Day85_RNN_Introduction

September 17, 2025

1 Recurrent Neural Networks RNN

Recurrent Neural Networks RNN are a type of neural network designed for sequential data such as text speech time series and video. Unlike feedforward neural networks RNNs have connections that form directed cycles which allow them to maintain memory of previous inputs. This makes RNNs very effective for tasks where context and order of inputs are important.

RNNs are widely used for natural language processing speech recognition machine translation and other sequence related tasks.

2 RNN Architecture

The key idea behind an RNN is that it processes input sequences step by step while maintaining a hidden state that carries information about previous inputs. This hidden state is updated at every step and passed to the next step along with the new input.

At each time step the RNN takes two inputs

- The current input vector xt
- The hidden state from the previous time step ht-1

It then produces

- The updated hidden state ht
- An output yt

The recurrence relation can be written as
ht = f(Wxh xt + Whh ht-1 + bh)
yt = Why ht + by

Here f is usually a nonlinear activation function such as tanh or ReLU.

3 Sequence to Sequence Neural Network

When RNNs are used for tasks like machine translation or chatbot building we use an encoder decoder architecture. This is called a sequence to sequence model.

3.1 Encoder

The encoder reads the input sequence one token at a time and compresses the entire sequence into a fixed length vector called the context vector. This vector represents the meaning or features of the input sequence.

3.2 Decoder

The decoder takes the context vector from the encoder and generates the output sequence step by step. At each step it predicts the next token in the sequence until the final output is produced. Decoders are also implemented using RNNs LSTMs or GRUs.

4 LSTM Long Short Term Memory

A problem with simple RNNs is that they suffer from vanishing or exploding gradients when trained on long sequences. This makes it hard for them to remember information from far in the past.

To solve this the Long Short Term Memory network LSTM was introduced. LSTMs have a special architecture that allows them to learn long range dependencies. They use gates to control the flow of information.

4.1 Gates in LSTM

Input gate

Controls how much new information from the current input flows into the memory cell

Forget gate

Controls how much of the past memory should be forgotten or retained

Output gate

Controls how much information from the memory cell should be passed to the hidden state

Memory cell

Acts as a conveyor belt that carries information across time steps with only minor modifications

These gates allow the LSTM to selectively remember or forget information which solves the vanishing gradient problem.

5 GRU Gated Recurrent Unit

The Gated Recurrent Unit GRU is a simplified version of the LSTM. It combines the input gate and forget gate into a single update gate and also has a reset gate.

5.1 Gates in GRU

Update gate
Decides how much of the past information needs to be passed along to the future

Reset gate
Decides how much of the past information to forget

GRUs are simpler and computationally faster than LSTMs while performing similarly well on many tasks.

6 Applications of RNN LSTM and GRU

Recurrent Neural Networks and their variants LSTM and GRU are widely used in tasks that involve sequential data where the order of information matters. Below are some important applications.

6.1 Natural Language Processing

RNNs are heavily used in text related tasks such as language modeling sentiment analysis machine translation text summarization and question answering. LSTM and GRU help in capturing long term dependencies in language which simple RNNs cannot handle effectively.

6.2 Next Word Prediction

Given a sequence of words the model predicts the next likely word. This is useful in predictive keyboards autocomplete systems and writing assistants. LSTMs are particularly effective in this application because they can remember context over many words.

6.3 Machine Translation

Using encoder decoder RNN architectures we can translate sentences from one language to another. The encoder processes the input sentence and the decoder generates the translated sentence in the target language.

6.4 Speech Recognition

RNNs can process audio signals as sequential data. By mapping audio frames to corresponding phonemes or words they can convert spoken language into text. LSTMs and GRUs are widely used in modern speech recognition systems.

6.5 Chatbots and Conversational AI

Chatbots use sequence to sequence RNN models where the encoder processes the user query and the decoder generates a human like response. They can be trained on large conversation datasets to learn natural conversation flow.

6.6 Text Summarization

RNNs can generate concise summaries of long documents. The encoder reads the entire text and the decoder outputs a shorter version that retains the main ideas.

6.7 Time Series Forecasting

RNNs can analyze patterns in time series data such as stock prices weather data and sales data to predict future values. LSTMs are preferred for this task as they capture long range temporal dependencies.

6.8 Music Generation

RNNs can be trained on sequences of musical notes to generate new pieces of music. Given a starting sequence the model can predict the next notes and create original compositions.

6.9 Handwriting Recognition and Generation

RNNs can be trained to recognize handwritten text by processing sequences of strokes or pixels. They can also generate handwriting styles when trained on a dataset of handwritten samples.

7 Summary of Applications

RNNs LSTMs and GRUs are extremely versatile and are used in tasks involving language speech vision and time series.
They are the backbone of many real world AI applications like Google Translate Siri predictive keyboards and conversational agents.

8 Summary

RNNs are powerful models for handling sequential data.
Basic RNNs capture short term dependencies but struggle with long sequences due to vanishing gradients.
LSTMs solve this with memory cells and gating mechanisms while GRUs provide a simpler alternative.
They are widely applied in next word prediction speech recognition and chatbot systems using encoder decoder architectures.

[ ]: