Class: RBE 577 – Machine Learning

Assignment: HW1

Students: Camilo Girgado & Jason Patel

9/28/2025

Encoder-Decoder Neural Network with Custom Loss Functions

This report analyzes the autoencoder neural network designed for maritime thruster control system as described in our reference paper, *Constrained Control Allocation for Dynamic Ship Control*. As part of the assignment, we worked to replicate both the methodologies and the results that were used and seen in the paper. The system itself uses a 3-layer encoder-decoder architecture to learn compressed representations of 3-dimensional force inputs while incorporating multiple constraint-based loss functions to ensure physically realistic and operationally safe thruster commands.

Methodology

Network Architecture

We implemented solution a symmetric autoencoder architecture with the following structure:

Encoder Path: 3 → 16 → 8 → 5

- Input layer: 3 dimensions (representing force components F1, F2, F3)

- First hidden layer: 16 neurons with ReLU activation and 10% dropout
- Second hidden layer: 8 neurons with ReLU activation and 10% dropout
- Bottleneck layer: 5 neurons with ReLU activation (encoded representation)

Decoder Path: 5 → 8 → 16 → 3

- Mirror architecture of encoder
- Reconstructs original 3-dimensional input from 5-dimensional encoded space
- Uses ReLU activations and 10% dropout in hidden layers

Data Generation Process

As described in section 3 of the reference paper, we synthetically generated the training data using the following physics-based approach:

1. **Input Parameters**: Random sampling from normal distributions for:

$$F_1 \in [-10000, 10000]N$$
$$F_2 \in [-5000, 5000]N$$
$$\alpha_2 \in [-180, 180]^\circ$$
$$F_3 \in [-5000, 5000]N$$
$$\alpha_3 \in [-180, 180]^\circ$$

Figure 1 – Physical Parameters

   - Force components: F1 ∈ (-10000, 10000), F2 ∈ (-5000, 5000), F3 ∈ (-5000, 5000)
   - Angles: α2, α3 ∈ (-180, 180)

2. **Transformation Matrix**: A 3×3 transformation matrix τ is constructed using trigonometric functions of the angles and geometric constants (l1=-14, l2=14.5, l3=-2.7, l4=2.7)

$$\boldsymbol{\tau} = \begin{bmatrix} 0 & c(\alpha_2) & c(\alpha_3) \\ 1 & s(\alpha_2) & s(\alpha_3) \\ l_2 & l_1 s(\alpha_2) & l_1 s(\alpha_3) \\ & -l_3 c(\alpha_2) & -l_4 c(\alpha_3) \end{bmatrix} \times \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix} = \boldsymbol{B}(\boldsymbol{\alpha})\boldsymbol{u}_f$$

Figure 2 – Transformation Matrix

3. **Output Calculation**: The result is computed as τ × F, representing the physical relationship between input forces and system response
4. **Dataset Size**: 1,000,000 samples for comprehensive training coverage

Multi-Component Loss Function

The system implements a sophisticated loss function combining five components:

L1 - Reconstruction Loss: Standard MSE between input and decoded output

L2 - Thruster Command Magnitude Limits: Penalizes commands exceeding physical limits

L3 - Rate Change Loss: Constrains rapid changes between consecutive commands using L3 norm

L4 - Power Consumption Loss: Minimizes power usage via |u|^(3/2) penalty on force components

L5 - Azimuth Sector Loss: Prevents thruster operation in forbidden angular sectors (-100° to -80° and 80° to 100°)

Training Strategy

- Optimizer: Adam with learning rate 0.001
- Batch Size: 100,000 samples
- Early Stopping: Terminates if test loss exceeds training loss for 200+ consecutive epochs
- Model Persistence: Saves best performing model based on training loss
- Train/Test Split: 80/20 random split with fixed seed for reproducibility

Hyperparameters

Network Architecture Parameters

- Input Dimensions: 3
- Encoder Hidden Layers: [16, 8]
- Bottleneck Dimensions: 5
- Decoder Hidden Layers: [8, 16]
- Output Dimensions: 3
- Activation Function: ReLU (all layers)
- Dropout Rate: 0.1 (10%)

- Data Type: torch.float32

Training Parameters

- Learning Rate: 0.001
- Optimizer: Adam
- Batch Size: 100,000
- Maximum Epochs: 1,000
- Early Stopping Patience: 200 epochs
- Train/Test Split: 80/20
- Random Seed: 42

Physical Constraint Parameters

- Force Limits: F1 ±10,000, F2/F3 ±5,000
- Angular Limits: ±180°
- Rate Change Thresholds: [1000, 1000, 1000, 10°, 10°]
- Forbidden Sectors: [-100°, -80°] and [80°, 100°]
- Power Law Exponent: 1.5

Results and Key Findings

1. Large batch sizes and comprehensive early stopping mechanisms promote stable convergence and prevent overfitting.
2. Dropout in both encoder and decoder helps reduce overfitting despite the large dataset, ensuring robustness.
3. Each additional loss term enforces realism but may slow convergence or conflict with pure reconstruction accuracy. Further tuning of loss weights could improve performance.
4. We noticed a difference in output when changing the hidden layer dimensions from 16–8 in the encoder and 8–16 in the decoder to 32-16 because this would directly affect the model's capacity to learn. The larger hidden layers allow the network to capture more nuanced nonlinear mappings between inputs and outputs, which in turn would reduce reconstruction error.

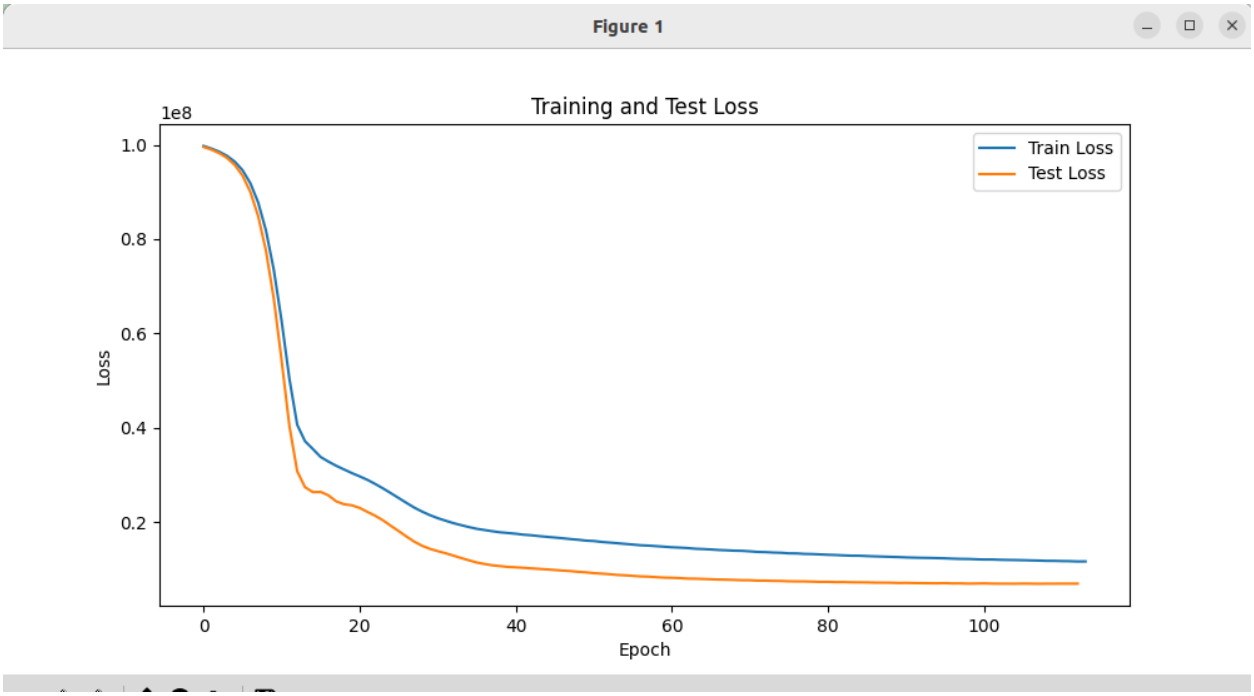

Figure Loss for 3-16-8-5

```
Epoch 0, Train Loss: 99746111.75065358, Test Loss: 99536976.23439676
Epoch 10, Train Loss: 62687212.94109174, Test Loss: 54327672.5437757
Epoch 20, Train Loss: 29736915.604277555, Test Loss: 23046524.933460947
Epoch 30, Train Loss: 20881533.813092582, Test Loss: 13895112.458647141
Epoch 40, Train Loss: 17576230.356521886, Test Loss: 10461301.316671547
Epoch 50, Train Loss: 16035343.878470453, Test Loss: 9223818.982958784
Epoch 60, Train Loss: 14711779.59702926, Test Loss: 8243264.6896416405
Epoch 70, Train Loss: 13848606.052918771, Test Loss: 7727494.721221866
Epoch 80, Train Loss: 13119745.488384841, Test Loss: 7379966.646607388
Epoch 90, Train Loss: 12561494.040853178, Test Loss: 7159925.38954743
Epoch 100, Train Loss: 12115958.740870789, Test Loss: 7056801.437825578
Epoch 110, Train Loss: 11785121.913697641, Test Loss: 7001108.278275012
Early stopping at epoch 113
Best Loss: 11616251.653734764 at epoch 112
```

Figure Training/Test Loss by Epoch for 3-16-8-5

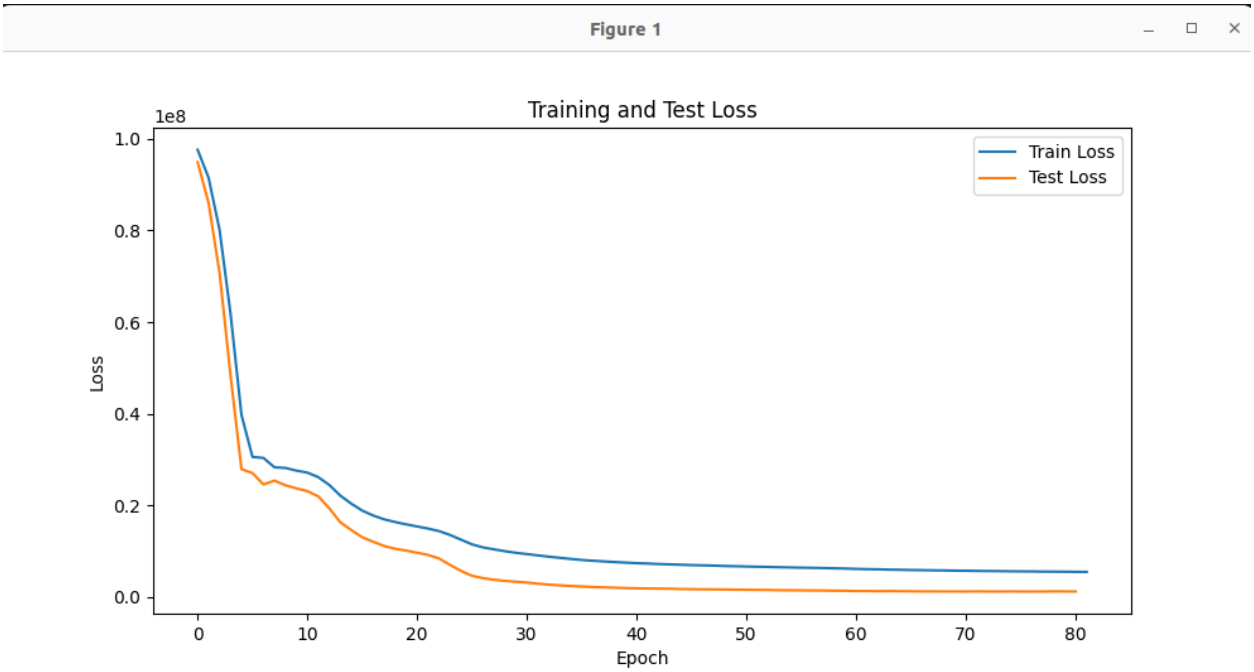

Figure Loss for 3-64-16-5

```
dev/rbe/RBE577_MachineLearning_Robotics/.venv/bin/python /home/mind/.vscode/extensions/ms-python.debugpy
pter/../../debugpy/launcher 54045 -- /home/mind/dev/rbe/RBE577_MachineLearning_Robotics/hw1/src/main.py
Epoch 0, Train Loss: 97577806.15070128, Test Loss: 94874053.18749152
Epoch 10, Train Loss: 27169088.578482248, Test Loss: 23139927.181965616
Epoch 20, Train Loss: 15425732.372192353, Test Loss: 9692361.36431209
Epoch 30, Train Loss: 9402631.664186755, Test Loss: 3193639.8135996116
Epoch 40, Train Loss: 7418502.088516362, Test Loss: 1942076.957702973
Epoch 50, Train Loss: 6692204.971390132, Test Loss: 1605308.7709603189
Epoch 60, Train Loss: 6168127.181570881, Test Loss: 1344036.775617118
Epoch 70, Train Loss: 5763009.241961816, Test Loss: 1236637.0956680123
Epoch 80, Train Loss: 5523147.506351368, Test Loss: 1236978.6323684028
Early stopping at epoch 81
Best Loss: 5469484.365489485 at epoch 81
```

Figure Training/Test Loss by Epoch for 3-64-16-5

Conclusion

Both the reference paper and our script demonstrate how neural networks can be guided by physics-inspired loss functions to produce solutions that are not only numerically accurate but also operationally feasible. The encoder-decoder framework successfully learns a compact latent space for thruster commands, while custom losses enforce safety, efficiency, and physical constraints.