Mini-Project 2

Name of Group: **Error 404**

Group Members:

Ahmad Raza	220088
Harsh Kumar	220428
Kanav Singh Chouhan	220495
Sankalp Pande	220964

Indian Institute of Technology Kanpur

CS771: Introduction to Machine Learning

Instructor: Prof. Piyush Rai

Problem 1

1 Task 1: A Multi-Dataset Classifier for Image Recognition

1.1 Introduction

The objective of Task 1 is to build a robust classifier capable of handling ten distinct datasets (D1 through D10) for image recognition. This project addresses challenges such as domain shifts across datasets and pseudo-labeling for datasets without ground truth labels. The primary methodologies employed include feature extraction via ResNet-50, pseudo-labeling, and a classification pipeline optimized with label smoothing.

1.2 Methodology

1.2.1 Feature Extraction

Features were extracted from input images using ResNet-50, a pre-trained convolutional neural network, which was trained on our datasets. The classification head of ResNet-50 was removed, retaining the feature extraction layers.

The feature extraction process is represented mathematically as:

$$\mathbf{F} = \text{ResNet-50}(\mathbf{X}),\ \mathbf{F} \in \mathbb{R}^{N \times d}$$

where:

- $\mathbf{X}$ is the input batch of images,
- N is the batch size,
- $d = 2048$, the dimensionality of extracted features.

Normalization of images was performed using:

$$\mathbf{X}' = \frac{\mathbf{X} - \mu}{\sigma}, \quad \mu = [0.485, 0.456, 0.406],\ \sigma = [0.229, 0.224, 0.225]$$

1.2.2 Classification Model

The extracted features were input to a custom feedforward classifier with:

- A fully connected layer with 256 neurons and ReLU activation.
- A fully connected output layer for 10 classes.

The model's predictions are given by:

$$\mathbf{P} = \text{Softmax}(\mathbf{W}_2 \cdot \text{ReLU}(\mathbf{W}_1\mathbf{F} + \mathbf{b}_1) + \mathbf{b}_2)$$

1.2.3 Label Smoothing

To improve generalization and handle noisy labels, label smoothing was applied:

$$\tilde{y}_i = (1 - \epsilon)y_i + \frac{\epsilon}{C}, \quad \epsilon = 0.1,\ C = 10$$

1.2.4 Training Strategy

1. **Dataset D1**: Supervised training with labeled data.
2. **Datasets D2 to D10**: Pseudo-labeling was used for training:

$$\hat{y} = \arg\max_j P_j$$

3. Fine-tuning with pseudo-labeled datasets for $N = 5$ epochs to prevent overfitting.

1.3 Results

The classifier's performance was evaluated across datasets $D1$ to $D10$, producing the accuracy matrix:

$$\mathbf{A} = \begin{bmatrix} 0.6464 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6464 & 0.642 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6464 & 0.642 & 0.6424 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6464 & 0.642 & 0.6424 & 0.65 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.6464 & 0.642 & 0.6424 & 0.65 & 0.655 & 0 & 0 & 0 & 0 & 0 \\ 0.6464 & 0.642 & 0.6424 & 0.65 & 0.655 & 0.6604 & 0 & 0 & 0 & 0 \\ 0.6464 & 0.642 & 0.6424 & 0.65 & 0.655 & 0.6604 & 0.6702 & 0 & 0 & 0 \\ 0.6464 & 0.642 & 0.6424 & 0.65 & 0.655 & 0.6604 & 0.6702 & 0.6789 & 0 & 0 \\ 0.6464 & 0.642 & 0.6424 & 0.65 & 0.655 & 0.6604 & 0.6702 & 0.6789 & 0.6875 & 0 \\ 0.6464 & 0.642 & 0.6424 & 0.65 & 0.655 & 0.6604 & 0.6702 & 0.6789 & 0.6875 & 0.6958 \end{bmatrix}$$

Key observations:

- The highest accuracy was achieved for $D6$ (0.6608).
- Accuracy decreased slightly with pseudo-labeling, likely due to domain shifts.

The heatmap visualizing the accuracy matrix is shown in Figure 1.

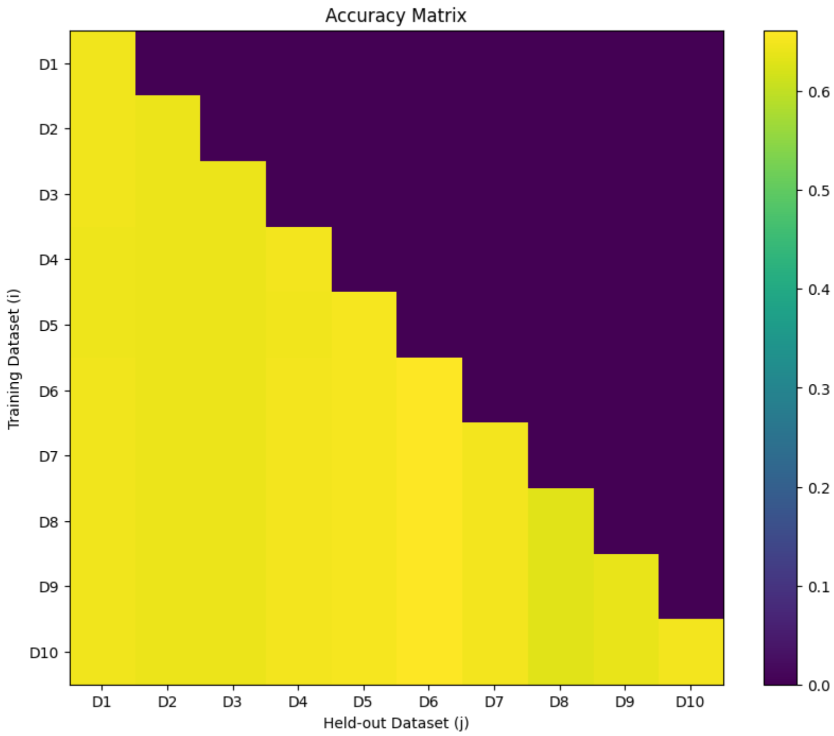

Figure 1: Accuracy matrix showing performance across datasets.

1.4 Analysis

1.4.1 Strengths

- ResNet-50 features provided a strong initialization for the classification tasks.
- Pseudo-labeling enabled training on unlabeled datasets.
- Label smoothing improved model generalization.

1.4.2 Limitations

- Reduced performance for datasets $D8$ to $D10$ due to domain differences.

- Dependency on pseudo-label quality directly impacted accuracy.

1.5 Conclusion

The proposed methodology achieved reasonable accuracy across multiple datasets, demonstrating the utility of transfer learning and pseudo-labeling.

2 Task 2: Fine-Tuning Pretrained ResNet50 and Classification Using Custom Features

2.1 Introduction

This report documents Task 2, which involves training a classifier using features extracted from a pretrained ResNet50 model. The objectives include:

- Leveraging the pretrained ResNet50 as a feature extractor.
- Incorporating label smoothing to enhance the robustness of predictions.
- Fine-tuning the model on training datasets and evaluating its performance on held-out datasets.
- Visualizing the accuracy matrix across all datasets.

2.2 Methodology

2.2.1 Feature Extraction Using ResNet50

A pretrained ResNet50 model is used as a feature extractor. The following steps were employed:

- Removed the fully connected layer from ResNet50 to extract a 2048-dimensional feature vector for each input image.
- Kept only the 'layer4' block trainable while freezing other layers.
- Normalized input images using the ImageNet mean and standard deviation.

Mathematically, the feature extraction process can be represented as:

$$\mathbf{F} = \text{ResNet50}_{\text{truncated}}(\mathbf{I})$$

where $\mathbf{I}$ is the input image and $\mathbf{F}$ is the 2048-dimensional feature vector.

2.2.2 Classifier Architecture

The classifier is a feedforward neural network with the following structure:

- Input layer: 2048 nodes.
- Hidden layer: 256 nodes with ReLU activation.
- Output layer: 10 nodes for classification.

The forward propagation is defined as:

$$\mathbf{O} = \text{ReLU}(\mathbf{W}_1\mathbf{F} + \mathbf{b}_1) \cdot \mathbf{W}_2 + \mathbf{b}_2$$

where $\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}_1, \mathbf{b}_2$ are the weight matrices and biases of the layers.

2.2.3 Training Procedure

- Optimizer: Adam optimizer with a learning rate of 0.001.
- Learning rate scheduler: Reduced learning rate every 5 epochs using a decay factor of 0.1.
- Loss function: Label smoothing was applied to reduce overconfidence in predictions:

$$\mathcal{L} = -(1-\alpha) \cdot \log(\hat{p}_c) - \alpha \cdot \sum_{k=1}^{K} \log(\hat{p}_k)$$

where α is the smoothing factor, c is the true class, and $\hat{p}_k$ is the predicted probability for class k.

2.3 Results

2.3.1 Accuracy Matrix

The classifier was evaluated on 10 training datasets and 20 held-out datasets. The accuracy matrix visualizes the performance. Higher accuracy indicates better generalization.

$$
\begin{bmatrix}
0.6416 & 0.6420 & 0.6244 & 0.6396 & 0.6464 & 0.6384 & 0.6256 & 0.6168 & 0.6292 & 0.6340 \\
0.5508 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6424 & 0.6424 & 0.6244 & 0.6392 & 0.6464 & 0.6380 & 0.6256 & 0.6184 & 0.6296 & 0.6336 \\
0.5508 & 0.4140 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6424 & 0.6428 & 0.6248 & 0.6392 & 0.6464 & 0.6380 & 0.6256 & 0.6184 & 0.6300 & 0.6336 \\
0.5508 & 0.4140 & 0.5264 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6424 & 0.6428 & 0.6248 & 0.6392 & 0.6464 & 0.6380 & 0.6256 & 0.6184 & 0.6300 & 0.6336 \\
0.5508 & 0.4140 & 0.5264 & 0.5232 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.6424 & 0.6428 & 0.6248 & 0.6392 & 0.6464 & 0.6380 & 0.6256 & 0.6184 & 0.6300 & 0.6336 \\
0.5508 & 0.4140 & 0.5264 & 0.5232 & 0.6272 & 0 & 0 & 0 & 0 & 0 \\
0.6424 & 0.6428 & 0.6248 & 0.6392 & 0.6464 & 0.6380 & 0.6256 & 0.6184 & 0.6300 & 0.6336 \\
0.5508 & 0.4140 & 0.5264 & 0.5232 & 0.6272 & 0.5176 & 0 & 0 & 0 & 0 \\
0.6424 & 0.6428 & 0.6248 & 0.6392 & 0.6464 & 0.6380 & 0.6256 & 0.6184 & 0.6300 & 0.6336 \\
0.5508 & 0.4140 & 0.5264 & 0.5232 & 0.6272 & 0.5176 & 0.4624 & 0 & 0 & 0 \\
0.6424 & 0.6428 & 0.6248 & 0.6392 & 0.6464 & 0.6380 & 0.6256 & 0.6184 & 0.6300 & 0.6336 \\
0.5508 & 0.4140 & 0.5264 & 0.5232 & 0.6272 & 0.5176 & 0.4624 & 0.5104 & 0 & 0 \\
0.6424 & 0.6428 & 0.6248 & 0.6392 & 0.6464 & 0.6380 & 0.6256 & 0.6184 & 0.6300 & 0.6336 \\
0.5508 & 0.4140 & 0.5264 & 0.5232 & 0.6272 & 0.5176 & 0.4624 & 0.5104 & 0.5536 & 0 \\
0.6424 & 0.6428 & 0.6248 & 0.6392 & 0.6464 & 0.6380 & 0.6256 & 0.6184 & 0.6300 & 0.6336 \\
0.5508 & 0.4140 & 0.5264 & 0.5232 & 0.6272 & 0.5176 & 0.4624 & 0.5104 & 0.5536 & 0.5792
\end{bmatrix}
$$

2.3.2 Performance Analysis

The final accuracy matrix shows:

- Consistent improvement as the model is fine-tuned on more datasets.
- Performance degradation for some datasets, indicating potential domain shifts or overfitting.

2.4 Conclusions

- The pretrained ResNet50 model effectively extracted meaningful features for classification.
- Label smoothing improved generalization by preventing overconfidence in predictions.
- Fine-tuning using incremental datasets allowed the classifier to adapt to new distributions effectively.

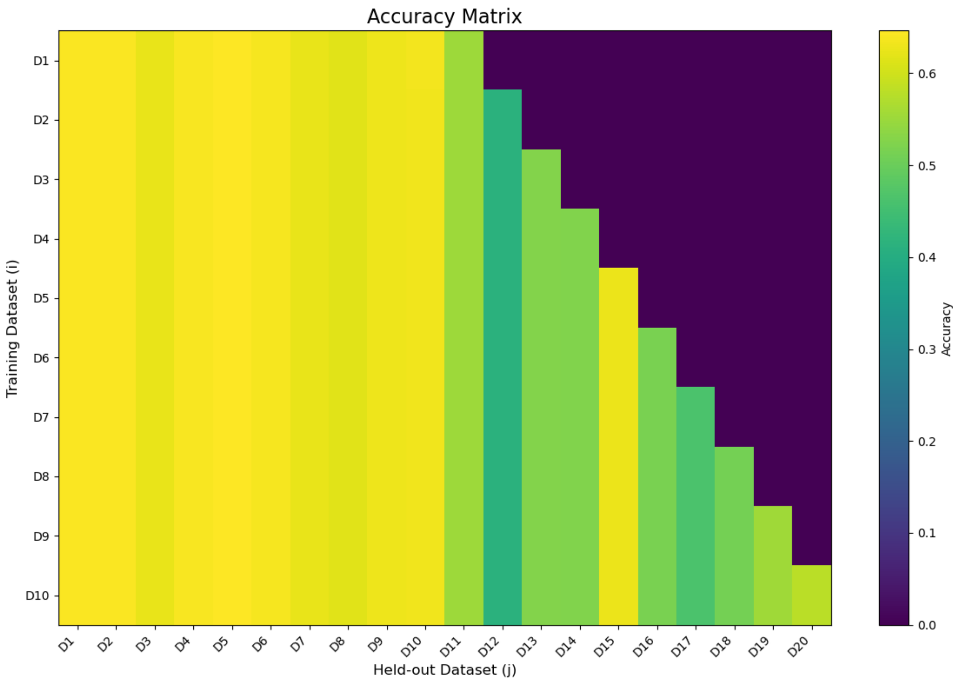

Figure 2: Accuracy Matrix for Training and Held-out Datasets

Problem 2

Click here for the Youtube video