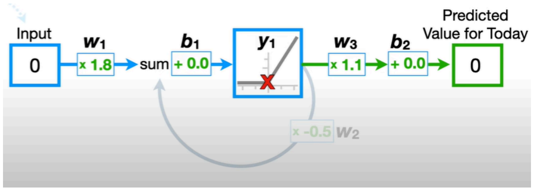
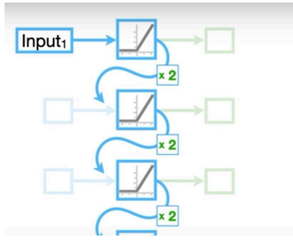
Basic Concepts of Recurrent Neural Networks

- Objective: Deal with not fixed input value lengths to make predictions. It has weights, biases and activation functions like other NNs but also has a feedback loop -> makes it possible to use sequential input values.
- We can make an unrolled diagram to visualise the inputs more clearly. The weights and biases are the same in each line.





22

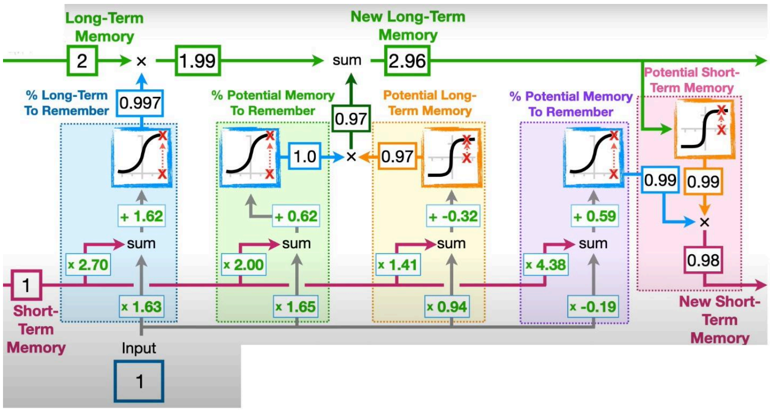
The Markov assumption is used to describe a model where the Markov property refers to the memoryless property of a stochastic process, which means that its future evolution is independent of its history. As we can see in the unrolled image this is the case of the RNN.

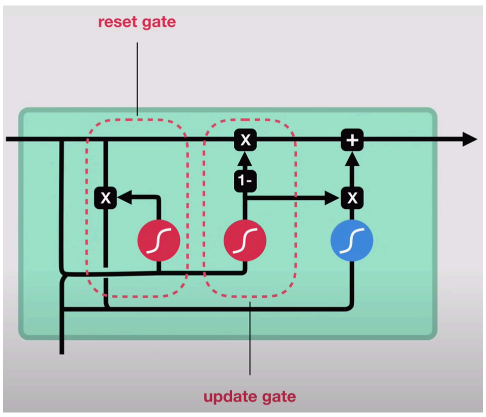
$$P(X_n = x_n \mid X_{n-1} = x_{n-1}, \ldots, X_0 = x_0) = P(X_n = x_n \mid X_{n-1} = x_{n-1}).$$

Different RNN architectures

1) LSTM

One problem with simple RNN is that when doing backpropagation for gradient descent, a term involving w2^(Number of unrolls) appears. This affects the step size when minimizing the loss function, vanishing the gradient if w2<1 and exploding it if w2>1.
A solution is Long short-term memories where each unit uses 2 separate paths to make predictions. One that involves short-term memory and another that involves long-term memory. This avoids having a weight connecting the units thus avoiding the problem mentioned before. It has added parameters to regulate the amount of long-term memory forgotten, update it with the input and set the new short-term memory.





2) *GRU (Gated Recurrent Unit):* are like LSTM with a gating mechanism to input or forget certain features, but lacks a context vector or output gate, resulting in fewer parameters than LSTM.
Has two gates:

- Update gate: decides what information to forget and what to add.
- Reset gate: decides how much passed information to forget.

The main advantage is that as there are fewer tensor operations, GRU are speeder to train than LSTM.
Usually, users use both types of architectures and given the results decide which one is better for their data set.

Case studies:

1) *Language*: Predicting the next word of a phrase is a typical RNN example, as the length of the phrases is not always the same. To give a logical prediction of the next word, some important words of the phrase have to be saved so LSTM is desired.

2) In *time series forecasting* RNN architectures have proven to be effective in capturing temporal dependencies within sequential data. For example in financial time series forecasting, LSTMs have demonstrated being able to model complex patterns and capture long-term dependencies.