Models of Memory, Brain and Cognitive Society

A. Kedarnath [1] Amay Bhargava [2] Nilay Beniwal [3] Roylan Pais [4] Jaswant Kumar [5] Falguni Yadav [6] Akshay Mehta [7]
Prajwal Arya [8] Deepalok Kaushik [9] Amartya S. Das [10] Aditya Prakash [11]

1.INTRODUCTION

Human memory can store large amounts of information. Nevertheless, recalling is often a challenging task. In this project we look at the past models of memory retrieval like hopfield network. After the classical models, we also develop a more realistic sequential neural network model for recall task(for eg. NTM, MANN). After that we will try to optimize and develop our own models of memory. So we first implemented the Hopfield model and then divided into two teams one on NTM and other on MANN.

2.HOPFIELD NETWORK

- For the revival of neural networks in the early 1980s, Hopfield played a pivotal role. This article mainly explains the author's understanding and experience of the Hopfield neural network, primarily based on its Python implementation.
- We generate some random patterns of a square matrix, and our Hopfield network will retrieve one of the patterns when it is given one of the patterns with noise.
- Result from one of our implementations :-

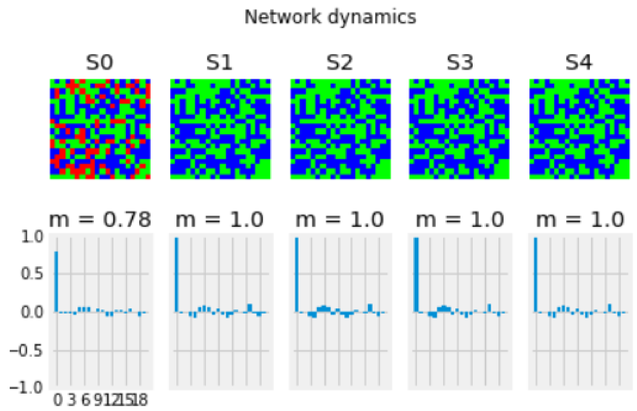

Figure 1. Evolution of pattern

3.1.NEURAL TURING MACHINE (NTM)

- NTMs are an example of MANNs, a new class of recurrent neural networks that decouples computation from memory that introduces an external memory unit resulting from which the model is differentiable and capable of being trained end-to-end.
- NTMs include a small collection of memory registers where the computation takes place and external memory storage accessible through read and write heads. A feedforward or a recurrent network can be used as a controller. A neural network controller is trained to generate a suitable output from an input sequence and data stored in memory by performing read/write operations.

3.2.TASKS

1. **Copy:** The NTM's ability to store and recall a long sequence of arbitrary information. We are giving random batches of "bits" sequences, ending with a delimiter as an input to our model. All the sequences within each batch have the same length. We get the copy of the input sequence without the delimiter flag.

Figure 2. Targets

Figure 3. Outputs

2. **Repeat Copy:** It is an extension of the Copy task where we require the network to copy the sequence a specific number of times and return the end of the sequence result. Basically NTM's ability to learn a simple nested function and execute a "for loop" is tested in the repeat copy task.
3. **Associative Recall:** Here NTM's capability for learning with another level complexity when one data item points to another by constructing a list so that we demand the network to return subsequent items when querying for one item.
4. **Priority Sort:** The NTM's ability to sort an input sequence is tested in the priority sort task. 20 random binary vectors corresponding with a scalar priority in the range [-1,1] are given as input. The expected output is the 16 input vectors with the highest priority are sorted by respective priority dataloader.

3.3.RESULTS

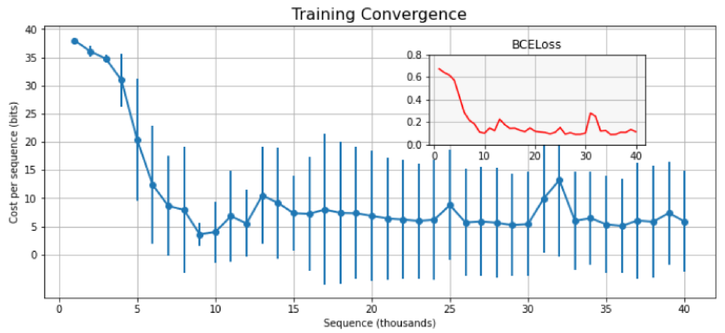

Figure 4. Training Convergence

4.1.MEMORY AUGMENTED NEURAL NETWORK(MANN)

- The idea of Meta Learning is that to allow a Neural Network to learn across previous tasks and to accomplish a new unseen task. Memory Augmented Neural Network (MANN) is one of them which inspired the use of external memory from Neural Turing Machine.
- MANN use two different weight vectors to perform read and write operations. Read operations in MANN are the same as NTM.
- The communication between Memory Module and the Controller Network is done by the Read Write Heads where while reading, the representations from memory is retrieved and while writing, a new memory is encoded.

4.2.MANN Components

Our MANN build should cover the state of Neural Network and the Neural Turing Machine output for next batch iteration processing. Our MANN State include

- Controller State,
- Read Vector List,
- Read Weight Vector List,
- Write Weight Vector List,
- Usage Weight Vector,
- Memory Matrix (updated given some specific time and addressing concern)

4.3.Results

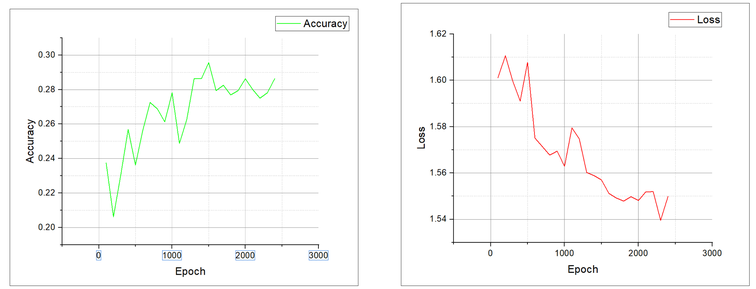

Figure 5. Accuracy and Loss Convergence

REFERENCES

1. https://arxiv.org/pdf/1807.08518.pdf
2. https://github.com/loudinthecloud/pytorch-ntm
3. http://proceedings.mlr.press/v48/santoro16.pdf
4. https://github.com/ash3n/One-shot-Memory-Augmented-NN

https://github.com/karthikedar/NTM

https://github.com/asd2204/MANN-Test