CS771- Intro to ML (Autumn 2024): Mini-project 2

Abhinav Singh 220038

Aditya Mathur 220068

Patel Rhut Dipakbhai 220756

Sahil Kumar Goyat 220937

Darshan Dinesh Sethia 220326

1 Problem - 1

The task provides 20 datasets (D_1–D_{20}) derived from CIFAR-10, where D_1 is labeled and D_2–D_{20} are unlabeled. Datasets D_1–D_{10} share a common input distribution, while D_{11}–D_{20} have distinct but related distributions. Additionally, labeled held-out datasets for D_1–D_{20} are provided for evaluation. The objective is to train image classification models using the datasets and evaluate their performance solely on the labeled held-out datasets, ensuring a fair assessment. Feature extraction methods may be used for processing inputs.

1.1 Task - 1

We are provided with first 10 datasets (D_1–D_{10}) from CIFAR-10, where D_1 is labeled and D_2–D_{10} are unlabeled. Using an LwP classifier, train a model f_1 on D_1, predict labels for D_2, and update f_1 to f_2 using D_2 without revisiting D_1. Repeat this sequentially for D_3–D_{10} to learn models f_1 to f_{10} ensuring the number of parameters remains constant. Evaluate each model on its corresponding held-out dataset and previous datasets, maintaining consistent accuracy across all evaluations.

1.1.1 Feature extraction and pre-processing

In the prep-rocessing stage, we first applied a pre-trained **ViT model** (Click here to see the Model), which was trained on the ImageNet-21k dataset and fine-tuned on ImageNet 2012, to extract feature representations from the raw CIFAR-10 images. The model was configured with an input shape of $32 \times 32 \times 3$ to match the image dimensions. The output features, were then reshaped to the size of 768 - dimensional vector for each image.

Next, we applied Linear Discriminant Analysis (LDA) to reduce the dimensionality of these feature vectors. It is better than PCA as the later approach only focuses on linear variance maximization. LDA maximizes the distance between means of classes and also minimizes the variation within each class. This approach results in feature vectors with a final shape of 9 dimensions.

The primary motivation for these dimensionality reduction steps was to facilitate faster computations, particularly when applying the Learning with Prototype (LwP) approach. In LwP, the operations on matrix like multipication and inverse (of Covariance Matrix) are critical steps, and using lower-dimensional data ensures that this operation can be performed more efficiently. Thus, LDA was therefore crucial for reducing computational complexity while preserving the essential discriminative features required for accurate classification.

1.1.2 Gaussian Mixture Model

The Gaussian Mixture Model method uses some concepts of **LDAuCID algorithm**[2] to create a probabilistic counterpart of LwP used in previous method. In this approach, we represent a class not only using it's mean, but also using it's covariance and prior probability.

The internal distribution of data is expressed in form of GMM. These parameters of distribution are estimated using MAP estimation of GMM are given by:

$$\mu_k = \frac{1}{|S_k|}\sum_{i=1}^{|S_k|} x_i \qquad \Sigma_k = \frac{1}{|S_k|}\sum_{i=1}^{|S_k|}(x_i-\mu_k)^T(x_i-\mu_k) \qquad \pi_k = \frac{|S_k|}{N}$$

where, μ_k, Σ_k and π_k are mean, covariance and prior respectively (internal distribution) of a class whose count is $|S_k|$.

The training process begins by constructing our first model f_1. This is done by finding internal distribution using GMM estimates for class parameters of dataset D_1. Then, in this method, since we are not storing our model at each step, we test our model f_i on unseen datasets $\hat{D}_1$,..,$\hat{D}_i$ just before updation to predict their labels.

The process of model updation in LDAuCID algorithm uses a method called buffer replay which stores some train examples of previous datasets. Since we aren't allowed that, we are using **generative buffer**. In this, we generate some samples using found internal distribution. To get better examples, we only take those which are above a certain threshold τ which is a hyper-parameter.

$$\text{Buffer} = x_{bi} : \log \pi_k - \frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x_{bi}-\mu_k)^T{\Sigma_k}^{-1}(x_{bi}-\mu_k) > \tau$$

We then experience replay by taking union of these generated examples with the new dataset that we need to perform classification on. This method helps model to remember previous datasets.

$$D_{i+1} = D_{i+1} \cup X_b^i$$

where, X_b^i is a set of buffer samples generated from the learned multi-modal internal distribution. For prediction, We use the same rule as that used in Gaussian class-conditional:

$$\hat{y} = \arg\max_k \log \pi_k - \frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x-\mu_k)^T{\Sigma_k}^{-1}(x-\mu_k)$$

were π_k, μ_k, Σ_k are MAP estimates computed as a result of model updation.

As the model progresses, we generate more and more buffer samples to replay. For ith dataset, we generate $recall * (i-1)$ number of samples, where recall is a fixed hyper-parameter. The performance of all models on their corresponding test datasets is shown in below table:

	$\hat{D}_1$	$\hat{D}_2$	$\hat{D}_3$	$\hat{D}_4$	$\hat{D}_5$	$\hat{D}_6$	$\hat{D}_7$	$\hat{D}_8$	$\hat{D}_9$	$\hat{D}_{10}$
f_1	95.84									
f_2	95.72	96.12								
f_3	95.04	95.56	95.48							
f_4	94.36	95.60	95.20	95.12						
f_5	93.76	95.48	94.84	94.56	95.16					
f_6	93.64	95.12	94.56	94.44	94.96	95.00				
f_7	93.88	94.96	94.64	94.28	94.76	94.56	95.12			
f_8	93.60	94.80	94.16	94.52	94.60	94.52	95.28	93.88		
f_9	93.40	94.72	94.16	94.24	94.60	94.52	94.84	93.60	93.88	
f_{10}	93.24	94.24	93.80	94.20	94.28	94.48	94.52	93.32	93.56	94.84

1.2 Task - 2

In Task 2, starting with f_{10} from Task 1, models f_{11} to f_{12} are learned sequentially using datasets D_{11} to D_{20} which may come from slightly different input distributions. The models are updated considering this distribution shift. Each model is evaluated on its corresponding held-out dataset and all previous ones, with accuracies reported in a 10×20 matrix.

1.1.1 Feature extraction and pre-processing

It is same as done in Task - 1.

1.1.2 RandMix Augmentation Technique

The RandMix augmentation technique generates augmented images by introducing controlled spatial and color perturbations combined with random noise. It leverages **Adaptive Instance Normalization (AdaIN)** layers to apply style-based transformations and convolutional layers with varying kernel sizes to create spatial distortions. These layers simulate variations in scale and detail, while random noise further diversifies the data.

The augmented images are produced by blending outputs from multiple perturbations using randomly assigned weights, ensuring unique transformations for each image. This process enhances data variability, helping the model generalize better to unseen inputs.

The augmented data is then concatenated with the original dataset, effectively doubling its size. RandMix enriches the dataset by creating diverse, realistic variations, making it an effective tool for improving model robustness and reducing overfitting.

1.1.3 Gaussian Mixture Model

The model follows a similar approach to the Gaussian Mixture Model (GMM) used in Task 1 but incorporates **RandMix augmentation** to enhance robustness. Initially, RandMix is applied to dataset D_1, generating augmented samples. Combining these with the original dataset results in 5000 samples, which are used to train the initial model f_1. The trained f_1 is then used to predict on D_2.

For D_2, RandMix is again applied, but this time a threshold of 0.8 is imposed on the maximum posterior probability. Specifically, for a sample x classified with probabilities $P(y_k \mid x)$ for classes k, it is included in the augmented set only if:

$$\max_k P(y_k \mid x) > 0.8.$$

Let x' denote the number of such selected samples. These x' samples are then combined with the original dataset D_2 and 2500 additional samples generated via the generative GMM buffer model (similar to Task 1) trained on the combined set. This process yields $2500 + x'$ new samples, resulting in a total of $5000 + x'$ samples. When applying a model to a new domain i.e to train f_2 , distinguishing between classes becomes challenging due to overlapping distributions and vague class boundaries. Clustering-based methods that use all training samples for centroid construction often suffer from noise caused by misclassified samples, leading to low-quality centroids and reduced pseudo-labeling accuracy. To mitigate this, we propose **Top-2 Pseudo Labeling (T2PL)**[1]. This mechanism leverages softmax confidence to assess the reliability of predictions, selecting the top 50% of samples with the highest confidence to construct robust class centroids, thereby improving pseudo-labeling accuracy and ensuring better centroid quality.

This iterative process continues, with each dataset D_i being augmented by the union of three components: the original data, RandMix-generated data, and generative model samples. By systematically incorporating augmented and generated samples, this method improves the robustness of the models $f_1, f_2, \ldots$, ensuring better generalization across datasets with varied distributions.

	$\hat{D}_1$	$\hat{D}_2$	$\hat{D}_3$	$\hat{D}_4$	$\hat{D}_5$	$\hat{D}_6$	$\hat{D}_7$	$\hat{D}_8$	$\hat{D}_9$	$\hat{D}_{10}$	$\hat{D}_{11}$	$\hat{D}_{12}$	$\hat{D}_{13}$	$\hat{D}_{14}$	$\hat{D}_{15}$	$\hat{D}_{16}$	$\hat{D}_{17}$	$\hat{D}_{18}$	$\hat{D}_{19}$	$\hat{D}_{20}$
f_1	94.72																			
f_2	94.64	94.84																		
f_3	94.32	94.96	94.16																	
f_4	93.96	94.52	93.92	94.04																
f_5	91.52	92.36	92.52	92.44	92.68															
f_6	89.52	90.40	90.56	90.48	90.36	90.36														
f_7	89.68	90.44	90.40	90.88	90.84	90.20	90.64													
f_8	89.36	90.20	89.88	90.24	90.60	89.96	89.96	89.52												
f_9	89.84	90.72	90.20	90.92	90.56	90.68	90.56	89.92	90.08											
f_{10}	89.24	90.08	89.984	90.36	90.44	90.32	90.24	89.72	89.56	90.24										
f_{11}	89.28	90.28	89.60	90.84	90.32	90.60	90.40	89.96	89.24	90.24	72.96									
f_{12}	89.12	90.40	89.16	90.40	89.22	90.16	89.56	89.72	88.52	90.72	72.64	60.92								
f_{13}	89.80	91.12	89.76	91.36	90.64	90.44	90.72	90.64	89.16	91.36	74.16	62.88	79.40							
f_{14}	90.04	90.80	89.72	90.96	90.35	90.28	90.72	90.44	89.48	91.24	74.08	63.16	79.56	84.12						
f_{15}	91.48	92.28	91.32	92.24	12.64	92.12	92.08	91.60	90.88	92.48	76.00	66.00	81.48	85.84	90.92					
f_{16}	88.08	88.72	87.96	89.00	88.84	88.08	89.60	88.28	87.68	89.12	70.68	59.36	77.00	81.68	87.36	75.64				
f_{17}	88.36	89.20	88.20	88.88	88.88	88.40	89.32	88.88	87.72	88.92	71.32	60.00	76.76	81.84	87.24	75.48	67.20			
f_{18}	88.00	89.20	88.08	88.84	88.164	87.88	89.48	88.88	87.64	88.76	70.96	59.64	77.16	81.88	87.56	75.76	66.88	73.52		
f_{19}	87.56	89.00	87.96	89.12	88.72	87.96	89.32	88.92	87.84	88.76	71.08	60.32	77.00	81.52	87.26	75.76	66.64	73.56	69.16	
f_{20}	89.08	90.56	89.32	90.44	89.88	89.64	90.28	89.56	89.00	90.68	72.36	62.08	78.44	82.76	88.68	77.20	68.52	74.76	70.76	81.76

2 Problem - 2

We have presented paper - 1 (Lifelong Domain Adaptation via Consolidated Internal Distribution (NeurIPS 2021)).(Click here to watch the video).

References

[1] Deja vu: Continual model generalization for unseen domains. In *(ICLR 2023)*.

[2] Lifelong domain adaptation via consolidated internal distribution. In *(NeurIPS 2021)*.