LaurentianUniversity
UniversitéLaurentienne

CPSC 5616: Robot Modelling Using LSTMs and BP

Presented by:

Haokun Zhang, 0424660, hzhang12@laurentian.ca

Pengyu Wang, 0425157, pwang2@laurentian.ca

Ziping Zhu, 0422426, zzhu4@laurentian.ca

Instructor:

Dr. Meysar Zeinali

- 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

How to model robot’s behaviour?

Atlas Gets a Grip | Boston Dynamics

Fig. 1: https://seekingalpha.com/news/3914254-sony-says-it-has-technology-to-make-humanoid-robots-still-looking-for-use-case-report

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)

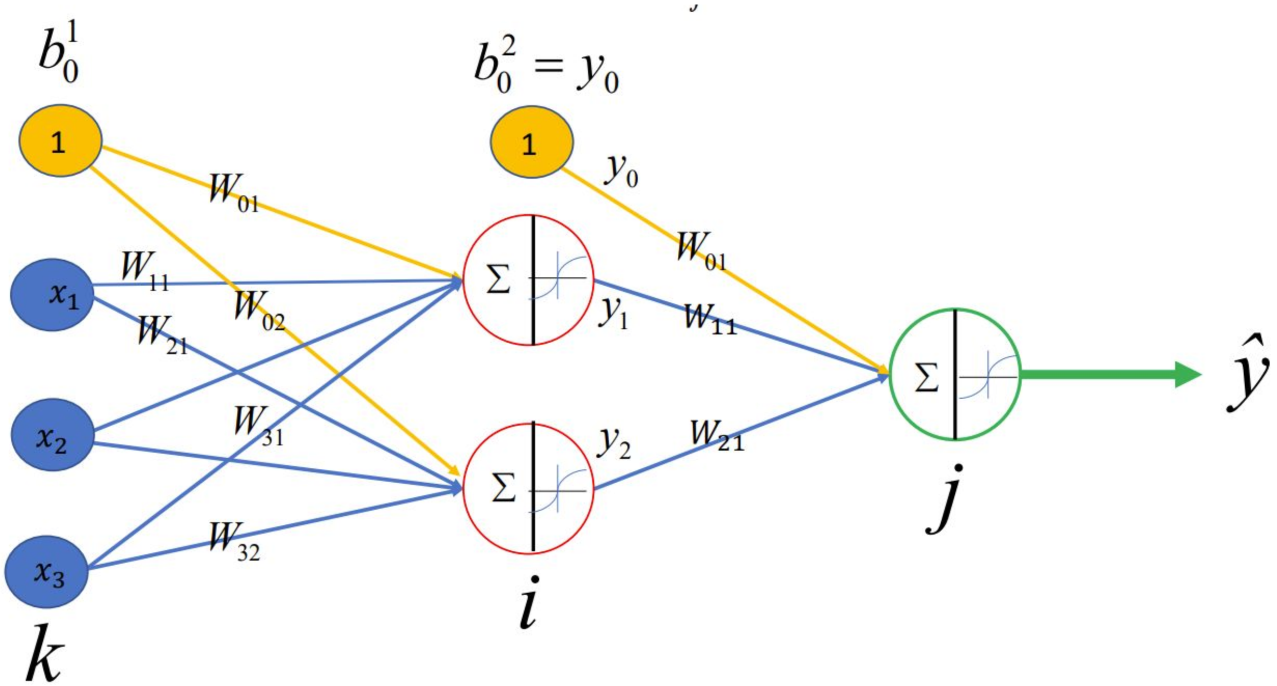


Fig. 2: CPSC 5616, Meysar Zeinali

(Cont.) Backpropagation (BP) and Multilayer Perceptron (MLP)

$$net^1 = W^1 x + b^1 \qquad (1)$$

$$y^1 = \sigma(net^1) \qquad (2)$$

$$net^2 = W^2 y^1 + b^2 \qquad (3)$$

$$y^2 = \sigma(net^2) \qquad (4)$$

$$net^3 = W^3 y^2 + b^3 \qquad (5)$$

$$\hat{y} = \sigma(net^3) \qquad (6)$$

$$E = \frac{1}{2}(y_d - \hat{y})^2 \qquad (7)$$

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

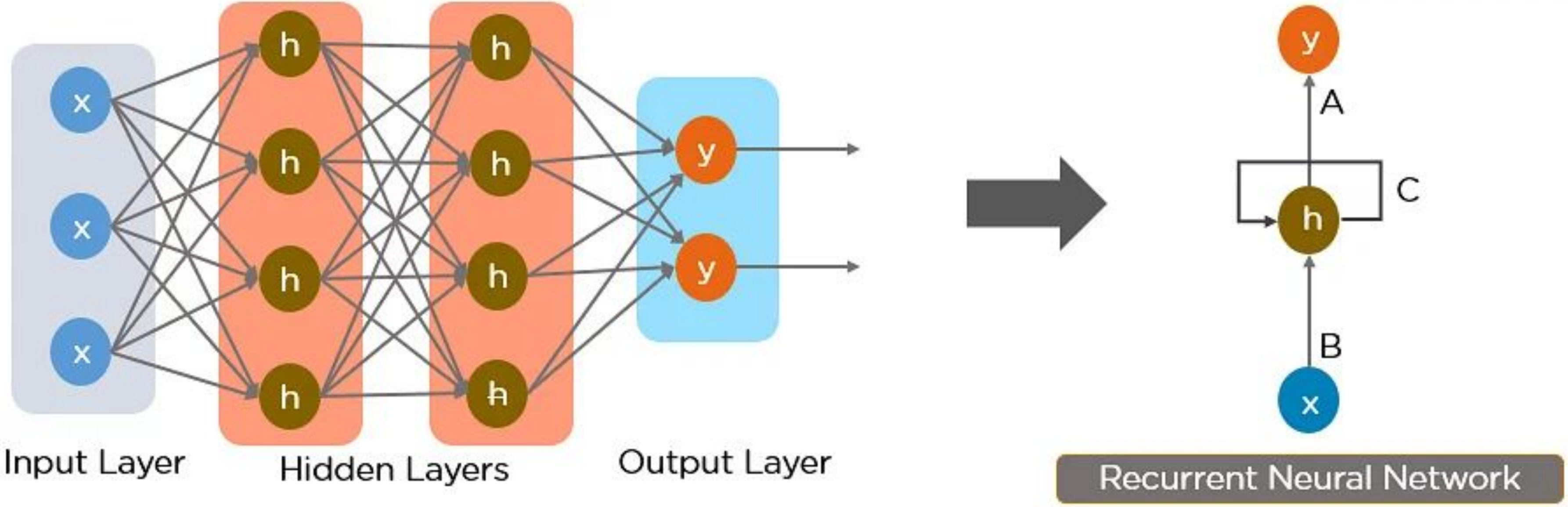
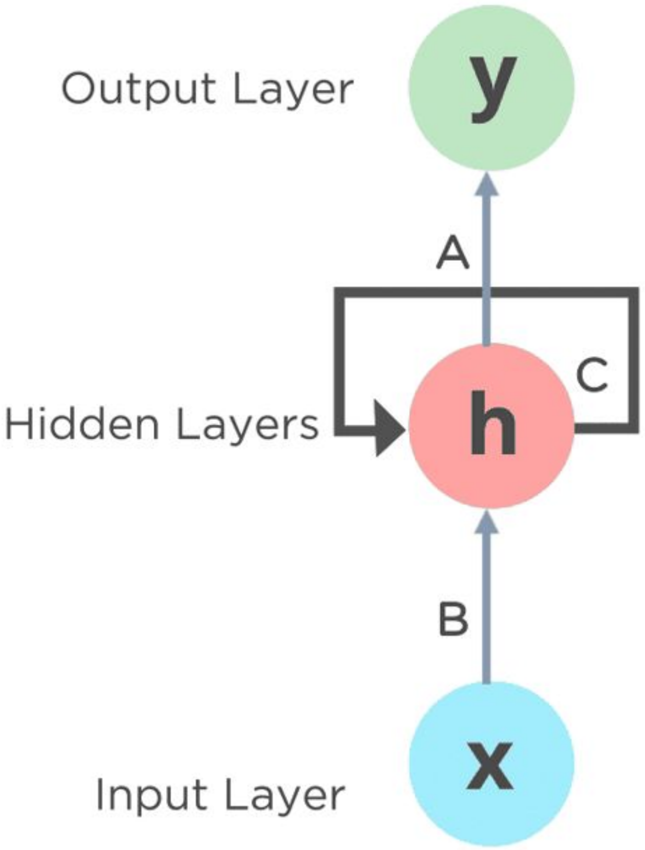
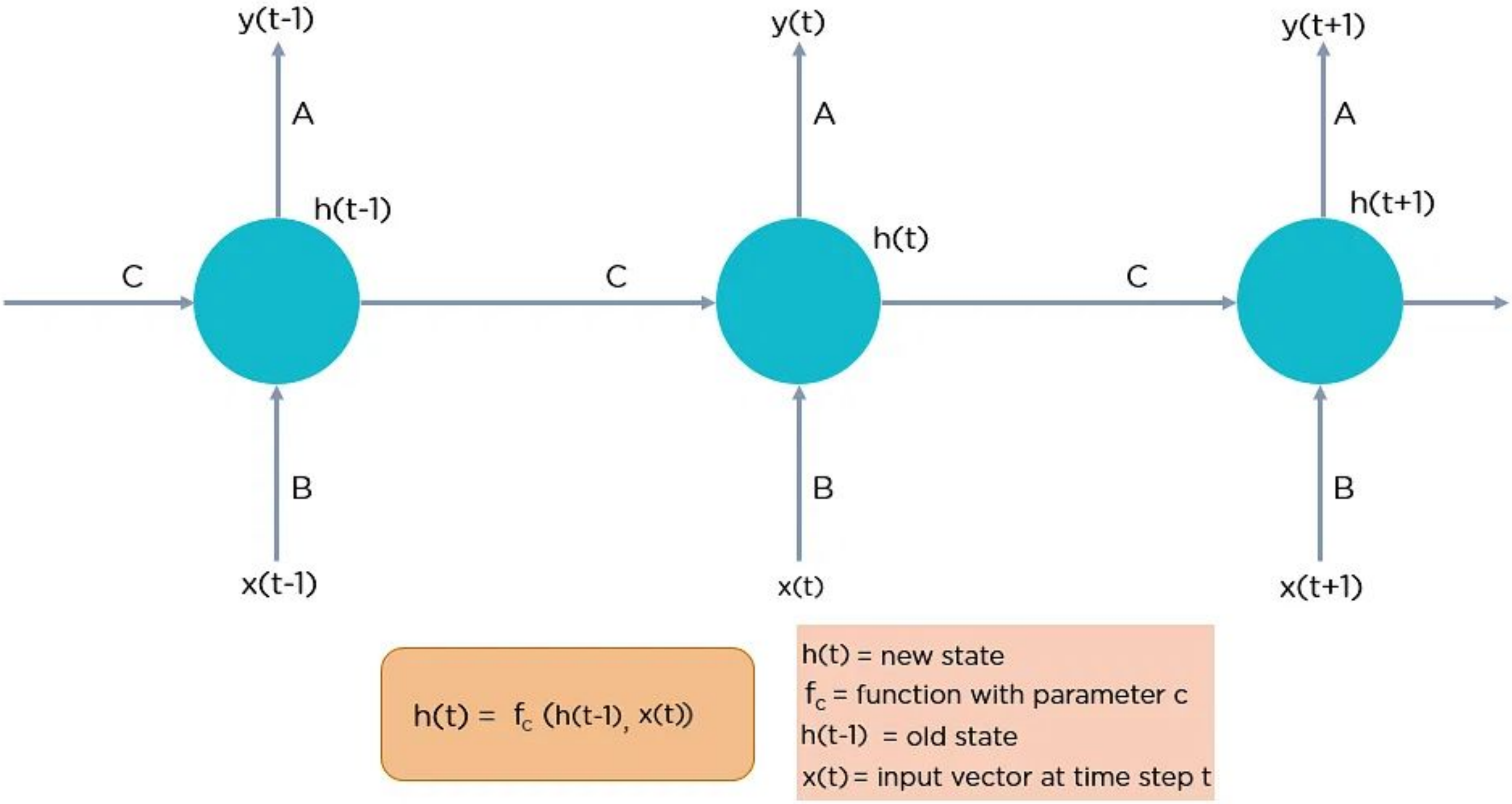


Fig. 3: https://www.simplilearn.com/tutorials/deep-learning-tutorial/rnn



A, B and C are the parameters

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



Fully connected Recurrent Neural Network

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

RNN vs. CNN

RNN (Recurrent Neural Network)

1. Less feature compatibility.
2. Text and speech Analysis.
3. Temporal /sequential data
4. Arbitrary input/ output lengths.
5. RNN, unlike feed-forward neural networks- can use their internal memory to process arbitrary sequences of inputs.
6. Time-series information, what a user spoke last would impact what he will speak next.

CNN (Convolutional Neural Network)

1. More potent than RNN.
2. Images and video processing.
3. Spatial data like images.
4. Fixed-size inputs and generates fixed size outputs.
5. Type of feed-forward artificial neural network with variations of multilayer perceptron' s designed to use minimal amounts of preprocessing.
6. Use of connectivity patterns between the neurons. It is affected by the organization of the animal visual cortex, whose individual neurons are arranged in such a way that they can respond to overlapping regions in the visual field.

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.

(Cont.) LSTMs - Long Short-Term Memory Networks

1. A type of RNN proposed by Hochreiter and Schmidhuber in 1997 as a solution to the problem of vanishing gradients
   a. Everyone cites that paper but really a crucial part of the modern LSTM is from Gers et al. (2000)
2. Only started to be recognized as promising through the work of S's student Alex Graves c. 2006
   a. Work in which he also invented CTC (connectionist temporal classification) for speech recognition
3. But only really became well-known after Hinton brought it to Google in 2013
   a. Following Graves having been a postdoc with Hinton

LSTM Structure

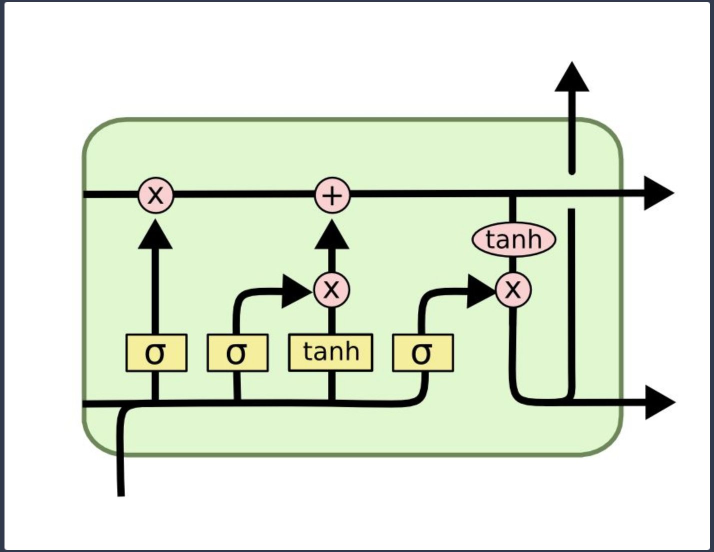


Fig. 6: https://www.google.com/url?sa=i&url=https%3A%2F%2Fcolah.github.io%2Fposts%2F2015-08-Understanding-LSTMs%2F&psig=AOvVaw0ssrwx_1f_O2CWjk4ZlVqP&ust=1680567188276000&source=images&cd=vfe&ved=0CBAQjRxqFwoTCOD5luS2jP4CFQAAAAAdAAAAABAR

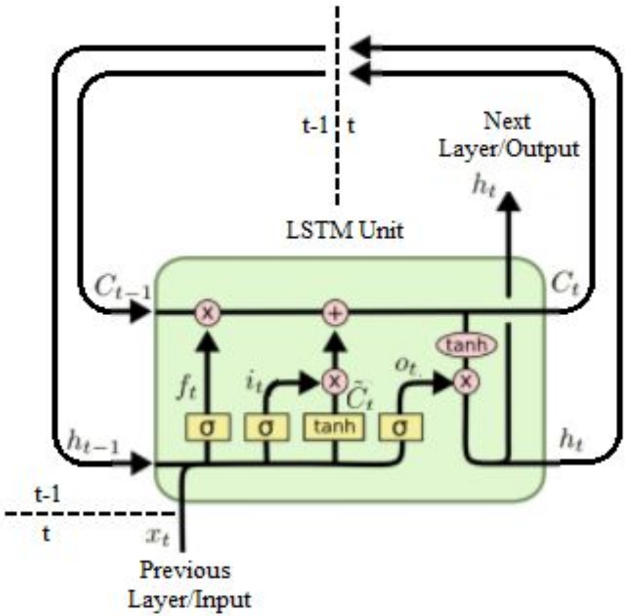
3 Sigmoid activation function

+

1 tanh activation function

1. Forget gate
2. Input gate
3. Cell state
4. Output gate

Understanding LSTM Networks



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \ + \ b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$
$$o_t = \sigma\left(W_o \ [h_{t-1}, x_t] \ + \ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

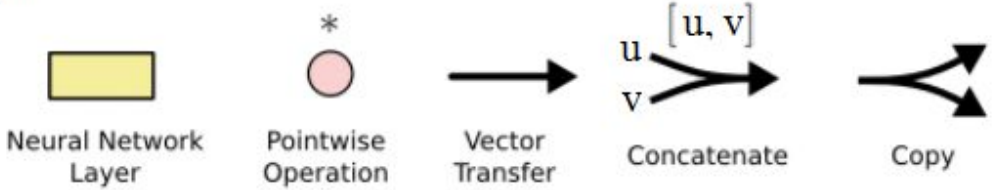


Fig. 5: https://www.google.com/url?sa=i&url=https%3A%2F%2Fcolah.github.io%2Fposts%2F2015-08-Understanding-LSTMs%2F&psig=AOvVaw0ssrwx_1f_O2CWjk4ZlVqP&ust=1680567188276000&source=images&cd=vfe&ved=0CBAQjRxqFwoTCOD5luS2jP4CFQAAAAAdAAAAABAR

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

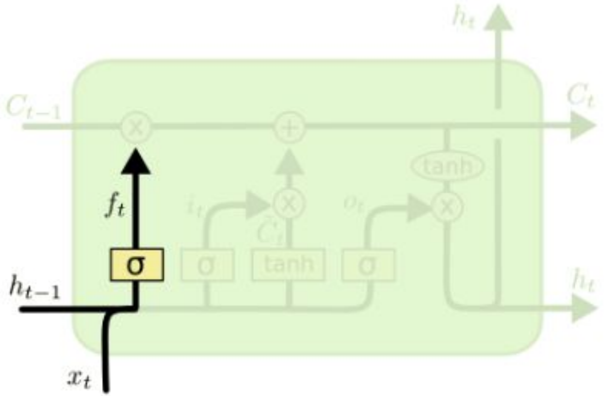
o_t: Output gate's output

h_t: Hidden state

x_t: current input

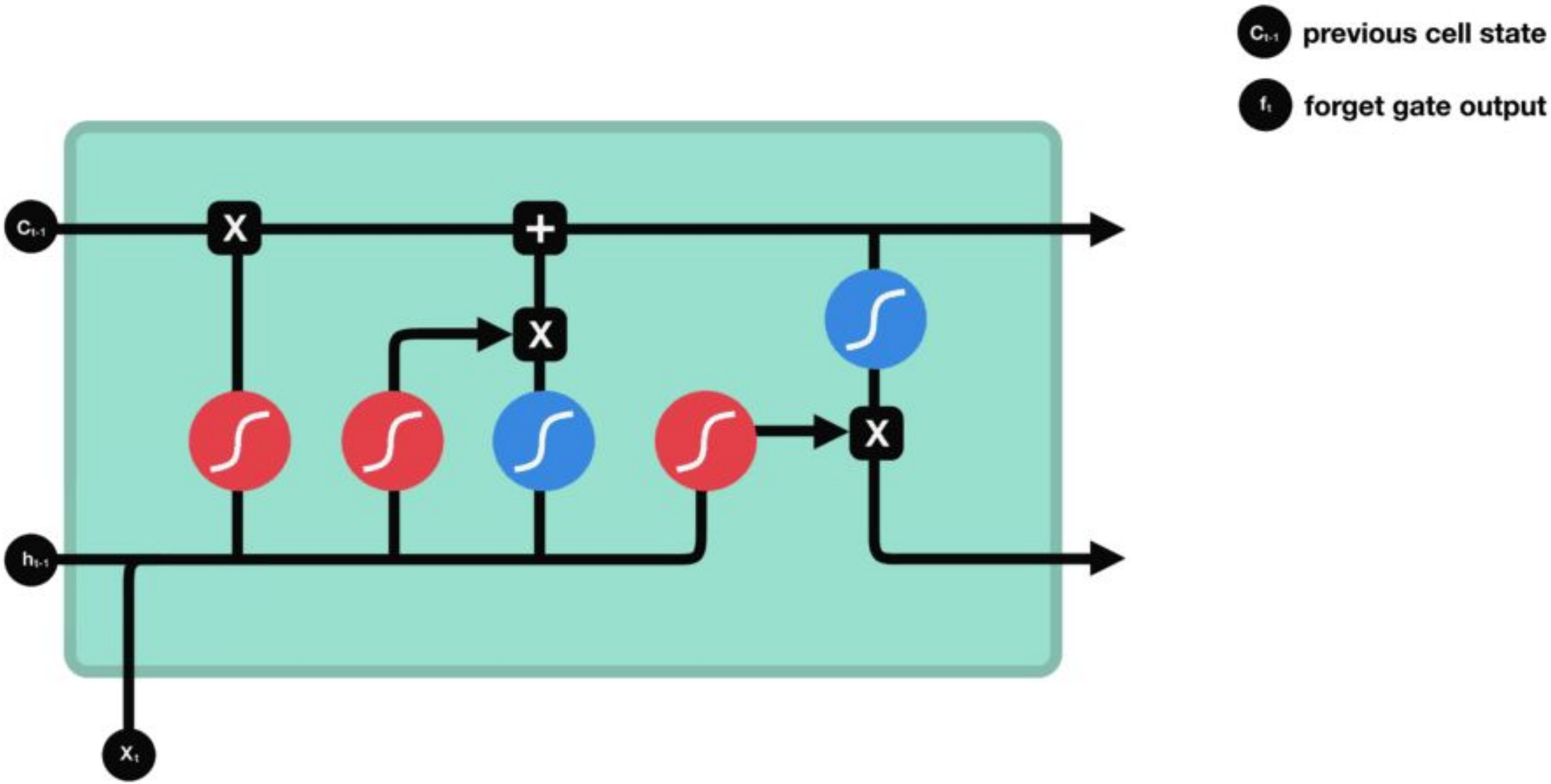
Walk through LSTMs

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.”

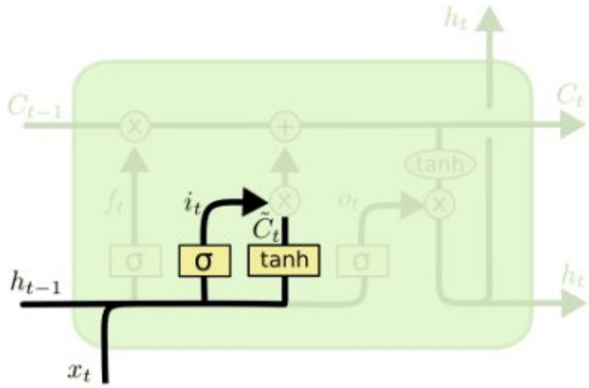


Forget Gate

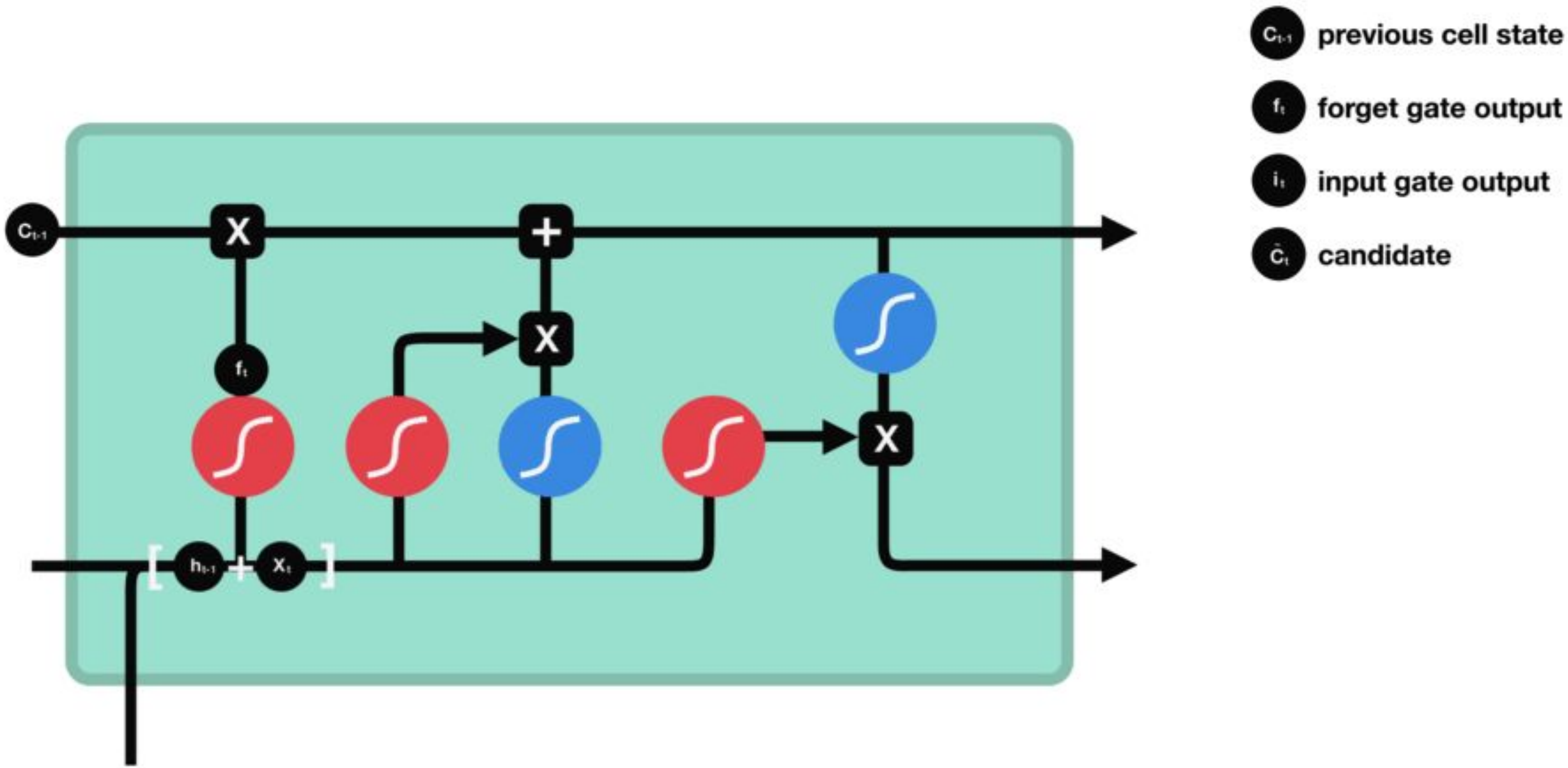
Walk through LSTMs

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.



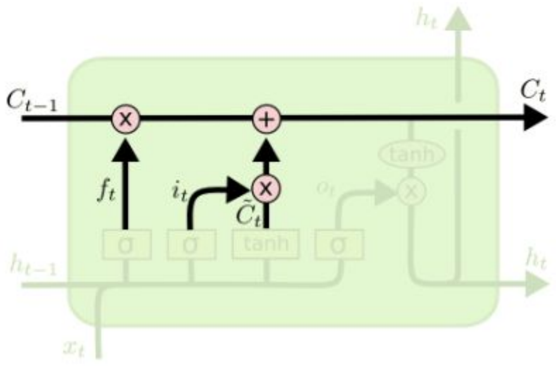
previous cell state
forget gate output
input gate output
candidate

Input Gate

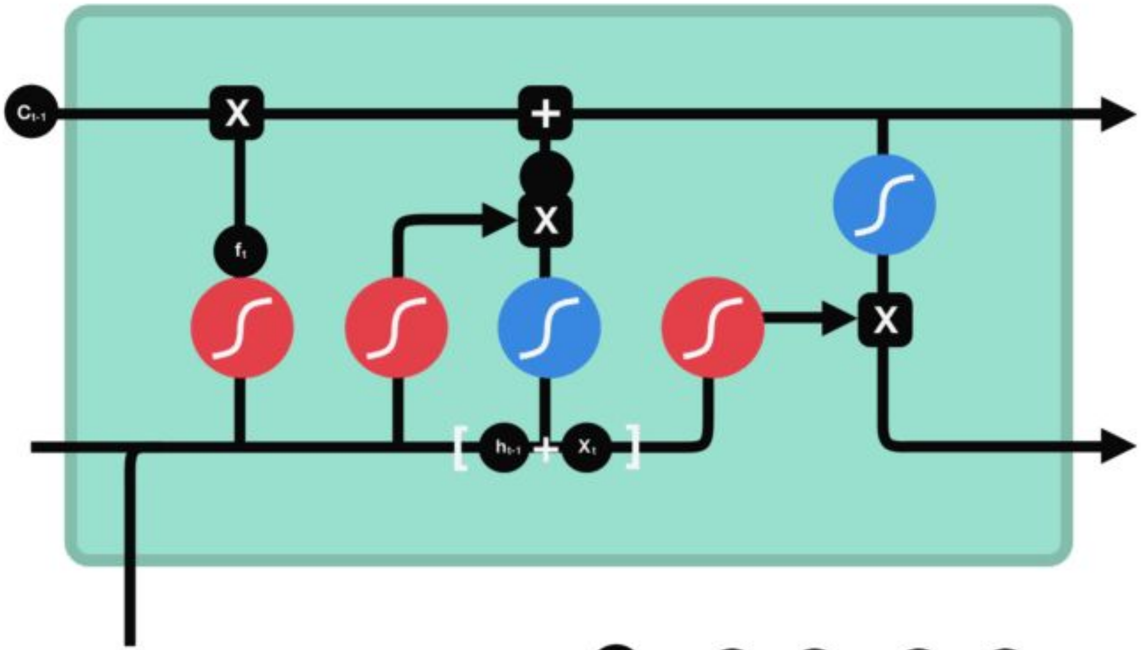
Walk through LSTMs

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.



- C_{t-1} previous cell state
- f_t forget gate output
- i_t input gate output
- $\tilde{C}_t$ candidate
- C_t new cell state

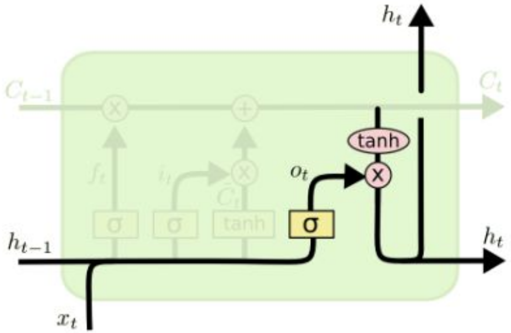
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Cell State

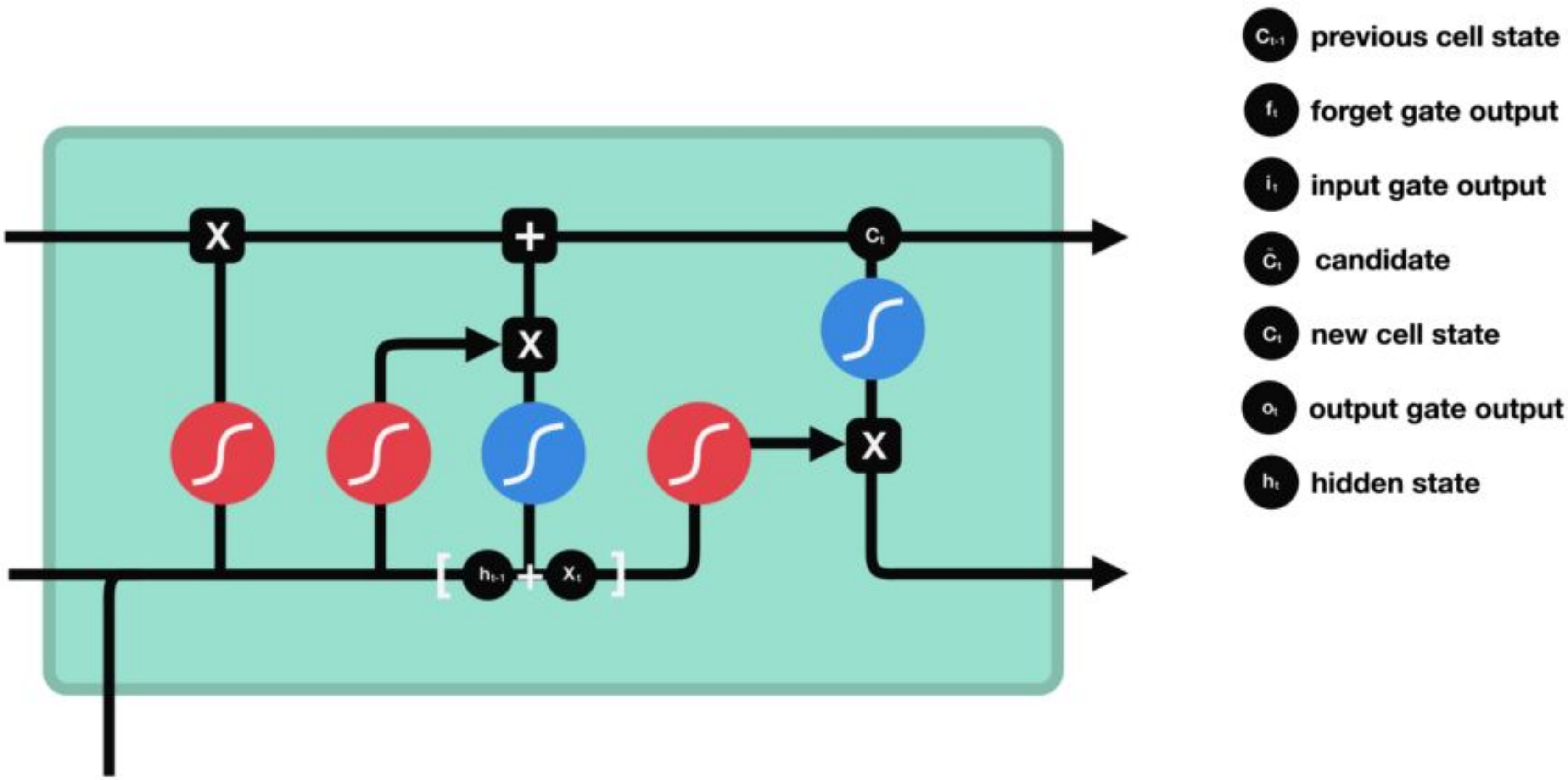
Walk through LSTMs

Output gate

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



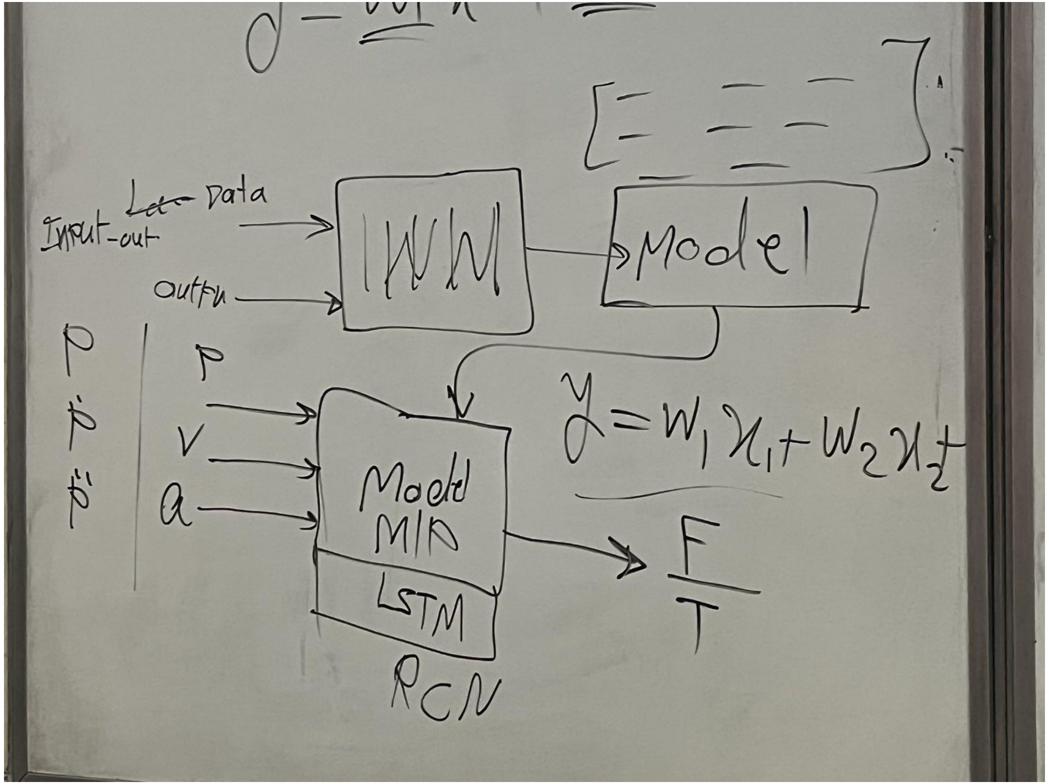
Finally, we need to decide what we're going to output. This output will be based on our cell state, but will be a filtered version. First, we run a sigmoid layer which decides what parts of the cell state we're going to output. Then, we put the cell state through **tanh** (to push the values to be between **-1** and **1**) and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to.



X
+
C_t
X
X
h_{t-1}
X_t
C_{t-1} previous cell state
f_t forget gate output
i_t input gate output
$\tilde{C}_t$ candidate
C_t new cell state
o_t output gate output
h_t hidden state

Output Gate

Prof's Explanation🤩



Implementation
&
Dataset

Dataset provided by Dr. Meysar Zeinali,

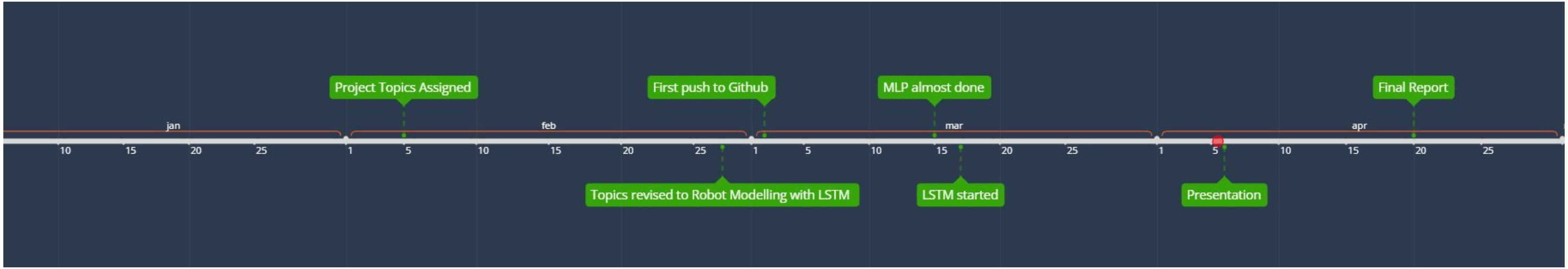
named as ***Robot Dataset_with_6 inputs and 2 Outputs.xlsx***

Dataset Description

Robot Dataset_with_6 inputs and 2 Outputs.xlsx

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


Project Topics Assigned
First push to Github
MLP almost done
Final Report
jan
feb
mar
apr
10
15
20
25
1
5
10
15
20
25
1
5
10
15
20
25
1
5
10
15
20
25
Topics revised to Robot Modelling with LSTM
LSTM started
Presentation

Milestones

RESULTS of MLP/BP

What parameters are?

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

Learning rate(Eta): 0.2

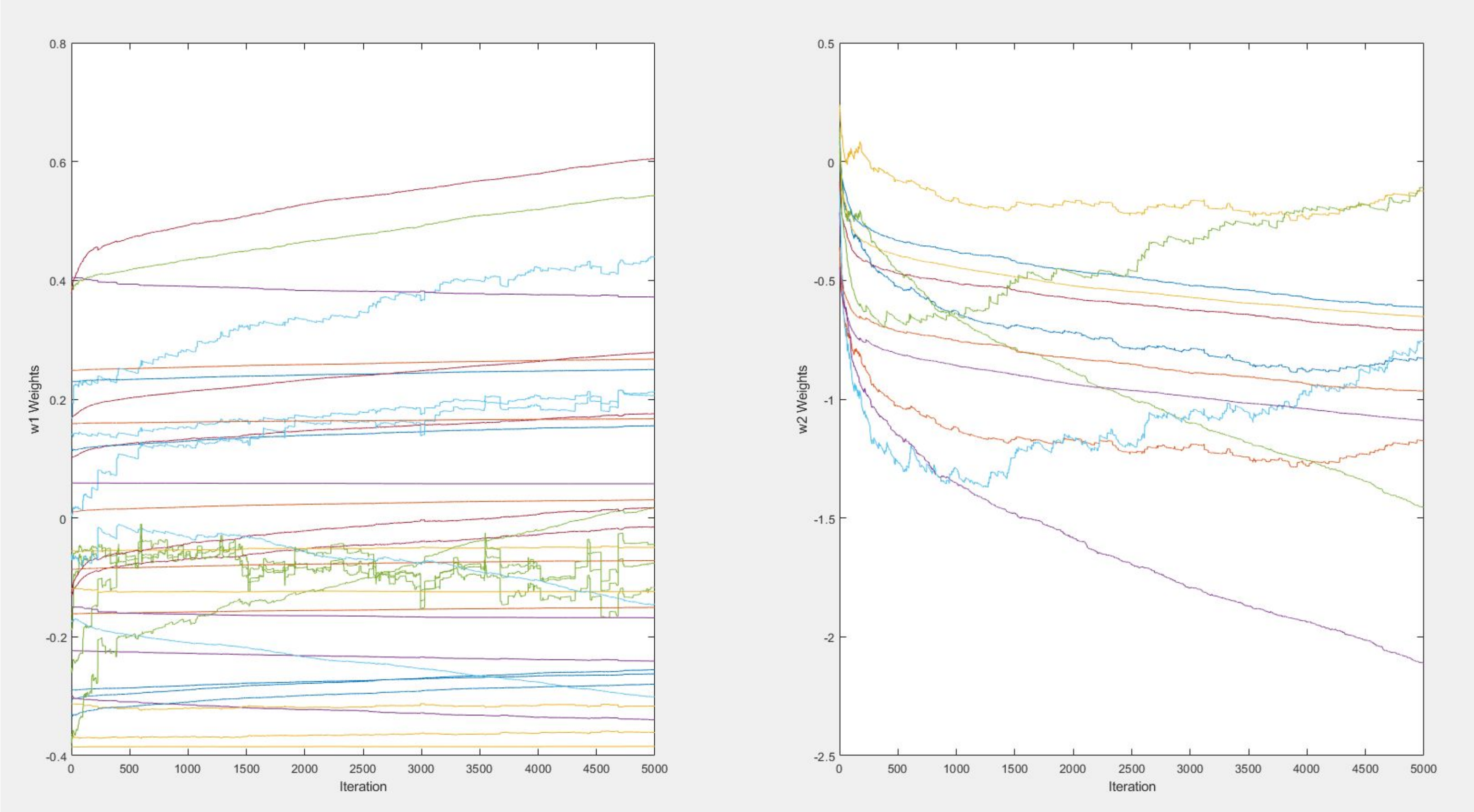
BIAS: 1

Neurons in hidden layer: 5

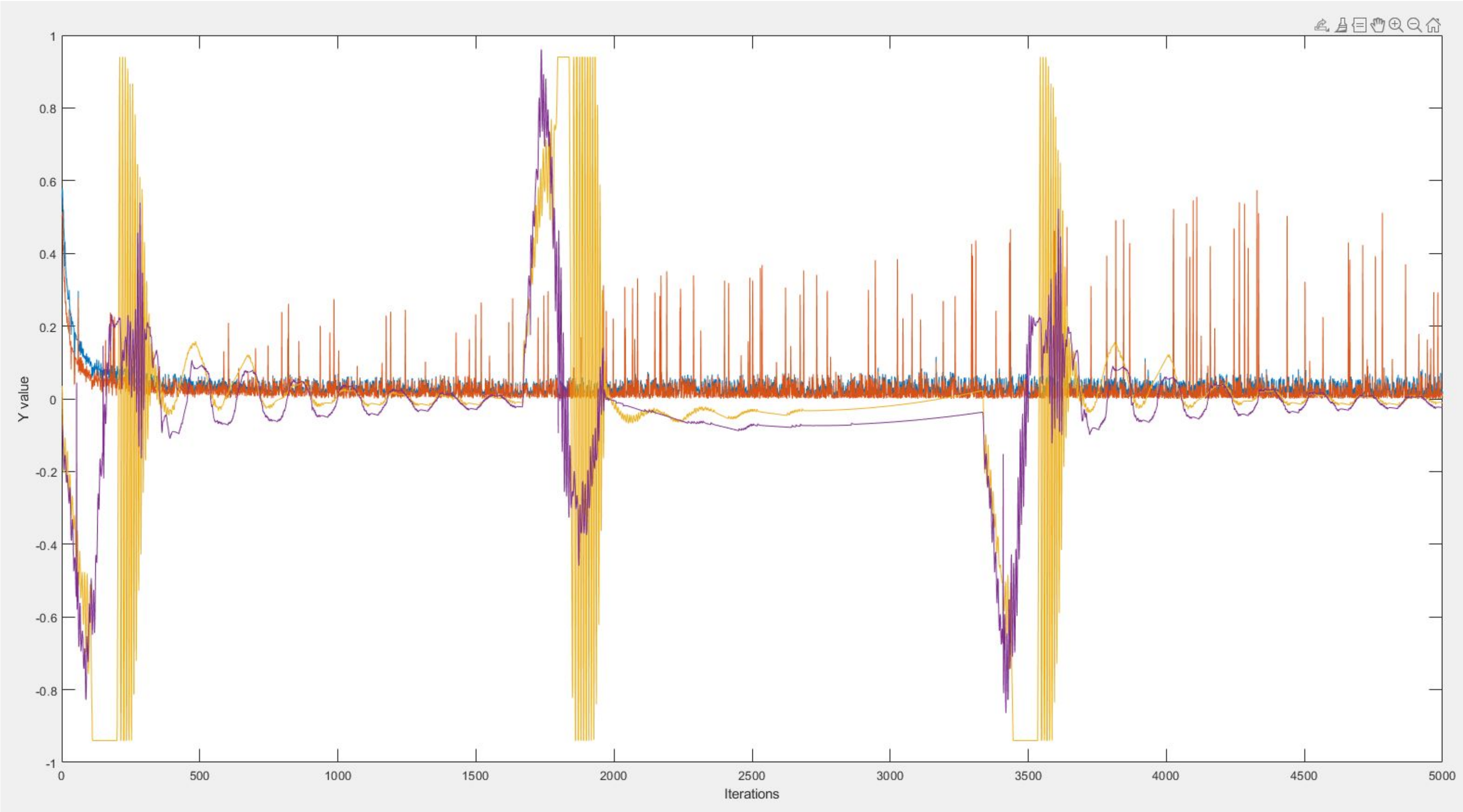
Layers: 1

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

BIAS = 1;
ETA = 0.2; % 0.1<ETA<0.4
```

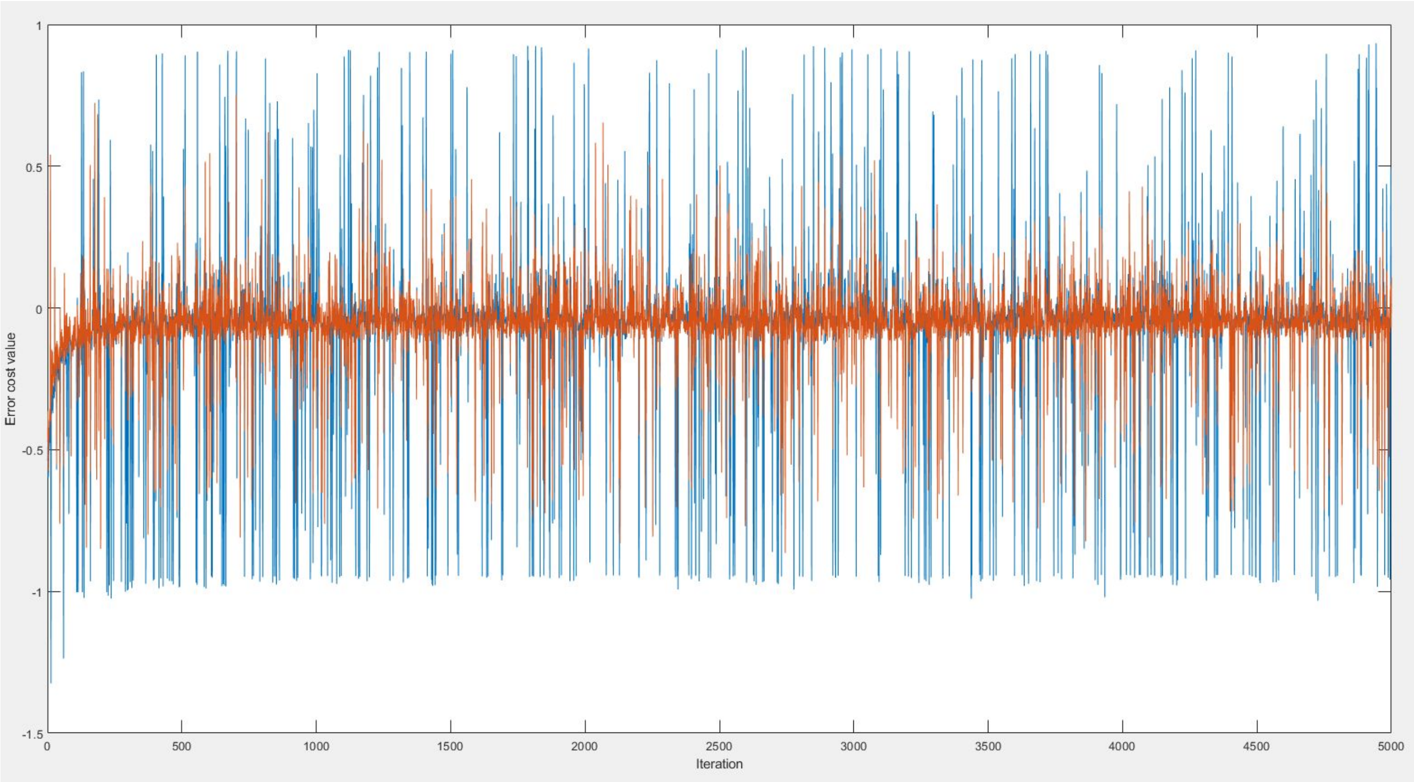


BP/MLP Weights of w_1 and w_2



MLP/BP Y values

BP/MLP Errors Cost Function Value



RESULTS of LSTM

What parameters are?

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

Learning rate(Eta): 0.1

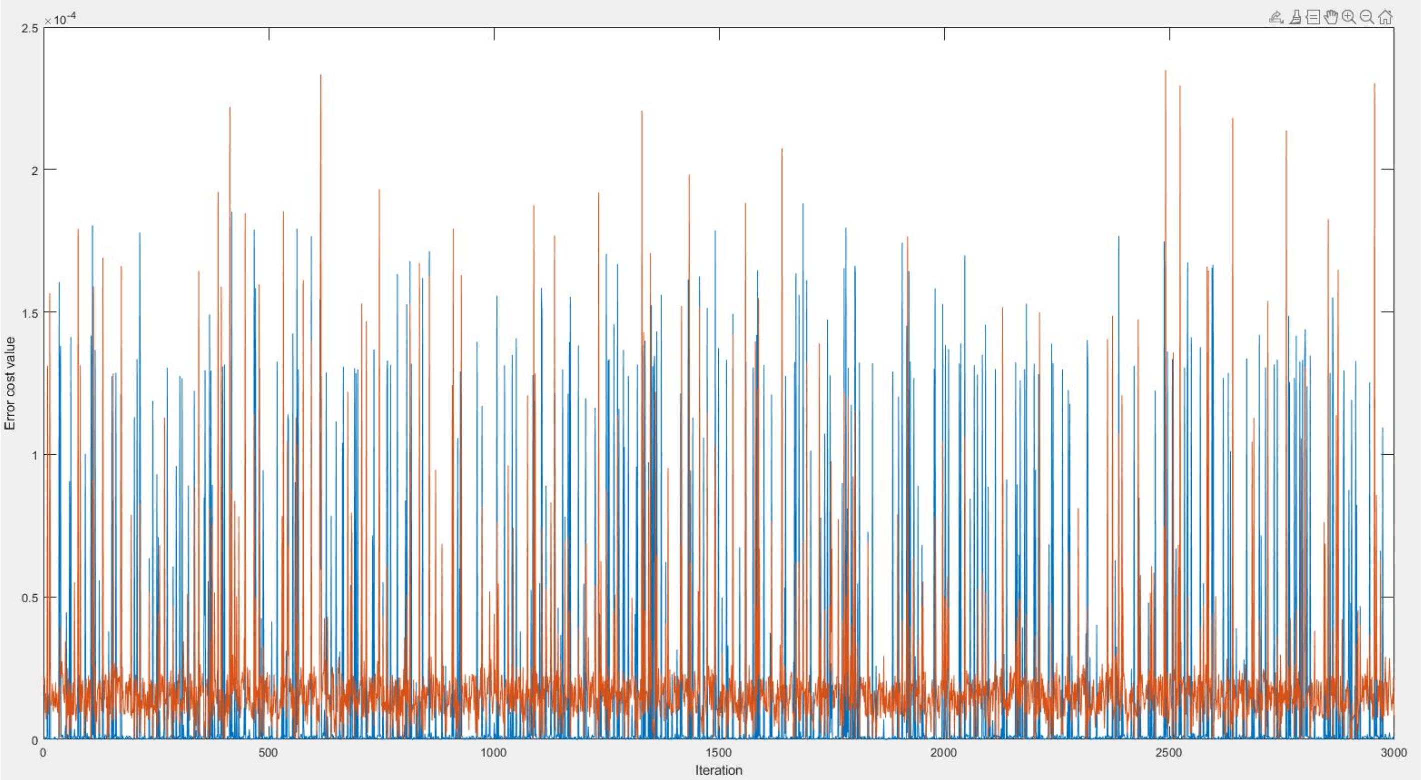
BIAS: 1

Neurons in hidden layer: 5

Layers: 1

```
neurons = IN-1; % % neurons, Range = 1 to L, Best = 2/3*L+N or L-1
Eta = 0.1;
Bias = 1;
```

BP/MLP 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…

Conclusion

Please be patient with our grand project…

Acknowledgement

Thanks for Dr. Meysar Zeinali teaching us and instructed our group's project at this term. Much appreciate everyone of our team spared no efforts on the work of this research. During the research period, we face challenges but never give up, the spirit of our team can cheer us up forever. Finally, thanks for everyone coming and listening.

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!