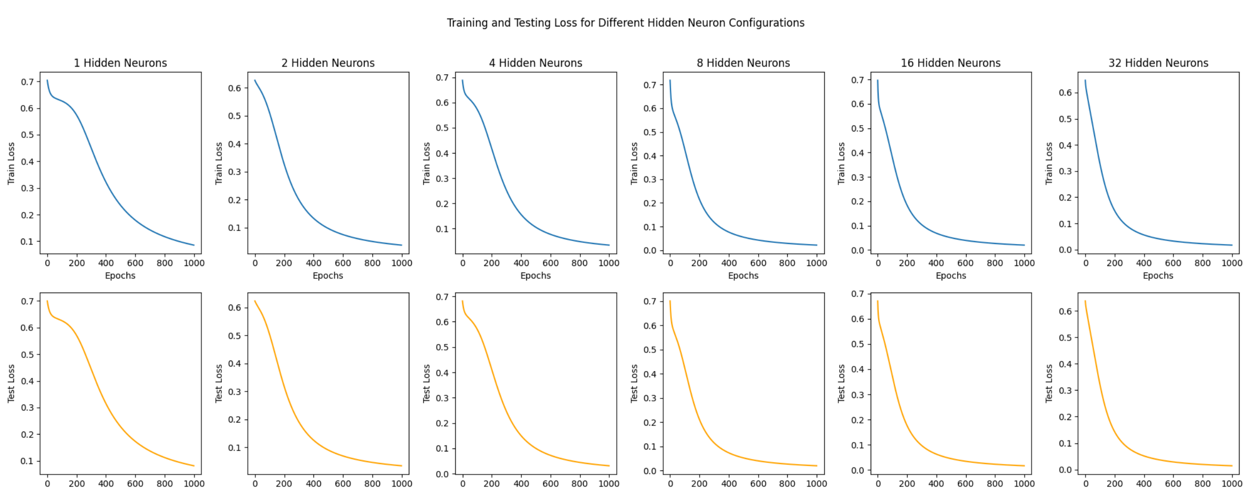
Lab 6 report (Neural Network)

- **Why is it important to use a random set of initial weights rather than initializing all weights as zero in a Neural Network?**
    - Initializing all weights to zero in a neural network would make all neurons learn the same features, leading to symmetry and preventing the network from learning effectively. Random initialization breaks this symmetry, allowing different neurons to learn unique features and enabling the network to converge to a useful solution.

- **How does a NN solve the XOR problem?**
    - A neural network solves the XOR problem by using hidden layers to create non-linear decision boundaries. The hidden layer allows the network to transform the input into a space where the XOR classes become separable. Through this transformation, the network can classify XOR inputs correctly.

- **Explain the performance of the different networks on the training and test sets. How does it compare to the logistic regression example? Make sure that the data you are refering to is clearly presented and appropriately labeled in the report**



- From the plots, we can observe the effect of increasing the number of hidden neurons on the training and test performance of the neural network. Across all configurations, the training and test losses decrease steadily over 1000 epochs, indicating that the models are learning to minimize error. However, the rate of convergence and final loss values vary slightly depending on the number of hidden neurons.

- **1 to 4 Hidden Neurons**: The models with fewer hidden neurons (1, 2, and 4) show a steady decline in both training and test losses. The loss values reach a reasonable low, indicating that the models are learning, though the convergence rate is slower compared to models with more hidden neurons.
- **8 to 32 Hidden Neurons**: Models with 8, 16, and 32 hidden neurons show faster convergence, achieving lower loss values early in training. The training and test losses drop sharply within the first few hundred epochs and continue to decrease gradually. This suggests that increasing the hidden neurons enables the network to capture more complex patterns, improving learning efficiency.
- **Comparison to Logistic Regression**: Logistic regression is a linear model, and therefore, it may not perform as well on data that requires non-linear decision boundaries. In contrast, the MLP with hidden layers provides greater flexibility to learn non-linear relationships. The MLP models, especially with more hidden neurons, show a clear improvement over logistic regression by minimizing losses further and learning more complex patterns in the data.

In summary, increasing the number of hidden neurons generally leads to faster convergence and lower training and test losses, improving the network's ability to capture complex relationships compared to logistic regression. However, this also increases model complexity, which may require careful tuning to avoid potential overfitting in different datasets