CPSC 5616: Robot Modelling Using LSTMs and BP

Instructor:

Dr. Meysar Zeinali

Presented by:

Haokun Zhang, 0424660, hzhang12@laurentian.ca

Pengyu Wang, 0425157, pwang2@laurentian.ca

Ziping Zhu, 0422426, zzhu4@laurentian.ca

- Apr. 06. 2023

Outline

1. Purpose
2. Review of BP/MLP
3. Introductions of RNN→LSTM
4. Walk through LSTMs
5. Works flow
6. Conclusion

Robot Modelling

Robot models simulate the **kinematic** and **dynamic** properties of manipulator robots and other rigid body systems. The models are ***rigidBodyTree*** objects containing ***rigidBody*** and ***rigidBodyJoint*** elements with joint transformations and inertial properties.

——MathWorks, Robot Models, R2023a

Backpropagation (BP) and Multi-Layer Perceptron (MLP)

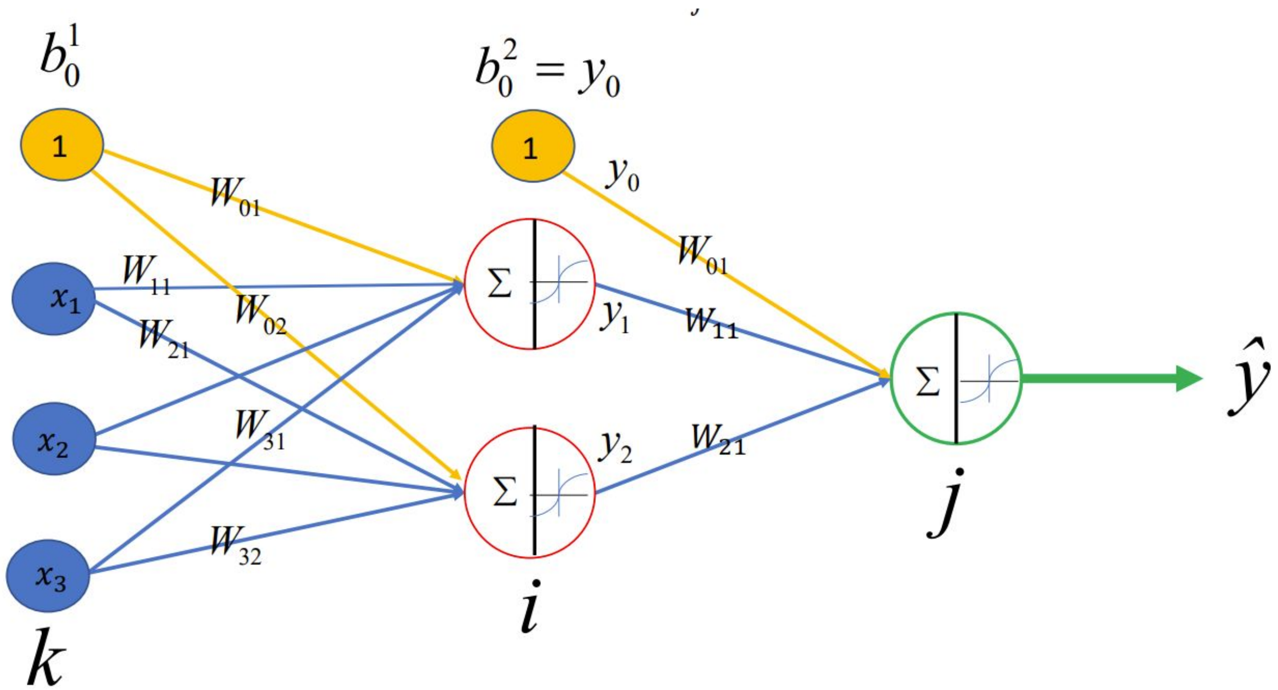


Fig. 2: CPSC 5616, Meysar Zeinali

What’s RNN (Recurrent Neural Networks)?

A family of neural networks that:

- Take sequential input of any length; apply the same weights on each step
- Can optionally produce output on each step

——Standford University, CS244N, Lecture 6: LSTM RNNs and Neural Machine Translation

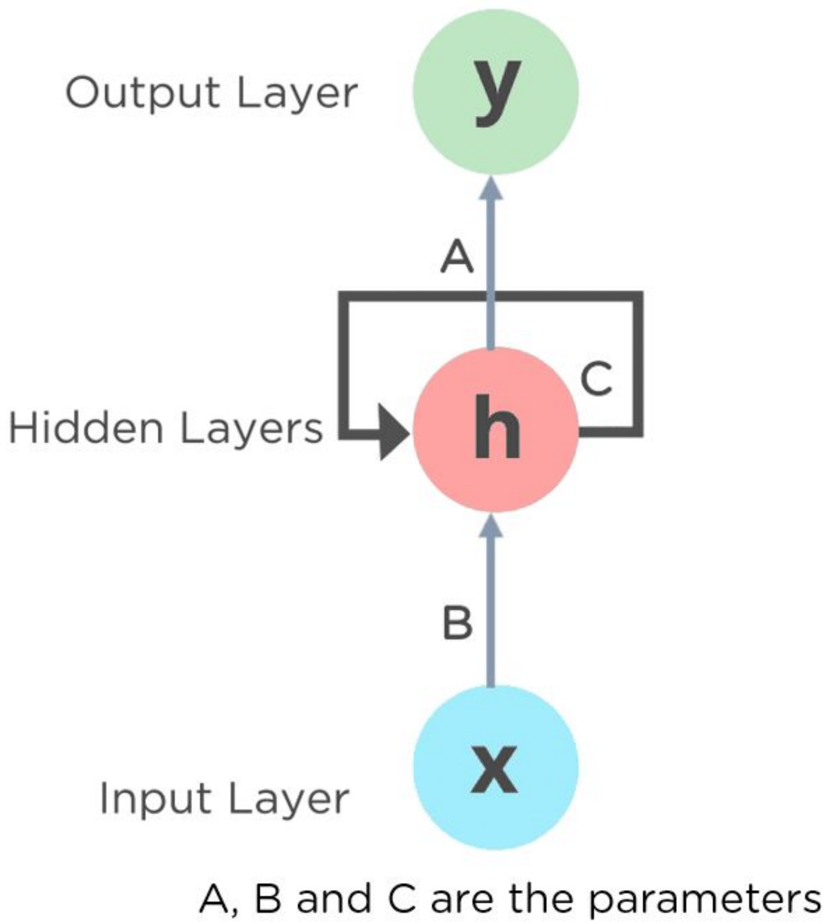
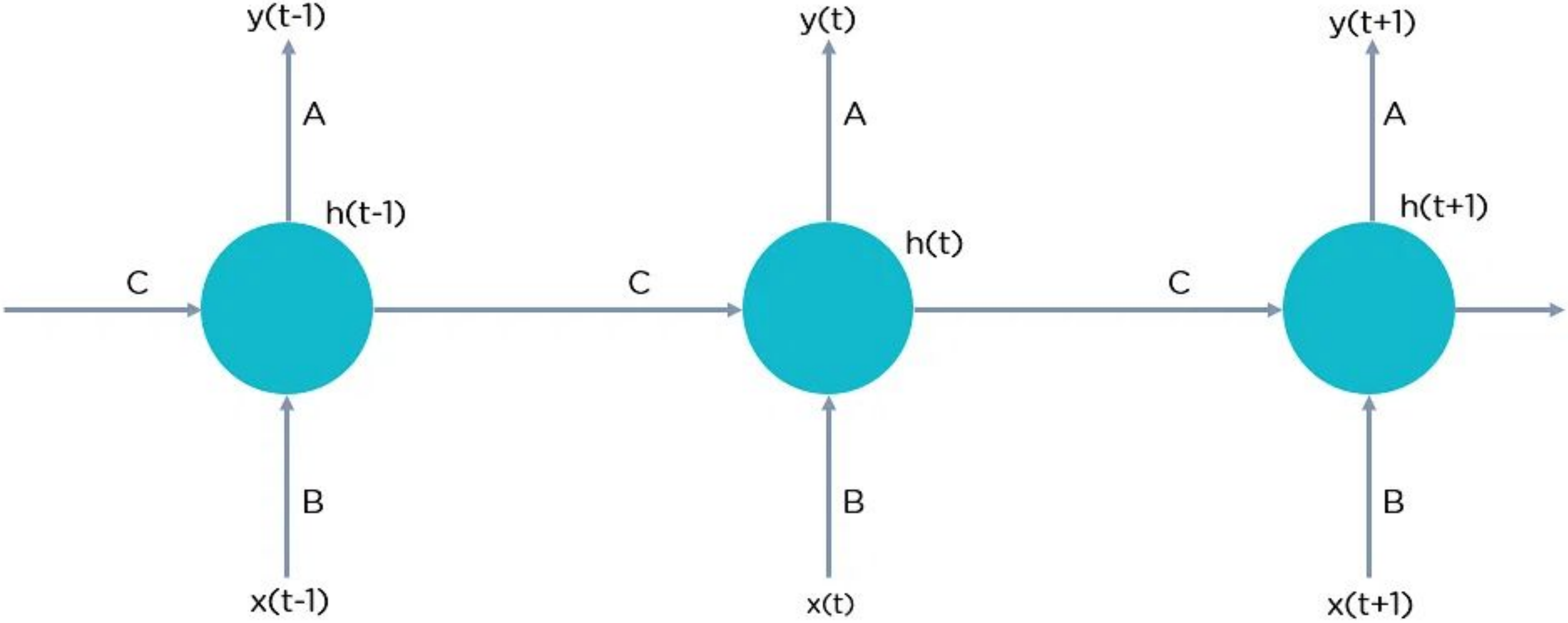


Fig. 4: https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn



$$h(t) = f_c\,(h(t-1), x(t))$$

$h(t)$ = new state
f_c = function with parameter c
$h(t-1)$ = old state
$x(t)$ = input vector at time step t

Fully connected Recurrent Neural Network

Fig. 5: https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn

LSTMs - Long Short-Term Memory Networks

Long short-term memory is an artificial neural network used in the fields of artificial intelligence and deep learning. Unlike standard feedforward neural networks, LSTM has feedback connections. Such a recurrent neural network can process not only single data points, but also entire sequences of data.

Understanding LSTM

Symbolics/Notations meanings

f_t: Forget gate output

i_t: Input gate output

$\hat{C}_t$: New candidate values

C_t: New cell state

C_{t-1}: Previous cell state

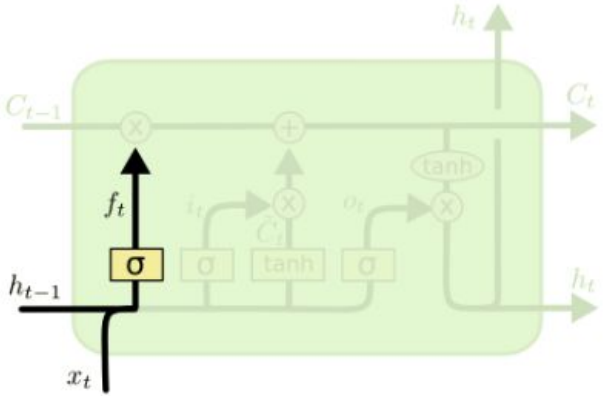
o_t: Output gate's output

h_t: Hidden state

x_t: current input

Walk through LSTMs

Forget gate



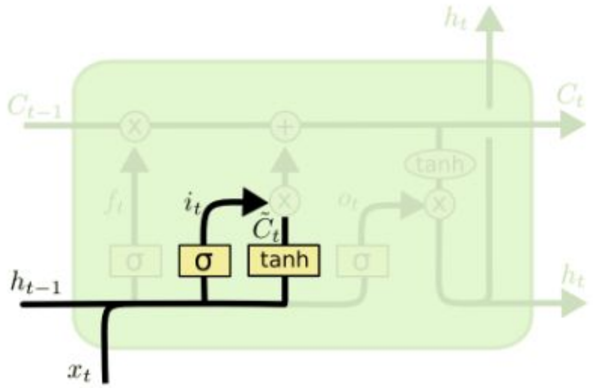
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$

It looks at $\boldsymbol{h}_{t-1}$ and $\boldsymbol{x}_{t}$, and outputs a number between **0** and **1** for each number in the cell state C_{t-1}. A **1** represents “completely keep this” while a **0** represents “completely get rid of this.”

Walk through LSTMs

Input gate & New candidate values

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

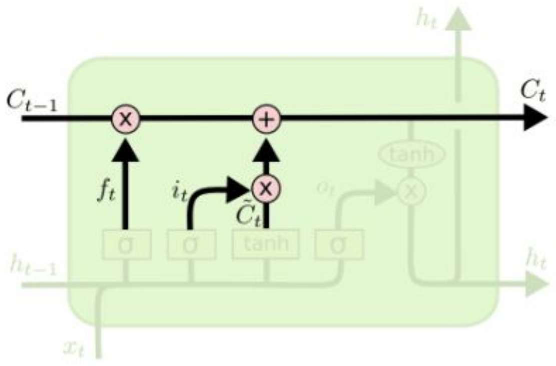


This has two parts. First, a sigmoid layer called the “input gate layer” decides which values we’ll update. Next, a tanh layer creates a vector of new candidate values, $\hat{C}_t$ that could be added to the state. In the next step, we’ll combine these two to create an update to the state.

Walk through LSTMs

Cell State

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

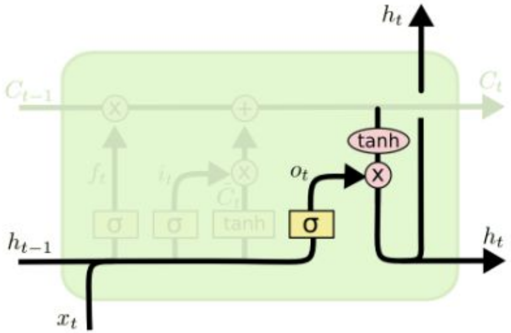


It's now time to update the old cell state, C_{t-1}, into the new cell state C_t. The previous steps already decided what to do, we just need to actually do it. We multiply the old state by $\boldsymbol{f_t}$, forgetting the things we decided to forget earlier. Then we add $i_t * \hat{C}_t$. This is the new candidate values, scaled by how much we decided to update each state value.

Walk through LSTMs

Output gate

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



Finally, we need to decide what we're going to output. This output will be based on our cell state, but will be a filtered version. First, we run a sigmoid layer which decides what parts of the cell state we're going to 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, so that we only output the parts we decided to.

Implementation & Dataset

Provided by Dr. Meysar Zeinali, named as Robot Dataset_with_6 inputs and 2 Outputs.xlsx

GitHub

MATLAB

1. 6 Inputs = 2 arms * 3
    a. P: position
    b. A: Acceleration
    c. V: Velocity
2. 2 Outputs
    a. Torque value, it is required to follow the desired trajectory.

—— Deep Learning-based Robot Control using Recurrent Neural Networks (LSTM; GRU) and Adaptive Sliding Mode Control, Patel.R, Zeinali. M, & Passi. K

What parameters are?

There are 3 parts of datasets, 60% for training, 20% for validation, 20% for testing.

Learning rate(Eta): 0.0001

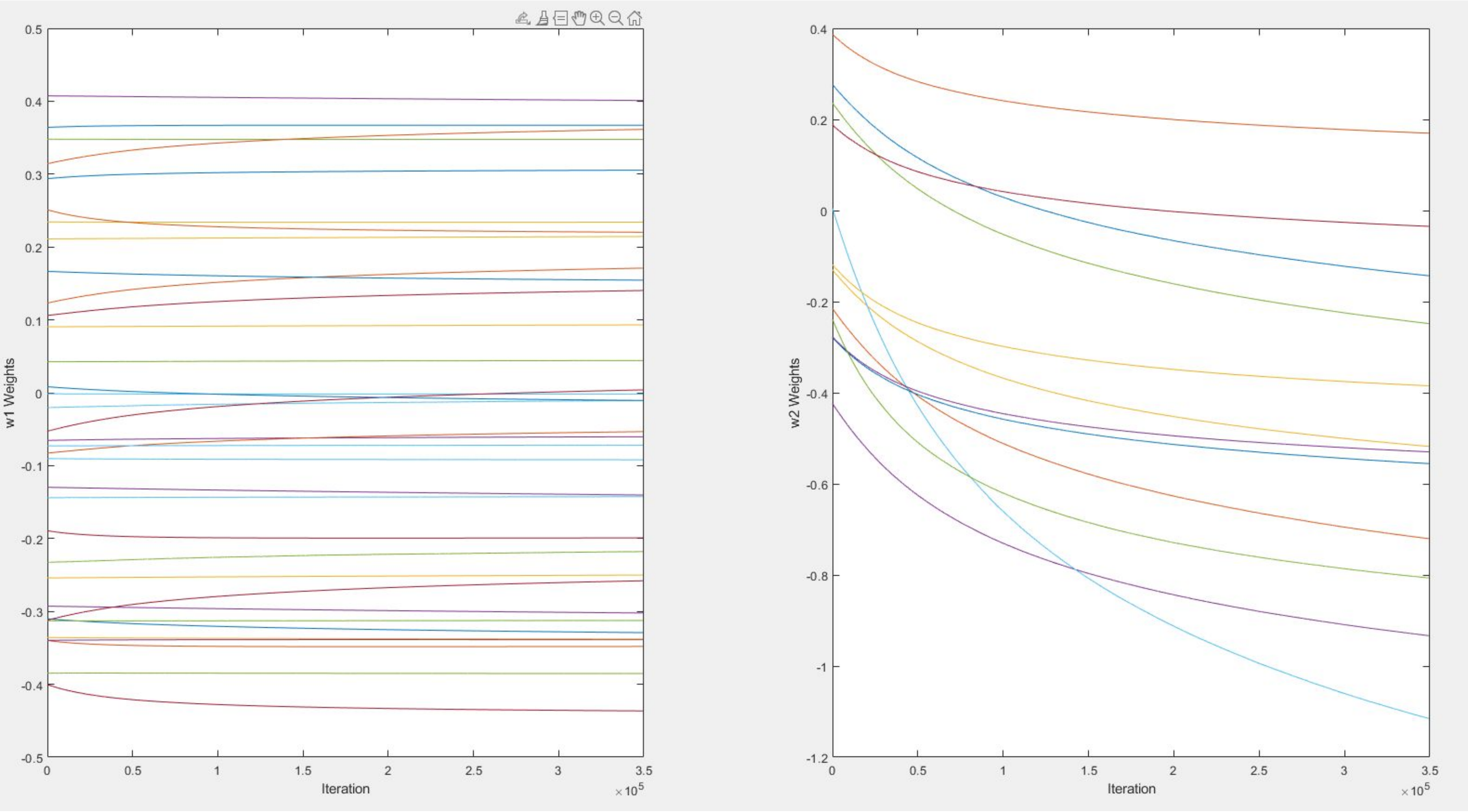
BIAS: 1

Neurons in hidden layer: 5

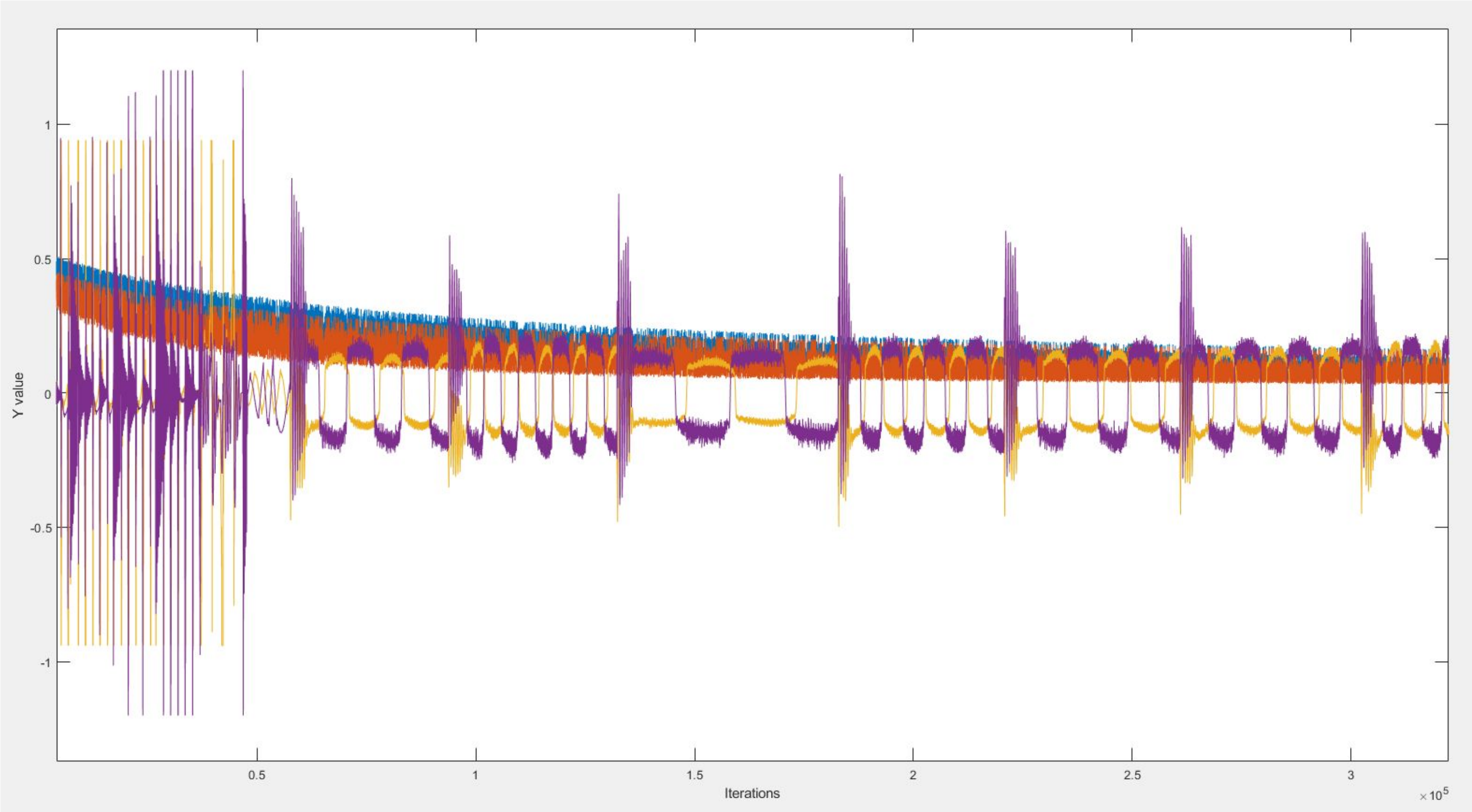
Layers: 1

```
L= 6; % 6 inputs
N = 2; % 2 outputs
M = L-1; % % neurons, Range = 1 to L, Best = 2/3*L+N or L-1

BIAS = 1;
ETA = 0.0001; % 0.1<ETA<0.4
```

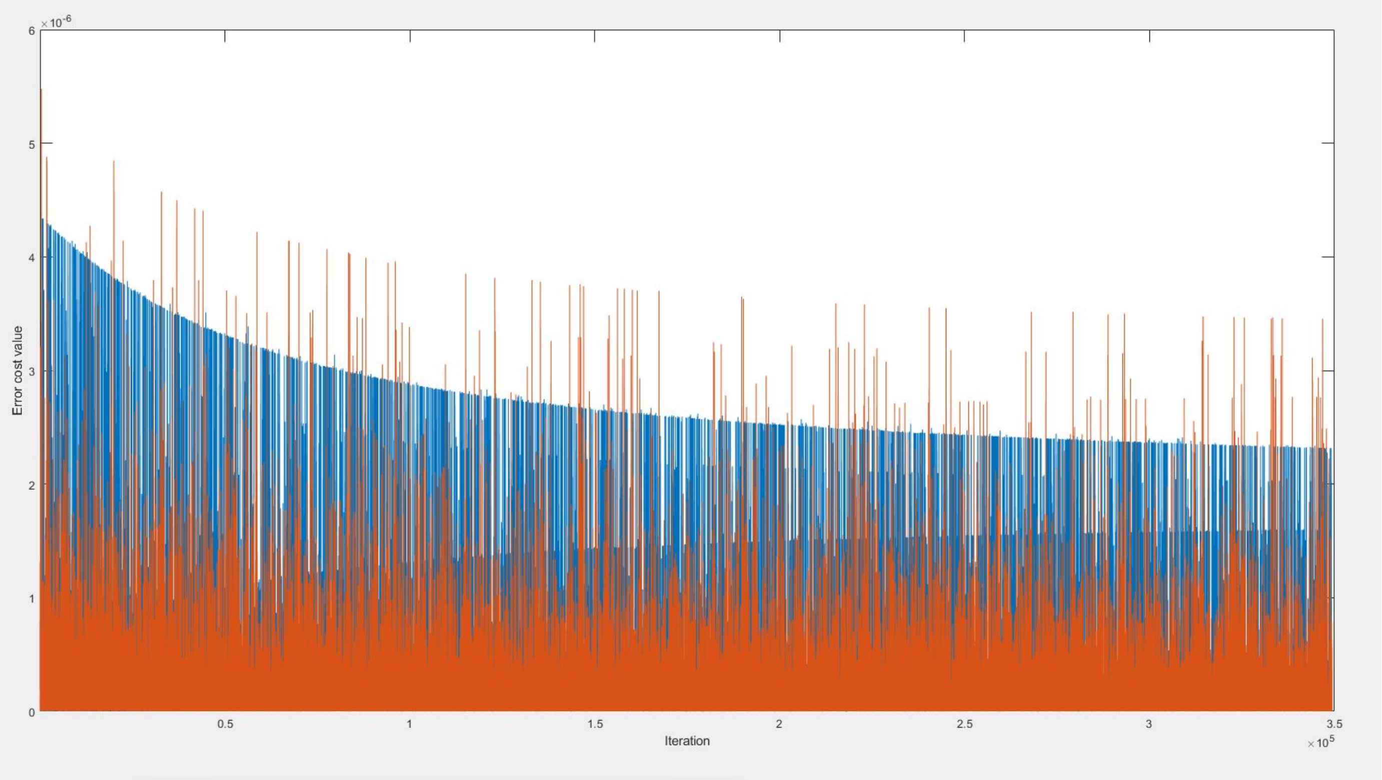


BP/MLP Weights of w_1 and w_2



MLP/BP Y values

BP/MLP Errors Cost Function Value



What parameters are?

There are 3 parts of datasets, 60% for training, 20% for validation, 20% for testing.

Learning rate(Eta): 0.0001

BIAS: 1

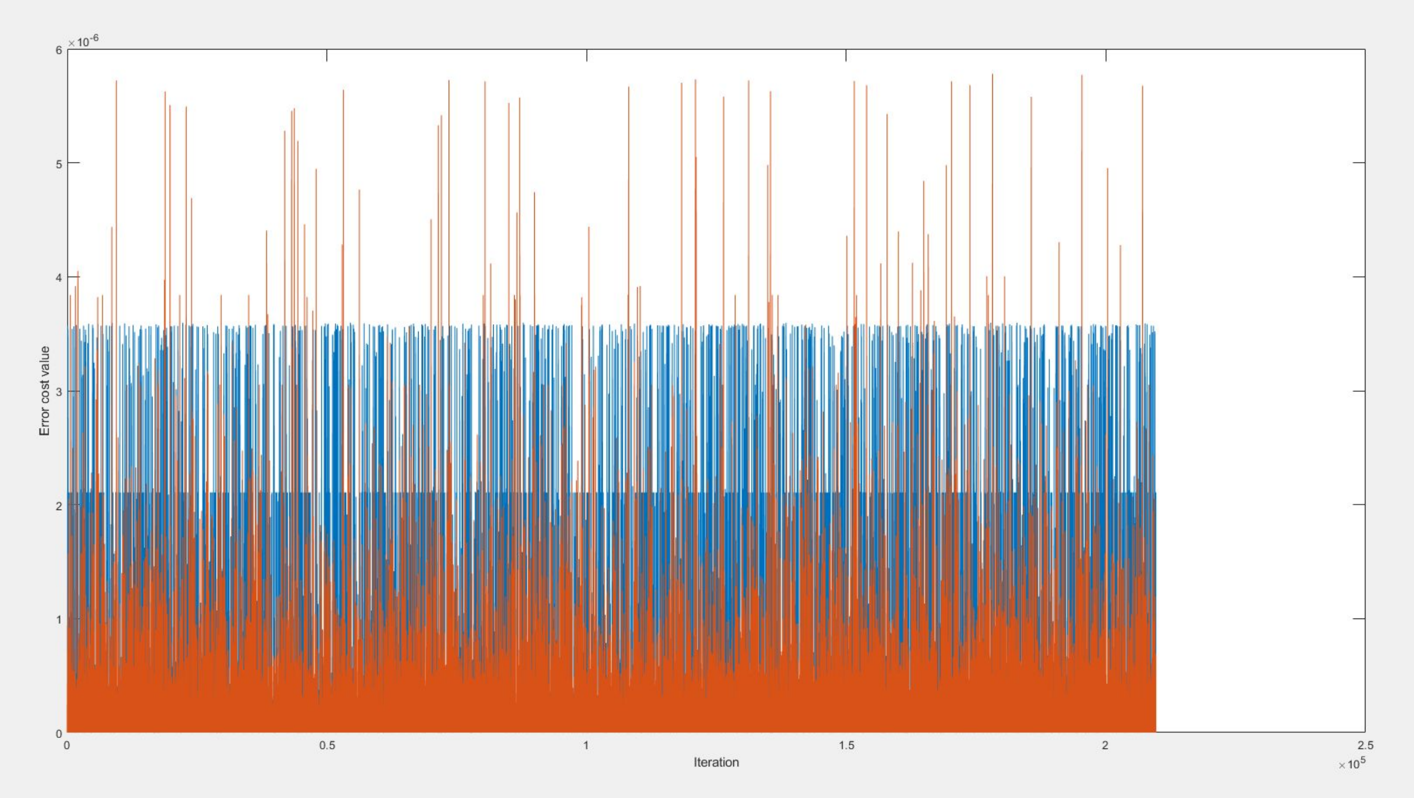
Neurons in hidden layer: 5

Layers: 1

```
neurons = IN-1; % % neurons, Range = 1 to L, Best = 2/3*L+N or L-1
Eta = 0.0001;
Bias = 1;

E = lstm(dataset,TRAIN,IN,neurons,OUT,Bias,Eta);
```

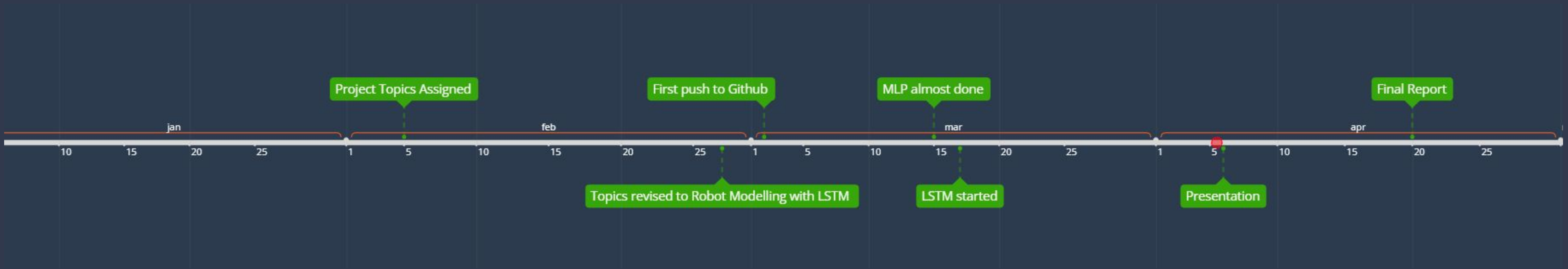
RESULTS of LSTM Errors Cost Function Value



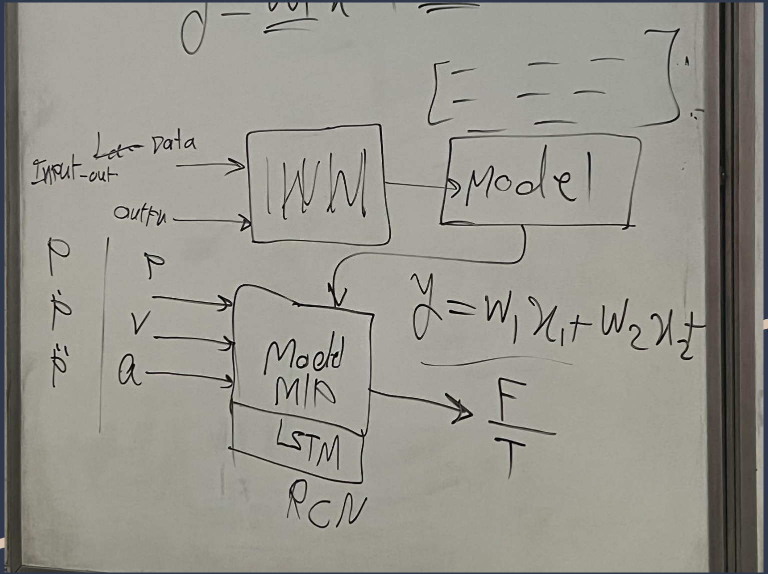
70%

We've done 70% of the whole project so far.

Some issues need to be fixed, and working on final report…



References



1. Patel, Raj & Zeinali, Meysar & Passi, Kalpdrum. (2021). Deep Learning-based Robot Control using Recurrent Neural Networks (LSTM; GRU) and Adaptive Sliding Mode Control. 10.11159/cdsr21.113.
2. Manning. C.(2023)*Natural Language Processing with Deep Learning, Lecture 6: LSTM RNNs and Neural Machine Translation* [PowerPoint slides]. https://web.stanford.edu/class/cs224n/slides/cs224n-2022-lecture06-fancy-rnn.pdf
3. Zeinali. M. (2023) *CPSC 5616 Machine and Deep Learning, Lecture 3, Chapter 3: Artificial Neural Network, Neuron, Perceptron, and Backpropagation Algorithm*[PowerPoint slides]
4. Biswal, A. (2023, February 14). Recurrent Neural Network(RNN) Tutorial: Types, Examples, LSTM and More. Simplilearn.com. https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn

Thank you!