Machine Learning Project Report

Arjun Krishnan (2022A4PS0617G)
Padi Sudeep (2022A4PS0527G)
Chaitanya Joshi (2022A7PS0730G)
Satyam Shukla (2022A3PS0527G)

1. Problem Statement

The objective of this project is to evaluate the capability of current Machine Learning (ML) algorithms in performing image transformations based on visual patterns, specifically using the IPARC dataset.

The hypothesis is: **A fully connected neural network can accurately predict which operator was used in an image transformation defined by a simple morphological operation.** The extent to which the ML model can achieve high accuracy in classifying input-output image pairs into any one of the eight different operations will determine the validity of this hypothesis.

2. Methodology

2.1 Data Preparation

- The dataset comprises multiple JSON files, each containing pairs of input and output images (15x15 pixels each) along with a transformation label.
- Transformations are defined by eight different operations using a structuring element (SE) for morphological processing (dilation using SE1–SE8).
- Preprocessing steps include:
  - Normalizing pixel values.
  - Converting the data into tensors suitable for deep learning models.

2.2 Model Architecture

- A neural network with three fully connected layers:
  - **Input Layer:** Flattens the input (2x15x15) images into a single-dimensional array.
  - **Hidden Layers:** Two hidden layers with ReLU activation.
  - **Output Layer:** Outputs one of the eight transformation types, using softmax activation.

2.3 Training Process

- The dataset was split into training (90%) and testing (10%) subsets.
- **Loss Function:** Negative Log-Likelihood Loss (NLLLoss).
- **Optimizer:** Adadelta optimizer with a learning rate scheduler to adjust the learning rate dynamically.
- **Evaluation Metrics:** Accuracy and loss per transformation class.

2.4 Adadelta Optimizer

The Adadelta optimizer was chosen for this project due to its advantages over traditional stochastic gradient descent (SGD). Adadelta is an extension of the Adagrad algorithm, which adapts the learning rate for each parameter individually, but unlike Adagrad, it does not accumulate the squared gradients over all time steps. Instead, Adadelta maintains a moving window of past squared gradients, which allows the learning rate to adjust dynamically over time.

Adadelta has the following advantages:

- **No Need for Manual Learning Rate Tuning:** It adjusts the learning rate dynamically based on the gradients, eliminating the need to tune the learning rate manually.
- **Efficient Memory Usage:** By using a moving average of squared gradients, it uses significantly less memory compared to storing all historical gradients.
- **Faster Convergence:** It has been shown to converge faster in many machine learning tasks compared to SGD, especially in cases where the dataset or model is large and complex.

For these reasons, Adadelta is well-suited for this task, where image transformations are learned, and the model needs to adapt quickly to various input patterns.

2.5 Model Selection

Various configurations were tested by changing the number of layers, batch size, learning rate, and gamma. Results are summarized below:

Model	Layers	Batch Size	Gamma	Learning Rate	ReLU
D1	2	32	0.7	0.5	Yes
D2	3	32	0.7	0.5	No
D3	4	64	0.6	1.0	Yes
D4	3	32	0.8	0.6	Yes

Table 1: Model configurations and key hyperparameters.

Based on average accuracy across all operations, **D1** performed best.

3. Experimental Results

3.1 Model Performance

The model was trained for 20 epochs, adjusting the learning rate dynamically using a learning rate scheduler with a gamma value of 0.7. The training accuracy increased while the training loss decreased over epochs.

Structuring Element (SE)	**Test Accuracy**
SE1	72.57%
SE2	53.09%
SE3	49.23%
SE4	69.48%
SE5	99.80%
SE6	99.79%
SE7	100.00%
SE8	100.00%

Table 2: Test accuracy for each structuring element (SE) of the D1 model.

3.2 Error Analysis

- The model performed best on simpler transformations such as SE5, SE6, SE7, and SE8, which involve straightforward visual patterns.
- It struggled with more complex transformations like SE3 and SE2, achieving the lowest accuracy for these operations.
- This indicates the need for more sophisticated feature extraction methods, such as convolutional layers, to handle intricate visual patterns.

4. Conclusion and Future Work

4.1 Conclusion

The hypothesis that "a fully connected neural network can accurately predict which operator was used in an image transformation defined by a simple morphological operation" is **FALSE**. The neural network demonstrated high accuracy for simpler transformations but failed to generalize effectively across all operations, especially those with complex patterns.

4.2 Future Work

- **Architectural Enhancements:** Introduce convolutional neural networks (CNNs) for better feature extraction.
- **Transfer Learning:** Incorporate pre-trained models to enhance training efficiency and generalization.

- **Hybrid Approaches:** Combine domain-specific symbolic reasoning with ML techniques for cases where neural networks fail.
- **Dataset Expansion:** Explore larger and more diverse datasets to test the scalability of the approach.

5. References

- The code for this project was adapted from: `https://github.com/pytorch/examples/blob/main/mnist/main.py`, with modifications made to match our specific requirements.