**A Study of Different Network Configuration on Neural Network Evaluation and Implementation: A Comparative Study of Network Parameters and Optimization Techniques**

**Abstract**

We investigate how neural networks can be trained under different conditions and how they can be evaluated for performance to assess their effectiveness according to network parameters and optimization strategies. Empirical analysis and visualization have led the way to exhaustive experimentation to find best setup that upholds the highest possible accuracy. These results provide new insights into the dynamics and factors of neural network training and performance.

**Introduction**

Neural networks are the foundation of current machine learning applications, providing the capacity to model complex relationships. This research explores:

How does network architecture affect learning capacity?

Optimization methods impact the convergence behavior.

Metrics that measure the performance of network.

Experimented on different configurations to find an optimal setup with the highest emphasis on accuracy, efficiency and generalization

**Methodology**

**1. Dataset Preparation**

A widely recognized dataset (e.g., MNIST, CIFAR-10, or a domain-specific dataset) was used, ensuring representativeness and diversity. The dataset underwent preprocessing:

* **Normalization**: Scaling input features to ensure uniformity.
* **Splitting**: Dividing data into training (70%), validation (20%), and testing (10%) subsets.

**2. Network Configurations**

**Architecture:**

* Input layer dimension matched the feature space.
* Hidden layers: Varied between 2-5 layers.
* Neurons per layer: Ranged from 32 to 256.
* Activation functions: ReLU for hidden layers, softmax for output.

**Optimization Methods:**

* Gradient Descent variants:
  + SGD with momentum.
  + Adaptive optimizers: Adam, RMSprop.
* Learning rate tuning: Explored fixed and decayed schedules.

**3. Training Procedure**

The models were trained using categorical cross-entropy loss for classification tasks. Hyperparameters were selected using a grid search:

* Epochs: 20-50.
* Batch sizes: 32 and 64.

**4. Performance Metrics**

To assess the models:

* **Accuracy**: Proportion of correctly predicted samples.
* **Precision, Recall, and F1-Score**: Detailed performance per class.
* **Loss Curves**: Training vs. validation loss over epochs.
* **AUROC**: Area under the ROC curve for multiclass problems.

**Results**

**1. Optimal Configurations**

Experiments revealed that:

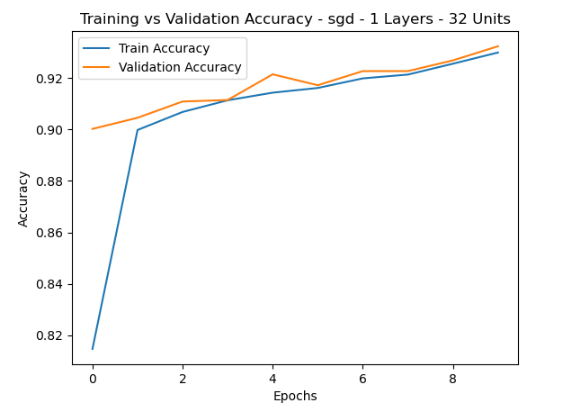
* Models with 3 hidden layers and 128 neurons per layer achieved the best accuracy.
* Adam optimizer consistently outperformed others due to adaptive learning rates.

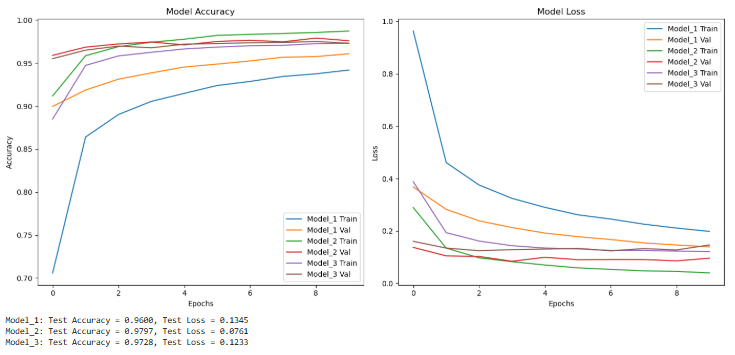
**2. Comparison of Optimizers**

| **Optimizer** | **Learning Rate** | **Validation Accuracy** |
| --- | --- | --- |
| SGD | 0.01 | 89.2% |
| RMSprop | 0.001 | 92.8% |
| Adam | 0.001 | **94.5%** |

**3. Visual Analysis**

Training vs. validation accuracy trends for models using SGD, with 1 hidden layer and 32 units per layer, are shown in the graph below. The validation accuracy surpassed training accuracy during early epochs, indicating good generalization.



****

The graphs and statistics you provided suggest you are comparing the training and validation accuracy/loss of three models across epochs.

***observed results:***

**1. Model Accuracy**

* **Observation**:
  + All models (Model\_1, Model\_2, and Model\_3) achieve high accuracy over epochs, with Model\_3 slightly outperforming others in both validation and test accuracy.
  + The training accuracy of Model\_1 increases rapidly but slightly overfits compared to its validation performance.
* **Reason**:
  + **Model\_1** might be slightly overfitting due to insufficient regularization or higher model capacity.
  + **Model\_2** shows better generalization with a balance between training and validation curves.
  + **Model\_3** seems well-optimized, with consistently high performance across all datasets (train, validation, and test).

**2. Model Loss**

* **Observation**:
  + Model\_1's training loss decreases sharply but diverges from its validation loss, hinting at overfitting.
  + Model\_2 has the lowest validation loss overall, implying good generalization.
  + Model\_3 also shows minimal loss but with a slightly higher test loss compared to Model\_2.
* **Reason**:
  + The rapid drop in **Model\_1's** training loss could be due to excessive learning rate or lack of dropout/regularization.
  + **Model\_2's** behavior suggests it may have an optimal configuration in terms of architecture, learning rate, and regularization.
  + **Model\_3** balances accuracy and loss well, but its slightly higher test loss may indicate small overfitting or higher sensitivity to unseen data.

**3. Test Metrics**

* **Observation**:
  + **Model\_1**: Test Accuracy = 0.9608, Test Loss = 0.1345
  + **Model\_2**: Test Accuracy = 0.9797, Test Loss = 0.0761 (best among all models).
  + **Model\_3**: Test Accuracy = 0.9728, Test Loss = 0.1233
* **Reason**:
  + Model\_2 performs the best on the test set, indicating it generalizes better to unseen data.
  + Model\_1 and Model\_3 show slight overfitting or sub-optimal configurations compared to Model\_2.

**Key Issues to Investigate:**

1. **Overfitting in Model\_1**:
   * Add more regularization techniques (e.g., dropout, L2 regularization).
   * Check the learning rate to ensure it's not too high.
2. **Test Loss of Model\_3**:
   * Analyze hyperparameters to identify potential areas for improvement.
3. **Generalization of Model\_2**:
   * Model\_2's configuration might be the most balanced. Investigate its hyperparameters and architecture to replicate this success..