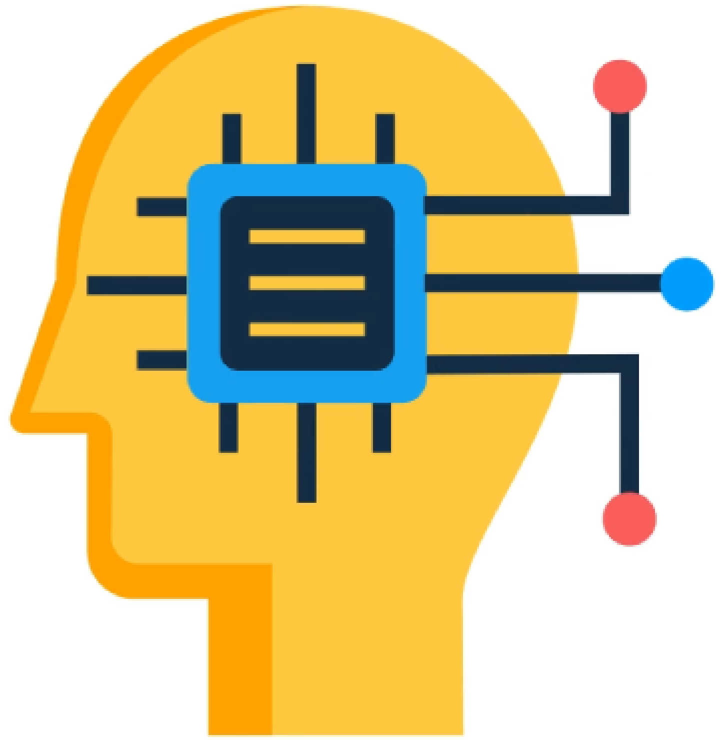
Sequential Modeling

The different types of sequential models are:

- RNNs
- Autoencoders
- Seq2Seq

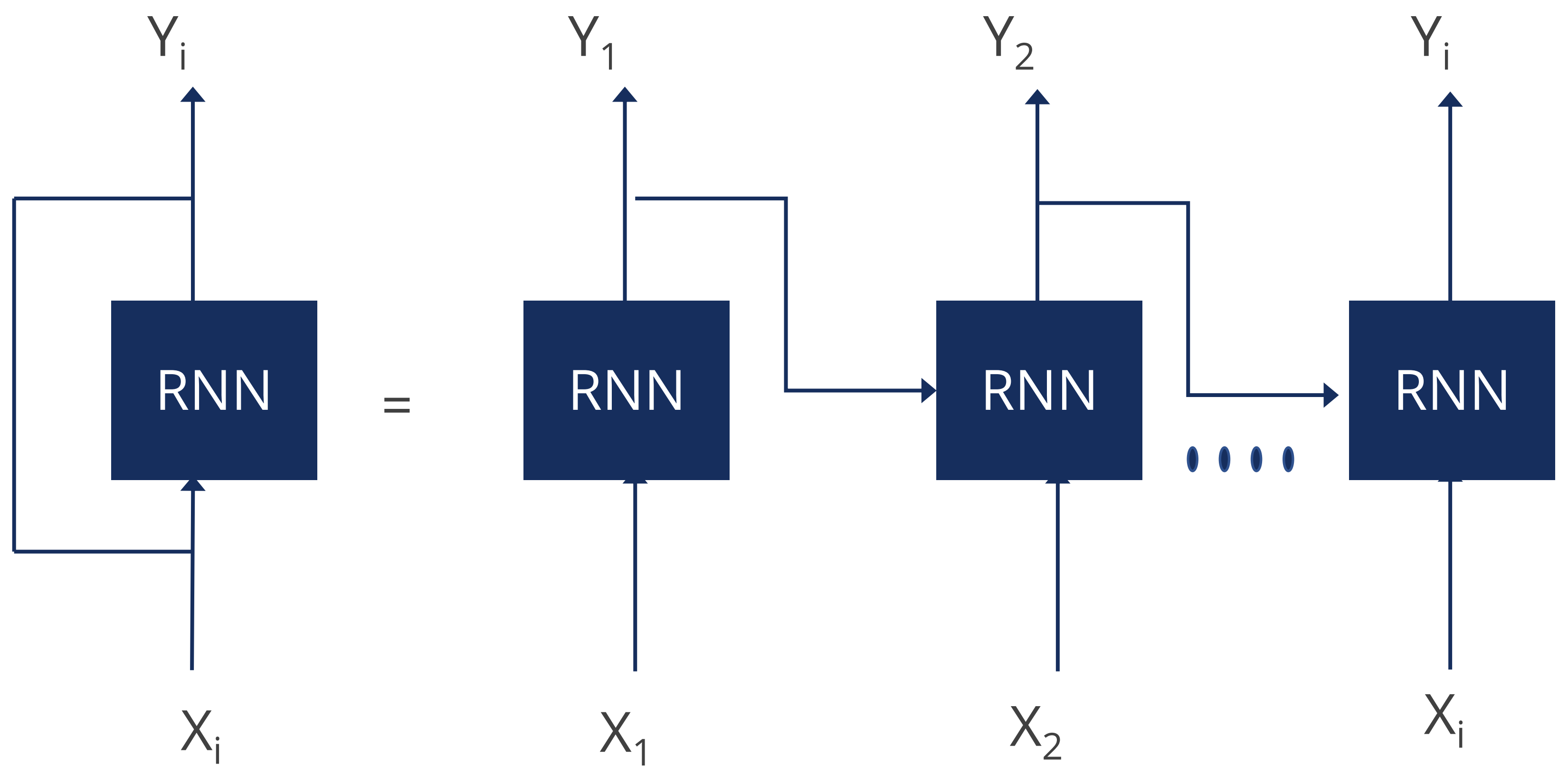
Sequential Modeling

To achieve memory-based learning, a unique artificial intelligence algorithm known as recurrent neural network (RNN) is used.



Sequential Modeling: Working

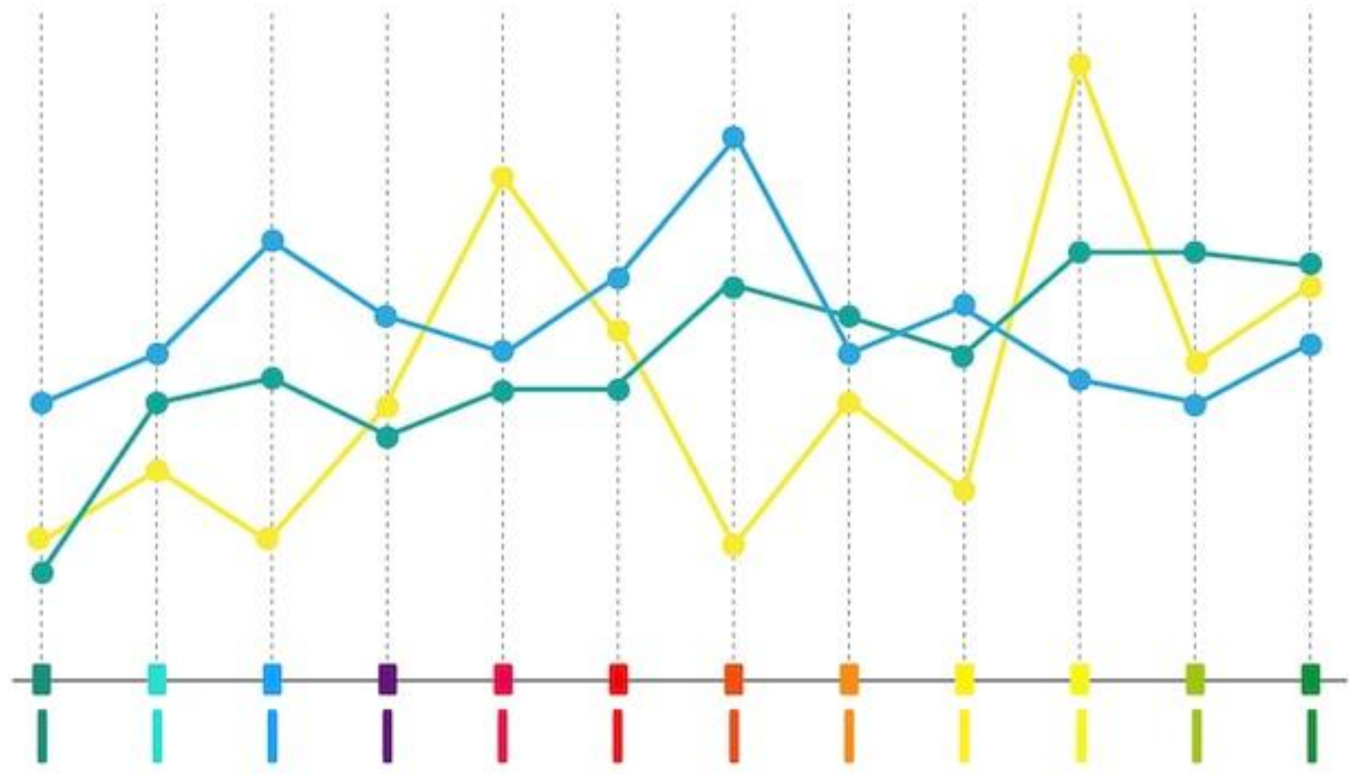
The following image shows the working of sequence modeling.



It is used to generate a sequence of values by analyzing a series of input values.

Sequential Modeling: Working

The input values may be a time-series data where variables vary over time.



It analyzes the recurring input values and sees them as discrete inputs for different problems or cases.

Real-World Applications

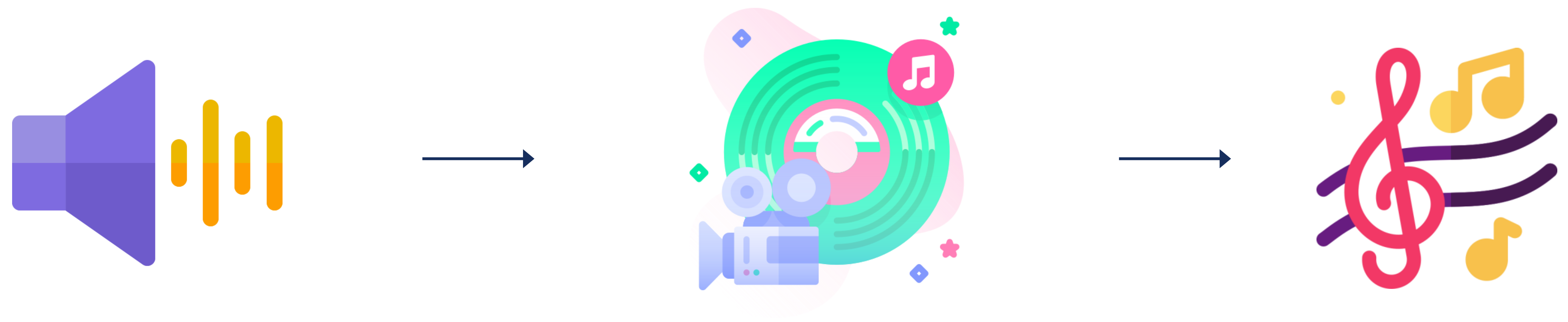
Sequence modeling can be used in natural language processing to process texts into valuable insights.

Example

An email spam classifier improves user experience by detecting unwanted or dangerous emails and ensures that the inbox stays tidy.

Real-World Applications

Another use-case could be a computer program that:



Takes an audio file as input

Recognizes its possible genre

Suggests a sequence of musical notes to complete the song

This could aid musical composition, as several musicians face problems in decrypting symbols early on in their careers.

Discussion: Sequential Modeling

- What is sequential modeling?

Sequential modeling in deep learning refers to the process of building and training models that can process sequential data, such as time series or text, sequentially.

- What are some real-world applications of sequential modeling?

It is used in speech recognition, machine learning, sentiment analysis, text generation, and so on.

Introduction to Recurrent Neural Networks

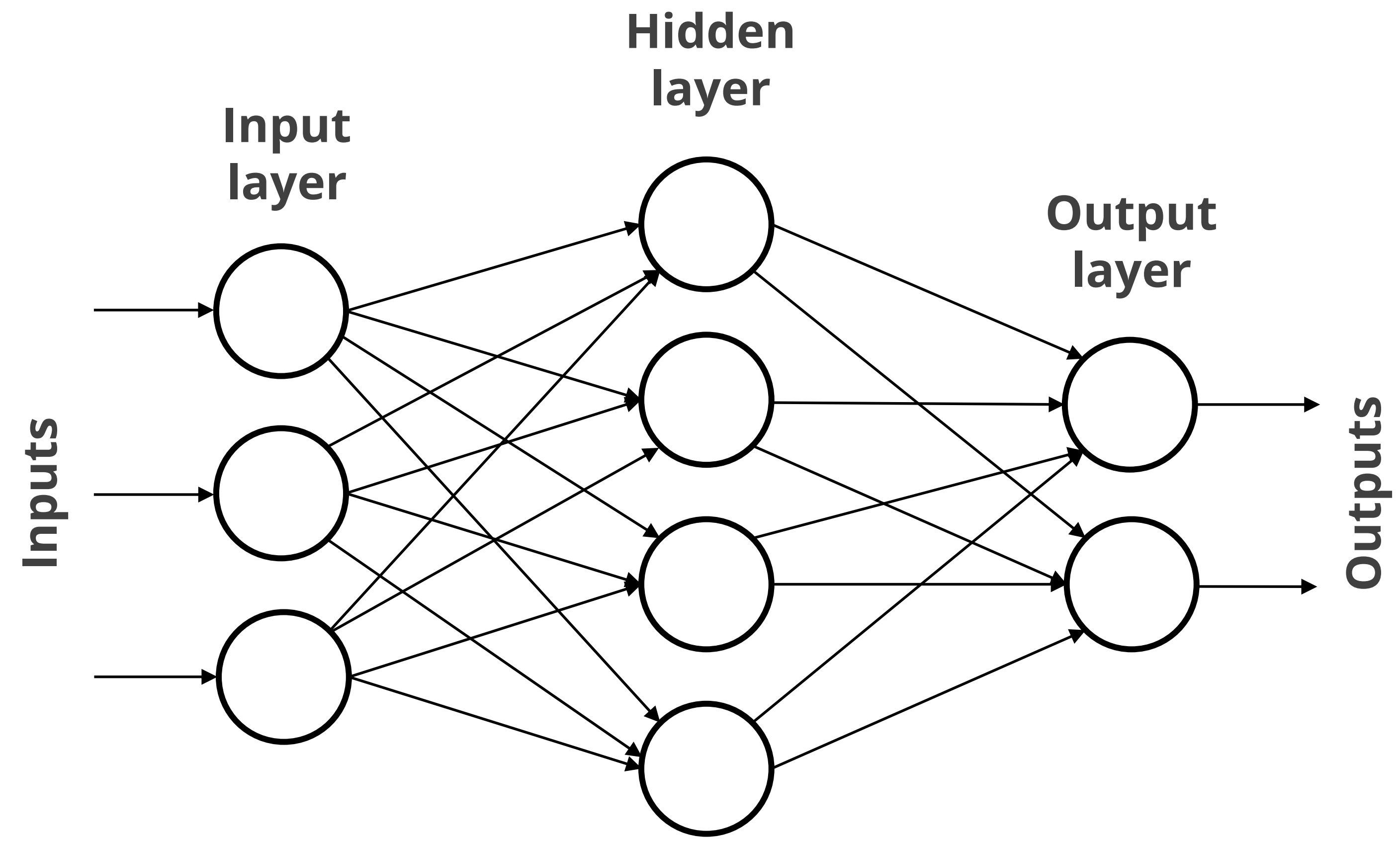
Discussion

Discussion: RNNs

- What are RNNs?
- What are the different types of RNNs?

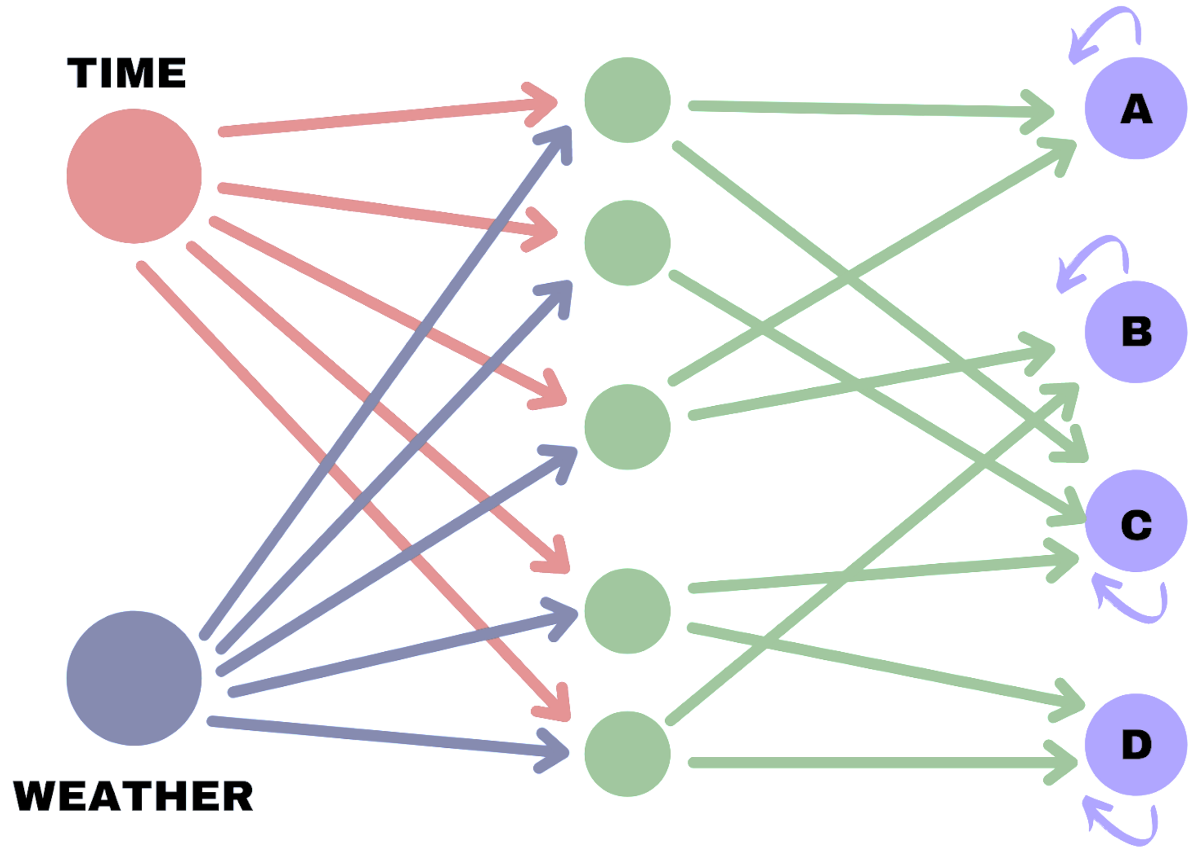
Feedforward Neural Networks in RNN

RNNs were introduced after feed-forward neural networks (FNNs) to address the limitation of processing sequential data.



Example: FNN in RNN

Consider the following image and create a computer program that suggests a sport to be played based on the time and the weather:



It follows a particular pattern and makes appropriate decisions.

Example: FNN in RNN

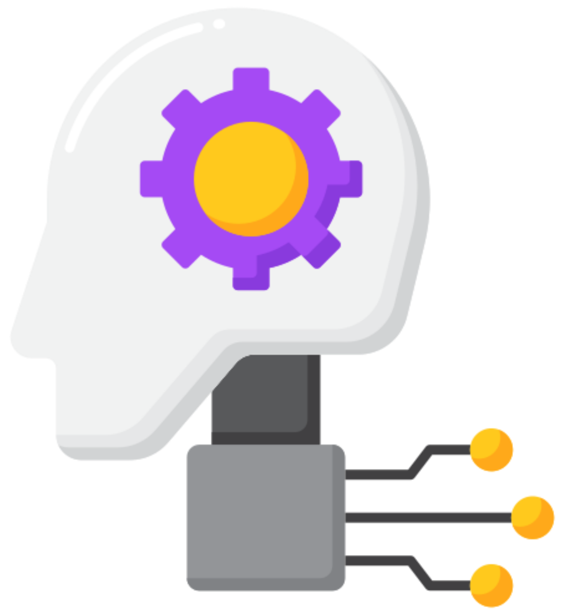
It might suggest football in the evening in clear weather or badminton when there is no wind.

An FNN would suggest the same sport for similar weather conditions every day.

An RNN would remember the user's choices from the previous day and suggest something different the next day.

Recurrent Neural Networks

A recurrent neural network (RNN) is a state-of-the-art algorithm used by Apple's Siri and Google's voice search.



It is the first algorithm that remembers its input owing to internal memory.

The ability to add the recent past to the present allows it to make precise predictions.

It is ideal for machine learning problems that involve sequential data, like natural language processing and time series forecasting.

Recurrent Neural Networks

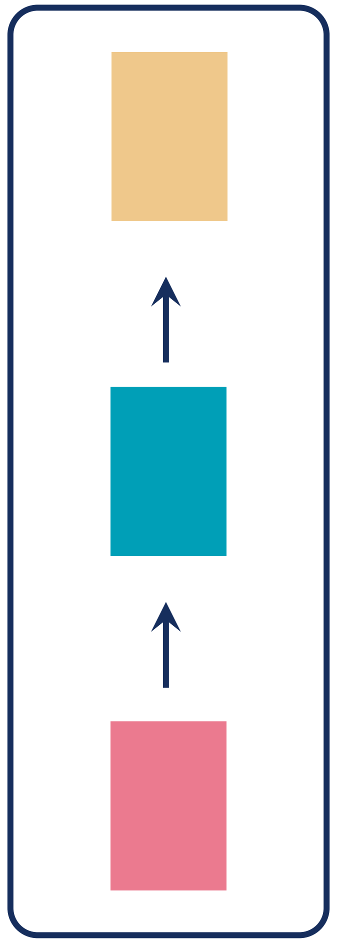
RNNs are mainly used for:

Sequence Classification	Sequence Labeling
Sentiment classification	Part of speech tagging
Video classification	Named entity recognition

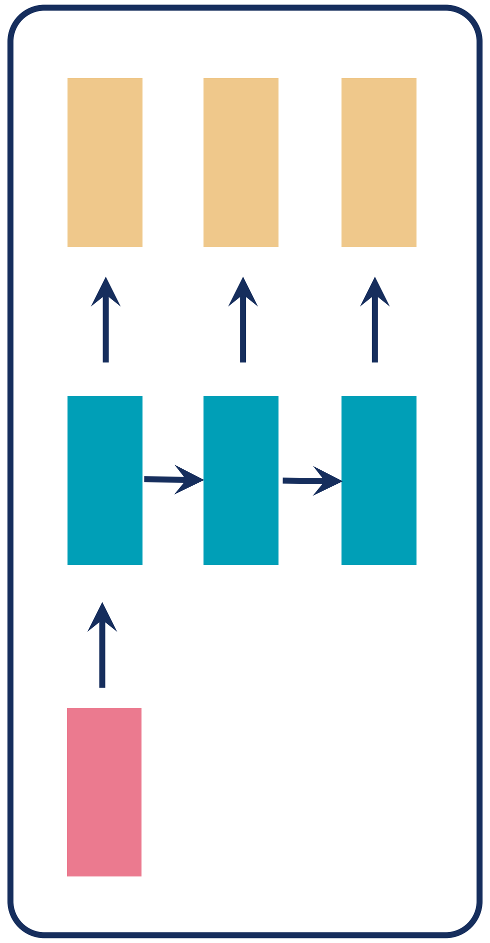
Recurrent Neural Networks

The different types of RNNs are:

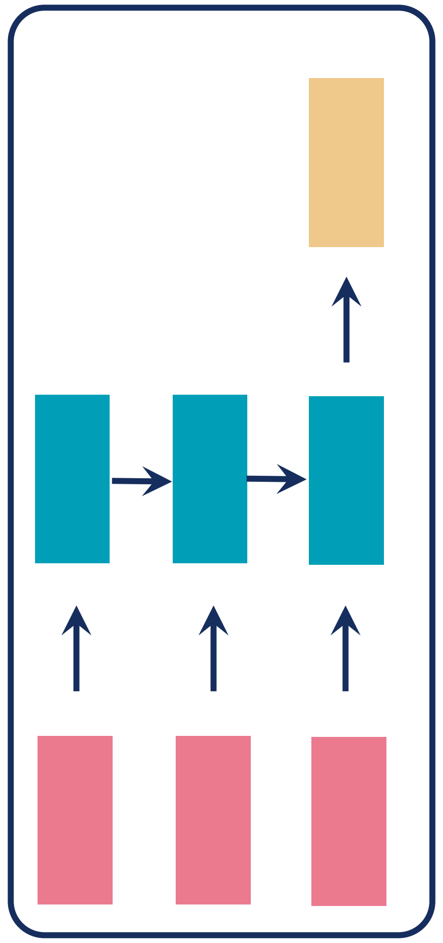
One-to-one



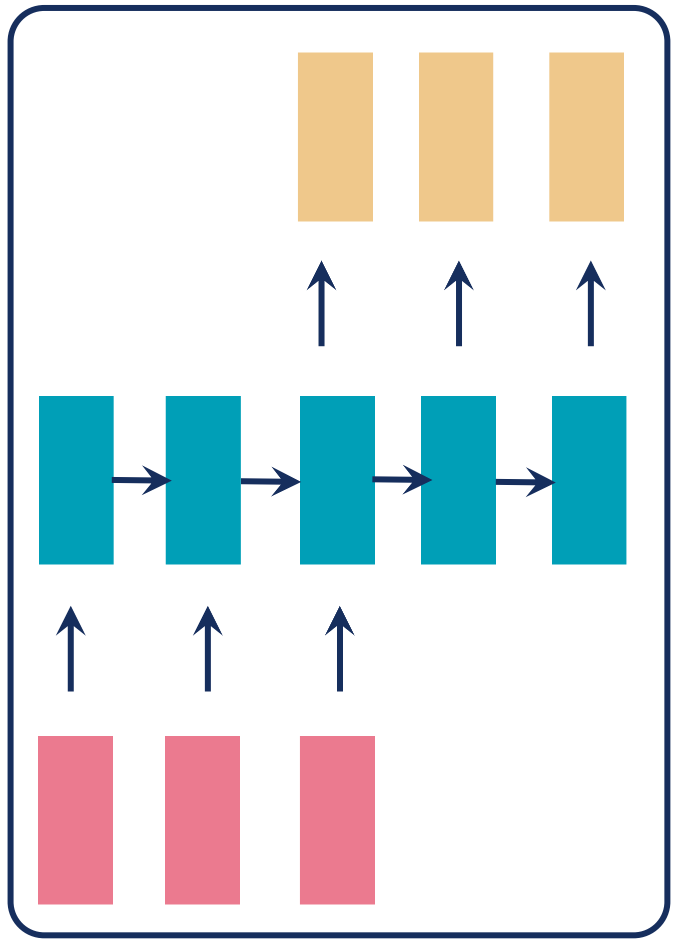
One-to-many



Many-to-one

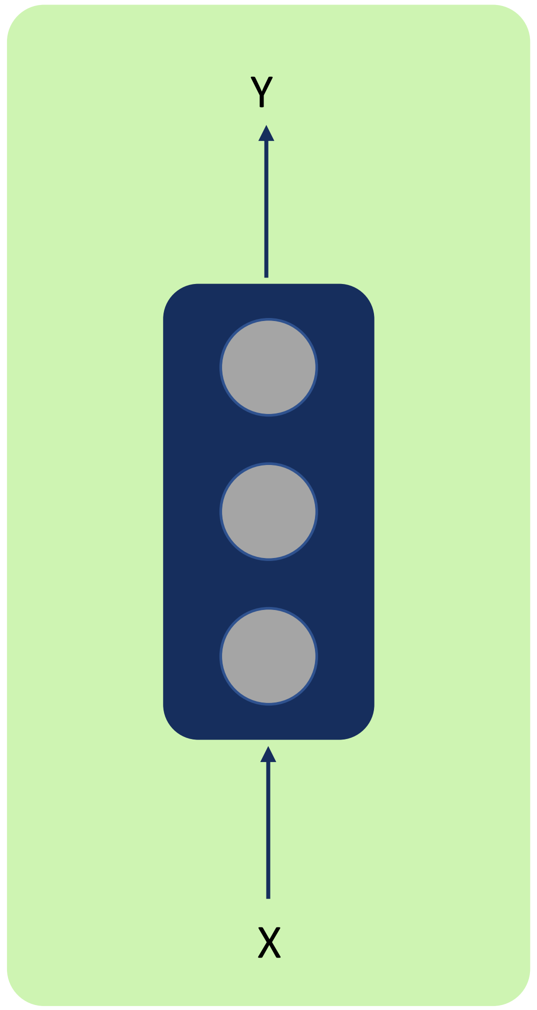


Many-to-many



One-to-One

One-to-one architecture forms the basis of feedforward neural networks.



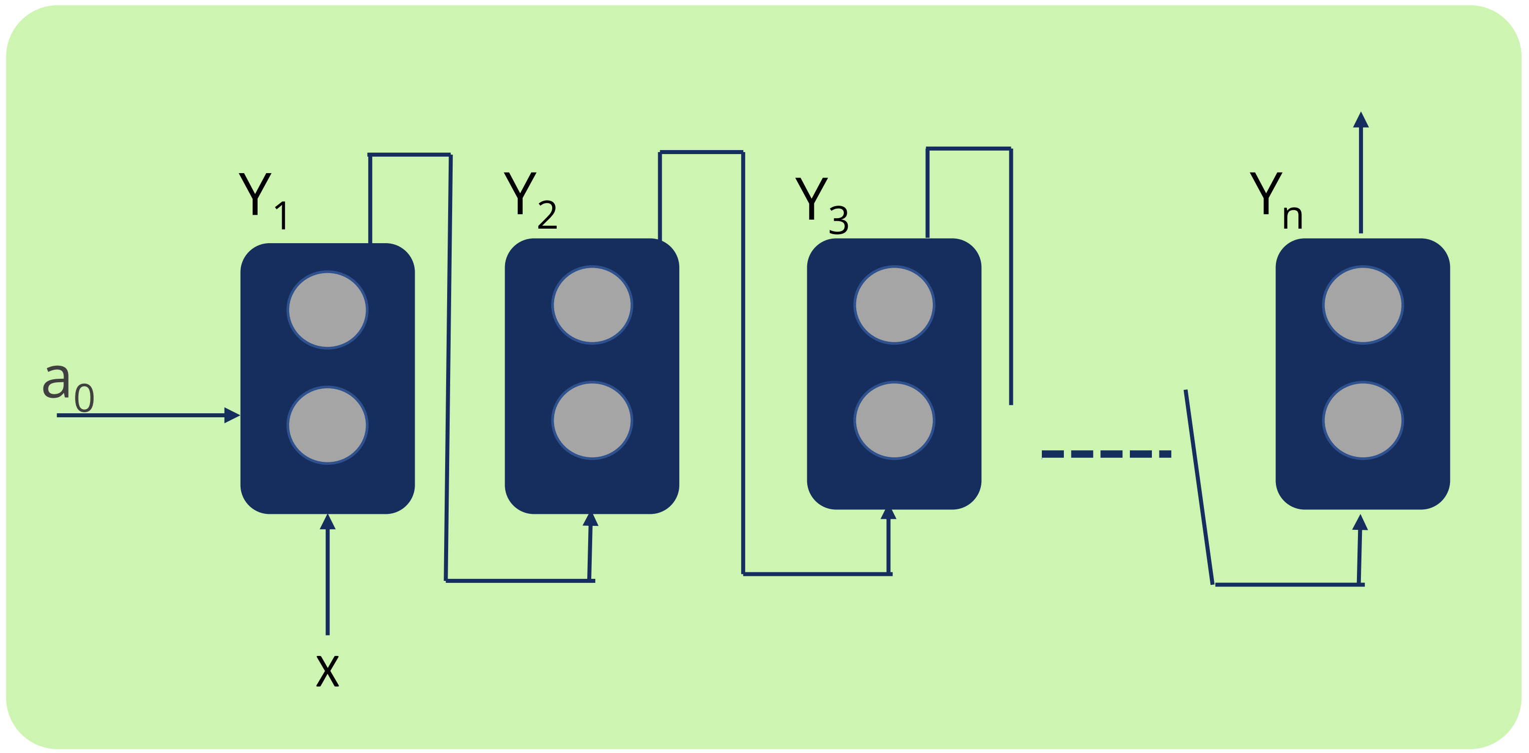
Example

An architecture that is used to classify the image of a musical symbol

Activation values are not needed as this is a straightforward scenario of a single input and a single output as seen in the figure.

One-to-Many

A single input drives the entire network.



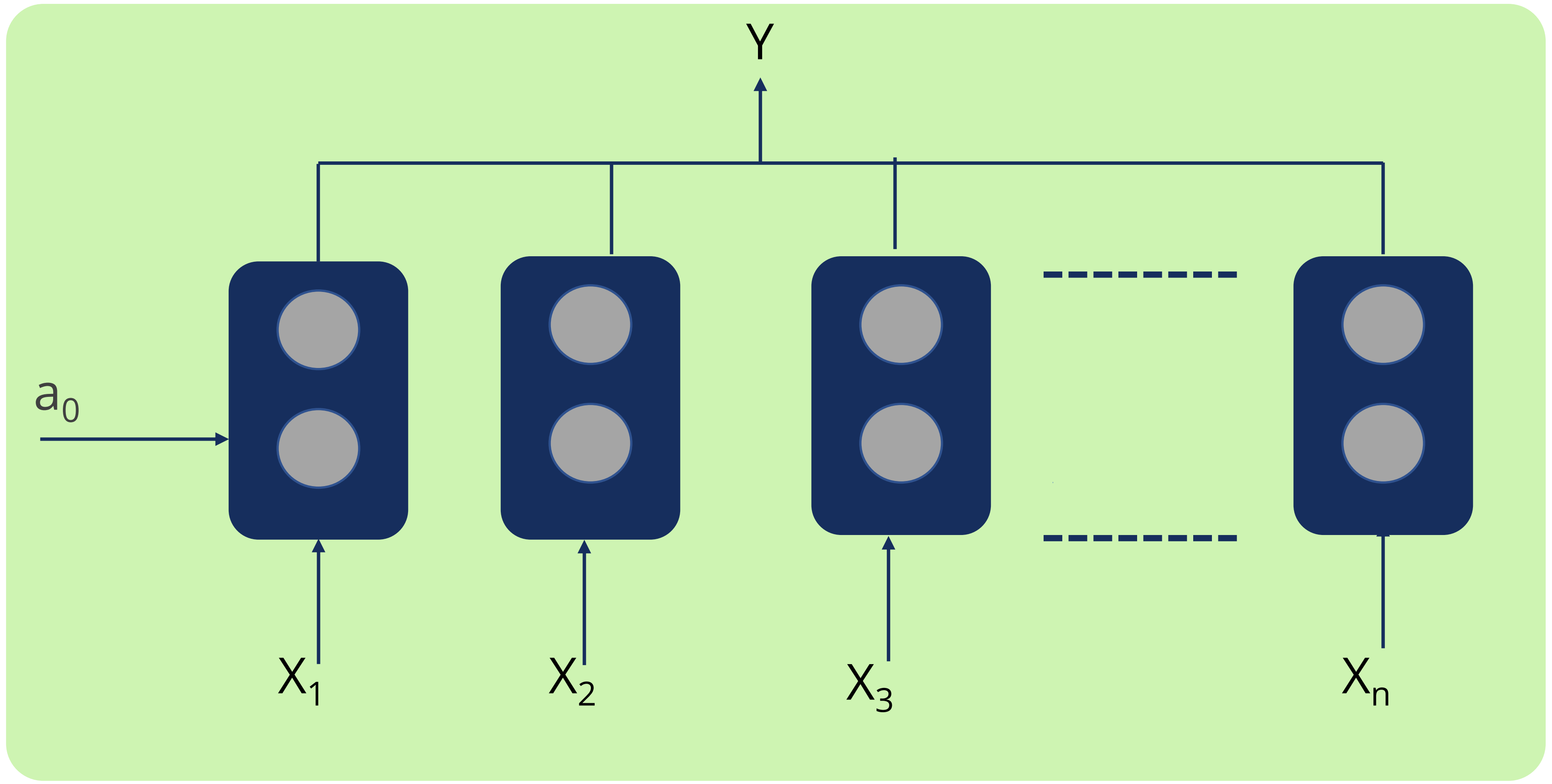
Example

Generating a sequence of musical notes based on the first note or the genre of music given as input

Once the input is given, The network produces a series of outputs, one for each time step.

Many-to-One

A sequence of multiple inputs is given to the network, which is instructed to predict a single output.

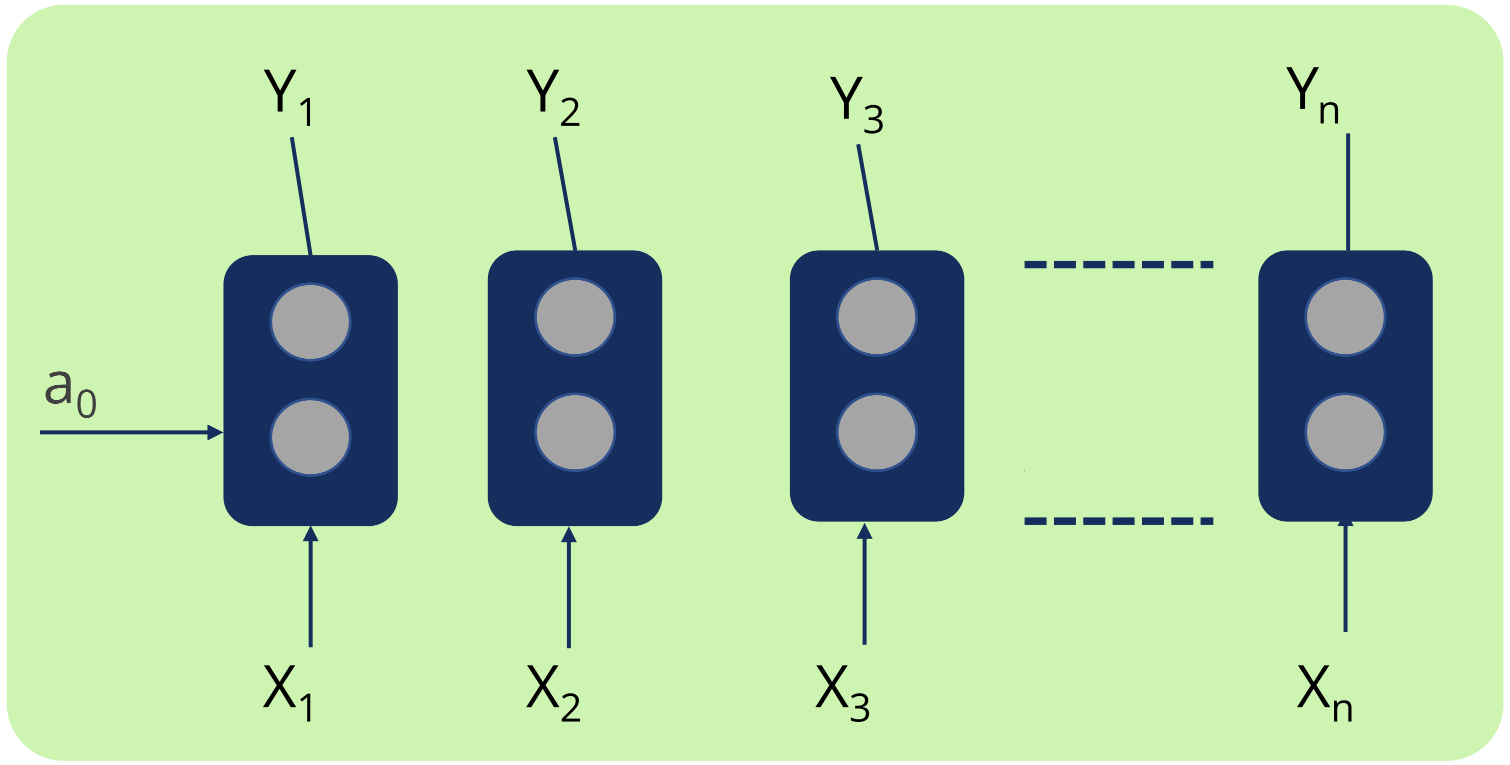


Example

The task of classifying the genre of music by giving a song as input

Many-to-Many

A sequence of multiple inputs is fed to the network to predict a sequence of outputs that may or may not be of the same length.



Example

An application for generating a string of musical symbols by giving the musical notes or chords as input

Such neural networks have two components: the encoder and the decoder.

Recurrent Neural Networks: Advantages

Some of the many advantages that RNN brings are:

The weights can be shared across time steps.

The model can process inputs of any length.

The model size isn't affected by the input size.

The model is designed to remember each piece of information and is helpful in any time-series predictor.

Recurrent Neural Networks: Disadvantages

Some of the disadvantages of RNN are:

Difficulty in training RNN models

Computation slow due to its recurrent nature

Issues with exploding and vanishing gradients

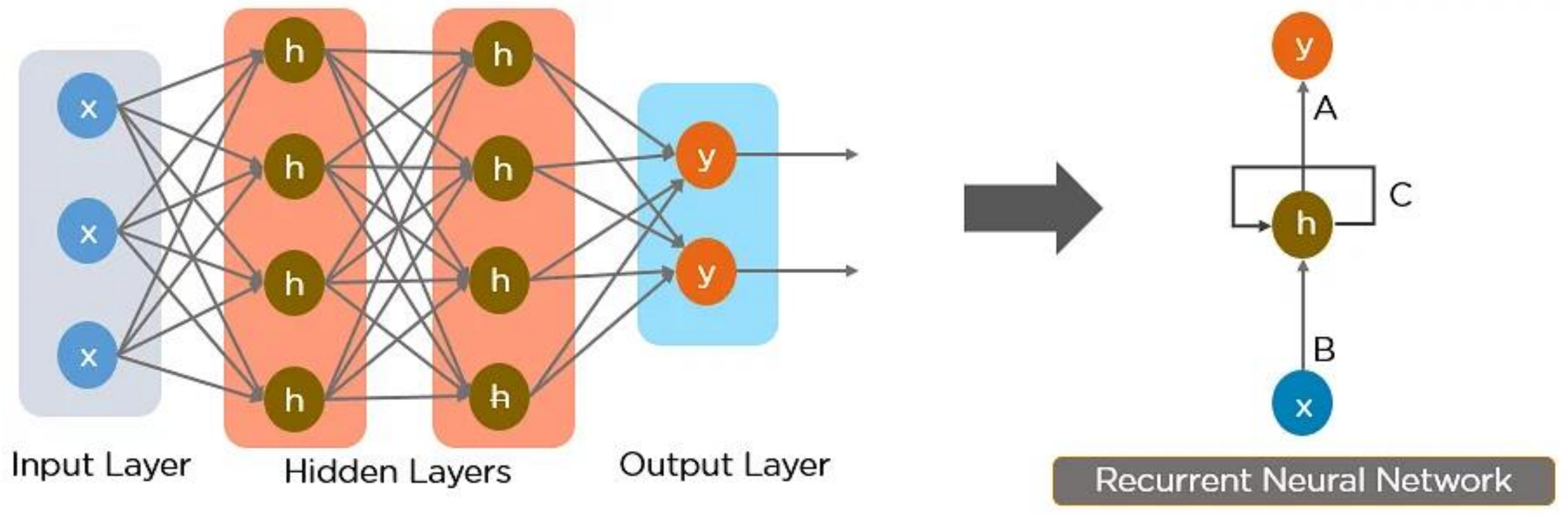
Discussion: RNNs

- What are RNNs?
  Recurrent Neural Networks (RNNs) are a type of artificial neural network designed to handle sequential data. Unlike traditional feedforward neural networks, RNNs have a recurrent structure that allows them to maintain an internal memory of previous inputs, enabling them to capture temporal dependencies in the data.
- What are the advantages of using RNNs?
  - RNNs excel at sequential modeling, capturing dependencies and patterns in sequences.
  - RNNs maintain the memory of previous inputs, enabling contextual understanding.
  - RNNs offer flexibility in handling variable-length inputs and outputs.

Architecture and Working of RNN

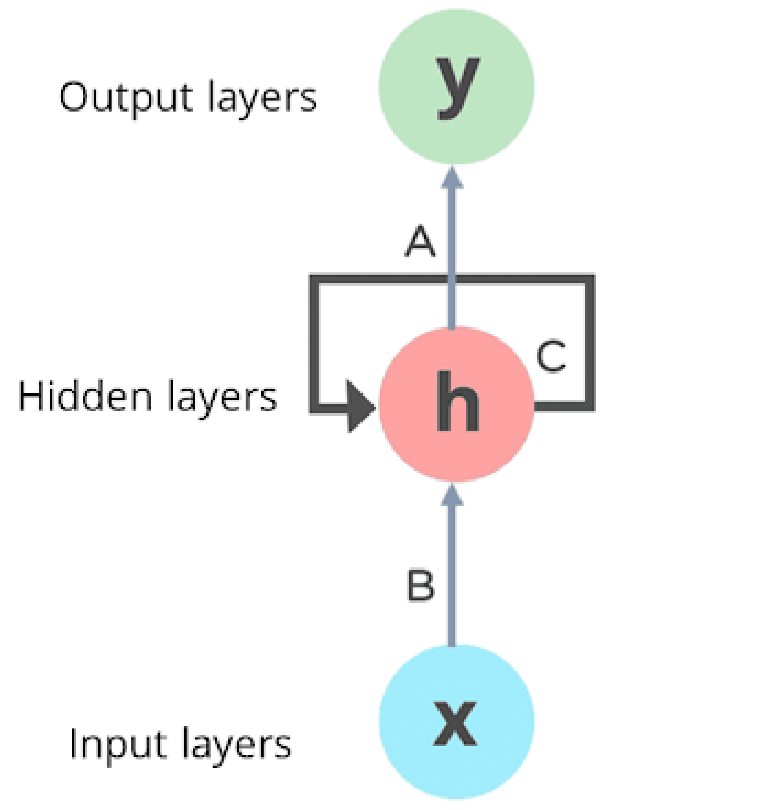
Architecture of RNN

The recurrent neural network (RNN) architecture is designed to process sequential data by capturing temporal dependencies and retaining the memory of previous inputs.



Architecture of RNN

It consists of three main components an input layer, a hidden layer with recurrent connections, and an output layer.

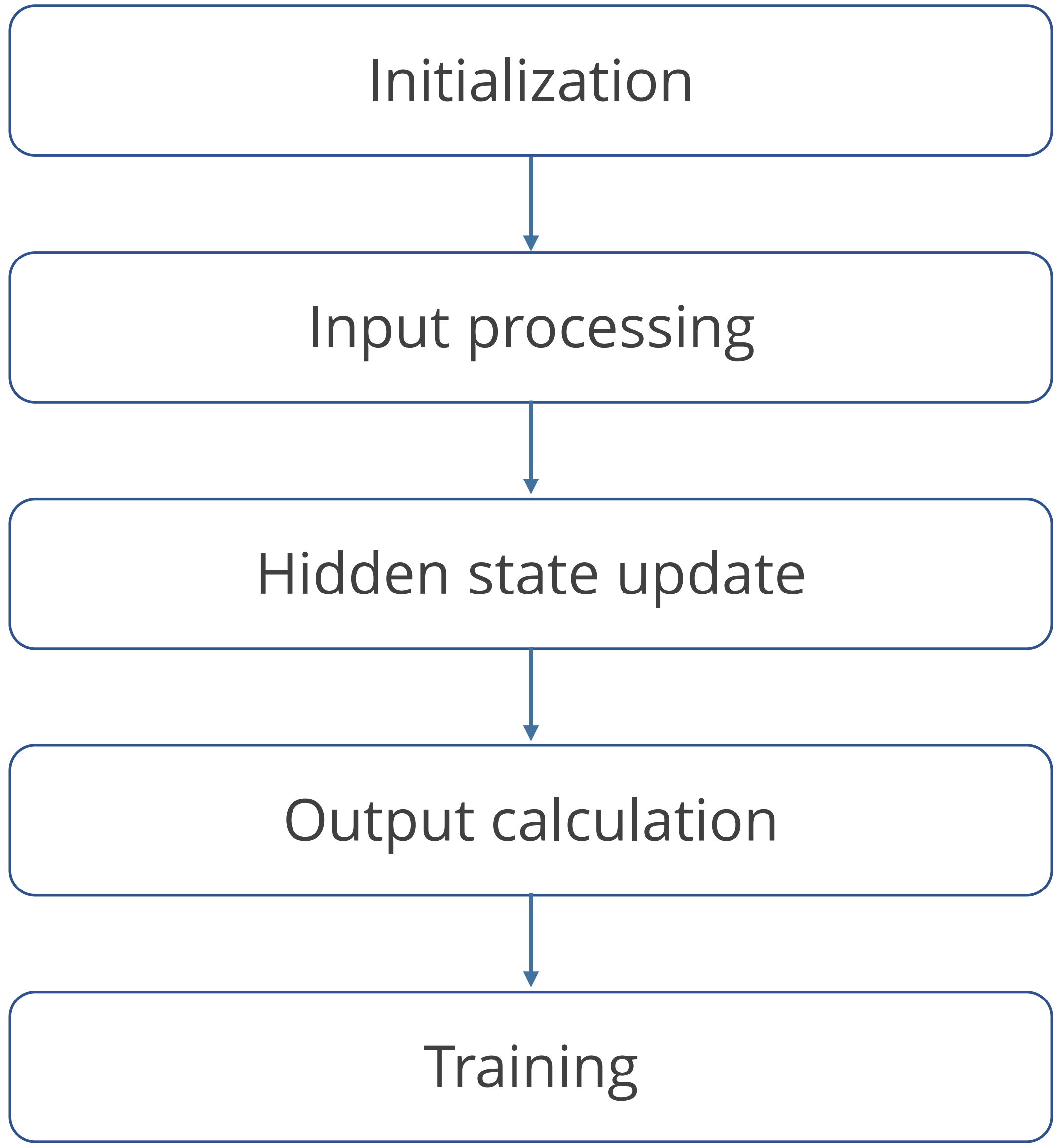


A, B and C are the parameters

- x is an input layer; h is the hidden layer, and y is the output layer.
- Parameters A, B, and C are utilized to enhance the model's output.
- The output at each time step is fed back into the network to improve the subsequent outputs.

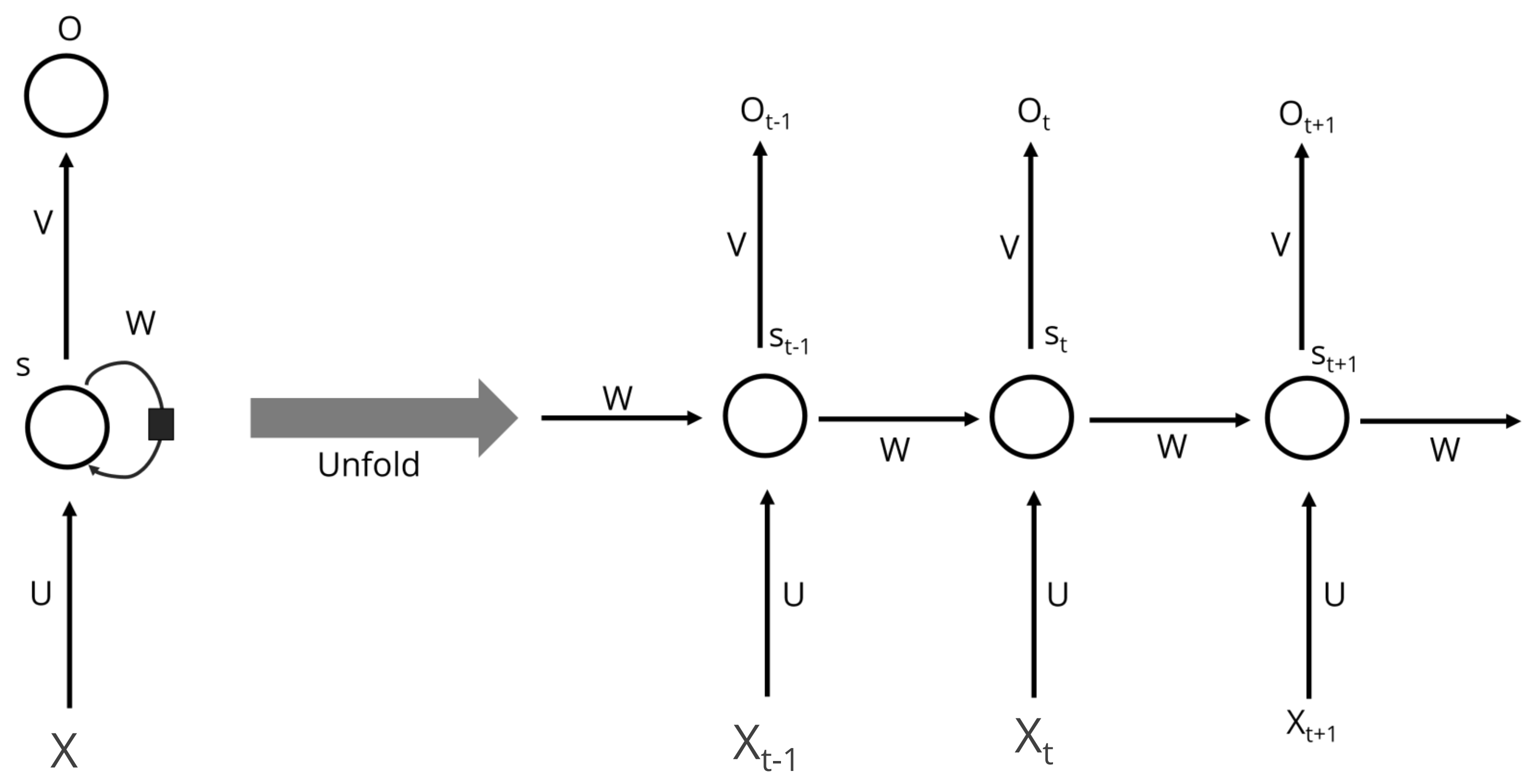
Working of RNN

The working of a recurrent neural network (RNN) can be described in a step-wise manner as follows:



Initialization

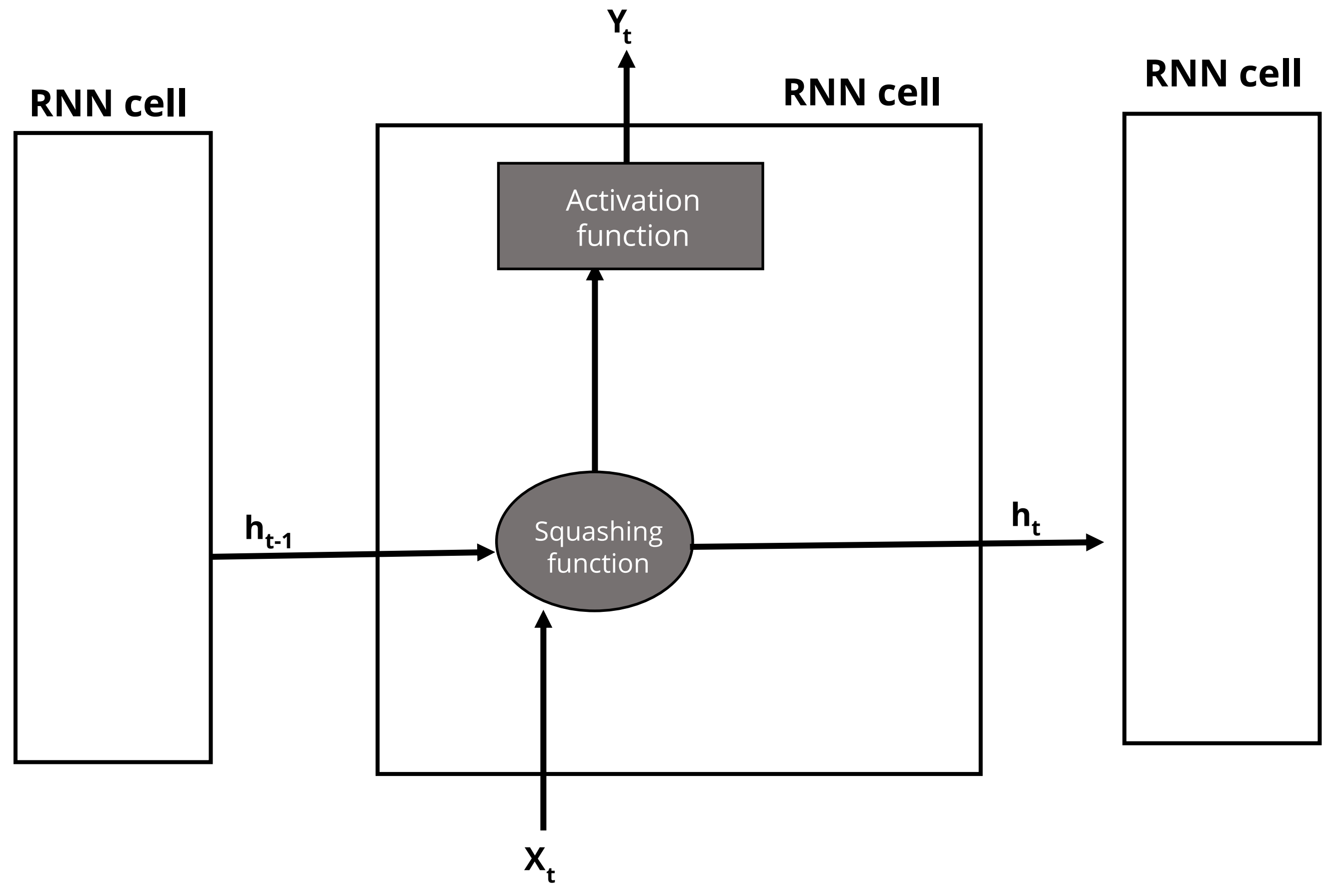
It begins with initialization, where the hidden state is set to a vector of zeros or random values, and the RNN architecture is defined, including the number of hidden units, and activation functions.



In the above diagram, X is the input state, s is the hidden state, and O is the output. The network's weights are U, V, and W.

Input Processing

During input processing, the input sequence is processed one element at a time.



X_t is the input state,
h_t is the current state,
and h_{t-1} denotes the previous state.

The current input is combined with the previous hidden state to capture the context and dependencies, and the activation of the hidden layer is computed based on this combined input.

Input Processing

Use the following equation to combine the current input with the previous hidden state:

$$a_t = (W_{xh} * x_t + W_{hh} * h_{t-1}) + b_h$$

Where,

W_{xh} represents the weight matrix for input-to-hidden connections,

W_{hh} represents the weight matrix for hidden-to-hidden connections

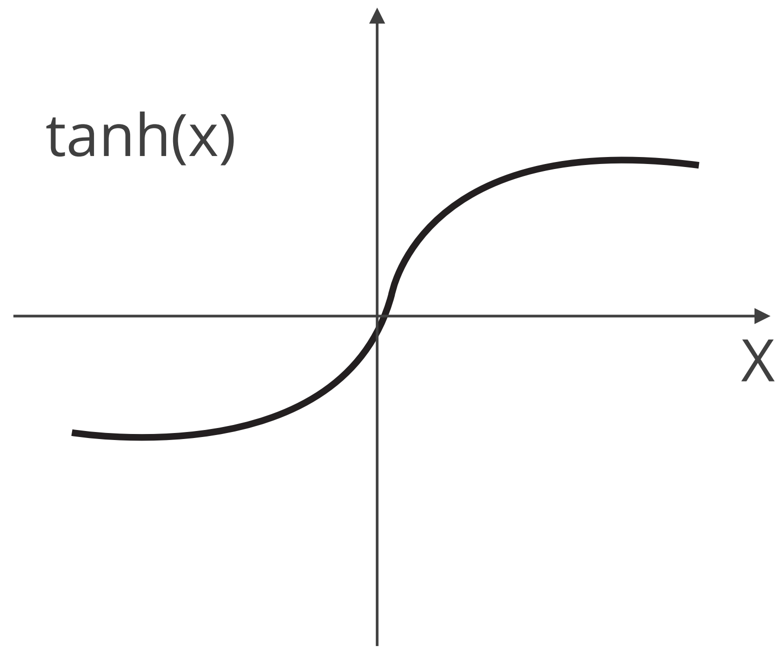
b_h represents the bias term for the hidden layer.

Hidden State Update

The hidden state is then updated using the computed activation of the hidden layer, allowing information to flow across different time steps.

Compute the activation of the hidden unit using a hyperbolic tangent activation function.

Tanh



$$h_t = \tan h(W_{hh}\, h_{t-1} + W_{Xh}\, X_t)$$

h_t= Hidden state
W_{hh}= Recurrent neuron weights
W_{Xh}= Input neuron weights

Output Calculation

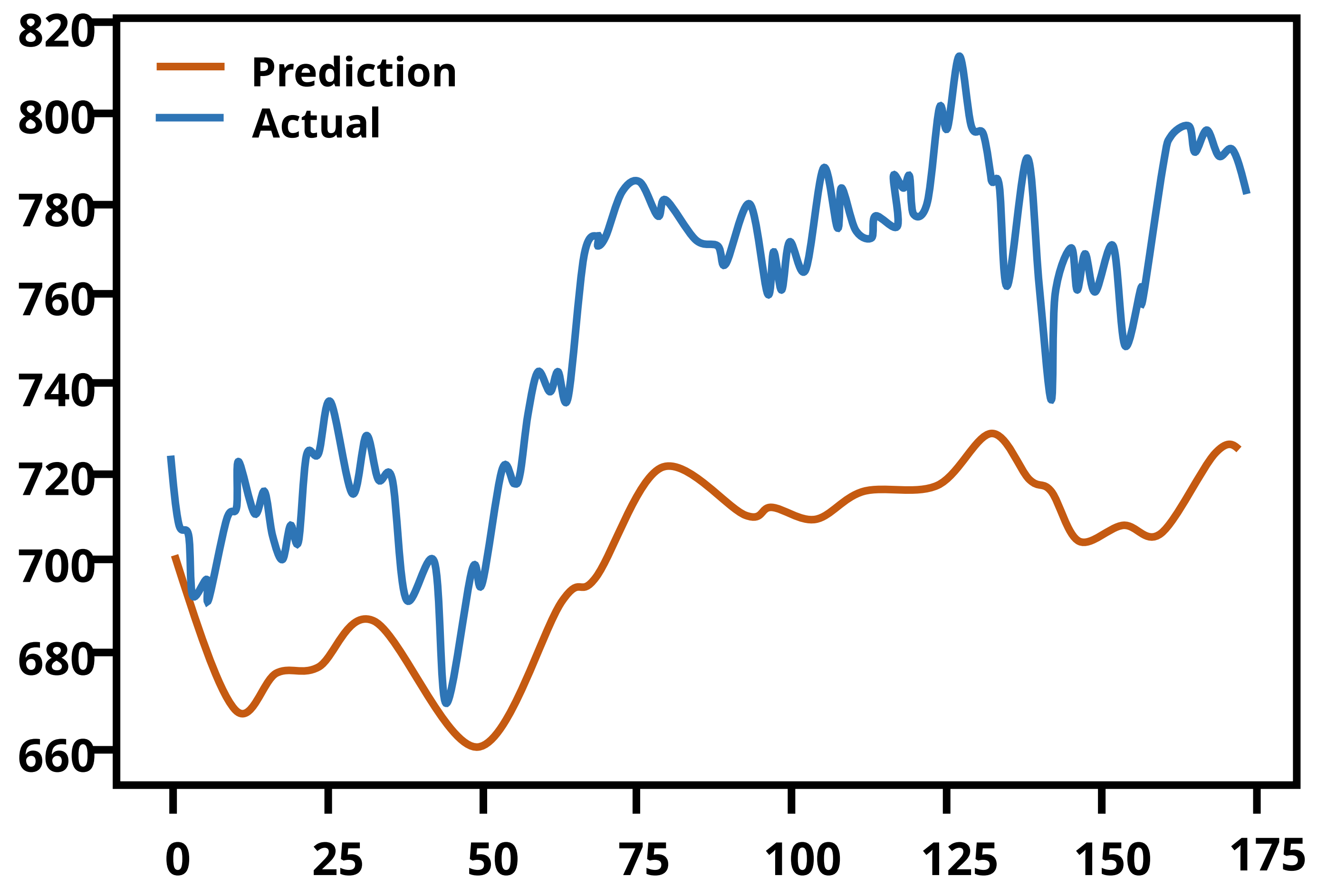
The output of the network is calculated based on the updated hidden state, and any necessary transformations or activation functions are applied to obtain the desired output format.

$$Y_t = W_{hy}\, h_t$$

Here, Y_t denotes the output and W_{hy} denotes the output layer weights.

Training

During training, the RNN calculates the loss between the predicted output and the target output, and the weights and biases of the RNN are updated using backpropagation through time (BPTT) and an optimization algorithm.



This process helps the RNN learn to make more accurate predictions and improve its performance.

Forward and Backward Propagation in RNN

Forward Propagation in RNN

The current song needs to be obtained based on two inputs:

Previous song

Time of the day

Once the inputs are passed through the network, they are multiplied by certain **weights** that convert them into a particular format, using which each neuron calculates its activation function.

Forward Propagation in RNN

The activated neurons pass value to the hidden layers until the output is obtained.

At time $T = T_i$,

$$O_i = F[\{(X_i * w) + (O_{i-1} * w'')\}]$$

O_i is the current output

X_i is the current input

w is the input weight

w” is the output weight

Forward Propagation in RNN

The final iteration gives Y’, which is the output obtained.

Suppose after passing (A, 18:00) as input, the network gives (A) as the output.

Backward propagation helps rectify this error.

Backward Propagation in RNN

With the predicted output (Y’) and the actual output (y), the loss function can be calculated as:

$$L = Y' - y$$

To minimize the loss function, gradient descent is applied by calculating the derivative of L with respect to the weights, using the chain rule.

Backward Propagation in RNN

Consider the following example of backward propagation in an RNN:

$$d(L)/d(w'') = \{d(L)/d(Y')\} * \{d(Y')/d(O_n)\} * \{d(O_n)/d(w')\} \ldots\ldots \{d(w')/d(w'')\}$$

Here,

d(L)/d(w”): Derivative of the loss function with respect to the weights

Y’: Predicted output

O_n: Output layer

w’ and w”: Weights in the model

Backward Propagation in RNN

The value of w” is reassigned as:

$$w'' = w'' - \eta * d(L)/d(w'')$$

Here, η is the learning rate.

The iterative process goes on for several epochs until the desired accuracy is achieved and all the weights have been reassigned.

Forward and Backward Propagation in RNN

Assume that Karen likes to listen to two songs, A and B, and listens to the songs in the sequence ABABABA and so on.

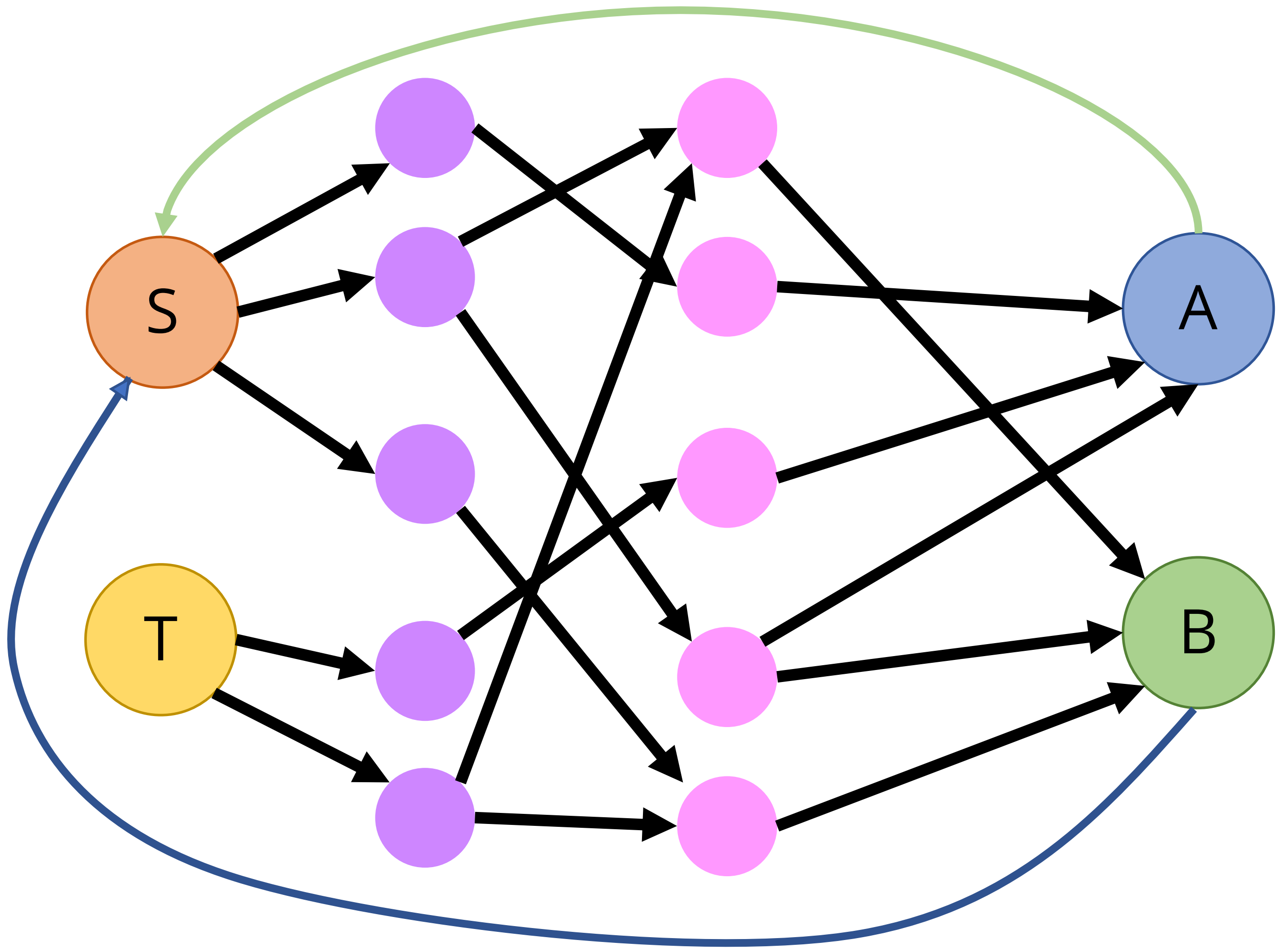
The decision-making algorithm works on the following conditions:

If it is switched on before noon, it repeats the previous song.

If it is switched on after noon, it changes the song.

Forward and Backward Propagation in RNN

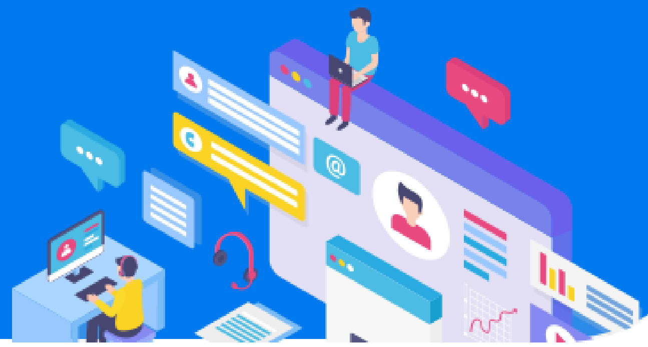
The following image shows how RNN drives the music engine.



T is the time of the day

S is the song

Assisted Practices



Let’s understand the concept of text classification with RNN using Jupyter Notebooks.

- 10.06.01_Text Classification Using RNN

Note: Please refer to the Reference Material section to download the notebook files corresponding to each mentioned topic

Long Short-Term Memory (LSTM)

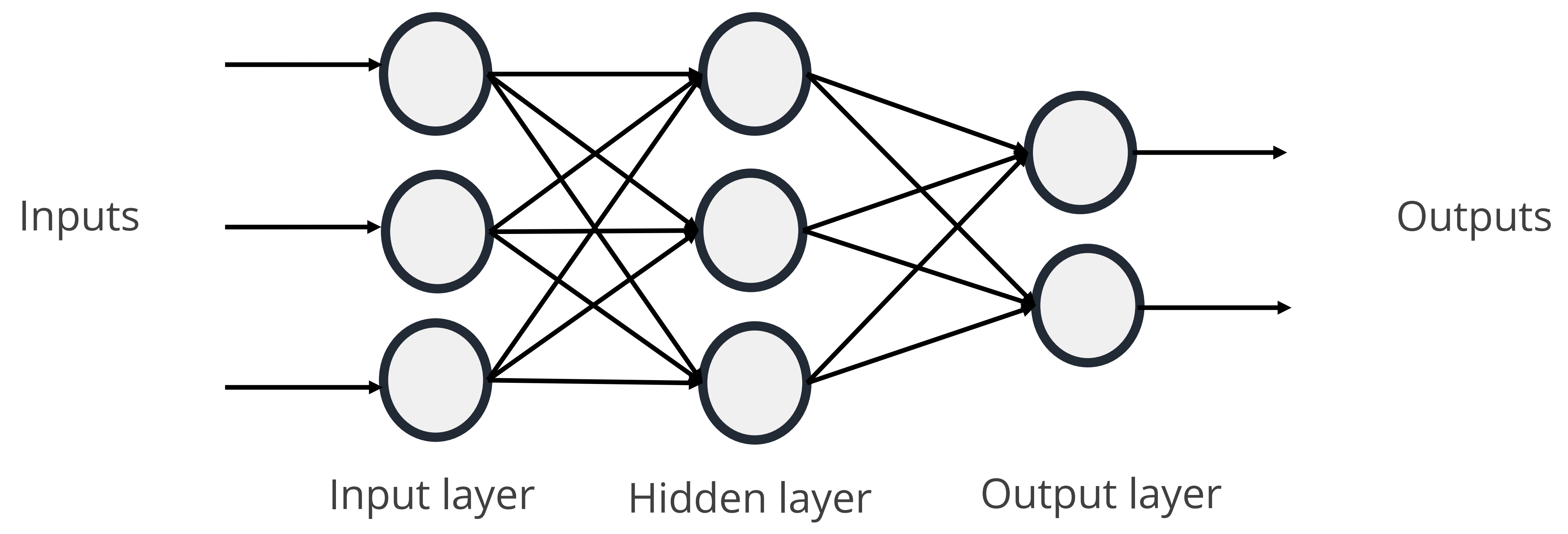
Discussion

Discussion: LSTM and CRNNs

- What is LSTM?
- What are some of the applications of CRNNs?

Long Short-Term Memory (LSTM)

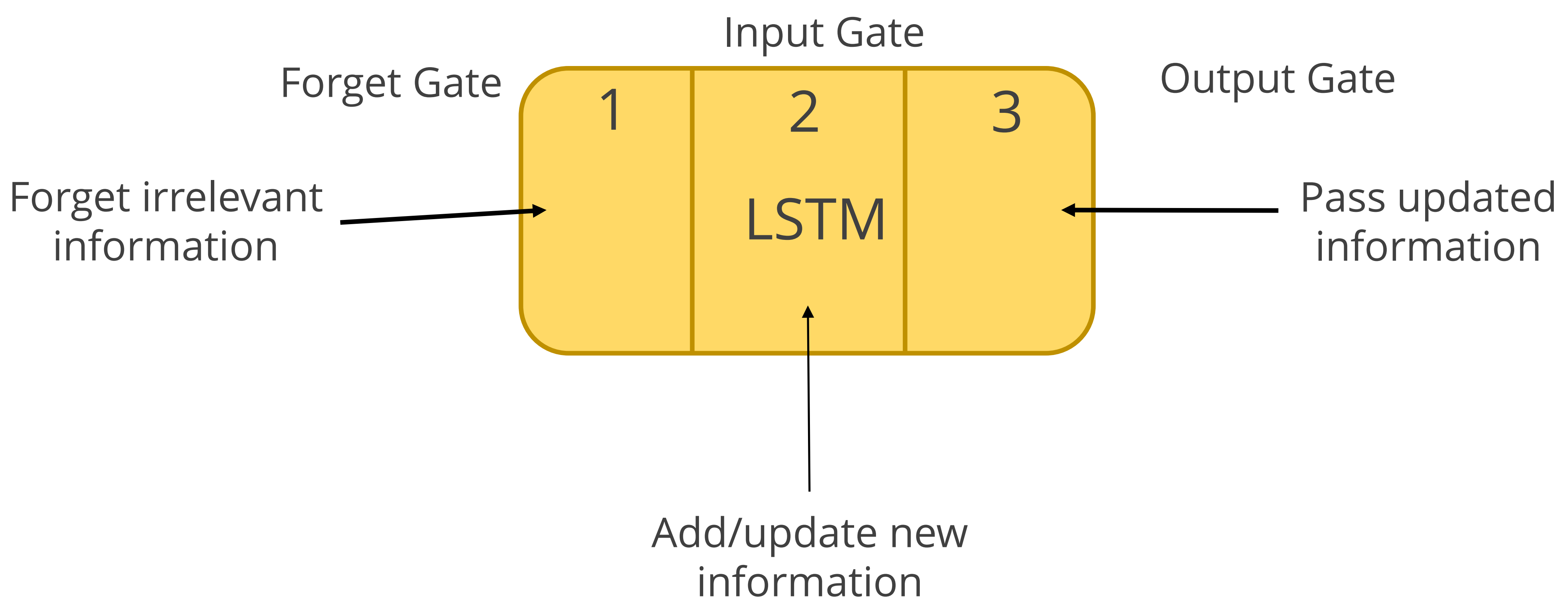
When back-propagating, recurrent neural networks have issues with exploding and disappearing gradients.



The modified versions of RNNs that help address the exploding and vanishing gradient problem are Long Short-Term Memory networks (LSTMs) and Gated Recurrent Units (GRUs).

Long Short-Term Memory (LSTM)

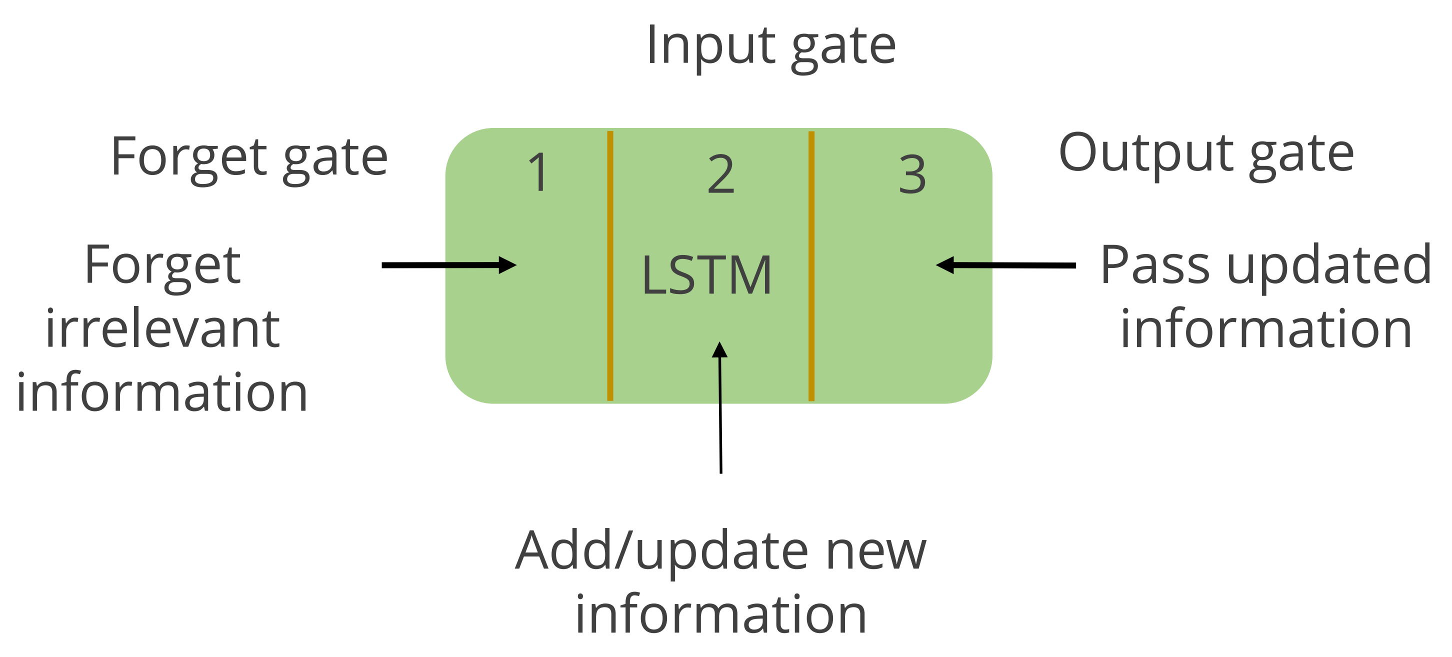
LSTM is an RNN architecture designed to capture and retain long-term dependencies in sequential data, overcoming the limitations of traditional RNNs.



LSTM networks are widely used in tasks such as natural language processing, speech recognition, time series analysis, and more.

Long Short-Term Memory (LSTM)

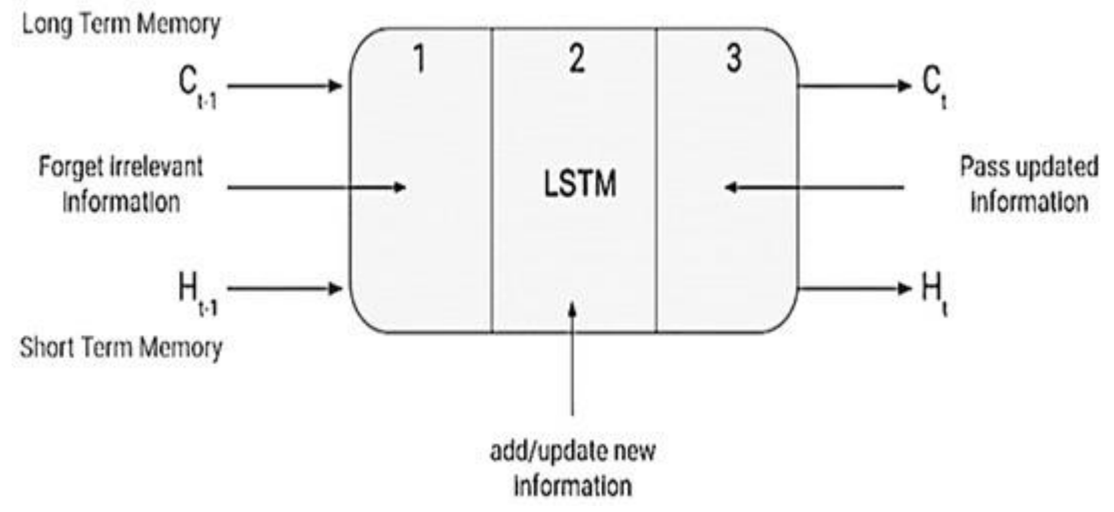
LSTM utilizes three types of gates to control the flow of information within the network:



- **Forget gate:** Determines which information to discard from the previous memory cell state.
- **Input gate:** Regulates which new information to incorporate into the current memory cell state.
- **Output gate:** Controls the output generated by the memory cell.

Hidden State

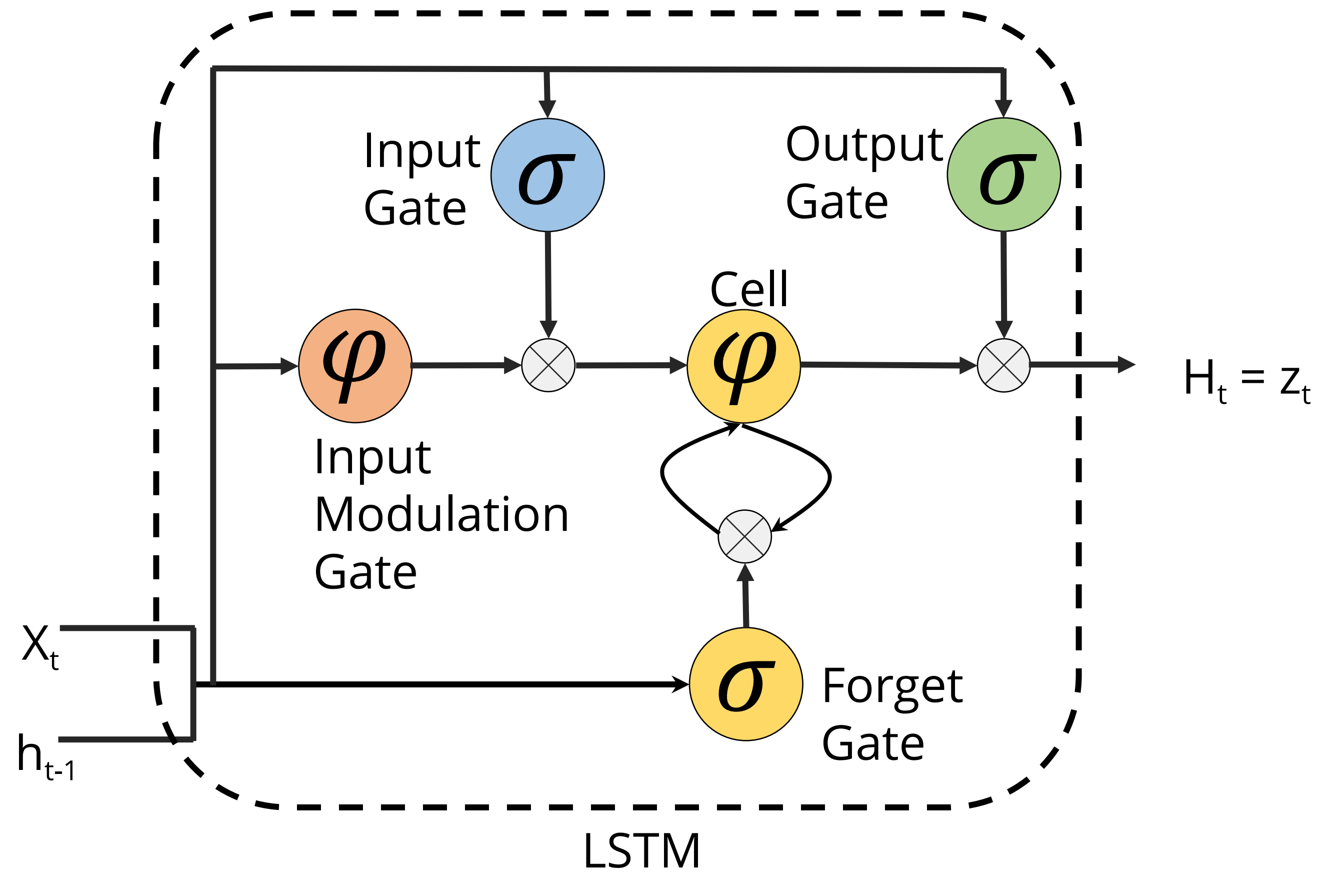
An LSTM, like a simple RNN, includes a hidden state.



- The previous timestamp's hidden state is denoted as H_{t-1}, while the current timestamp's hidden state is denoted as H_t.
- In addition to the hidden state, LSTMs have a cell state.
- The cell state at the previous timestamp is represented as C_{t-1}, and at the current timestamp, it is represented as C_t.
- The cell state enables LSTMs to capture and retain long-term dependencies in sequential data.

Forget Gate

The forget gate controls memory retention. Closing it erases prior memories, and opening it allows all memories to flow through.



The forget gate factor determines which information from the previous memory cell state should be forgotten.

Forget Gate

The equation for the forget gate is expressed as follows:

$$ft = \sigma(X_t * U_f + H_{t-1} * W_f)$$

Here,

ft: Forget gate value at time t

X_t: Input value at time t

U_f: Weight matrix associated with the input X_t

H_{t-1}: Hidden state at the previous time (t-1)

W_f: Weight matrix associated with the hidden state H_{t-1}

Input Gate

The equation for the input gate is expressed as follows:

$$I_t = \sigma(X_t * U_i + H_{t-1} * W_i)$$

Here,

I_t: Input gate value at time t

X_t: Input value at time t

U_i: Weight matrix connecting the input X_t to input gate I_t

H_{t-1}: Hidden state at the previous time (t-1)

W_i: Weight matrix connecting H_{t-1} to input gate I_t

Output Gate

The equation for the output gate is expressed as follows:

$$O_t = \sigma(X_t * U_O + H_{t-1} * W_O)$$

Here,

O_t: Output gate value at time t

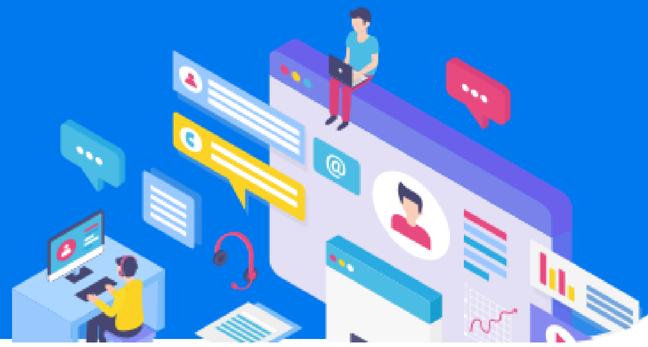
X_t: Input value at time t

U_O: Weight matrix connecting the input X_t to output gate O_t

H_{t-1}: Hidden state at the previous time (t-1)

W_O: Weight matrix connecting H_{t-1} to output gate O_t

Assisted Practices



Let’s understand the concept of text classification with LSTM using Jupyter Notebooks.

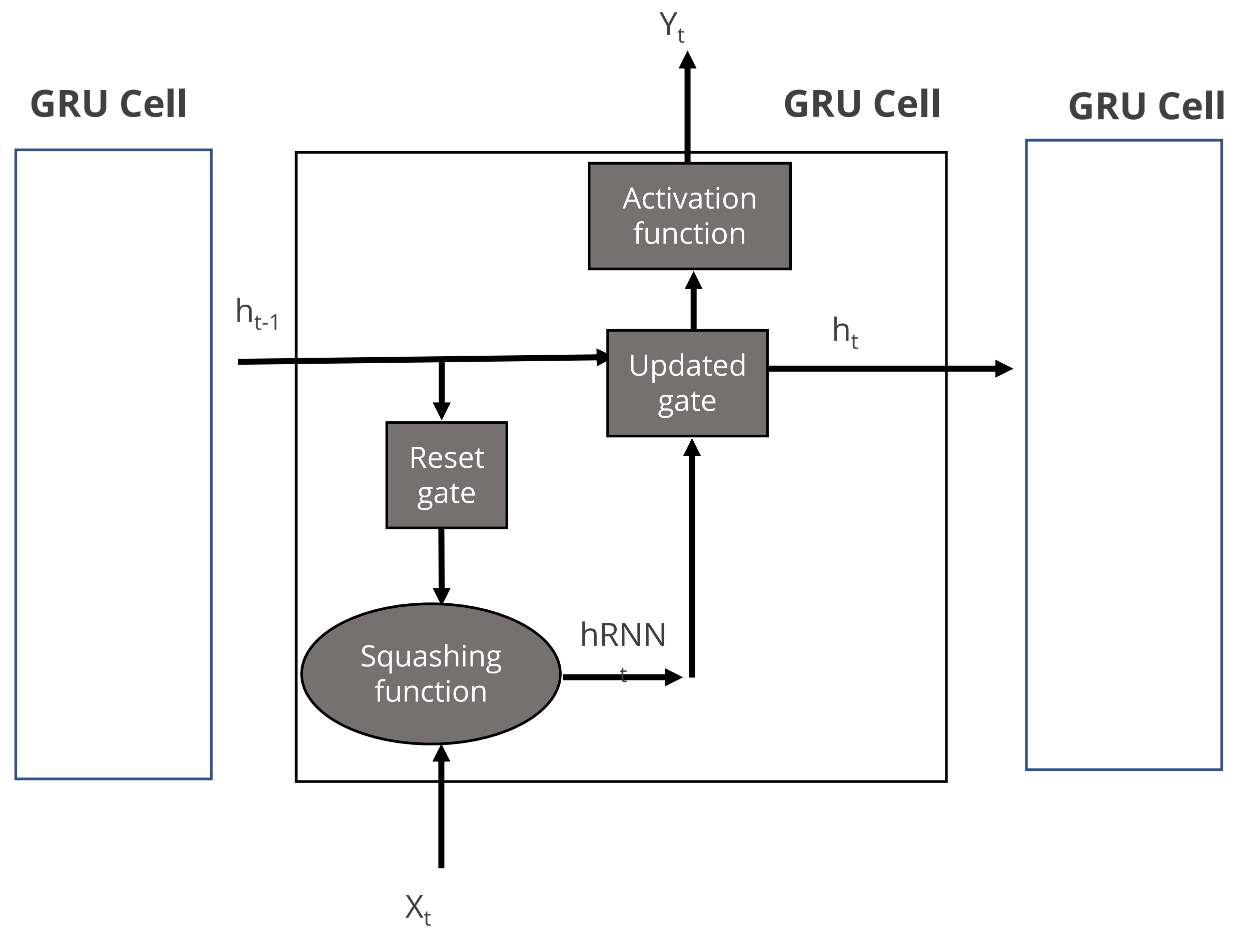
- 10.06.02_Text Classification Using LSTM

Note: Please refer to the Reference Material section to download the notebook files corresponding to each mentioned topic

Gated Recurrent Network (GRU)

Gated Recurrent Network (GRU)

GRUs are a type of RNN architecture that incorporate gating mechanisms to regulate the flow of information between neural network cells.



However, GRUs are relatively newer compared to LSTMs and have shown better performance with a simpler architecture.

Gated Recurrent Network (GRU)

The update gate, like an LSTM's input and forget gate, determines whether the information is retained or discarded.

The reset gate allows us to control the relevance of the previous cell state by deciding the extent to which past data should be ignored.

The current hidden state (h_t) is computed by taking the Hadamard product of the update gate and the previously hidden state vector (h_{t-1}).

Introduction to Hybrid Modeling

Hybrid Modeling

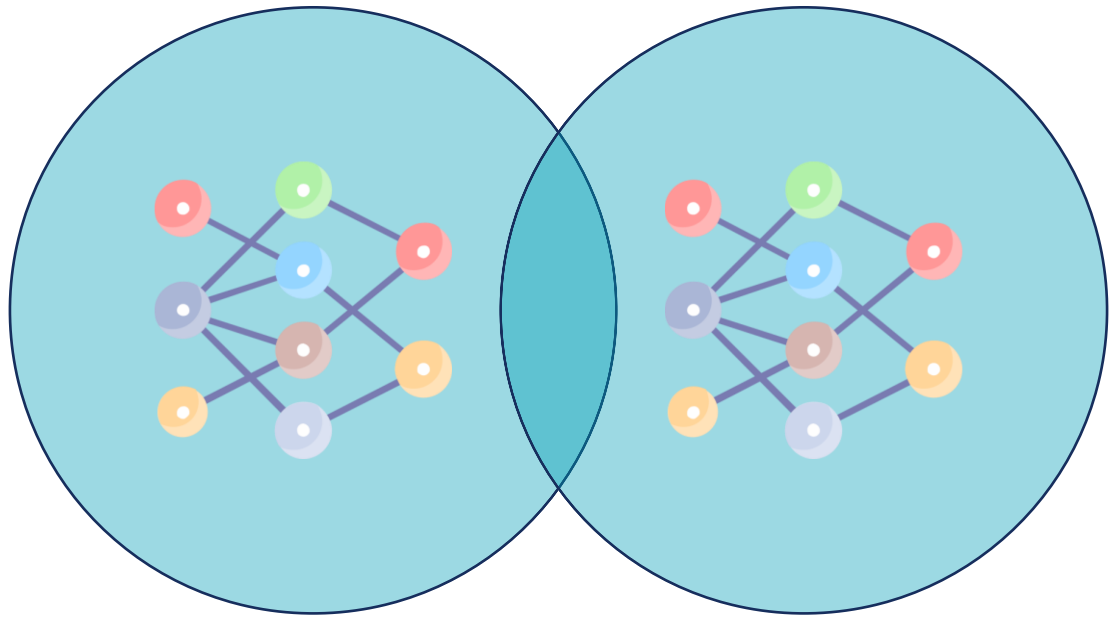
It is the practice of employing two different neural network models and merging them to achieve a sequence of tasks.

One of its well-known implementations is the concept of convolutional recurrent neural networks (CRNN).

The convolution network is mostly used for spatial analysis such as extracting essential features from an image.

Hybrid Modeling

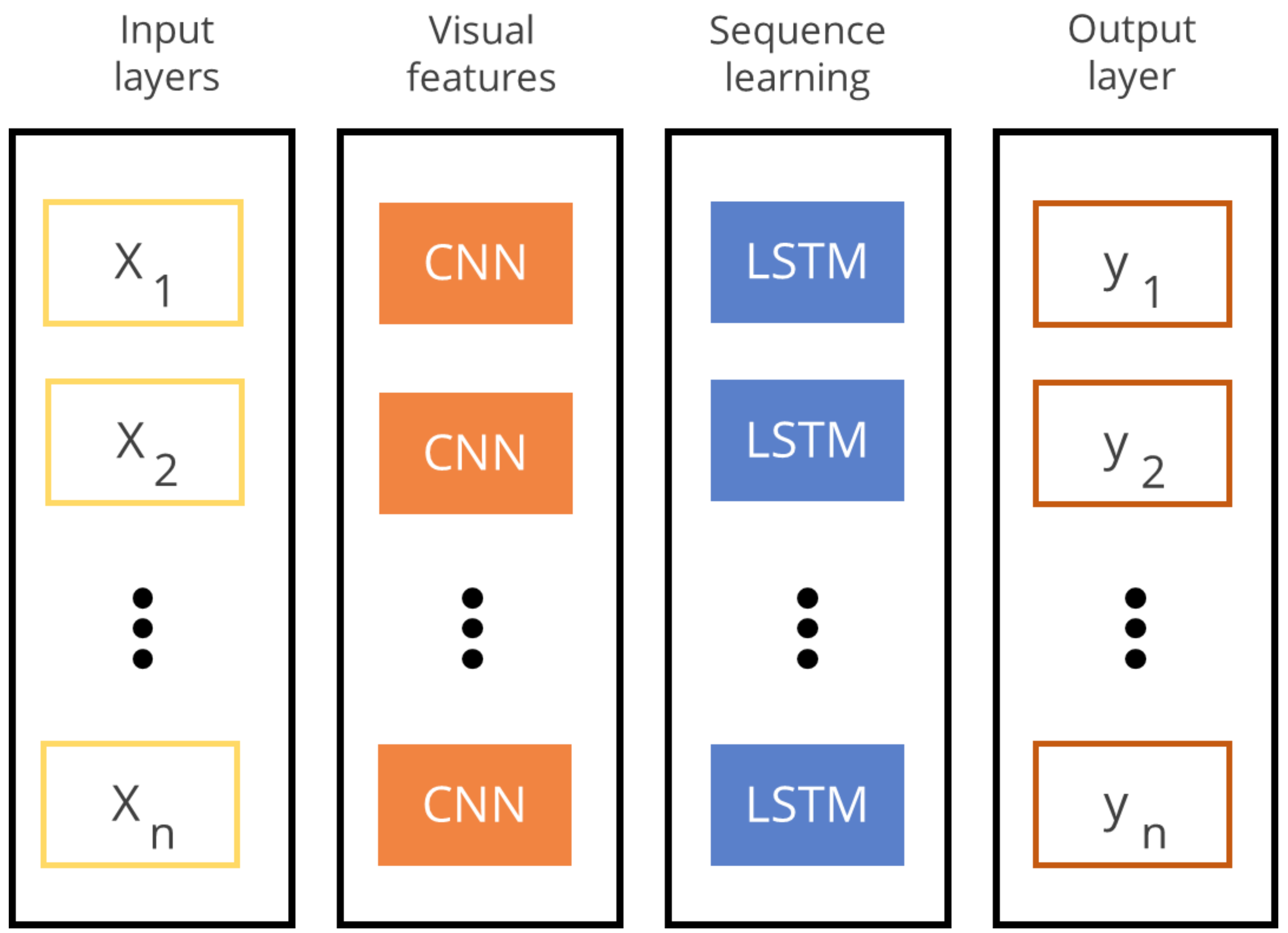
The recurrent network is used for temporal analysis, such as finding links between the extracted features that influence the output.



By merging the two networks, the machine can learn patterns in the sequential data provided and form appropriate predictions.

Hybrid Modeling: Example

A model predicts the next digit in a sequence of handwritten digits, and the input given is < 1, 2, 4, 8, ... >.



The figure here shows it's working.

Hybrid Modeling: Example

It performs the following:

It slices the image of the digits into numerous segments.

The CNN first extracts the essential features of each segment, which are then passed into the RNN.

The RNN recursively analyzes each feature, taking into consideration the previous input.

The algorithm decodes the output, and the model predicts 16 as the next digit in the sequence.

Architecture of a CNN-RNN Hybrid Model

Convoluted Neural Network

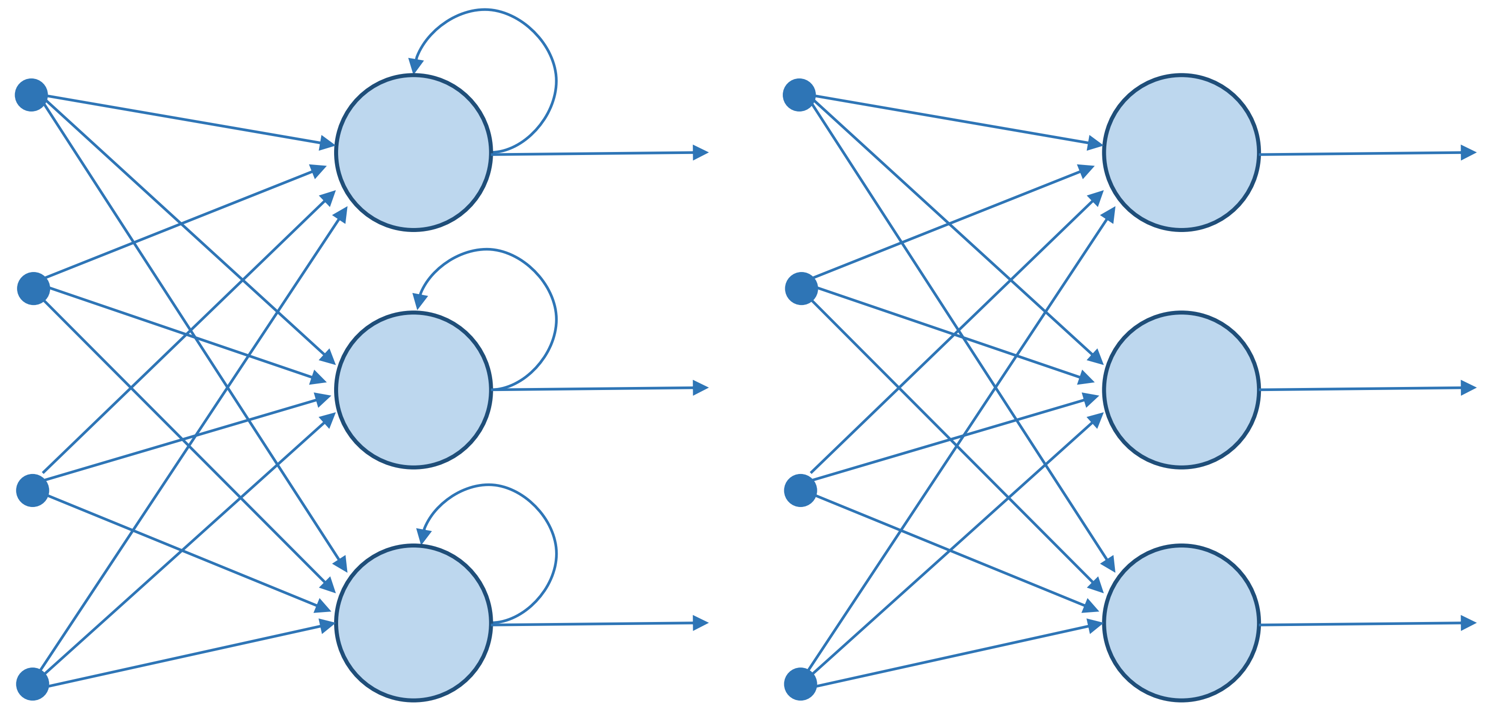
A traditional, dense neural network flattens a matrix into a vector, which results in a spatial loss.

For more significant matrices, the number of weights would be vast, thus increasing the computational time and data needed.

CNN prevents this by passing a filter matrix that extracts relevant information.

Recurrent Neural Network

While feedforward neural networks only consider the present input, an RNN utilizes the previous output as well.



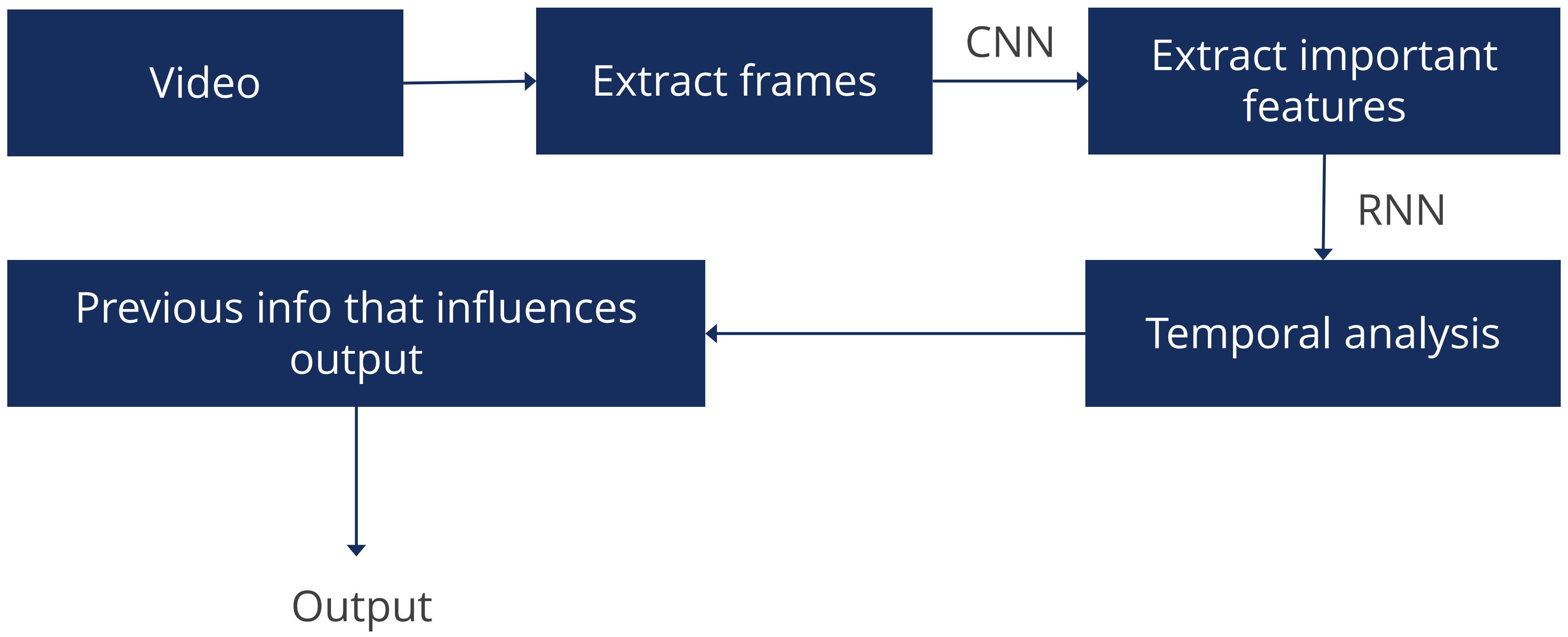
Recurrent neural network   Feed-forward neural network

It prevents temporal loss and allows the model to recognize the output features.

Applications of CRNN

Video Classification Using CRNN

Consider performing a video classification using the CRNN hybrid model:


Video
Extract frames
CNN
Extract important features
RNN
Temporal analysis
Previous info that influences output
Output

A video is a series of images.

Video Classification Using CRNN

Extract frames and pass them to the CNN, which extracts the essential features in each frame.

Feed the features to the RNN, which analyzes them in order and considers the previous information to find links between them that influence the output.

Such networks can be employed in a theft detection system.

Building a Theft Detection System Using CRNN

Let us create a theft detection program based on surveillance footage.

Dataset: Anomaly Detection dataset (UCF-Crime, CRCV)

It is curated by the Center for Research in Computer Vision, University of Central Florida, and contains several anomaly videos, including theft and arson, along with regular footage.

Building a Theft Detection System Using CRNN

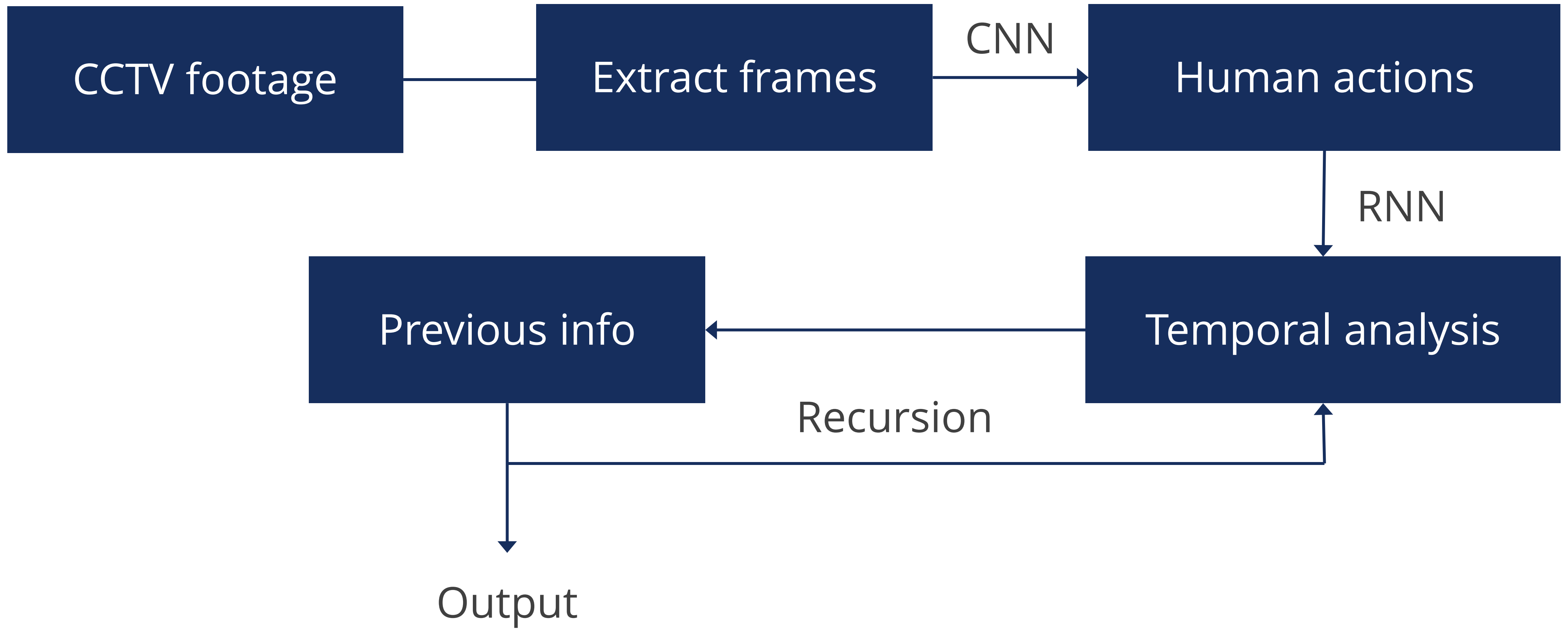
While criminal cases are not rare, they are infrequent compared to other unusual cases.

Security systems employ CCTV cameras for surveillance, but rely on human monitors to observe inconsistencies, resulting in costly errors.

A theft detection system based on computer vision renders human involvement redundant to solve issues.

Building a Theft Detection System Using CRNN

In the proposed working principle, individual frames from each surveillance video are extracted.



These are passed to the CNN, which extracts relevant information, like moving lots of appliances and valuable objects simultaneously.

Building a Theft Detection System Using CRNN

Features are passed to the RNN, which brings a temporal context. The features could be:

Time

Presence of the owner in the video

Clothes worn by the people

This enables the computer to decide if a certain instance is an act of theft.

Discussion: LSTM and CRNNs

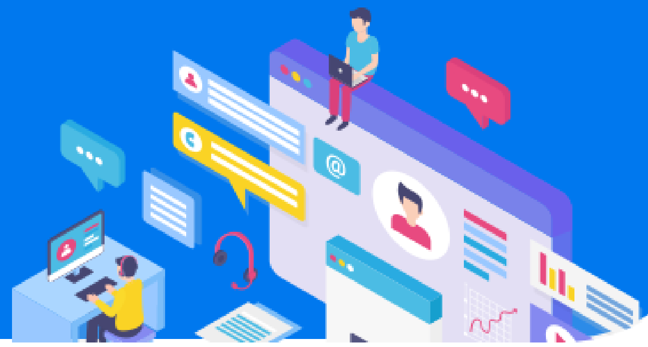
- What is LSTM?

LSTM stands for Long Short-Term Memory, and it is a type of recurrent neural network (RNN) architecture that is particularly effective in modeling sequential data and capturing long-term dependencies.

- What are some of the applications of CRNNs?

CRNNs have found applications in various domains such as text, speech, handwriting recognition, gesture recognition, and action recognition.

Assisted Practices



Let’s understand the concept of video classification with the hybrid model using Jupyter Notebooks.

- 10.11_Video Classification Using Hybrid Model

Note: Please refer to the Reference Material section to download the notebook files corresponding to each mentioned topic

Key Takeaways

- A sequence modeling program can model, interpret, and predict various types of sequential data.
- The recurrent neural network is a state-of-the-art algorithm that adds the recent past to the present, allowing it to make precise predictions.
- A recurrent neural network has four types of architecture: one-to-one, one-to-many, many-to-one, and many-to-many.
- Hybrid modeling refers to employing two different neural network models and merging them to achieve a sequence of tasks.

Knowledge Check

Knowledge Check

1

What is the purpose of padding in sequential modeling?

A. To make all the sequences of the same length

B. To reduce the number of sequences in the dataset

C. To remove stop words from the sequences

D. To turn the tokens into lists of sequences

Knowledge Check

1

What is the purpose of padding in sequential modeling?

A. To make all the sequences of the same length

B. To reduce the number of sequences in the dataset

C. To remove stop words from the sequences

D. To turn the tokens into lists of sequences

The correct answer is **A**

Padding is used in sequential modeling to make all the sequences of the same length.

Knowledge Check

2

What is sequential modeling?

A. A process used to analyze a series of input values to generate a sequence of values

B. A process used to analyze a series of output values to generate a sequence of input values

C. A process used to analyze a series of output values to generate a sequence of output values

D. A process used to analyze a series of input values to generate a sequence of input values

Knowledge Check

2

What is sequential modeling?

A. A process used to analyze a series of input values to generate a sequence of values

B. A process used to analyze a series of output values to generate a sequence of input values

C. A process used to analyze a series of output values to generate a sequence of output values

D. A process used to analyze a series of input values to generate a sequence of input values

The correct answer is **A**

Sequential modeling is a process used to generate a sequence of values by analyzing a series of input values.

Knowledge Check

3

Which neural network model is mostly used for spatial analysis, such as extracting essential features from an image?

A. Recurrent neural network

B. Convolutional neural network

C. Dense neural network

D. None of the above

Knowledge Check

3

Which neural network model is mostly used for spatial analysis, such as extracting essential features from an image?

A. Recurrent neural network

B. Convolutional neural network

C. Dense neural network

D. None of the above

The correct answer is **B**

The convolution network is mostly used for spatial analysis, such as extracting essential features from an image.

Thank You!