Reflection Report: Model Comparison and Improvement Analysis

Model	Metric	Train	Test	Total Params	Training Time(s)	Model Details
Baseline	R2 Score	0.82	0.82	2209	114.53	2 hidden layers, ReLU activation, 10 epochs, batch size 32
Three HL	R2 Score	0.9	0.9	5953	115.73	3 hidden layers, ReLU activation, 10 epochs, batch size 32
Four HL	R2 Score	0.93	0.93	17537	405.79	4 hidden layers, ReLU activation, 50 epochs, batch size 32
Regularized model1	R2 Score	0.93	0.93	18433	323.97	4 hidden layers with batch normalization & dropout after each activation, L2 regularization, ReLU, 100 epochs, batch size 32, early stopping
Regularized model2	**R2 Score**	**0.92**	**0.91**	**6529**	**282.34**	4 hidden layers with batch normalization & dropout after 2nd hidden layer only, L2 regularization, ReLU, 100 epochs, batch size 32, early stopping
leaky relu	R2 Score	0.93	0.93	18433	416.98	4 hidden layers with Leaky ReLU, batch normalization & dropout after 2nd hidden layer only, L2 regularization, 100 epochs, batch size 32, early stopping

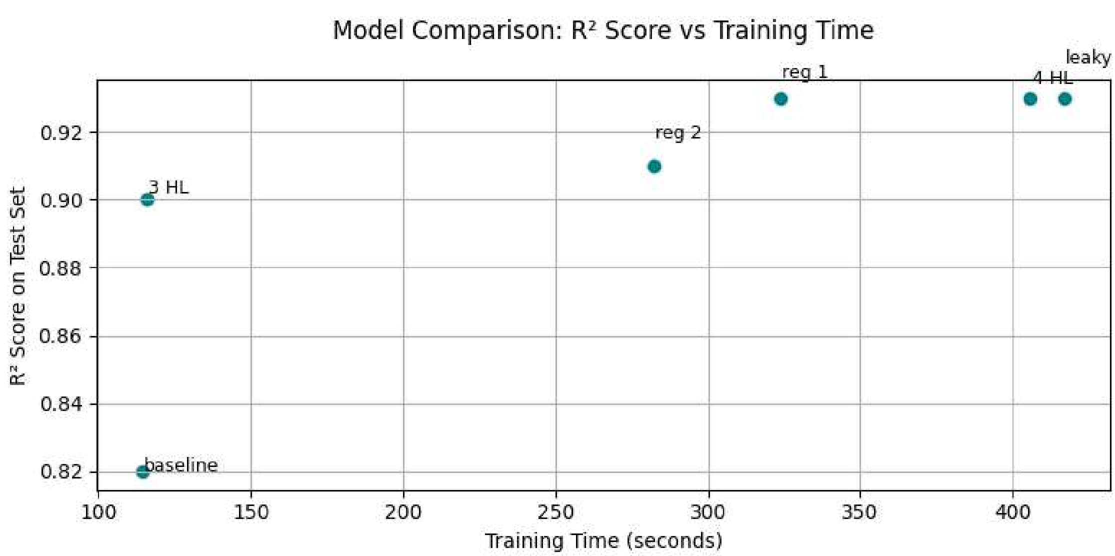

Fig 1.1. R2 Score (Test Set) VS training time

I began with a original model as baseline using **2 hidden layers**, ReLU activation, and a total of **2,209 parameters**. This model achieved an **R^2 score of 0.82** on the test set with a training time of approximately **115 seconds**.

To enhance performance, I experimented by gradually increasing the number of hidden layers and applying advanced techniques like **batch normalization**, **dropout**, and **regularization**. As seen in both the results table and scatter plot, deeper architectures led to better R^2 scores, although at the cost of increased training time.

Key model performance comparisons:

- **Three HL** (3 layers): $R^2 = 0.90$ | Time ≈ 116s
- **Regularized Model 2** (4 layers): R^2 = **0.91** | Time ≈ 282s
- **Regularized Model 1** (4 layers): R^2 = **0.93** | Time ≈ 324s
- **Four HL** (4 layers): $R^2 = 0.93$ | Time ≈ 406s
- **Leaky ReLU Model**: $R^2 = 0.93$ | Time ≈ 417s

Although **Four HL** and the **Leaky ReLU model** slightly outperformed others, they required noticeably **more training time**. **Regularized Model 2**, on the other hand, provided an excellent balance between performance and efficiency, making it my selected **optimal architecture**.

This model incorporated **batch normalization and dropout** layers after the second hidden layer, combined with **L2 regularization**. These techniques help improve model generalization by reducing overfitting:

- **Dropout** randomly disables neurons during training, forcing the model to learn more robust features.
- **Batch normalization** stabilizes and accelerates training by normalizing activations.
- **L2 regularization** discourages large weight values, promoting simpler models.

In conclusion, while deeper networks can capture more complex patterns, the performance gains tend to plateau. **Regularized Model 2** stands out by achieving the **best balance between test performance and training time**. It maintains a strong **R^2 score of 0.91, MAPE 6.9, MSE 42559.13, and** while keeping training time relatively low compared to deeper, more complex models. This balance is supported by the **effective use of regularization, batch normalization, and dropout**, which together help prevent overfitting and enhance generalization. Among all tested architectures, **Regularized Model 2 offers the most practical trade-off**, making it the best choice for deployment where both performance and efficiency are critical.