Programming Assignment – 1

Sohan Reddy Jala – SE21UCAM012

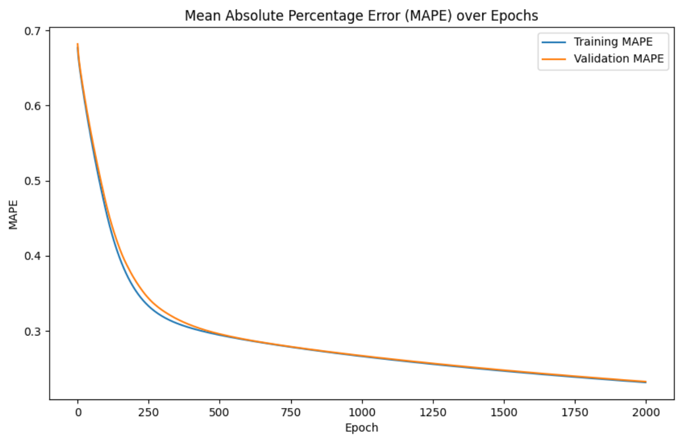
Siddhartha Reddy Paduri – SE21UCAM011

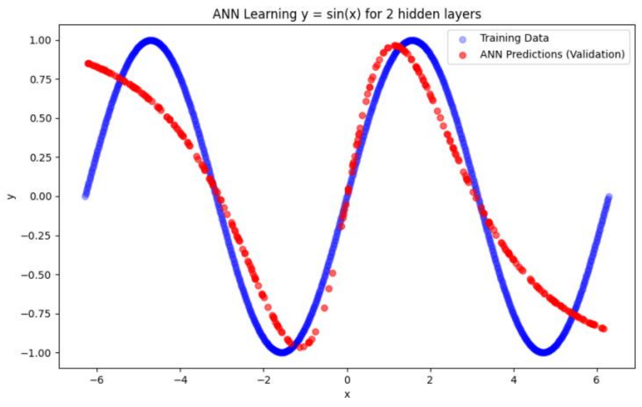
Rithvik Chowdary Vadlamudi – SE21UCSE165

Toy Problem

ANN Architecture

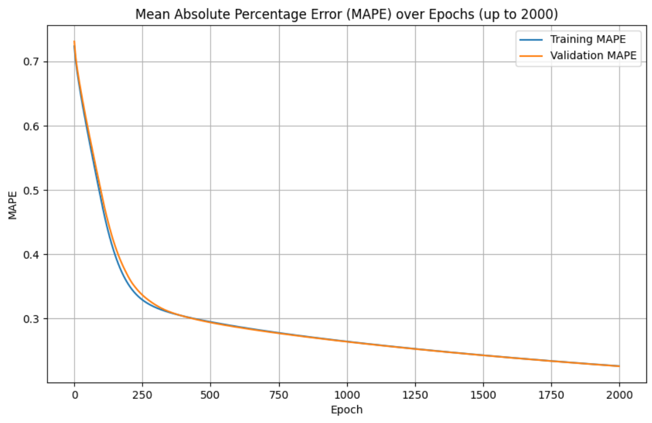
2 Hidden Layers:

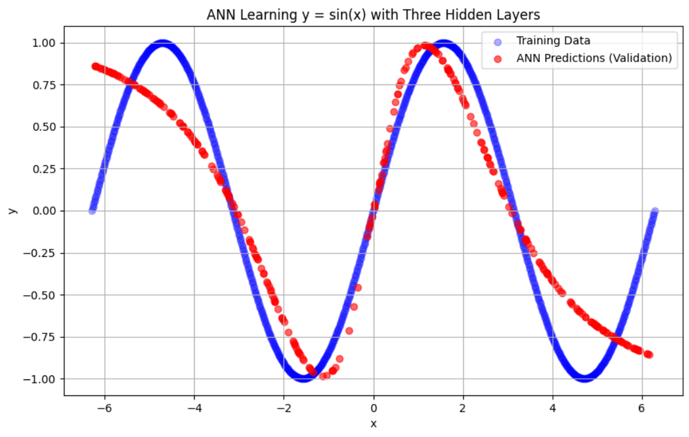




At Epoch 2000, MSE: 0.079348, MAPE Train: 0.2315, MAPE Val: 0.2325

3 Hidden Layers:





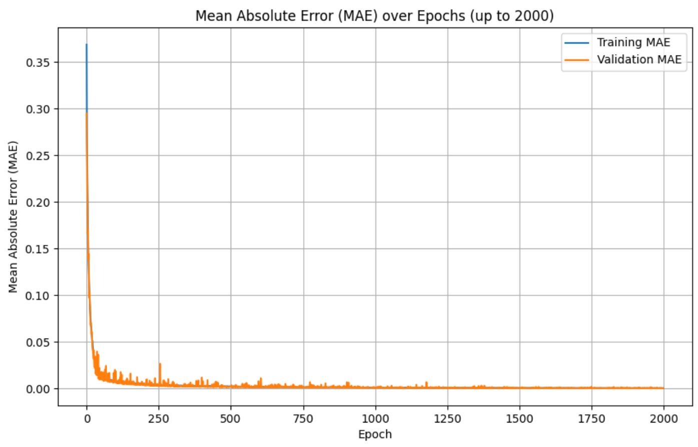
At Epoch 2000, MSE: 0.079348, MAPE Train: 0.2315, MAPE Val: 0.2325

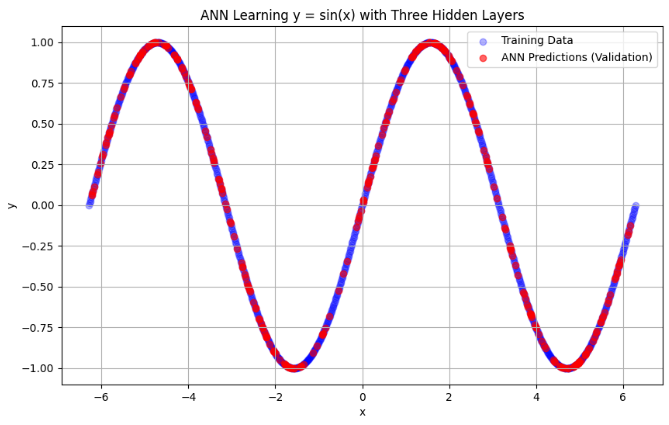
- Faster Convergence**:** The 3-hidden-layer model shows quicker convergence in the early stages (within the first 200 epochs) compared to the 2-hidden-layer model. Faster convergence means that the model is able to learn and generalize the patterns in the data more efficiently.
- Smaller Gap Between Training and Validation MAPE**:** The gap between the training and validation MAPE is smaller in the 3-hidden-layer model, which indicates better generalization. A smaller gap suggests that the model is not overfitting the training data and is capable of performing well on unseen data (validation set).
- More Stable Performance**:** The 3-hidden-layer model exhibits more stable performance as the number of epochs increases. The alignment between the training and validation MAPE curves in the 3-hidden-layer model remains tighter across the epochs, showing consistent performance and less fluctuation.

Both models achieve similar final MAPE values (around 0.27-0.3), but the 3-hidden-layer model achieves these values with less variance between training and validation sets, making it more reliable in terms of performance on unseen data.

Granulation of training data(3-Hidden Layers)

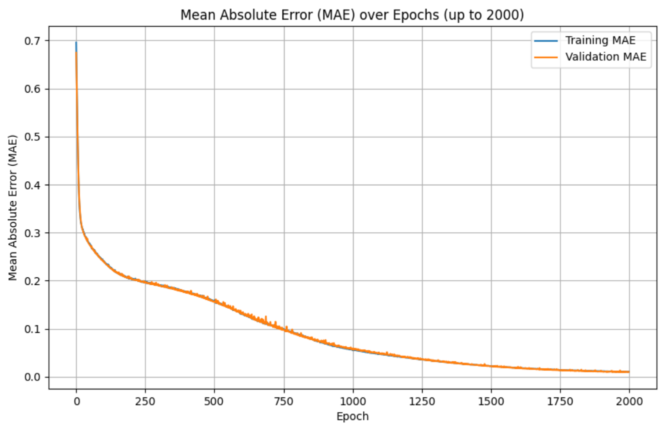
Batch Size = **1**

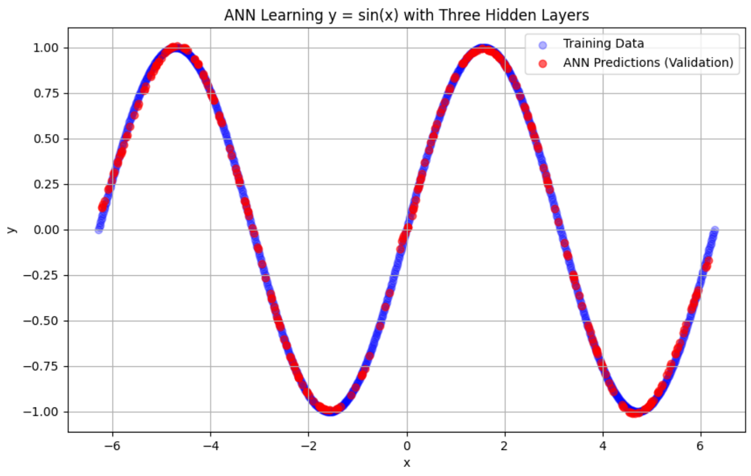




```
Epoch 2000, LOSS: 0.000001, MAE Train: 0.000558, MAE Val: 0.000441
```

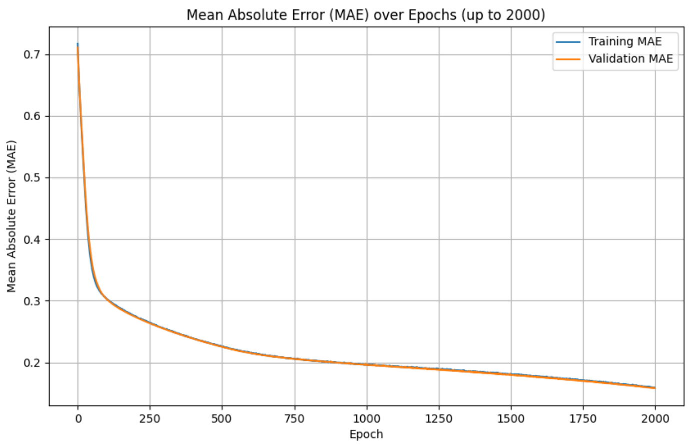
Batch Size = **64**

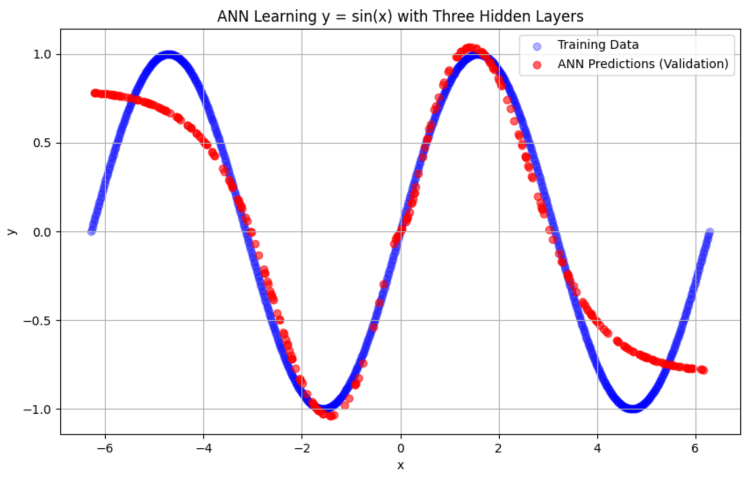




Epoch 2000, LOSS: 0.000264, MAE Train: 0.010379, MAE Val: 0.010174

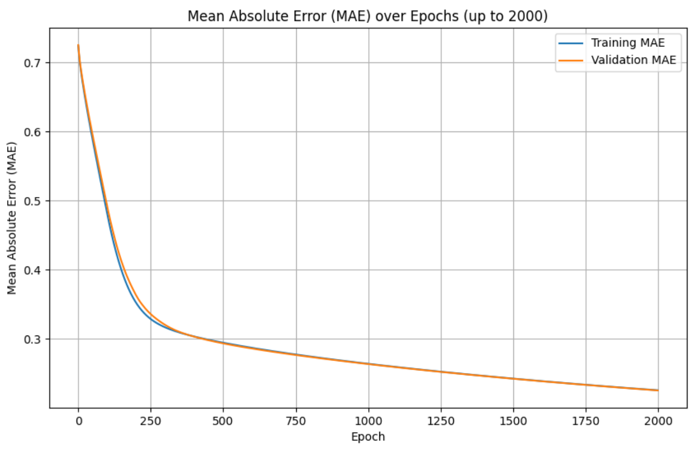
Batch Size = **256**

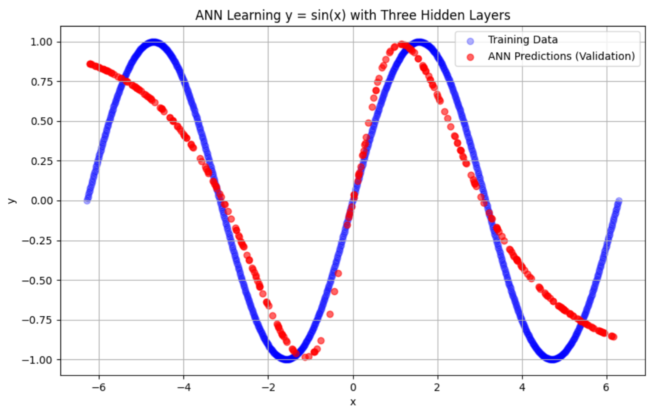




Epoch 2000, LOSS: 0.048762, MAE Train: 0.158803, MAE Val: 0.158239

<u>Batch Size</u> = **Full**





Epoch 2000, LOSS: 0.077305, MAE Train: 0.225575, MAE Val: 0.225289

Comparison of different batch sizes

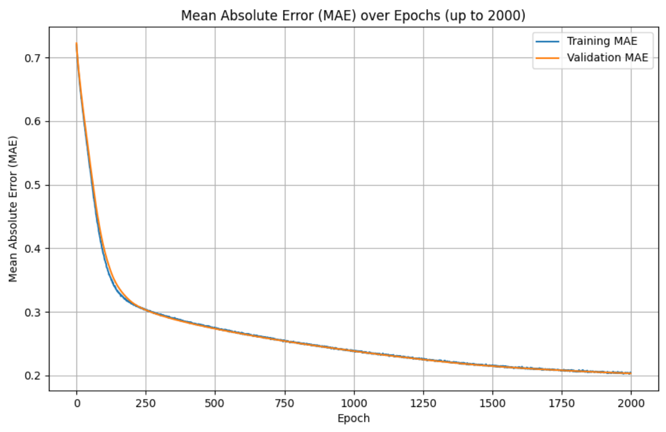
BATCH SIZE	CONVERGENCE SPEED	STABILITY(MAE)	PREDICTION ACCURACY	EXPLAINATION
1	Fast but Noisy	Noisy, Stabilizes lately	Decent, but deviations	Fast updates but noisy convergence. Struggles in some parts of the sine function
64	Fast and smooth	Very Smooth, Stabilizes early	BEST, nearly perfect fit	Best balance between frequent updates and smooth convergence. Best prediction accuracy
256	Moderate speed	Smooth, stabilizes around 500 epochs	Good ,some deviations	Slower convergence, visible prediction errors in some regions.
Full	Slow but very smooth	Extremely smooth, slower convergence	Good, similar to batch size 256	Extremely smooth learning curve but slower due to fewer updates.

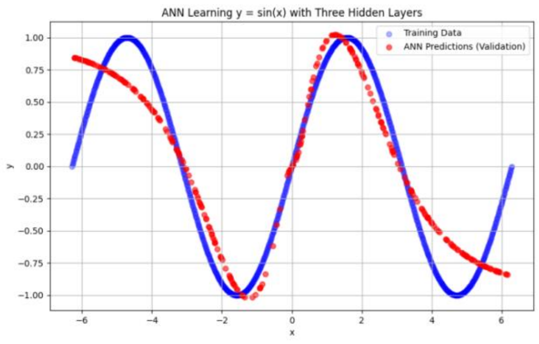
Batch Size = 64 offers the best trade-off between convergence speed, prediction accuracy and stability.

Activation Function (only using tanh as activation function)

Learning rate parameter and L2 Regularization

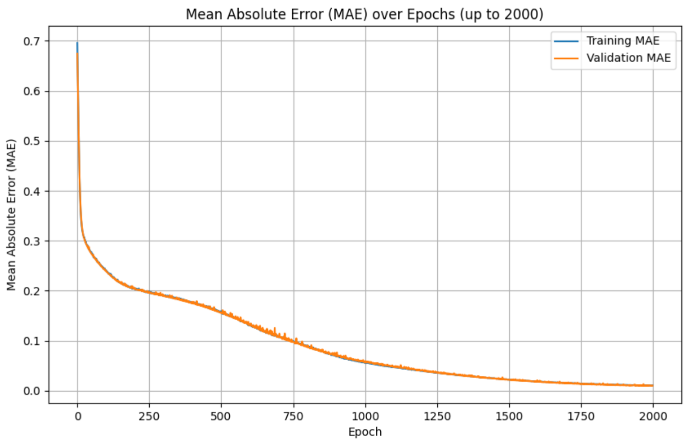
<u>Take η as 0.001</u>

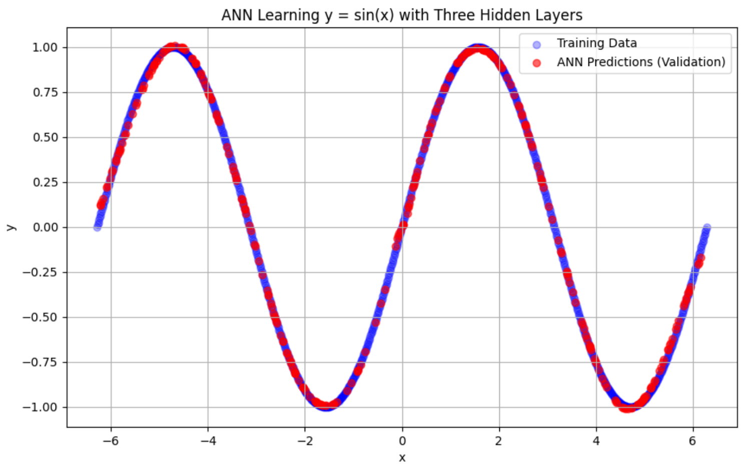




```
Epoch 2000/2000, MSE: 0.068081, MAE Train: 0.203885, MAE Val: 0.203446
```

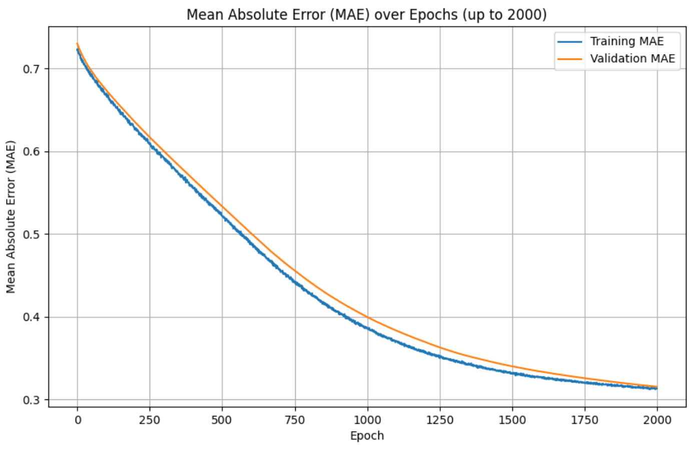
Take η as 0.01

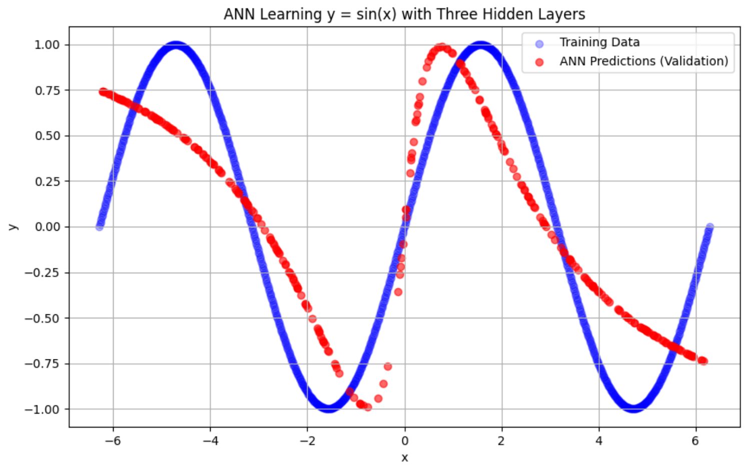




```
Epoch 2000, LOSS: 0.048762, MAE Train: 0.158803, MAE Val: 0.158239
```

Take η as 0.0001





```
Epoch 2000, MSE: 0.123657, MAE Train: 0.313433, MAE Val: 0.315525
```

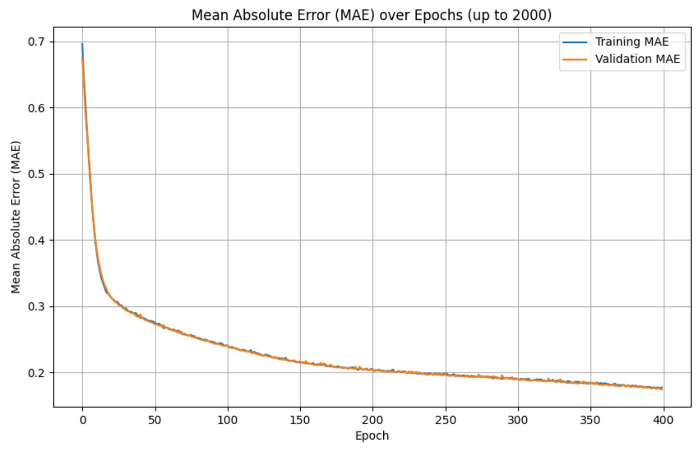
Comparison between different rates

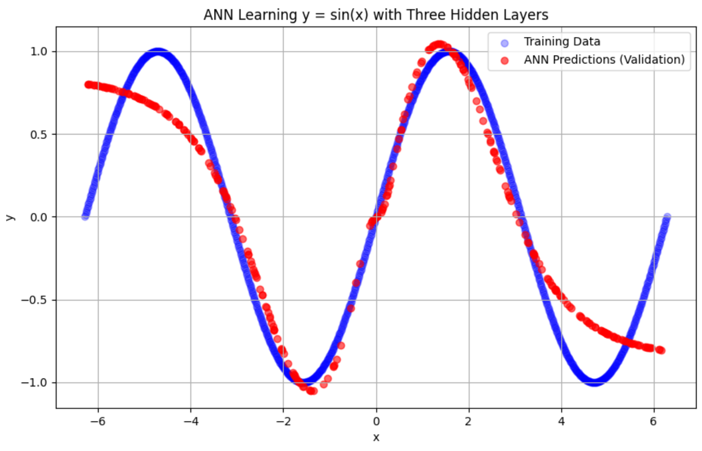
LEARNING RATE	CONVERGENCE SPEED	MAPE STABILITY	PREDICTION ACCURACY	TRAINING VS VALIDATION ALIGNMENT
η = 0.01	Fastest	Very Smooth	BEST(minimal deviation)	Excellent alignment(best generalization)
η = 0.001	Moderate	Very smooth	Good, slight deviations	Good alignment, but slower convergence
η = 0.0001	Slowest	Very smooth	Weakest ,visible deviation	Alignment good, but learning too slow

Best learning rate = 0.01 as it is optimal among the three. It provides balance of fast convergence and accurate predictions while maintaining stable and smooth learning.

Stopping Criteria

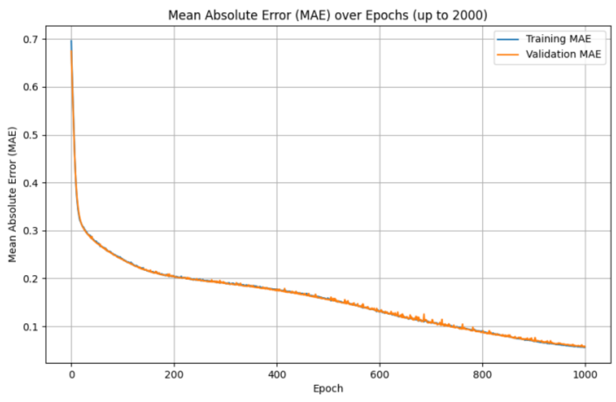
Stopping Criteria = **400 epochs**

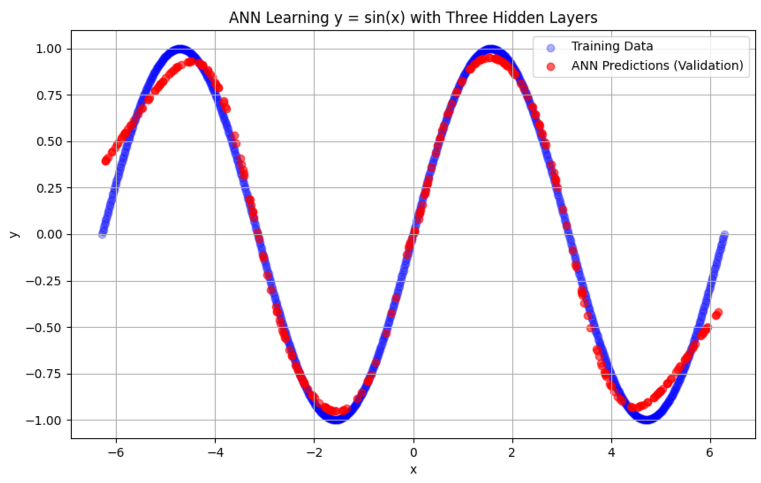




```
Epoch 400, LOSS: 0.056175, MAE Train: 0.177593, MAE Val: 0.174520
```

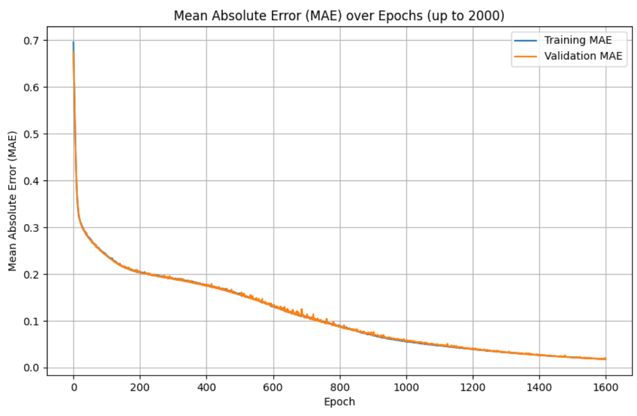
Stopping Criteria = **1000 epochs**

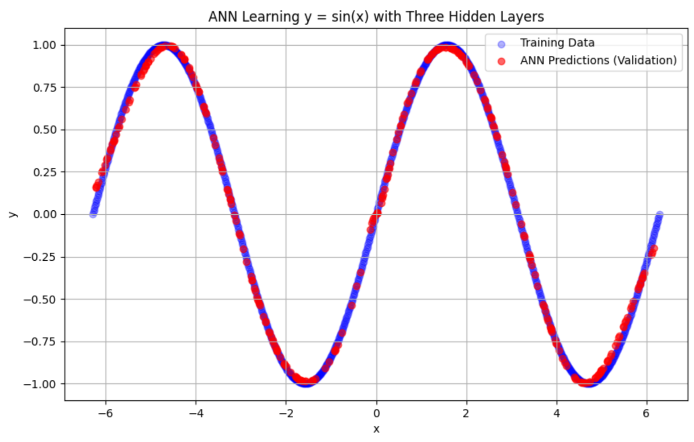




Epoch 1000, LOSS: 0.006846, MAE Train: 0.056459, MAE Val: 0.057544

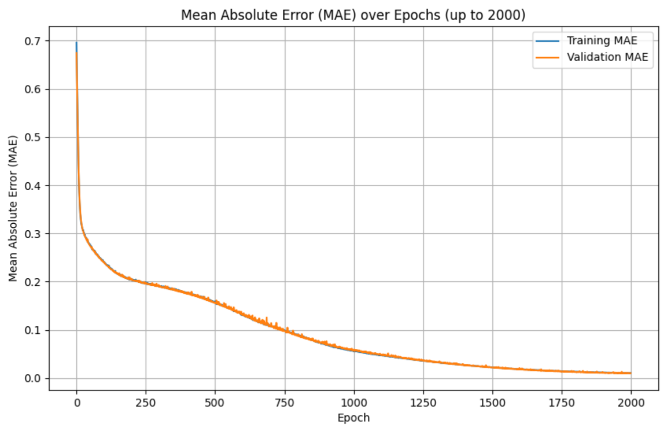
Stopping Criteria = **1600 epochs**

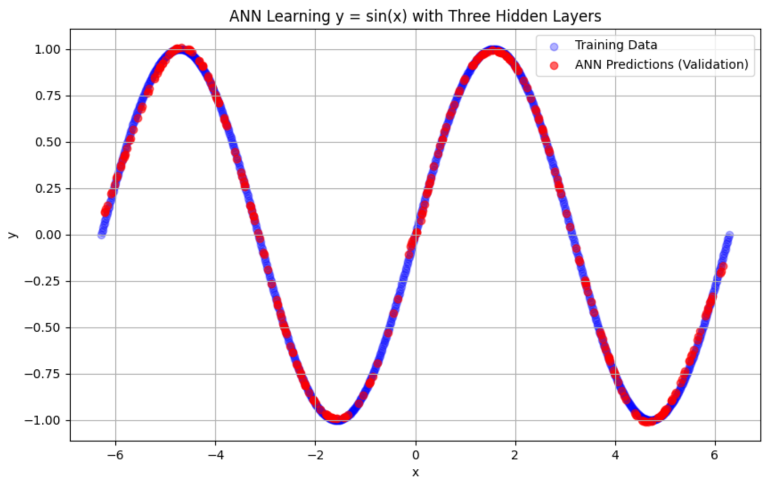




Epoch 1600, LOSS: 0.000705, MAE Train: 0.018472, MAE Val: 0.018578

Stopping Criteria = **2000 epochs**





```
Epoch 2000, LOSS: 0.000264, MAE Train: 0.010379, MAE Val: 0.010174
```

Comparison between stopping criteria

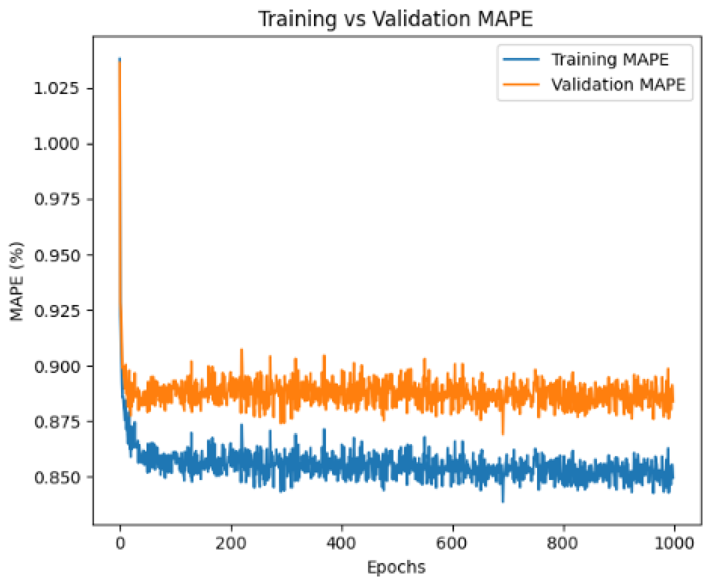
NO OF EPOCHS	CONVERGENCE(MAPE)	STABILITY	PREDICTION ACCURACY	GENERALIZATION
400	Early-stage, still reducing	Slight Fluctuations	Poor, visible deviations, underfitting	WEAK
1000	Moderate convergence	Improving, but not optimal	Better fit but still deviating	MODERATE
1600	Almost fully converged	Minimal fluctuations	Close to sine wave, slight deviations	GOOD
2000	Fully converged	Very Stable	BEST FIT, minimal deviations	EXCELLENT

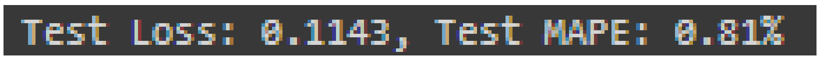
Best stopping criteria = 2000 epochs as it provides the best balance of convergence, stability and prediction accuracy. The network has fully learned the sine wave pattern by this point, and the MAPE is minimized.

ANN on data of a Combined Cycle Power Plant

ANN Architecture

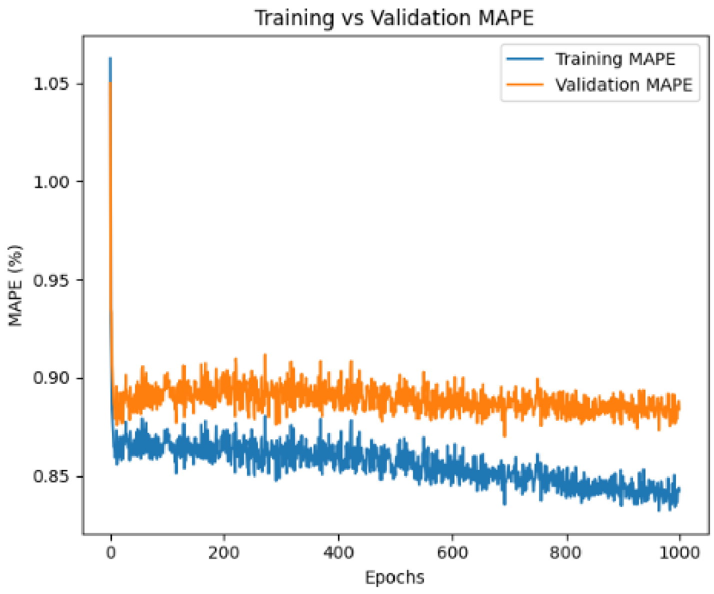
2 Hidden Layers:

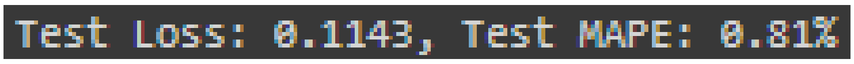




```
Test Loss: 0.1143, Test MAPE: 0.81%
```

3 Hidden Layers:



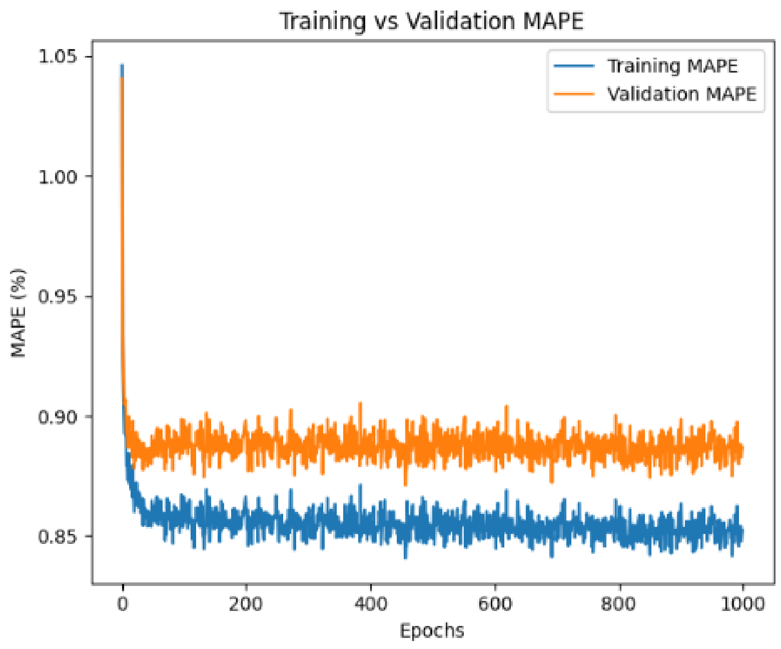


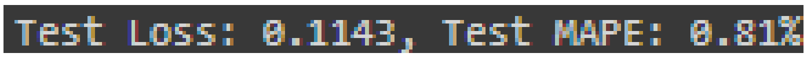
```
Test Loss: 0.1143, Test MAPE: 0.81%
```

There is not much of a significant change in the MAPE between 2 and 3 hidden layers, but since the goal is minimizing MAPE, even though the 3-Layer ANN performs better, the validation MAPE has high variance which indicates potential overfitting. The 2-Layer ANN would be better to work with, as it balances both training and validation performance, leading to a more robust model.

Granulation of training data(2-Hidden Layers)

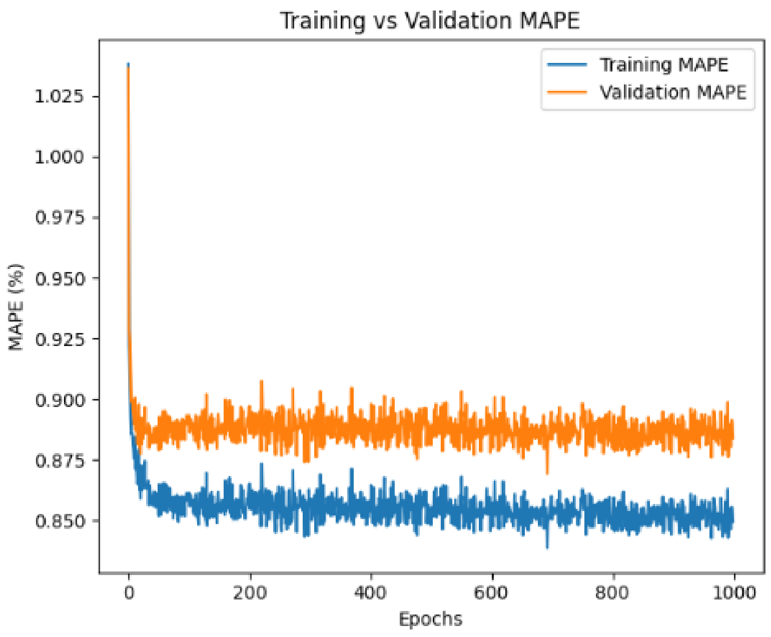
Batch Size = **1**

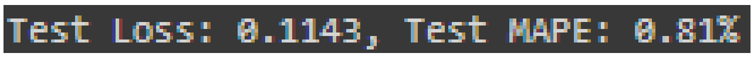




```
Test Loss: 0.1143, Test MAPE: 0.81%
```

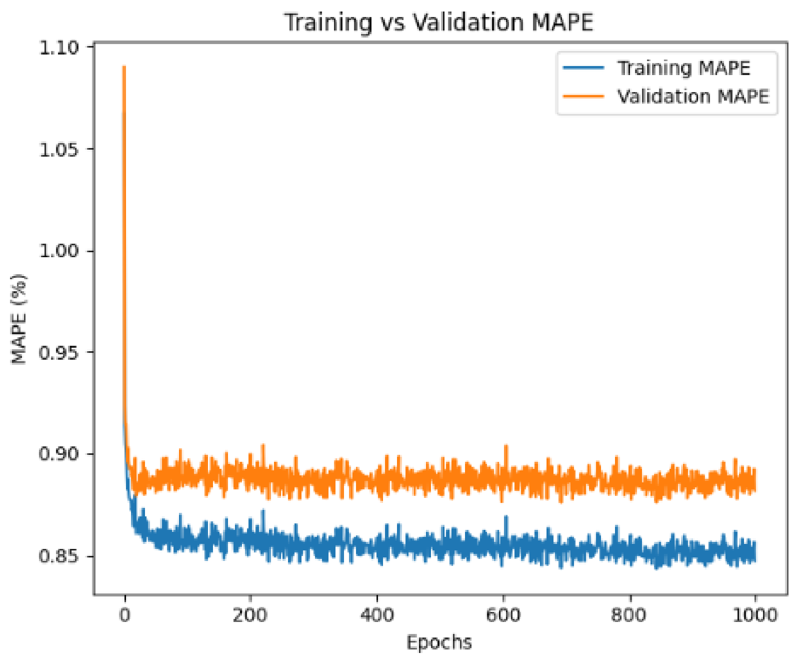
Batch Size = **64**

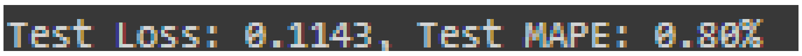




```
Test Loss: 0.1143, Test MAPE: 0.81%
```

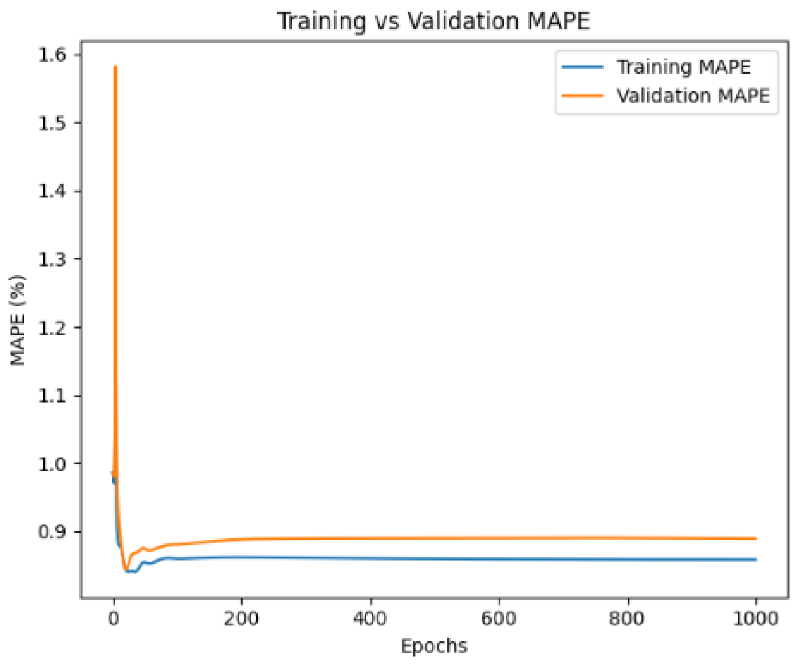
Batch Size = **256**

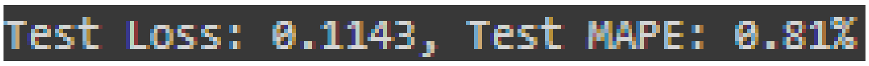




```
Test Loss: 0.1143, Test MAPE: 0.80%
```

Batch Size = **Full**





```
Test Loss: 0.1143, Test MAPE: 0.81%
```

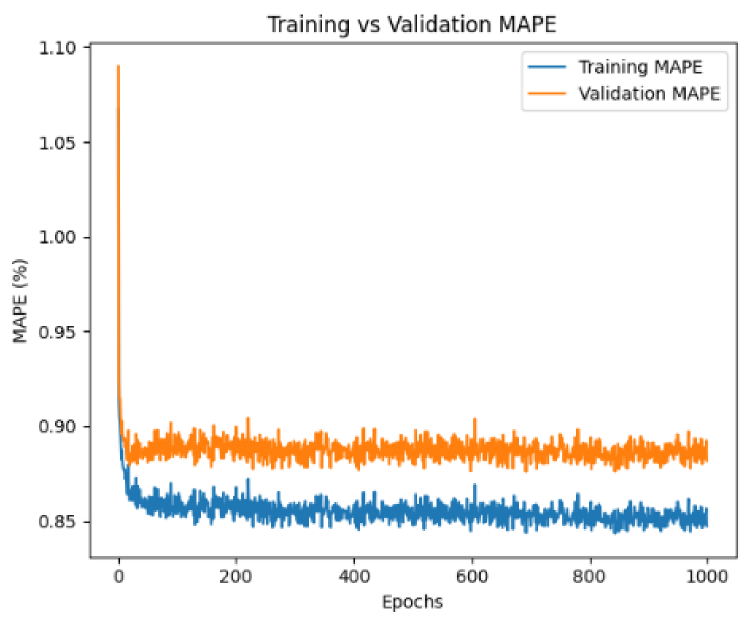
Comparison of Batch Sizes

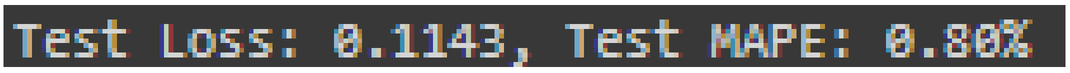
- Batch Size 1 provides too much variance, making it unreliable for this task due to noisier updates and less consistent validation performance.
- Batch Size 64 shows improvement but still presents fluctuations in the validation MAPE, although it balances updates and generalization better than batch size 1.
- Batch Size 256 strikes a good balance between smooth convergences; frequent enough updates, and good generalization, making it the best choice for most scenarios.
- Full Batch Size offers the smoothest convergence with minimal variance but may lead to slower updates and potentially getting stuck in local minima.

We will proceed with Batch Size being 256 as it offers a good balance between stability, speed of convergence, and generalization without the overfitting risks or slowdowns of the full batch size.

Activation Function (Batch Size = 256)

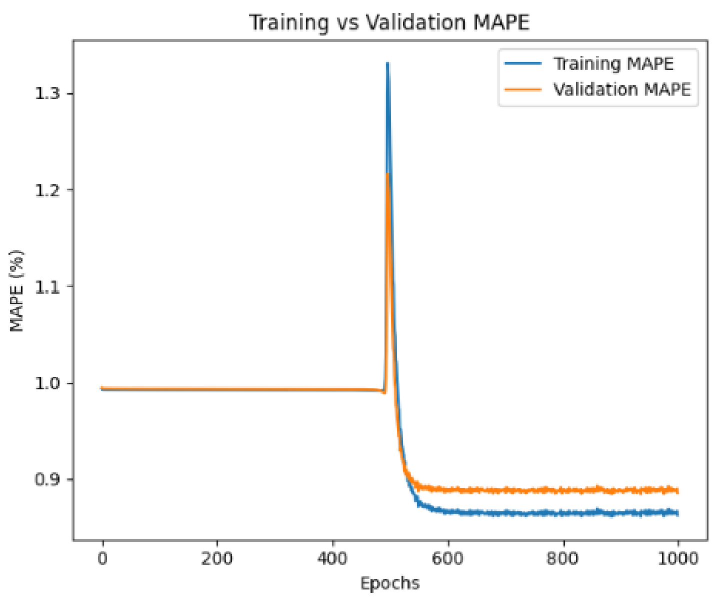
Tanh for all hidden layers and Logistic for output layer:

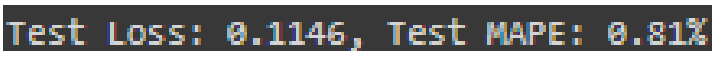




```
Test Loss: 0.1143, Test MAPE: 0.80%
```

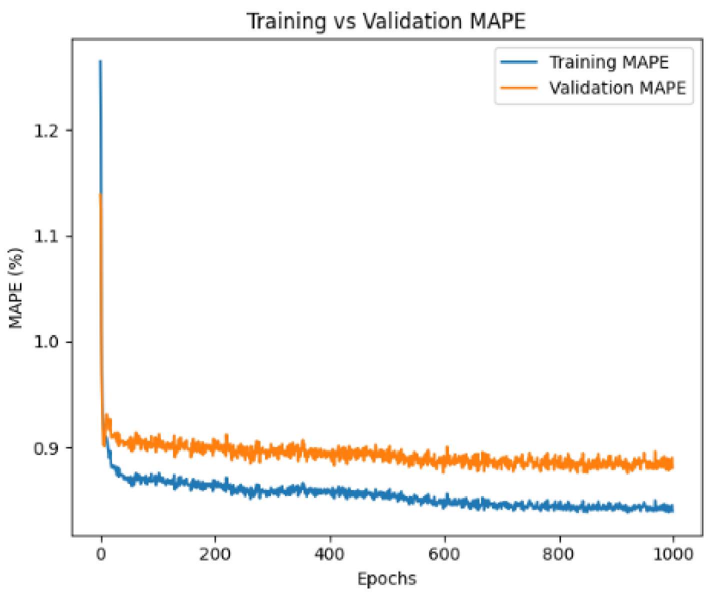
<u>Logistic for all hidden layers and logistic for output layer:</u>

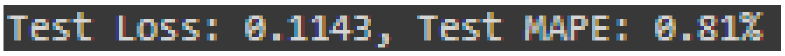




```
Test Loss: 0.1146, Test MAPE: 0.81%
```

<u>Relu for all hidden layers and logistic for output layer:</u>





```
Test Loss: 0.1143, Test MAPE: 0.81%
```

- <u>Graph 1: Tanh for All Hidden Layers and Logistic for Output Layer MAPE behavior</u>: While the validation MAPE stays higher at roughly 0.90%, the training MAPE declines and stabilizes at roughly 0.85%. Throughout the training process, there is a constant discrepancy between the validation and training MAPE. The validation MAPE continuously stays higher, which suggests that while the model is performing well on the training data, it may have a small overfitting problem.

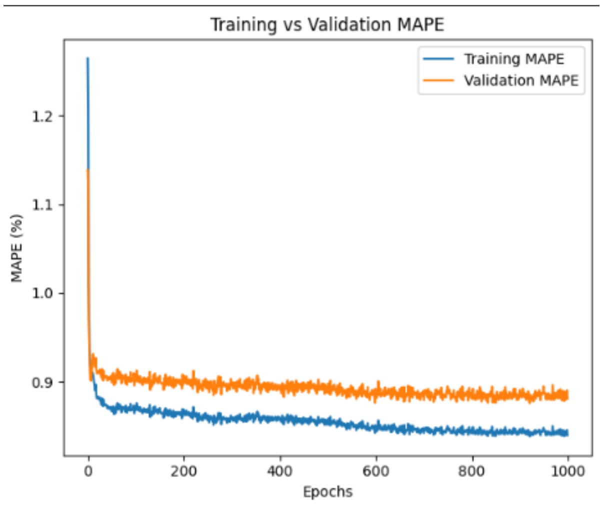
- Graph 2: Logistic for All Hidden Layers and Logistic for Output Layer MAPE behavior: Following a notable peak at approximately 500 epochs, both the training and validation MAPEs exhibit a dramatic decline at 600 epochs. Both, though, settle at roughly 0.90%. This behavior is peculiar—especially considering the abrupt spike—and suggests that there may have been instability during training. Although the final performance is not too poor, this technique is less reliable because the spike indicates that the model may have run into problems such as gradient explosion.
- Graph 3: ReLU for All Hidden Layers and Logistic for Output Layer MAPE behavior: While the validation MAPE stays at roughly 0.90%, the training MAPE stabilizes at roughly 0.85%. Though marginally less than in Graph 1, the difference between the training and validation MAPEs is nonetheless persistent. Overall Performance: The model's performance is comparable to that of Graph 1. Compared to Tanh, the ReLU activation function typically aids in faster convergence and is less prone to vanishing gradient issues. The continuous gap, however, still points to a slight overfitting problem.

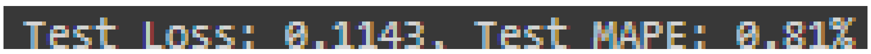
  Overall, the first and the third graph perform similarly. Between both of them, Graph 3 is the best choice as the ReLU is known for faster convergence and performs better in deeper networks. It shows stable performance without major instability, and while it has a similar gap as Graph 1, its overall stability and speed of convergence make it a preferable choice.

Learning rate parameters and L2 regularization

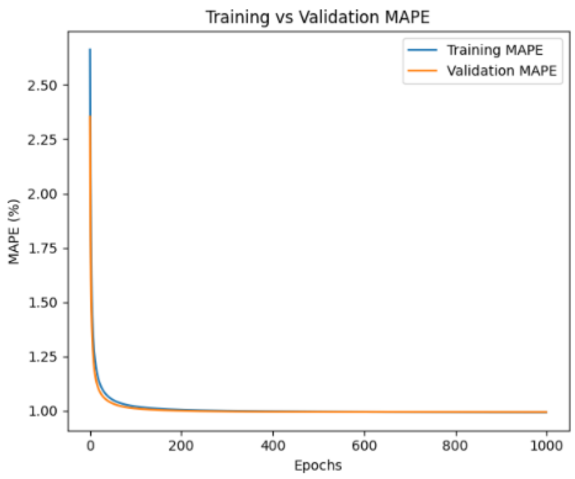
L2 Regularization(learning rate = 0.001)

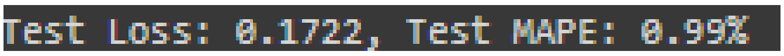
Lambda = **0**




```
Test Loss: 0.1143, Test MAPE: 0.81%
```

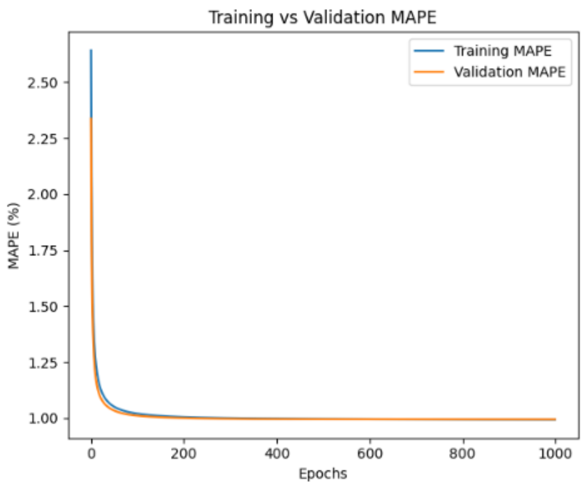
Lambda = **0.1**

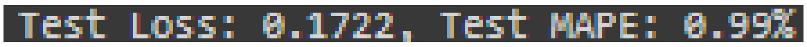




```
Test Loss: 0.1722, Test MAPE: 0.99%
```

Lambda = **0.9**





```
Test Loss: 0.1722, Test MAPE: 0.99%
```

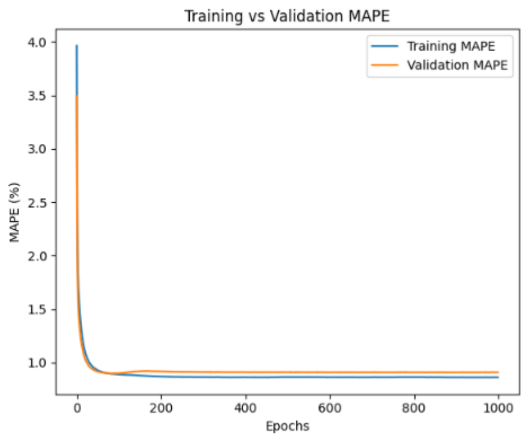
- Graph 1 (lambda = 0): The model is trained in this instance without regularization. The training error keeps going down while the validation error stabilizes at a higher value, indicating a slight overfitting pattern in the graph. This suggests that the model might be memorizing the training data too much, leading to a lack of generalization.
- Graph 2 (lambda = 0.1): According to the graph, MAPE consistently decreases and stabilizes at roughly the same level for both training and validation. This implies that there is little to no overfitting and that the model generalizes well. The training and validation lines are almost identical, indicating that regularization is helping to control overfitting and improve the model's generalization.

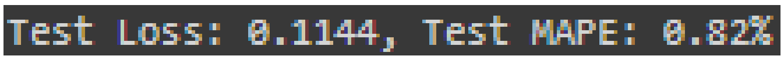
- Graph 3 (lambda = 0.9): The model is underfitting the data due to the high regularization. The flat training and validation curves indicate that the model is not able to significantly reduce the error.The model has become too simple due to excessive regularization, making it less capable of learning from the data.

The second graph (lambda = 0.1) is the best choice. It balances the training and validation MAPE curves well, showing that the model generalizes better to unseen data without overfitting or underfitting.

Learning rate parameters with lambda = 0

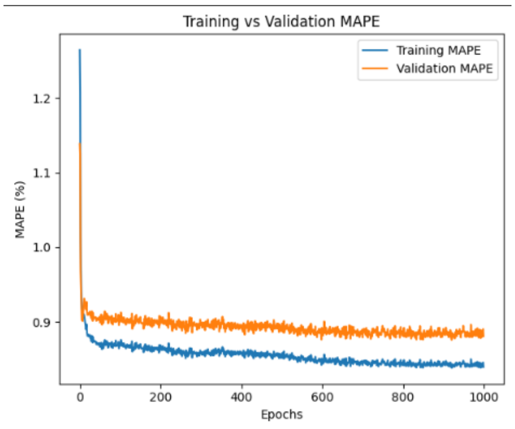
Learning rate = **0.0001**

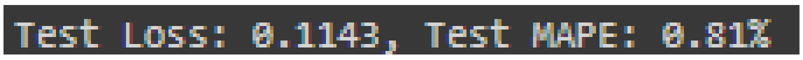




```
Test Loss: 0.1144, Test MAPE: 0.82%
```

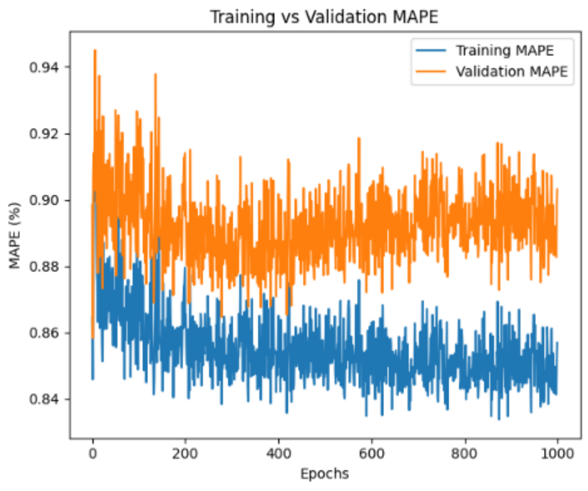
Learning rate = **0.001**

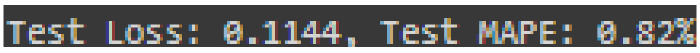




```
Test Loss: 0.1143, Test MAPE: 0.81%
```

Learning rate = **0.01**





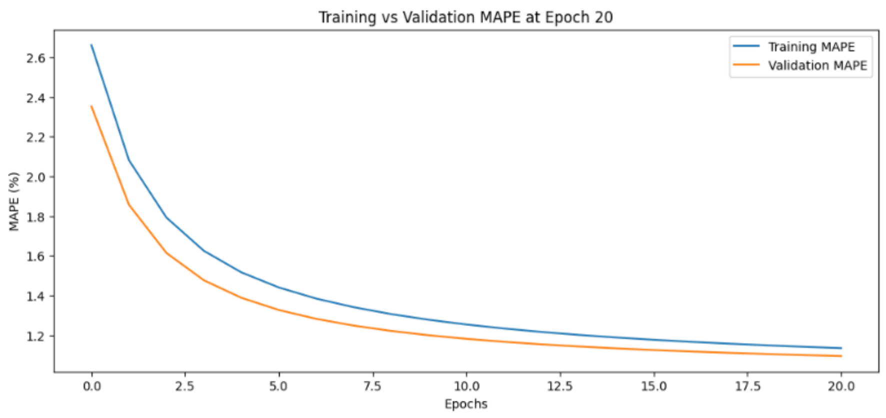
```
Test Loss: 0.1144, Test MAPE: 0.82%
```

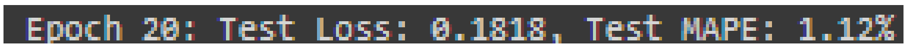
- Learning rate – 0.0001: The MAPE curves for training and validation both rapidly decline and then stabilize at 1.0%. The close alignment of the curves suggests that the model prevents overfitting and has good generalization. However, because of the slow learning rate, it converges very slowly. The training and validation curves for this model are nearly identical, demonstrating its excellent generalization. However, the final MAPE is approximately 1.0%, which is higher than anticipated.
- Learning rate – 0.001 : The validation MAPE stays at roughly 0.90%, but the training MAPE drops to about 0.85%. While the model is learning well, there appears to be a slight overfitting, as indicated by the small but consistent gap between the two curves. The training and validation curves of this model converge rather well, with a small gap between them, and it exhibits respectable generalization. This rate of learning achieves a good balance between accuracy and speed of convergence.
- Learning Rate = 0.01 : The MAPE curves for training and validation exhibit more erratic behavior. The validation MAPE varies between 0.90% and 0.92%, whereas the training MAPE oscillates between 0.86% and 0.88%. There is a discernible and stable difference between the two curves, suggesting that this model is more overfit than the others. Faster convergence is achieved at the expense of stability with this learning rate. In comparison to the other graphs, the larger gap between the training and validation curves indicates that the model is overfitting more.

The best option is learning rate = 0.001. It offers a good balance between speed of convergence and accuracy, with a final MAPE that is lower than that of Graph with learning rate of 0.0001 and a smaller gap than learning rate of 0.01.

Stopping criteria

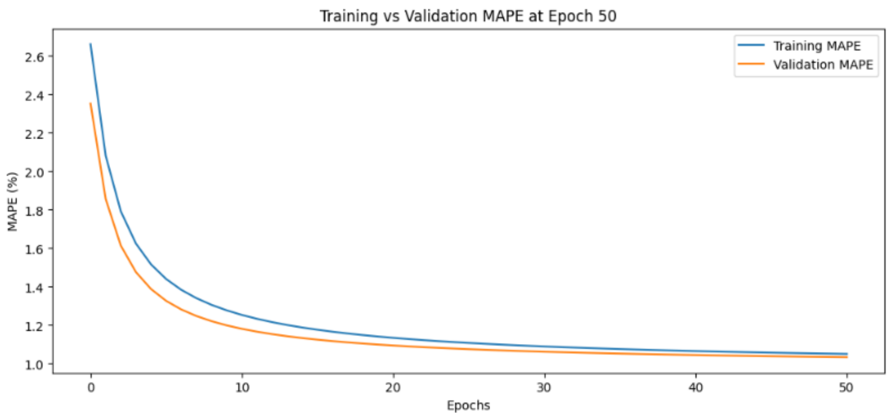
Stopping criteria = **20**

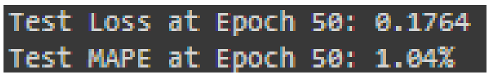




```
Epoch 20: Test Loss: 0.1818, Test MAPE: 1.12%
```

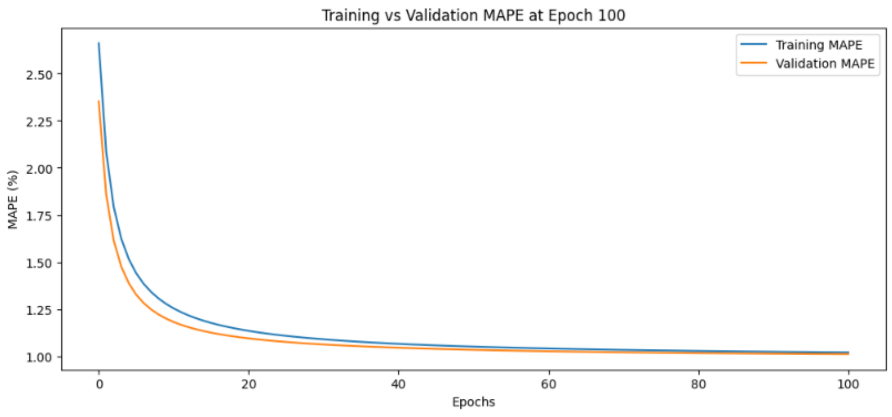
Stopping criteria = **50**

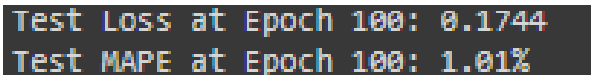




```
Test Loss at Epoch 50: 0.1764
Test MAPE at Epoch 50: 1.04%
```

Stopping criteria = **100**





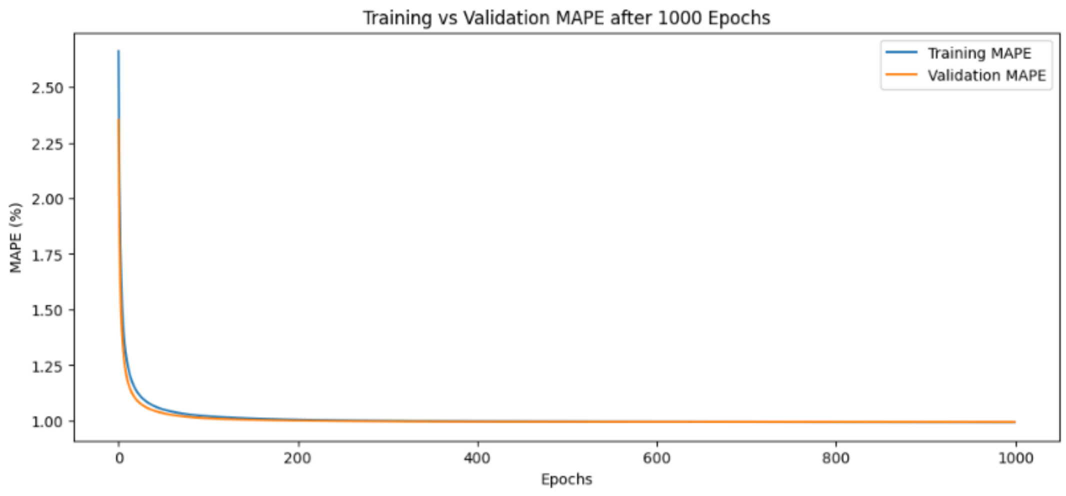
```
Test Loss at Epoch 100: 0.1744
Test MAPE at Epoch 100: 1.01%
```

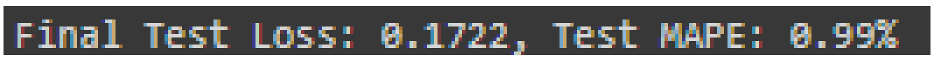
- <u>Epoch 20</u> – Although they haven't completely stabilized yet, the training and validation MAPE values are currently declining. Although the error has decreased, it appears that there is still room for improvement because the lines haven't reached a plateau. This might be too early to stop as the model is still learning and improving.
- <u>Epoch 50</u> - The model is clearly generalizing well because the difference between the training and validation errors is not that great, even though the training and validation MAPEs have both continued to decline. Although there are fewer small fluctuations, the lines are more stable. This could be a reasonable stopping point, as the performance seems to have stabilized compared to epoch 20, and there is a balance between learning and overfitting risk.
- <u>Epoch 100</u> – The model has fully learned the pattern, as evidenced by the smooth lines and nearly identical values for the training and validation MAPE. Comparing this to 50 epochs, there isn't much of an improvement, and overfitting may become more likely.

Epoch 50 seems to be the optimal stopping point. The model has learned well, with reduced training and validation errors, and the risk of overfitting is minimized compared to 100 epochs. Epoch 50 provides a good trade-off between error minimization and efficiency.

Optimal Parameters

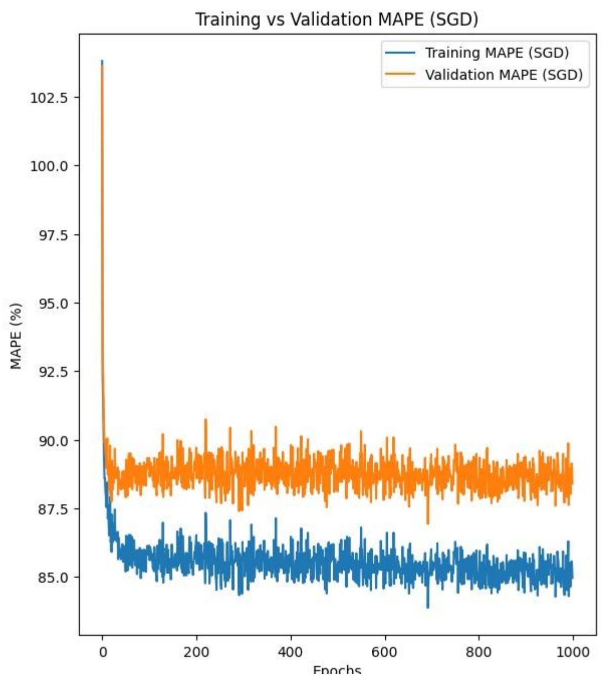
This includes batch size = 256, the function being ReLU for all hidden layers and logistic for output layers, learning rate = 0.01, lambda = 0.1 and the stopping criteria is 100 epochs. The following is the final graph with all these parameters applied:

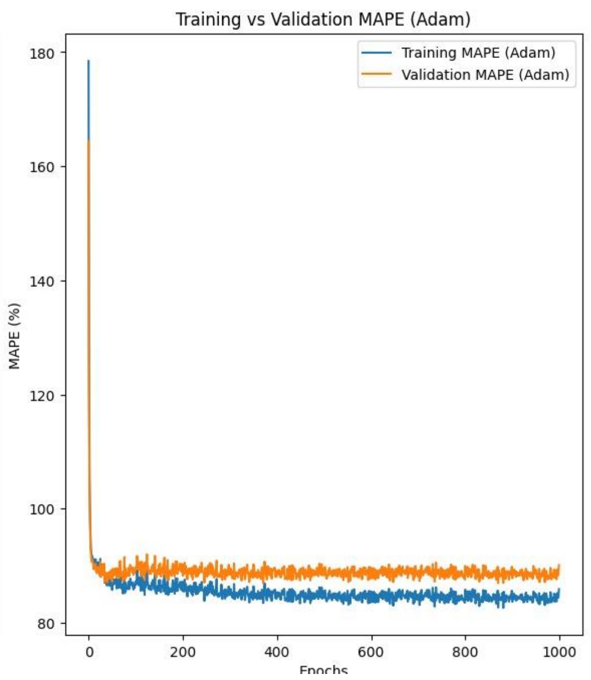




```
Final Test Loss: 0.1722, Test MAPE: 0.99%
```

Bonus Section





Using Adam optimization

1. Training MAPE Observations:
    - Adam Optimizer: The plot for Adam typically demonstrates a sharper initial decline in training MAPE compared to the SGD with momentum curve from the earlier plot. This behavior aligns with Adam's known tendency for quicker convergence.
    - SGD with Momentum: In the previous visualization, the line representing SGD with momentum (blue) likely exhibited a more gradual reduction in training MAPE over the epochs.
2. Validation MAPE Insights:
    - Adam Optimizer: Ideally, the validation MAPE for the Adam optimizer in the new plot should show a consistent downward trajectory, potentially achieving lower MAPE values more quickly than the SGD with momentum. This is due to Adam's effectiveness in managing noisy gradients and tackling non-convex optimization challenges.
    - SGD with Momentum: The previous plot for SGD with momentum (red line) may have displayed greater fluctuations in validation MAPE, possibly indicating a slower or less consistent downward trend compared to Adam.
3. Overall Convergence Comparison:
   The results using Adam likely reveal faster convergence on both training and validation MAPE compared to those using SGD with momentum. This supports Adam's reputation for accelerated convergence, particularly in scenarios where SGD with momentum may face difficulties.

In summary, the Adam optimization algorithm tends to provide quicker and more stable convergence while being less sensitive to hyperparameter choices compared to SGD with momentum. However, it may demand more memory due to the additional computations involved. Conversely, while SGD with momentum is simpler, it may take longer to converge and requires careful tuning of hyperparameters to achieve optimal results.