Detailed Report on Custom Local-Global Feature Model for MNIST Classification

Introduction
This report presents the design, implementation, and evaluation of a custom neural network architecture for classifying handwritten digits from the MNIST dataset. The architecture focuses on capturing both local and global features in the images to improve classification performance. The report includes a mathematical formulation of the model, performance metrics, and a comparison with a baseline model.

Problem Statement
The task was to design a custom neural network that:
1. **Captures Local Details**: Extracts and learns features from small neighborhoods of pixels.
2. **Captures Global Details**: Correlates local features to understand the overall structure of the image.
3. **Unified Architecture**: Combines local and global feature extraction into a single model.

The goal was to achieve better performance than a baseline model and analyze the results.

Methodology
1. **Data Loading and Preprocessing**: The MNIST dataset was loaded and normalized. The dataset was split into training, validation, and test sets.
2. **Model Architecture**:
   - **Patch Extraction**: Random patches were extracted from the input images.
   - **Local Feature Encoder**: Each patch was processed through dense layers to extract local features.
   - **Global Feature Aggregator**: Self-attention mechanisms were used to capture relationships between local features and aggregate them into a global representation.
   - **Classification**: The global representation was passed through dense layers to produce the final classification.
3. **Training**: The model was trained using the Adam optimizer and categorical cross-entropy loss.
4. **Evaluation**: Performance metrics such as accuracy, precision, recall, and F1 score were computed. Misclassified examples were analyzed to understand the model's shortcomings.

Mathematical Formulation

1. **Local Feature Extraction**:
   - Given an input image $X \in \mathbb{R}^{28 \times 28}$, extract N patches $\{p_1, p_2, \ldots, p_N\}$.
   - For each patch p_i, compute a local feature representation:

$$f_i = \phi(W_2 \cdot \mathrm{ReLU}(W_1 \cdot \mathrm{flatten}(p_i) + b_1) + b_2)$$

   - This produces a set of local feature vectors $F = \{f_1, f_2, \ldots, f_N\}$.
2. **Global Feature Aggregation**:
   - Apply self-attention to capture relationships between local features:

$$A = \mathrm{MultiHeadAttention}(F, F)$$

$$F' = \mathrm{LayerNorm}(F + A)$$

   - Further process the attended features:

$$G = \mathrm{LayerNorm}(W_G \cdot F' + b_G)$$

   - Aggregate the global features into a single representation:

$$g = \frac{1}{N} \sum_{i=1}^{N} G_i$$

3. **Classification**:
   - Pass the global representation g through dense layers:

$$h = \mathrm{ReLU}(W_h \cdot g + b_h)$$

$$h' = \mathrm{Dropout}(h)$$

$$h'' = \mathrm{LayerNorm}(h')$$

$$y = \mathrm{softmax}(W_y \cdot h'' + b_y)$$

Results and Analysis

1. **Baseline Model**:
   - Test Accuracy: 0.9806
   - Training Time: 81.47 seconds
   - Inference Time: 0.68 seconds

2. **Local-Global Model**:
   - Test Accuracy: 0.8761
   - Training Time: 557.44 seconds
   - Inference Time: 1.79 seconds

3. **Performance Metrics**:
   - Accuracy: 0.8761
   - Precision: 0.8751
   - Recall: 0.8749
   - F1 Score: 0.8746

4. **Misclassified Examples**:
   - The model struggled with digits that have similar shapes, such as 4 and 9, 2 and 7, and 3 and 8.
   - Misclassifications suggest that the model may not be capturing subtle differences in the curvature or orientation of certain digits.

Comparison with Baseline Model

- **Performance Metrics**: The baseline model outperformed the custom model in all metrics (accuracy, precision, recall, F1 score).
- **Training and Inference Time**: The custom model took significantly longer to train and infer compared to the baseline model.
- **Confusion Matrix**: The confusion matrix difference showed that the custom model had more misclassifications across all digit classes.

Conclusion and Recommendations

1. **Model Shortcomings**:
   - The custom model's performance was lower than the baseline model, indicating that the local-global feature extraction approach may not be as effective for the MNIST dataset.
   - The model struggled with distinguishing between digits that have similar shapes.

2. **Improvement Suggestions**:
   - **Data Augmentation**: Introduce data augmentation techniques to increase the diversity of the training data.
   - **Advanced Architectures**: Experiment with more advanced architectures like Convolutional Neural Networks (CNNs) that are better suited for image data.
   - **Class Weighting**: Adjust the class weights to give more importance to harder-to-classify examples.

   - **Ensemble Methods**: Use ensemble methods to combine the predictions of multiple models, which can help in reducing misclassifications.

Final Thoughts

The custom local-global feature model achieved a lower accuracy compared to the baseline model, highlighting the challenges in designing effective feature extraction mechanisms for image data. Future work should focus on improving the model's ability to capture subtle differences between similar digits and reducing the computational complexity of the model.