Maths Behind LSTMs

1 Motivations

LSTM stands for Long-Short Term Memory. They are a type of Random Neural Network (RNN) and help avoid the vanishing/exploding gradient problem that RNNs commonly suffer from when they're unrolled.

In an LSTM we have a long-term memory and a short-term memory and both are utilised to help us make our predictions.

2 Process

At each time step of the LSTM. The long-term and short-term memories are initialised as 0 for the first time step and are updated throughout. The values for the long and short-term memories for the following time steps are determined by the output of the previous step. The input value is the corresponding value of the predictor variable for this time step.

2.1 Forget Gate

In the first stage we begin by doing,

$$x = \text{(STM Value * Its Weight)} + \text{(Input Value * Its Weight)} + \text{Bias}$$

This value of x is then put into the Sigmoid Activation Function. The formula for the Sigmoid Activation Function is,

$$f(x) = \frac{e^x}{e^x+1}$$

We then multiply the value returned by the Sigmoid Activation Function by the value we have for the long-term memory to get an updated LTM.

This first stage determines how much of the long-term memory should be kept.

2.2 Input Gate

In the next step, we first calculate x in the same manner that we did earlier and input this into the Tanh Activation function. The formula for the Tanh Activation Function is,

$$f(x) = \frac{e^x-e^{-x}}{e^x+e^{-x}}$$

This is our value for the potential LTM.

Then, again, we caluclate x in the same way as before and then input this into the Sigmoid Activation Function. We'll call this percentage potential retained for referential purposes.
Lastly, we do,

$$\text{Additional LTM} = \text{Potential LTM * Percentage Potential Retained}$$

and add this onto the value we have for the LTM to get a New long-term memory.

2.3 Output Gate

In the last stage we input our New LTM into the Tanh Activation Function. This is our potential STM.

We then calculate the percentage potential retained as before. Lastly we do,

$$\text{New LTM} = \text{Potential STM} * \text{Percentage Potential Retained}$$

We now have our new values for the short-term and long-term memories that can be used for the next time step.

2.4 Next Steps

The above process is repeated for each time step, with the new input being the value of the predictor variable for this time step.

3 Weights

Weights are crucial in LSTMs, at each stage of the process each value we use will have its own weight. These weights are determined during the training process to minimise the loss function of the model. At each time step we need to use the same weights for each stage within the time step.

4 How we're using the LSTM

We'll be looking at two LSTM models to predict pollution in Beijing, a univariate and a multivariate one. For the univariate model, it will be relatively simple and perform as above with the input value being the pollution of each hour. The multivariate model is a bit more complex as there are multiple features being inputted at each time step, in our case there will be eight.

5 Why is this appropriate for our task

LSTMs are appropriate for the task at hand since there are many independent calculations going on within the model at each time step and these can be parallelised in order reduce the strain on any one processing unit and speed up a fairly complex model.

6 References

`https://towardsdatascience.com/lstm-gradients-b3996e6a0296`
`https://medium.com/@leenabora1/simplified-math-behind-complex-lstm-equations-66cff0d52d78`