Reading 11

★ Recurrent Neural Networks

- Often thought of as a stepping stone to Long-Short Term Memory Networks and Transformers.

- Scenario: Creating a NN to predict stock prices.

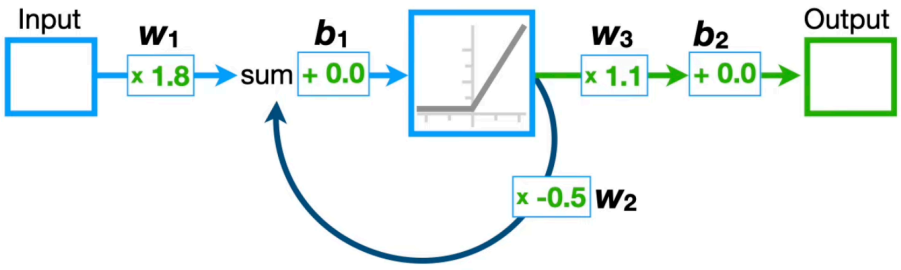
- So we need the NN to be flexible in terms of how much sequential data we use to make a prediction.
  ↳ Recurrent Neural Networks allow for this.

- The key idea in RNNs is that they also have feedback loops.
  ↳ the feedback loop makes it possible to use sequential input values, like stock market prices collected over time, to make predictions.

- In our scenario we have 4 cases:
  1. If yesterday and today the price is low, then tomorrow should be low
  2. yesterday = low, today = medium → tomorrow = high
  3. yesterday = high, today = medium → tomorrow = high
  4. yesterday = high, today = high → tomorrow = high

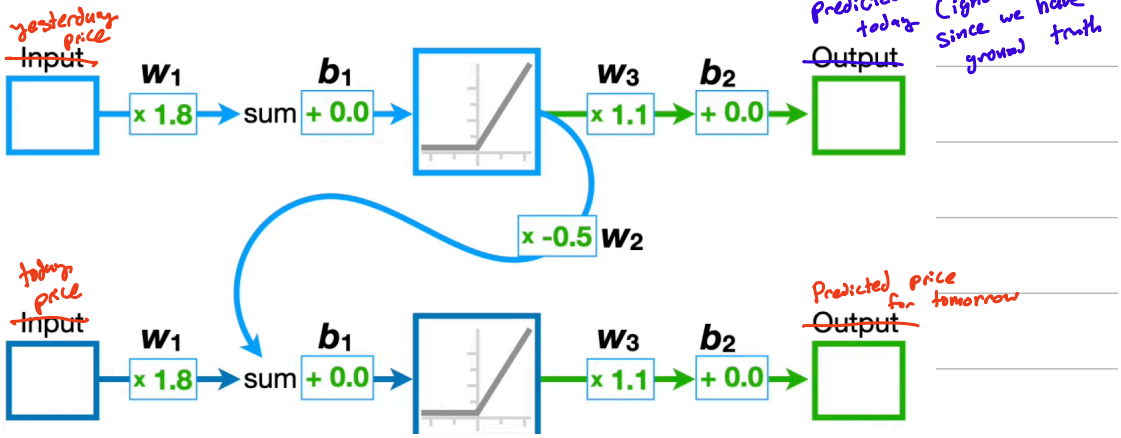
- first step is to scale values s.t. low = 0, medium = 0.5, high = 1
- If you pass in yesterdays price and skip the feedback loop, you get the prediction for today.

- RNN Diagram (for this scenario):

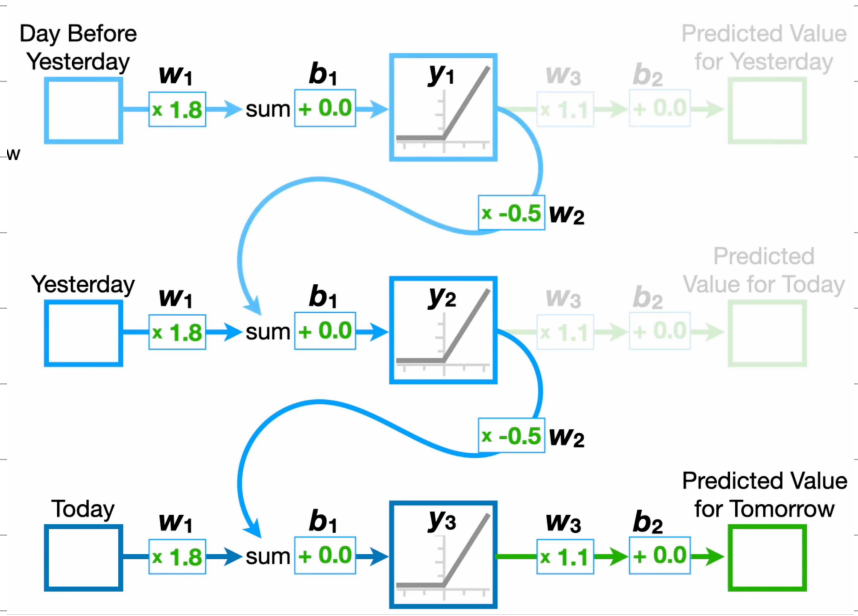


- The feedback loop allows both yesterdays and todays value to influence the prediction.

- You can think about the feedback loop as pointing to the sum in the second pass of the RNN:



- What if we want to use 3 days of data to predict?
  - ↳ Just keep unrolling !!!



- Regardless of how many times we unroll a recurrent neural network, the weights and biases are shared across every input.
  - ↳ i.e. only input and output changes for each "unrolled" NN.
  - ↳ i.e. the # of times you unroll does NOT increase the number of weights and biases that we have to train.

- But, the more you unroll a RNN, the harder it is to train.
  - ↳ because of the vanishing / exploding gradient problem.

- Exploding gradient: when the gradient contains large numbers which lead to large steps in gradient descent. (so we may just keep bouncing around the loss function and never find the min)

- Vanishing gradient: when the gradient contains small numbers, so steps in gradient descent become small.
  - ↳ often leads to max # of steps in gradient descent being reached before we find the min of loss func.

- Long-Short Term memory networks help out with mitigating the exploding/vanishing gradient problem.