Intro to ML (Autumn 2024): Mini-project 2

Akshay Toshniwal (241110005), Devang Agarwal (241110019),
Prakhar Mandloi (241110051), Roshan Kumar (241110058), Suyamoon Pathak (241110091)

1 Exploratory Data Analysis

The dataset includes 20 CIFAR-10 subsets, with D1–D10 sharing a common input distribution and D11–D20 having distinct distributions. However, upon closer examination, it was observed that all the datasets from D_{11} to D_{20} contained the same images but with various transformations applied to them. These transformations included blurring, tweaking sharpness, saturation, and contrast, as well as introducing missing or white pixels in some images, resulting in each dataset having different distribution.

- **Approaches Taken:** For both Task 1 and Task 2, we experimented with feature extraction using various pre-trained deep neural networks such as ResNet18, ResNet50, VGG19, and EfficientNet models. After extracting features, two approaches for LwP were considered: one using Euclidean distance and another using Mahalanobis distance. In Task 1, Mahalanobis distance provided slightly better accuracy at the cost of increased computation time. In Task 2, an LwP model with Euclidean distance, initialized with the previous prototypes, achieved better accuracies than Mahalanobis distance. This is likely because the datasets in Task 2 contained transformations and noise that reduced the reliability of covariance matrices, making the simpler Euclidean distance more robust.

2 Links to Extracted Features:

Kindly download the following extracted features from the link below. For D1 to D10, store them in the same folder as the code with the folder name 'feat_embedding_1_10'.

For D11 to D20, store them in the same folder as the code with the folder name 'feat_embedding_11_20'.

For datasets D1 to D10: https://drive.google.com/file/d/1BQCdCQfBXaNxz6023J2jTHq6FiW_CZkE/view?usp=sharing

For datasets D11 to D20: https://drive.google.com/file/d/1IIoRo6UCqbmy5MjL2kATUcEZF8C8iu0q/view?usp=sharing

3 Problem 1: Task 1

The objective of Task 1 is to sequentially train a lightweight prototype-based classifier (LwP) on subsets of the CIFAR-10 dataset. The process begins with the labeled dataset D_1, and the model is iteratively updated using pseudo-labeled datasets D_2 to D_{10}. The key steps are as follows:

- **Feature Extraction:** A pre-trained EfficientNet-B3 model (trained on ImageNet-1k at a resolution of 300x300) with its fully connected (FC) layer removed was employed to extract feature representations. The architecture consists of several stages of convolutions, including the use of Mobile Inverted Bottleneck Convolutions (MBConv), depthwise separable convolutions, and squeeze-excitation blocks. For feature extraction, we specifically use the output of the adaptive average pooling layer after the last convolutional block. This pooling layer produces fixed-size feature vectors (with dimensions $1 \times 1 \times 1536$ for each image), which are then flattened to serve as the feature embeddings for the next LwP models. The extracted features were cached for computational efficiency, and transformations applied included:

- Conversion to PIL format (to ensure compatibility with `torchvision.transforms`, for transformations like resizing and normalization.) and resizing to 300×300 pixels (to match the input resolution expected by the pre-trained EfficientNet-B3 model).
- Normalization to the ImageNet mean and standard deviation (`mean` = [0.485, 0.456, 0.406]; `std` = [0.229, 0.224, 0.225]).

In the provided architecture diagram for EfficientNet-B3, the penultimate layer is the MBConv6, k3x3, IRC block at the $7 \times 7 \times 384$ resolution, located just before the Conv1x1 layer. This is the layer we are utilizing for feature extraction. After this, we add pooling, which reduces the feature map to a 1D embedding. The image is taken from the source: Classification_of_Remote_Sensing_Images_Using_EfficientNet-B3_CNN_Model_with_Attention

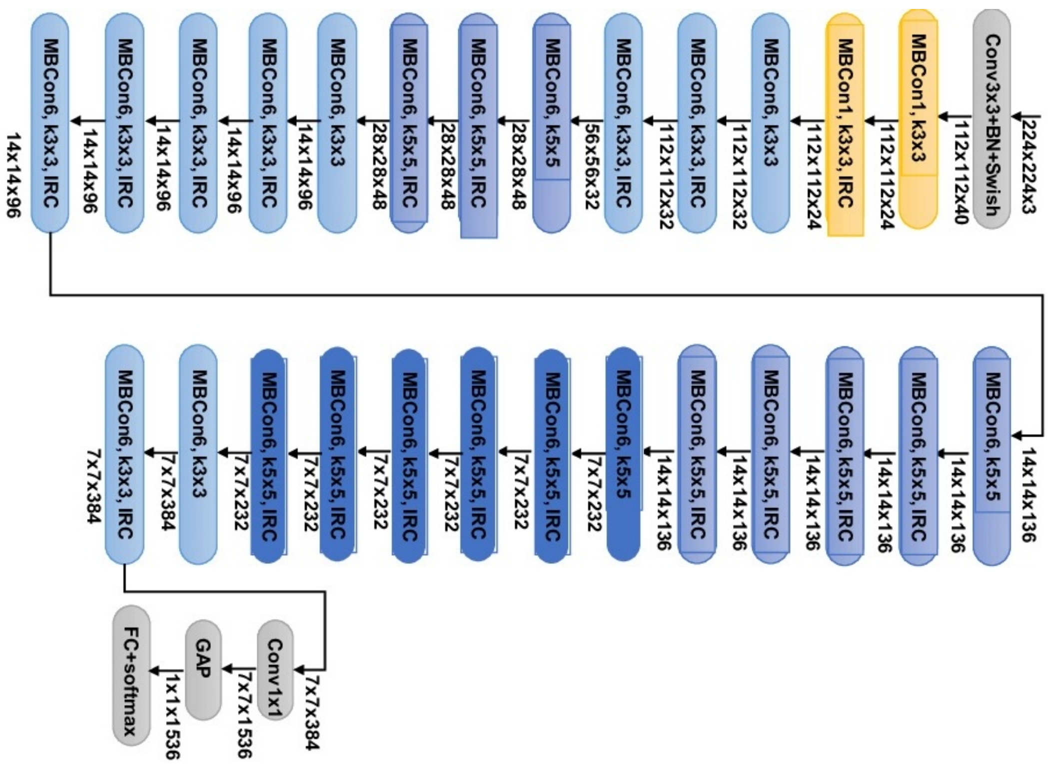

Figure 1: EfficientNet-B3 Architecture

- **Initial Training:** The labeled dataset D_1 was used to train the initial LwP classifier. The LwP classifier was designed to use **Mahalanobis distance** for classification, with the following components:
  - **Prototypes:** Mean feature vectors computed for each class.
  - **Covariance Matrices:** Regularized covariance matrices ($\text{cov} + 10^{-6} \cdot I$) were inverted and stored for each class.
- **Sequential Updates:** For each subsequent dataset D_i ($i = 2$ to 10), the following steps were performed:
  - Pseudo-labels for D_i were predicted using the current LwP model.
  - The classifier was updated iteratively by:

* **Prototype Update:** Class prototypes were refined using weighted averaging:

$$\mu_{\text{new}} = 0.85 \cdot \mu_{\text{old}} + 0.15 \cdot \mu_{\text{new data}}$$

where μ_{old} is the existing prototype and $\mu_{\text{new data}}$ is the mean feature vector from the new dataset.

* **Covariance Matrix Update:** Covariance matrices were updated by combining the old and new covariance inverses:

$$\Sigma_{\text{new}}^{-1} = 0.85 \cdot \Sigma_{\text{old}}^{-1} + 0.15 \cdot \Sigma_{\text{new data}}^{-1}$$

where Σ_{old}^{-1} is the inverse covariance matrix of the previous dataset, and $\Sigma_{\text{new data}}$ is computed as:

$$\Sigma_{\text{new data}} = \text{Cov}(X_{\text{new}}) + 10^{-6} \cdot I$$

The regularization term $10^{-6} \cdot I$ ensures numerical stability in the covariance computation.

- **Evaluation:** At each step, the model was evaluated on all the held-out datasets from $\hat{D}_1$ upto $\hat{D}_i$, corresponding to D_i, and their performance were recorded. Accuracies are reported in a matrix format, where each row corresponds to a model ($f_1, \dots, f_{10}$) and each column corresponds to a held-out dataset ($\hat{D}_1, \dots, \hat{D}_{10}$). The accuracy matrix obtained is shown in Table 1 below.

Table 1: 10x10 Accuracy Matrix for Task 1 Models

	$D\hat{1}$	$D\hat{2}$	$D\hat{3}$	$D\hat{4}$	$D\hat{5}$	$D\hat{6}$	$D\hat{7}$	$D\hat{8}$	$D\hat{9}$	$D\hat{10}$
f_1	90.96	-	-	-	-	-	-	-	-	-
f_2	91.24	92.72	-	-	-	-	-	-	-	-
f_3	91.00	93.04	91.80	-	-	-	-	-	-	-
f_4	90.48	92.96	91.76	91.44	-	-	-	-	-	-
f_5	90.28	92.84	91.76	91.08	91.24	-	-	-	-	-
f_6	89.88	92.32	91.68	91.04	91.28	91.08	-	-	-	-
f_7	89.92	92.08	91.40	90.96	91.12	90.76	90.76	-	-	-
f_8	89.48	91.80	91.16	90.92	90.68	90.68	90.68	90.96	-	-
f_9	89.00	91.32	90.72	89.96	90.32	90.04	90.20	90.20	89.56	-
f_{10}	88.72	90.80	90.36	89.60	89.76	89.52	89.40	89.72	89.28	89.32

4 Problem 1: Task 2

For this task, the process begins with the dataset D11 and the model is iteratively updated using the predicted labels of D11 upto D20. The key steps are as follows:

- **Feature Extraction:** The same pre-trained EfficientNet-B3 model, with its fully connected (FC) layer removed, was employed to extract feature representations. The extracted features were cached for computational efficiency, and similar transformations as task 1 were applied.
- **Sequential Updates:** For each subsequent dataset D_i ($i = 11$ to 20), pseudo-labels for D_i were predicted using the current LwP model (which now uses **Euclidean Distances**) and then the classifier was updated iteratively by updating the prototypes in the same way as done in task 1.
- **Evaluation:** At each step, the model was evaluated on all the held-out datasets from $\hat{D}_1$ upto $\hat{D}_i$, corresponding to D_i, and their performance were recorded. Accuracies are reported in a matrix format, where each column corresponds to a model ($f_1, \dots, f_{10}$) and each row corresponds to a held-out dataset ($\hat{D}_1, \dots, \hat{D}_{10}$). The accuracy matrix obtained is shown in Table 2 below.

Table 2: Final Accuracy Table for Models f_{11} to f_{20}

	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}	f_{17}	f_{18}	f_{19}	f_{20}
$D\hat{1}$	86.80	85.88	85.48	85.44	85.12	84.96	84.80	84.40	84.44	84.40
$D\hat{2}$	88.56	87.84	87.48	87.20	87.20	86.96	86.60	86.32	86.12	86.04
$D\hat{3}$	88.88	88.68	88.40	88.00	87.64	87.36	87.04	86.76	86.52	86.64
$D\hat{4}$	88.52	87.88	87.72	87.68	87.52	87.08	86.92	86.56	86.72	86.48
$D\hat{5}$	88.08	87.44	87.28	87.12	86.96	86.40	86.20	86.04	85.48	85.56
$D\hat{6}$	88.24	87.84	87.44	87.20	87.28	87.00	86.76	86.44	86.00	85.92
$D\hat{7}$	87.84	87.40	87.00	86.88	86.68	86.40	86.16	85.80	85.36	85.40
$D\hat{8}$	88.12	87.88	87.56	87.40	87.08	86.72	86.36	86.16	86.20	86.08
$D\hat{9}$	87.08	86.48	86.08	85.68	85.36	85.24	84.96	84.60	84.16	84.24
$D\hat{10}$	87.88	87.60	87.16	86.84	86.68	86.24	86.08	86.00	85.44	85.40
$D\hat{11}$	72.44	71.96	71.72	71.56	70.96	70.68	70.60	70.60	70.44	69.76
$D\hat{12}$	-	54.28	54.20	53.80	53.52	53.12	52.60	52.60	52.40	52.36
$D\hat{13}$	-	-	79.64	79.32	79.24	79.00	78.80	78.48	78.56	77.84
$D\hat{14}$	-	-	-	85.60	85.52	85.12	84.92	84.72	84.28	84.04
$D\hat{15}$	-	-	-	-	87.00	86.60	86.52	86.32	85.44	85.20
$D\hat{16}$	-	-	-	-	-	74.04	73.84	73.44	73.36	73.00
$D\hat{17}$	-	-	-	-	-	-	79.36	79.16	78.52	78.28
$D\hat{18}$	-	-	-	-	-	-	-	76.48	76.28	76.36
$D\hat{19}$	-	-	-	-	-	-	-	-	66.08	65.80
$D\hat{20}$	-	-	-	-	-	-	-	-	-	83.80

5 Problem 2

The link for our paper presentation's video is: https://youtu.be/xQdT7259VjI

6 Conclusion

In this project, we tackled the challenges of sequential learning and domain adaptation using subsets of the CIFAR-10 dataset. Starting with labeled data from D_1, we developed a lightweight prototype-based classifier (LwP) leveraging Mahalanobis distance for D_1 to D_{10}. For D_{11} to D_{20}, we used LwP with Euclidean distance to handle the variations in distributions. Both approaches utilized features extracted using a pre-trained EfficientNet-B3 model.

The iterative updates ensured consistent model performance across held-out datasets while balancing adaptation to new datasets. For the diverse distributions in D_{11} to D_{20}, we effectively addressed transformations such as blurring, changes in sharpness, contrast, and missing pixels. This approach demonstrated the feasibility of sequential learning in evolving data environments while minimizing performance degradation on earlier datasets.

References

[1] Lifelong Domain Adaptation via Consolidated Internal Distribution. NeurIPS 2021.

[2] Deja vu: Continual model generalization for unseen domains. ICLR 2023.

[3] Classification_of_Remote_Sensing_Images_Using_EfficientNet-B3_CNN_Model_with_Attention

[4] PyTorch: An open source machine learning framework. Retrieved from `https://pytorch.org/`.

[5] NumPy: The fundamental package for scientific computing with Python. Retrieved from `https://numpy.org/`.

[6] Scikit-learn: Machine learning in Python. Retrieved from `https://scikit-learn.org/stable/`.