California State University, Dominguez Hills
Department of Computer Science
CSC 595

Professor: Dr. Benyamin Ahmadnia
bahmadniayebosari@csudh.edu

Spring 2025

Copyright Notice

Table of Contents

Common Activation Functions

Sigmoid Function

- The sigmoid function is commonly used in RNNs. It has a range between 0 and 1, which makes it useful for binary classification tasks. The formula for the sigmoid function is:

$$\sigma(x) = 1 / (1 + e^{(-x)})$$

Hyperbolic Tangent (tanh) Function

- The tanh function is also commonly used in RNNs. It has a range between -1 and 1, which makes it useful for non-linear classification tasks. The formula for the tanh function is:

$$\tanh(x) = (e^x - e^{(-x)}) / (e^x + e^{(-x)})$$

Rectified Linear Unit (ReLU) Function

- The ReLU function is a non-linear activation function that is widely used in deep neural networks. It has a range between 0 and infinity, which makes it useful for models that require positive outputs. The formula for the ReLU function is:

$$\text{ReLU}(x) = \max(0, x)$$

Leaky ReLU Function

- The Leaky ReLU function is similar to the ReLU function, but it introduces a small slope to negative values, which helps to prevent “dead neurons” in the model. The formula for the Leaky ReLU function is:

$$\text{Leaky ReLU}(x) = \max(0.01x, x)$$

Softmax Function

- The softmax function is often used in the output layer of RNNs for multi-class classification tasks. It converts the network output into a probability distribution over the possible classes. The formula for the softmax function is:

$$\text{softmax}(x) = e^x / \sum(e^x)$$

Backpropagation through Time

Backpropagation through Time

- In a typical RNN, one input is fed into the network at a time, and a single output is obtained. But in backpropagation, you input the current and the previous inputs. This is called a timestep; one timestep will consist of many time series data points entering the RNN simultaneously.

- Once the neural network has trained on a time set and given you an output, that output is used to calculate and accumulate the errors. After this, the network is rolled back up, and weights are recalculated and updated, keeping the errors in mind.

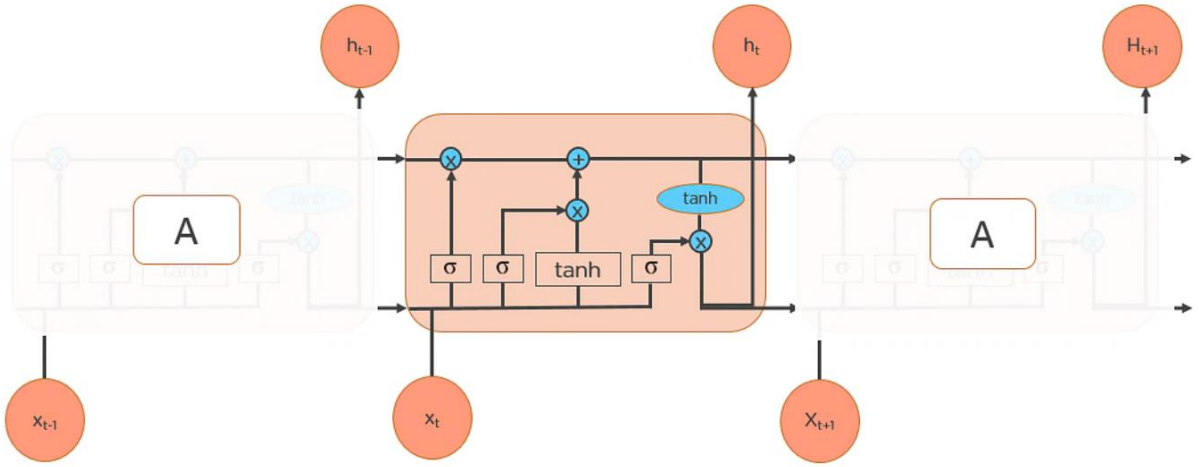
Variant RNN Architectures

LSTM Networks

- LSTM Network
- GRU Network
- Bidirectional RNN
- Encoder-Decoder RNN
- Attention Mechanisms

Long Short-Term Memory Networks

- LSTMs are a special kind of RNN — capable of learning long-term dependencies by remembering information for long periods is the default behavior.
- All RNN are in the form of a chain of repeating modules of a neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.



Long Short-Term Memory Networks

- The first step in the LSTM is to decide which information should be omitted from the cell in that particular time step. The sigmoid function determines this. It looks at the previous state (h_{t-1}) along with the current input x_t and computes the function.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

f_t = forget gate
Decides which information to delete that is not important from previous time step

Long Short-Term Memory Networks

- In the second layer, there are two parts. One is the sigmoid function, and the other is the tanh function. In the sigmoid function, it decides which values to let through (0 or 1). The tanh function gives weight to the values that are passed, deciding their level of importance (-1 to 1).

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

i_t = input gate
Determines which information to let through based on its significance in the current time step

Long Short-Term Memory Networks

- The third step is to decide what the output will be. First, we run a sigmoid layer, which decides what parts of the cell state make it to the output. Then, we put the cell state through tanh to push the values to be between -1 and 1 and multiply it by the output of the sigmoid gate.

$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

o_t = output gate
Allows the passed in information to impact the output in the current time step