Web Neural Network with Complete DiGraphs

Frank Li (angli23@cs.washington.edu)

Background

- Most neural networks are linear and sequential.
- RNNs and LSTMs have recurrence but they are based on a predetermined structure, i.e. takes state or memory from one previous node and passes on to the next node.
- General structure of models are directional, has a clear input and output and some ordered layer in between.
- Networks don't recognize continuous data; time or order of data are often treated as just another dimension.

Question: What if we create a neural network with full connections that are recurrent and cyclical?

Related work / sources of inspiration:

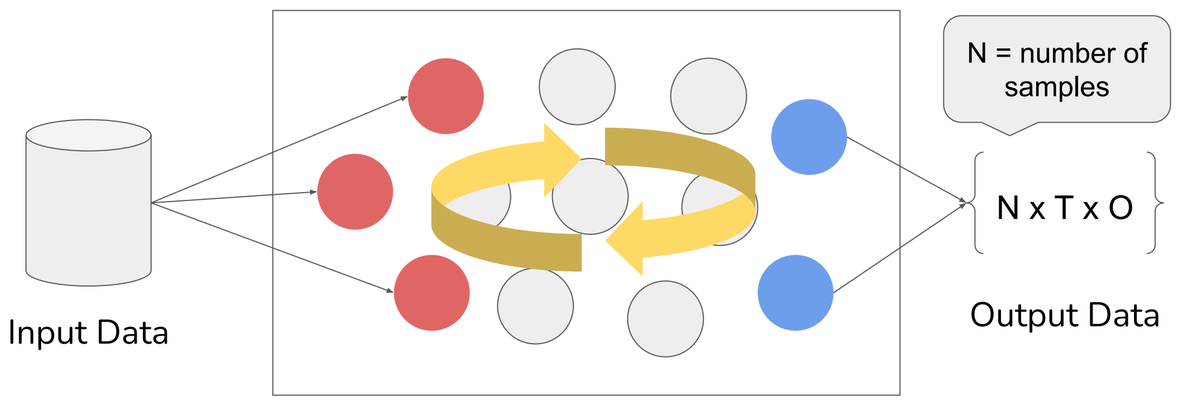
- LSTM: Solving the vanishing gradient problem in RNNs by using gates to control the flow of information.
- Randomly Wired Neural Network (Xie et al., 2019): neural architecture search through stochastic network generation, creating randomly wired graph-structured networks.
- Spiking Neural Networks (Chowdhury et al., 2021): allow neurons to operate on continuous signal data like synapses between neurons in a biological brain.

Model Design

Summary: a neural network model that aims to mimic the biological brain more closely by structuring the network as a complete directed graph that processes continuous data for each time step.

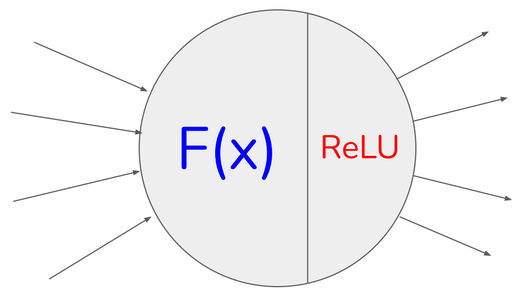
Hyperparameters: number of total nodes in the graph (Q), input dimension (I), output dimension (O), maximum timesteps (T), optionally, dropout rate (R).

Example: Q = 12, I = 3 (red nodes), O = 2 (blue nodes)



Complete directed graph (edges not shown)

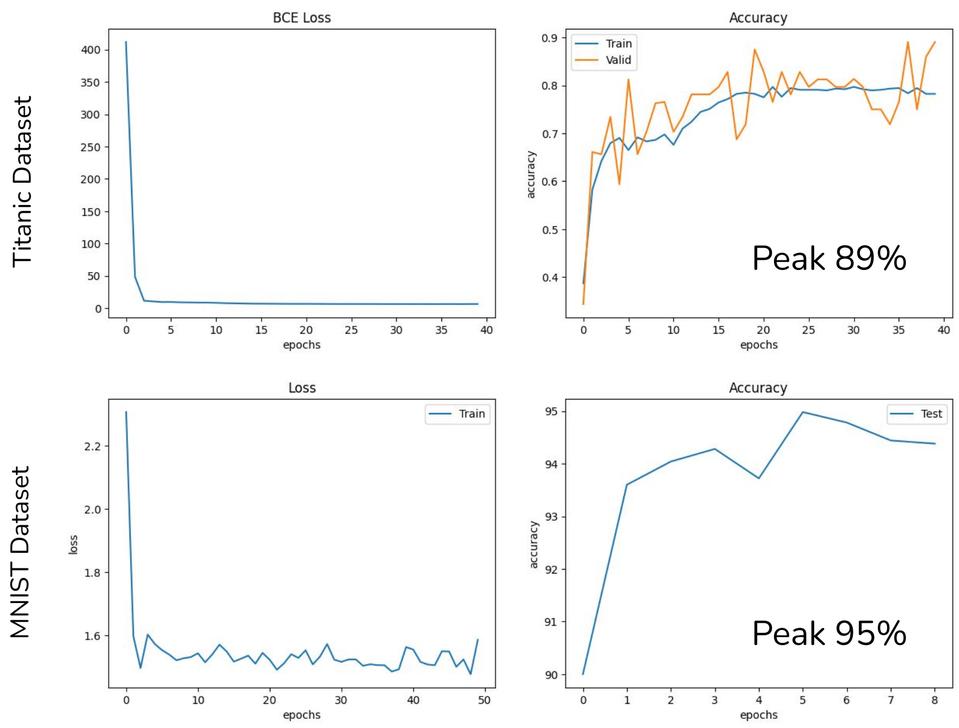
Input neurons will take in data from input every timestep and output be read from output neurons every timestep. All neurons (including input and output neurons) will take in inputs from all other neurons, perform calculations, then output to all other neurons (hence a complete graph).



- Input and output nodes are also just normal nodes, sampled from the full collection of nodes according to the hyperparameters I and O.
- Hence Q >= I + O needs to be true.
- F(x) is just a linear function Wx+b, with input x size Q.

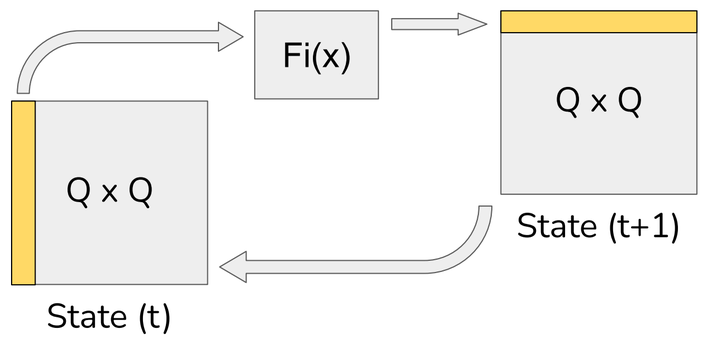
Methodology & Results

The model was trained on small datasets from Kaggle and the MNIST dataset to illustrate the performance of the web neural network. Due to a lack of time and compute resources, the models were only trained on a small number of epochs to simply verify the validity of the model. Also, hyperparameters used in training are not fully optimized as training a recursive model takes very long, making it difficult to iterate on.



Implementation

Let the state of nodes be an **adjacency matrix** where the (row, col) element indicates the output of the row-th neuron to the col-th neuron.



For every node i:
- Get inputs (i-th column)
- Pass through Fi(x)
- Store output (i-th row)

Where Fi(x) will be a linear function (nn.Linear) followed by a ReLU activation.

Naive solution: iterate N times for every node.
- Slow, complexity of O(TQ), with very high Q since Q is at least I + O.
- Also requires Q linear functions (nn.Linear).

Vectorized solution: utilize batch matrix multiplication (tensor.matmul) function to compute the outputs for every node together.
- Allows us to remove one layer of loop and only use one nn.Linear!
- Suppose W in Fi(x) has dimension (Q, Q) and i-th column is (Q), then we have Wx = (Q, Q) x (Q) -> (Q). We need to repeat this for all Q.
- Utilizing the property (A, B, x, y).matmul(A, B, y, z) = (A, B, x, z), we can form operation (N, Q, Q, Q).matmul(N, Q, Q, 1) to calculate outputs in batch.
- We just need to remember to transpose the output matrix and store it in the next state (outputting row-wise and inputting column-wise).

Future Improvements

The model can be adapted to **language modelling problems** just like recurrent neural networks. This can test the ability of web neural networks to preserve contextual information. Also, with more time and compute, we can test **hyperparameter optimization** methods on the model to visualize how they affect model performance.

References

Sayeed Shafayet Chowdhury, Nitin Rathi, and Kaushik Roy. One Timestep is All You Need: Training Spiking Neural Networks with Ultra Low Latency, Oct. 2021. arXiv:2110.05929 [cs].

Saining Xie, Alexander Kirillov, Ross Girshick, and Kaiming He. Exploring Randomly Wired Neural Networks for Image Recognition. 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 1284–1293, Seoul, Korea (South), Oct. 2019. IEEE.