CS771 Mini Project-2 Fall-24

Group 29: The Outliers

Himalaya Kaushik
241110029

Keshav Banka
241110032

Pritindra Das
241110054

Tafazzul Nadeem
232110401

Yuvraj Raghuvanshi
241110084

1 Introduction

Our project focusses on unsupervised domain adaptation where a learned model after training on a single supervised domain is continually adapted to perform well on new unseen domains without significant loss of accuracy in previous domains (catastropic forgetting). The problem is also restrictive in the sense that the previous domain data is not available during adaptation to new domain and only Learning with Prototypes model has to be used for classification. There are two tasks to solve one being adapting to datasets coming from same distribution and other to adapt to distinct but similar distributions. At the end a presentation link is provided that explains a paper on continual learning and domain adaptation.

2 Problem-1

We were provided with 20 CIFAR-10[3] based training datasets: $D_1, D_2, \ldots, D_{20}$. The first 10 datasets (D_1 to D_{10}) share the same input distribution, while the remaining 10 (D_{11} to D_{20}) come from distinct but similar distributions. Only D_1 is labeled, with the rest unlabeled. Additionally, 20 labeled held-out datasets ($\hat{D}_1, \hat{D}_2, \ldots, \hat{D}_{20}$) are provided strictly for evaluation purposes. There are two tasks in this problem. Task-1 deals with continual adaptation for datasets coming from same distribution and Task-2 is for continual adaptation in distinct domains.

2.1 Task-1

Using D_1 learn a model say f_1 using an LwP classifier and continually update the model on subsequent datasets D_2 to D_{10} to get f_2 to f_{10} and calculate accuracies on previous heldout sets $\hat{D}_1, \hat{D}_2, \ldots, \hat{D}_{10}$ after each adaptation. For instance, after training f_1 evaluate on $\hat{D}_1$ to $\hat{D}_3$.

2.1.1 Initial models

Since it was image classification task, we tried to use a pre-trained CNN as a feature extractor. Initially, we tried pre-trained ResNet-50[2] model which was trained on Imagenet[1] dataset to extract feature embeddings for each image. The final fully connected layer of the ResNet model is replaced with an identity layer, ensuring that the output is a high-dimensional feature vector instead of class predictions. Each input image is normalized and passed through the ResNet model in evaluation mode, generating a meaningful representation in the feature space. Using the extracted features, class prototypes are computed by averaging the feature vectors of all images belonging to the same class. The final evaluation accuracy was 57.4% for $\hat{D}_1$. Using this learned model we predicted the pseudo labels of unlabeled dataset D_2. Pseudo-labels are assigned based on L_2 distances from class means from previous model in feature space. We evaluated on the heldout sets but did not get promising results.

Next we experimented with another pre-trained VGG-11[5] model to extract features. In this case the model prototypes were updated using a weighted averaging strategy with a parameter λ to balance old and new data. For incremental learning, i.e. λ weight for old and (1-λ) for new prototypes. The performance was evaluated on the heldout datasets and accuracy matrix is maintained to track performance across the incremental learning process, but even after tuning the parameters the evaluation accuracy did not improve more than 10%.

2.1.2 Final best model

The final and best approach utilizes a pre-trained EfficientNet-B0[6] model trained on ImageNet[1] dataset to extract meaningful feature representations of 1280 dimensions from input images. The EfficientNet model is modified by removing its final classification layer. Images from the datasets are preprocessed to match EfficientNet's input requirements through resizing, normalization, and transformation. The extracted features containing high level representations are used as input to a simple LwP classifier to predict the labels.

It implements a continual learning framework where the LwP classifier is updated incrementally using newly encountered datasets. Initially, prototypes are computed from the labeled dataset D_1 as the mean feature vector for each class. Subsequent datasets $D_2, D_3, \ldots, D_{10}$ are unlabeled, and their features are classified using the latest prototypes to generate pseudo-labels, i.e. pseudo labels for D_5 obtained from f_4. These pseudo-labels are used to update the prototypes iteratively by incorporating the **moving average of the cluster means** as shown in Equation 1:

$$f_t^{\text{clusterMeans}} = \left(\frac{t-1}{t}\right) f_{t-1}^{\text{clusterMeans}} + \frac{1}{t} d_t^{\text{clusterMeans}} \tag{1}$$

where,
$f_t^{\text{clusterMeans}}$ is the model's cluster means after update using the dataset D_t and
$d_t^{\text{clusterMeans}}$ is the cluster means using pseudolabels of D_t,

The model's performance is evaluated on the held-out datasets ($\hat{D}_1, \hat{D}_2, \ldots, \hat{D}_i$), where the extracted features of heldout datasets are classified using the f_i^{th} prototypes , and accuracy is computed. An accuracy matrix (Table. 1) tracks the performance of each incremental model on all held-out datasets, providing insights into the model's ability to retain knowledge and generalize across datasets. As evidenced from the matrix the accuracy is above 80% for all datasets and maximum drop in accuracy is less than 3% after domain adaptation making this model the best model of all.

	$\mathbf{D_1}$	$\mathbf{D_2}$	$\mathbf{D_3}$	$\mathbf{D_4}$	$\mathbf{D_5}$	$\mathbf{D_6}$	$\mathbf{D_7}$	$\mathbf{D_8}$	$\mathbf{D_9}$	$\mathbf{D_{10}}$
$\mathbf{f_1}$	84.00	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
$\mathbf{f_2}$	83.44	84.28	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
$\mathbf{f_3}$	83.44	83.96	82.80	0.0	0.0	0.0	0.0	0.0	0.0	0.0
$\mathbf{f_4}$	82.32	83.48	82.48	83.76	0.0	0.0	0.0	0.0	0.0	0.0
$\mathbf{f_5}$	82.32	83.28	82.28	83.56	83.16	0.0	0.0	0.0	0.0	0.0
$\mathbf{f_6}$	82.08	83.36	82.24	83.36	83.24	82.8	0.0	0.0	0.0	0.0
$\mathbf{f_7}$	81.84	83.32	82.20	83.20	83.08	82.84	82.28	0.0	0.0	0.0
$\mathbf{f_8}$	81.76	83.16	82.16	83.20	82.92	82.76	82.32	82.52	0.0	0.0
$\mathbf{f_9}$	81.96	83.00	81.96	83.04	82.72	82.88	82.12	82.52	81.04	0.0
$\mathbf{f_{10}}$	81.68	82.88	81.92	82.96	82.76	82.64	81.84	82.40	80.92	83.08

Table 1: Accuracy matrix of the final model for Task 1

2.2 Task-2

Using f_{10} learned from Task-1 continually update the model on subsequent distinct but similar datasets D_{11} to D_{20} to get f_{11} to f_{20} and calculate accuracies on previous heldout sets $\hat{D}_1, \hat{D}_2, \ldots, \hat{D}_{20}$ after each adaptation.

2.2.1 Initial Models

Since the distributions were different for D_{11} to D_{20}, consideration of the spread or covariances of the data seemed to be a good idea to start with. At first, we tried to implement an incremental

learning framework for datasets D_{11} to D_{20} using Mahalanobis distance for LwP classification. Since EfficientNet-B0 proved to be the best model for Task-1, we used it to extract features and computed Gaussian Mixture Model (GMM) parameters (mean and covariance) for each class. These parameters were then used to calculate Mahalanobis distances in pseudo label prediction and simple averaging of means and covariances were to be done after each update. To test this approach we first wanted to know the accuracy we can achieve on D_1. So we trained the f_1 model to get mean and covariance matrices of feature space and used this to predict the labels of heldout set $\hat{D_1}$. The accuracy obtained was 27%, way below the LwP with L_2 distance approach. Hence, we did not delve further. We also tried incorporating PCA for dimensionality reduction to remove noise in the feature space in hope for alignment of distribution in lower dimensional space but failed to achieve accuracy above 15%.

After that, we thought of first doing clustering in the feature space and then assigning the cluster labels as the one which are closest to the cluster means labels from previous model, i.e. if a cluster from the clustering algorithm is nearest to cluster '2' of the previous model the cluster label is assigned the value '2'. We used the k-means++ algorithm first and after that a simple k-means with cluster means initialized to previous model cluster means to do clustering in feature space. After getting the new cluster means a simple averaging was done to update the model. Both these approaches led to good accuracies in the range of 70% for datasets D_1 to D_{10} but 57% for D_{11} dataset and so on.

2.2.2 Final Best Model

Since, the accuracies were drastically dipping from 84 to 70% in the clustering approach we thought of using the best approach we got for Task-1 for this task too. Hence, for dataset D_{11}, features are first extracted using a pre-trained EfficientNet-B0[6] model trained on ImageNet[1] dataset, with the classification layer removed and then pseudo labels predicted using the model f_{10} a simple LwP classifier with Euclidian distance. Cluster means for D_{11} are then calculated by averaging the feature vectors for each pseudo label. To ensure a smooth transition while retaining past knowledge, the updated prototypes are computed as a **moving average of the old prototypes and the newly calculated cluster means** as done in the best model for Task-1. This ensured that older prototypes retain a diminishing but bigger influence as compared to new dataset. The same process is repeated for subsequent datasets D_{12} to D_{20} using previously learned model f_{t-1} to get pseudo labels.

The results of this approach are summarized in the accuracy matrix, which measures the model's performance on held-out datasets $\hat{D_1}$ to $\hat{D_{20}}$ after training on each incremental dataset. The accuracy matrix with rows corresponding to models f_{11} to f_{20} and columns to datasets D_1 to D_{20} in Table 2 shows that the evaluation accuracy remains almost constant as we update our models from f_{11} to f_{20} on previous datasets. This model achieved the best performance among all the models we tried in terms of both accuracy and retention.

	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}	D_{11}	D_{12}	D_{13}	D_{14}	D_{15}	D_{16}	D_{17}	D_{18}	D_{19}	D_{20}
f_{11}	79.4	80.6	79.4	81.0	81.2	80.8	79.5	80.4	78.1	81.3	65.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
f_{12}	77.9	78.1	77.7	79.5	78.8	79.1	77.1	77.8	74.6	79.3	63.0	55.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
f_{13}	77.3	77.4	77.0	78.4	78.2	78.0	76.0	77.4	74.0	78.4	62.6	54.9	68.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
f_{14}	76.7	77.2	76.8	78.1	78.2	78.1	75.4	77.4	73.8	78.0	62.0	54.8	68.0	71.2	0.0	0.0	0.0	0.0	0.0	0.0
f_{15}	76.7	77.5	76.7	78.5	78.5	78.0	75.5	77.3	74.1	79.9	62.0	54.3	68.0	71.3	77.0	0.0	0.0	0.0	0.0	0.0
f_{16}	76.8	77.0	76.7	78.2	78.2	78.3	75.8	77.3	73.8	77.7	62.0	54.0	67.7	71.5	76.7	64.2	0.0	0.0	0.0	0.0
f_{17}	76.5	76.8	76.4	77.7	78.0	78.0	75.4	77.2	73.7	77.7	61.6	53.8	67.4	71.0	76.4	64.0	65.8	0.0	0.0	0.0
f_{18}	76.0	76.8	76.3	77.4	77.8	77.7	75.3	76.9	73.4	77.3	61.2	53.5	67.2	70.8	76.1	63.9	65.4	65.3	0.0	0.0
f_{19}	76.0	76.2	76.0	77.0	77.5	77.2	74.8	76.8	72.7	76.8	61.0	53.0	66.8	70.5	75.7	63.8	65.0	65.1	57.3	0.0
f_{20}	76.1	76.4	76.1	77.1	77.6	77.6	75.0	76.9	73.1	76.6	61.1	52.9	67.0	70.8	75.7	63.8	65.2	65.2	57.2	74.1

Table 2: Accuracy matrix of the final model for Task 2

3 Problem-2

The presentation video for the paper **"Lifelong Domain Adaptation via Consolidated Internal Distribution"** [4] providing an overview of the methodology and key insights of the paper is accessible via the provided link: Click here for Paper presentation Link

4 Conclusion

This project implementation demonstrates an effective approach to incremental learning, leveraging feature extraction via EfficientNet and continuously updating the prototypes using a moving average technique. This method ensures adaptability to new datasets while maintaining performance on previously seen data, as evidenced by the accuracy matrix while evaluating across sequential datasets. Our approach also highlights the significant effectiveness and versatility of simple models, such as Learning with Prototypes. Despite their simplicity, these models demonstrated the ability to achieve performance that surpassed that of more complex and sophisticated techniques. This serves as a reminder of the value of simplicity in model design, particularly when balancing computational efficiency and predictive power. At the end a paper on continual learning and domain adaptation was presented to explain the topic and it's key insights.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. IEEE, 2009.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 770–778, 2016.

[3] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, 2009.

[4] Mohammad Rostami. Lifelong domain adaptation via consolidated internal distribution. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 11172–11183. Curran Associates, Inc., 2021.

[5] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations (ICLR)*, 2015.

[6] Mingxing Tan and Quoc V Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning (ICML)*, pages 6105–6114. PMLR, 2019.