Supplementary Material: Attaining Class-level Forgetting in Pretrained Model using Few Samples

Pravendra Singh[1,⋆[0000−0003−1001−2219]], Pratik Mazumder[2,⋆[0000−0003−1103−1884]], Mohammed Asad Karim[3,⋆[0000−0001−9664−8706]]
[1]IIT Roorkee, India [2]IIT Kanpur, India [3]Carnegie Mellon University, USA
pravendra.singh@cs.iitr.ac.in, pratikm@cse.iitk.ac.in, mkarim2@cs.cmu.edu

1 Real World Scenario Where the RCRMR-LD Problem Can Occur

A real-world scenario where our proposed RCRMR-LD problem can arise is federated learning [3]. In the federated learning setting, there are multiple collaborators that have a part of the training data stored locally, and a model is trained collaboratively using these private data without sharing or collating the data due to privacy concerns. Suppose organization A has a part of the training data, and there are other collaborators that have other parts of the training data for the same classes. Organization A collaboratively trains a model with other collaborators using federated learning. After the model has been trained, a few classes may become restricted in the future due to some ethical or privacy concerns, and these classes should be removed from the model. However, the other collaborators may not be available or may charge a huge amount of money for collaborating again to train a fresh model from scratch. In this case, organization A does not have access to the full training data of the non-excluded/remaining classes that it can use to re-train a model from scratch in order to exclude the restricted classes information. This clearly shows that the RCRMR-LD problem is present in federated learning.

2 Process for Selecting the Restricted Class Relevant Parameters

In this section, we provide a detailed description of our process for selecting the restricted class relevant parameters. First, we apply a data augmentation technique f, not used during training, to the images of the given restricted class. Next, we combine the predictions for these images and perform backpropagation. Finally, we select the parameters with the highest absolute gradient as the relevant parameters for the corresponding restricted class. Specifically, for a given restricted class, we choose all the parameters from each network layer such

⋆ All the authors contributed equally.

that pruning these parameters will result in the maximum degradation of model performance on that restricted class. We use a process similar to the binary search for automatically selecting the parameters with the highest absolute gradient.

We first create a list of parameters in each layer, sort them in descending order according to the absolute gradient values, and check if zeroing out the weights of the first 20% parameters from this list for a particular layer leads to low accuracy (less than 10% for ResNet-164 on CIFAR-100) for that class. If the accuracy is not low after zeroing out the chosen parameters, then we select double the number of parameters chosen earlier and repeat this process. However, if the accuracy is low after zeroing out the chosen parameters, we still need to check if a low accuracy can be achieved by zeroing out fewer parameters. To check this, we reduce the number of parameters by half the difference between the current and previously chosen number of parameters and observe the effect of zeroing out these parameters. If the accuracy is low for the reduced parameters, then we stop the process with the current set of parameters as the relevant parameters for the current restricted class. If the accuracy is not low for the reduced parameters, then we take the previously chosen higher number of parameters as the relevant parameters of the layer for the current restricted class. We repeat this process for all the restricted classes to obtain the relevant parameters for each restricted class in all the layers. The combined set of the relevant parameters for all the excluded classes is referred to as the restricted/excluded class relevant parameters. Please note that this process is just for identifying the parameters relevant to the restricted classes, and their weights are restored after this process.

3 Baselines

We propose 9 baseline models for the RCRMR-LD problem and compare our proposed approach with them. The baseline 1 involves deleting the weights of the fully-connected classification layer corresponding to the excluded classes. Baselines 2, 3, 4, 5 involve training the model on the limited training data of the remaining classes. Baselines 6, 7, 8, 9 involve fine-tuning the model on the available limited training data. The details about the baselines are provided below:

Original model: It refers to the original model that is trained on the complete training set containing all the training examples from both the excluded and non-excluded classes. It represents the model that has not been modified by any technique to remove the excluded class information.

Baseline 1 - Weight Deletion (WD): It refers to the original model with a modified fully-connected classification layer. Specifically, we remove the weights corresponding to the excluded classes in the fully-connected classification layer so that it cannot classify the excluded classes.

Baseline 2 - Training from Scratch on Limited Non-Restricted Class data (TSLNRC): In this baseline, we train a new model from scratch using the limited training examples of only the non-excluded classes. It uses the complete

training schedule as the original model and only uses the classification loss for training the model.

Baseline 3 - Training from Scratch on Limited Non-Restricted Class data with KD (TSLNRC-KD): This baseline is the same as baseline 2, but in addition to the classification loss, it also uses a knowledge distillation loss to ensure that the non-excluded class logits of the model (student) match that of the original model (teacher).

Baseline 4 - Training of Original model on Limited Non-Restricted Class data (TOLNRC): This baseline is the same as baseline 2, but the model is initialized with the weights of the original model instead of randomly initializing it.

Baseline 5 - Training of Original model on Limited Non-Restricted Class data with KD (TOLNRC-KD): This baseline is the same as baseline 4, but in addition to the classification loss, it also uses a knowledge distillation loss.

Baseline 6 - Fine-tuning of Original model on Limited data after Mapping Restricted Classes to a Single Class (FOLMRCSC): In this baseline approach, we first replace all the excluded class labels in the limited training data with a new single excluded class label and then fine-tune the original model for a few epochs on the limited training data of both the excluded and remaining classes. In the case of the examples from the excluded classes, the model is trained to predict the new single excluded class. In the case of the examples from the remaining classes, the model is trained to predict the corresponding non-excluded classes.

Baseline 7 - Fine-tuning of Original model on Limited data after Mapping Restricted Classes to a Single Class with KD (FOLMRCSC-KD): This baseline is the same as baseline 6, but in addition to the classification loss, it also uses a knowledge distillation loss to ensure that the non-excluded class logits of the model (student) match that of the original model (teacher).

Baseline 8 - Fine-tuning of Original model on Limited Non-Restricted Class data (FOLNRC): In this baseline approach, we fine-tune the original model for a few epochs on the limited training data of non-excluded/remaining classes. The model is trained to predict the corresponding non-excluded classes of the training examples.

Baseline 9 - Fine-tuning of Original model on Limited Non-Restricted Class data with KD (FOLNRC-KD): This baseline is the same as baseline 8, but in addition to the classification loss, it also uses a knowledge distillation loss.

4 Experiments

4.1 Datasets

For the RCRMR-LD problem setting, we modify the CIFAR-100 [2], CUB-200 [7] and ImageNet-1k [5] datasets. In order to simulate the RCRMR-LD problem setting with limited training data, we choose the last 20 classes of the CIFAR-100

dataset as the excluded classes and take only 10% of the training images of each class. Similarly, we choose the last 20 classes of the CUB-200 dataset as the excluded classes with only 3 training images per class. For ImageNet-1K, we choose the last 100 classes as the excluded classes with 5% of the training images to simulate the limited data available for this problem setting.

4.2 Implementation Details

In this section, we provide all the details required to reproduce our experimental results. We use the ResNet-20 [1], ResNet-56, ResNet-164 architectures for the experiments on the CIFAR-100 dataset. We use the standard data augmentation methods of random cropping to a size of 32×32 (zero-padded on each side with four pixels before taking a random crop) and random horizontal flipping, which is a standard practice for training a model on CIFAR-100. In order to obtain the original and FDR models for the CIFAR-100 dataset, we train the network for 300 epochs with a mini-batch size of 64 using the stochastic gradient descent optimizer with momentum 0.9 and weight decay $1e-4$. We choose the initial learning rate as 0.1, and we decrease it by a factor of 5 after the 90, 150, 210, 240, and, 270 epochs. For the CIFAR-100 experiments with ERwP using the ResNet-20, ResNet-56, and ResNet-164 architectures, we use learning rate= $1e-4$, $\beta = 10$ and optimize the network for 10 epochs. Since the available limited training data is only 10% of the entire CIFAR-100 dataset, therefore, our ERwP approach is approximately $30 * 10 = 300\times$ faster than the FDR method.

For the experiments on the ImageNet dataset, we use the ResNet-18, ResNet-50, and MobileNet-V2 architectures. We use the standard data augmentation methods of random cropping to a size of 224×224 and random horizontal flipping, which is a standard practice for training a model on ImageNet-1k. In order to obtain the original and FDR models for the ImageNet dataset, we train the network for 100 epochs with a mini-batch size of 256 using the stochastic gradient descent optimizer with momentum 0.9 and weight decay $1e-4$. We choose the initial learning rate as 0.1, and we decrease it by a factor of 10 after every 30 epochs. For evaluation, the validation images are subjected to center cropping of size 224×224. For the ImageNet-1k experiments (5% training data) with ERwP using the ResNet-50 architecture, we optimize the network for 10 epochs with a learning rate of $9e-5$ using $\beta = 200$. For the ERwP experiments using the ResNet-18 architecture, we optimize the network for 10 epochs using $\beta = 200$ with an initial learning rate of $1.1e-4$ and a learning rate of $1.1e-5$ from the third epoch onward. In the case of the ERwP experiments with the MobileNet-V2 architecture, we optimize the network for 10 epochs using $\beta = 400$ with an initial learning rate of $1.5e{-}4$ and a learning rate of $1.5e{-}5$ from the third epoch onward. Since the available limited training data is only 5% of the entire ImageNet-1k dataset, therefore, our ERwP approach is approximately $20 * 10 = 200\times$ faster than the FDR method. For the experiments on the CUB-200 dataset, we use the ResNet-50 architecture pre-trained on the ImageNet dataset. In order to obtain the original and FDR models for the CUB-200 dataset, we train the network for 50 epochs with a mini-batch size of 64 using the stochastic gradient descent

Table 1: Experimental results on the CUB-200 dataset with ResNet-50 architecture for the RCRMR-LD problem with 20 excluded classes using only 3 training images per class

Methods	$\texttt{FA}_\texttt{e}$	$\texttt{FPA}_\texttt{e}$	$\texttt{CA}_\texttt{ne}$
Original	85.20%	84.69%	77.37%
No Training			
Baseline 1 - WD	0.00%	84.69%	77.64%
Full Train Schedule			
Baseline 2 - TSLNRC	0.00%	30.27%	27.56%
Baseline 3 - TSLNRC-KD	0.00%	35.54%	31.66%
Baseline 4 - TOLNRC	0.00%	60.37%	64.60%
Baseline 5 - TOLNRC-KD	0.00%	68.37%	70.48%
Only Fine-Tuning			
Baseline 6 - FOLMRCSC	53.40%	77.38%	74.39%
Baseline 7 - FOLMRCSC-KD	60.88%	81.12%	75.14%
Baseline 8 - FOLNRC	84.86%	84.18%	76.85%
Baseline 9 - FOLNRC-KD	84.35%	85.20%	77.70%
ERwP (Ours)	0.77%	48.89%	75.45%

optimizer with momentum 0.9 and weight decay $1e-3$. We choose the initial learning rate as $1e-2$, and we decrease it by a factor of 10 after epochs 30 and 40. For the CUB-200 experiments (3 images per class, i.e., 10% training data) with ERwP using the ResNet-50 architecture, we optimize the network for 10 epochs with a learning rate of $1e-4$ using $\beta = 10$. Since the available limited training data is only 10% of the entire CUB-200 dataset, therefore, our ERwP approach is approximately $5 * 10 = 50\times$ faster than the FDR method.

In our proposed approach, we use $\kappa = 2$ for all the experiments (see Sec. 5.5 in the supplementary material). We use a popular Pytorch implementation[1] for performing knowledge distillation. We run all the experiments 5 times and report the average accuracy. We perform all the experiments using the Pytorch [4] and Python 3.0. We use 4 GeForce GTX 1080 Ti graphics processing units for our experiments.

4.3 CUB-200 Results

Table 1 reports the experimental results for different approaches to the RCRMR-LD problem over the CUB-200 dataset using the ResNet-50 architecture. Our proposed ERwP approach achieves a constraint accuracy $\texttt{CA}_\texttt{ne}$ that is very close to that of the original model even though we use only 3 images per class for optimizing the model. It achieves close to 0% forgetting accuracy $\texttt{FA}_\texttt{e}$ and achieves a $\texttt{FPA}_\texttt{e}$

[1] https://github.com/peterliht/knowledge-distillation-pytorch/blob/master/model/net.py

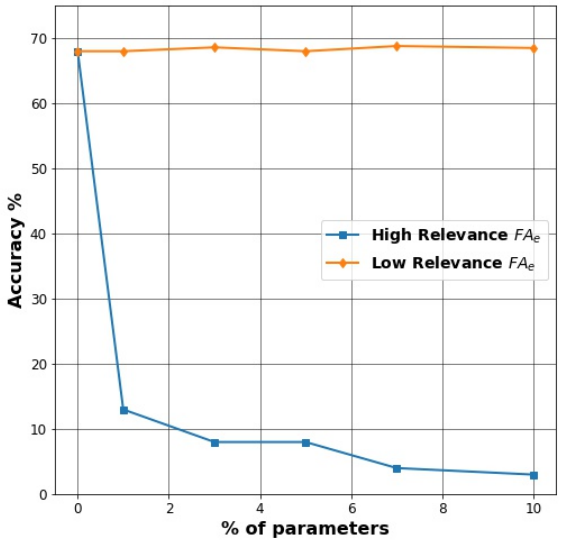


Fig. 1: Ablation to validate our approach for identifying relevant model parameters for a random restricted class of CIFAR-100

that is significantly lower than that of the original model by an absolute margin of 35.80%. Our ERwP approach outperforms all the baseline approaches. Further, our ERwP requires only 10 epochs to remove the excluded class information from the model. Since the available limited training data is only 10% of the entire CUB dataset, therefore, our ERwP approach is approximately $5 * 10 = 50\times$ faster than the FDR method that is trained on the full training data for 50 epochs.

5 Ablation Studies

5.1 Ablation on Our Approach of Identifying the Restricted Class Relevant Parameters

We perform ablation experiments to verify our approach of identifying the highly relevant parameters for any restricted class. We perform these experiments on the CIFAR-100 dataset with the ResNet-56 architecture and report the forgetting accuracy $\texttt{FA}_\texttt{e}$ for the randomly chosen excluded class. Please note that in this case, only the chosen class of CIFAR-100 is the restricted class and all the remaining classes constitute the non-excluded classes. In order to show the effectiveness of our approach, we sort the absolute gradients of the parameters in the model (obtained through backpropagation for the excluded class augmented images) and choose a set of high relevance and low relevance parameters. We then prune/zero

Table 2: Significance of ERwP components

$\mathcal{L}_c^{ne}$	$\mathcal{L}_c^{e}$	$\mathcal{L}_{kd}$	$\texttt{FA}_\texttt{e}$	$\texttt{FPA}_\texttt{e}$	$\texttt{CA}_\texttt{ne}$
✓	✗	✗	66.50%	68.19%	69.79%
✓	✓	✗	0.00%	24.40%	6.45%
✓	✓	✓	0.00%	47.84%	69.32%

out these parameters and record the forgetting accuracy. Fig. 1 demonstrates that as we zero out the high relevance parameters, the forgetting accuracy of the excluded class drops by a huge margin. It also shows that as we zero out the low relevance parameters, there is only a minor change in the forgetting accuracy of the excluded class. Therefore, the parameters relevant to the excluded class receive large gradient updates as compared to the other parameters. This validates our approach for identifying the high relevant parameters for the restricted classes.

5.2 Significance of the Components of the Proposed ERwP Approach

We perform ablations on the CIFAR-100 dataset using the ResNet-56 model to study the significance of the $\mathcal{L}_c^e$, $\mathcal{L}_c^{ne}$ and $\mathcal{L}_{kd}$ components of our proposed ERwP approach. Table 2 indicates that optimizing the restricted class relevant parameters using only $\mathcal{L}_c^{ne}$ cannot significantly remove the information regarding the restricted classes from the model. Applying $\mathcal{L}_c^{ne}$ along with $\mathcal{L}_c^e$ significantly reduces the forgetting accuracy $\texttt{FA}_\texttt{e}$ and forgetting prototype accuracy $\texttt{FPA}_\texttt{e}$ but also significantly reduces the constraint accuracy $\texttt{CA}_\texttt{ne}$. Finally, applying the $\mathcal{L}_{kd}$ loss along with $\mathcal{L}_c^{ne}$ and $\mathcal{L}_c^e$ significantly reduces $\texttt{FA}_\texttt{e}$ and $\texttt{FPA}_\texttt{e}$ while maintaining the constraint accuracy $\texttt{CA}_\texttt{ne}$ very close to that of the original model.

5.3 Ablation on the Number of Excluded Classes

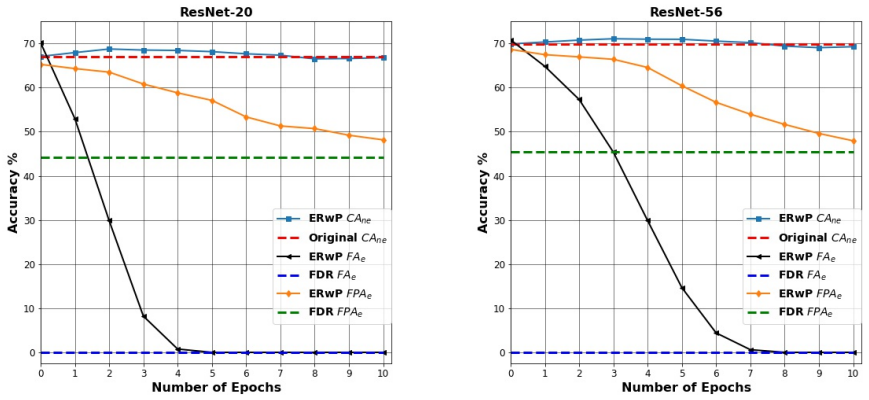
We report the experimental results for our approach for different splits of excluded and remaining classes of the CIFAR-100 dataset in Table 3. We observe that our ERwP performs well for all the splits for both the ResNet-20 and ResNet-56 architectures.

5.4 Performance of ERwP over Training Epochs

We analyze the change in the performance of the model after every epoch of our proposed ERwP approach in Fig. 2 for the CIFAR-100 dataset with 20 excluded classes using the ResNet-20 and ResNet-56 architectures. For both architectures, we observe that as the training progresses, ERwP maintains the constraint accuracy close to that of the original model and forces the forgetting accuracy to drop to 0%. ERwP also forces the forgetting prototype accuracy to keep dropping and makes it similar to the FDR model.

Table 3: Experimental results on the CIFAR-100 dataset using ResNet-56 for ERwP with different numbers of excluded classes. # R/E → no. of non-excluded classes / no. of excluded classes

# R/E	Methods	ResNet-20		ResNet-56	
		$\mathtt{FA_e}$	$\mathtt{CA_{ne}}$	$\mathtt{FA_e}$	$\mathtt{CA_{ne}}$
60/40	Original	68.18%	67.35%	69.98%	70.11%
	ERwP	0.00%	67.03%	0.00%	69.98%
70/30	Original	67.83%	67.61%	69.60%	70.26%
	ERwP	0.00%	67.25%	0.00%	69.81%
80/20	Original	70.15%	67.06%	70.80%	69.88%
	ERwP	0.00%	66.84%	0.00%	69.32%
90/10	Original	67.90%	67.66%	68.40%	70.24%
	ERwP	0.00%	67.26%	0.00%	69.69%
95/5	Original	66.20%	67.76%	67.00%	70.22%
	ERwP	0.00%	67.55%	0.00%	69.63%



(a) (b)

Fig. 2: Plots denoting the performance of our proposed ERwP during the optimization process for forgetting 20 excluded classes from CIFAR-100 using a) ResNet-20 and b) ResNet-56 architectures

5.5 Ablation Experiments for β and κ

We perform ablation experiments to identify the most suitable values for the hyper-parameters β and κ for our proposed ERwP. The ablation results in Tables 4, 5,

Table 4: Experimental results on the CIFAR-100 dataset with ResNet-20 architecture for the RCRMR-LD problem with 20 excluded classes using our proposed ERwP with different values of β

β	Methods	$\texttt{FA}_\texttt{e}$	$\texttt{CA}_\texttt{ne}$
-	Original	70.15%	67.06%
8	ERwP	0.00%	66.03%
9	ERwP	0.00%	66.23%
10	ERwP	0.00%	**66.84**%
11	ERwP	0.00%	66.58%
12	ERwP	0.00%	66.15%

Table 5: Experimental results on the CIFAR-100 dataset with ResNet-20 architecture for the RCRMR-LD problem with 20 excluded classes using our proposed ERwP with different values of κ

κ	Methods	$\texttt{FA}_\texttt{e}$	$\texttt{CA}_\texttt{ne}$
-	Original	70.15%	67.06%
1.0	ERwP	0.00%	66.05%
1.5	ERwP	0.00%	66.08%
2.0	ERwP	0.00%	**66.84**%
2.5	ERwP	0.00%	66.30%
3.0	ERwP	0.00%	66.23%

validate our choice of hyper-parameter values considering the forgetting accuracy and the constraint accuracy of the resulting model.

5.6 Effect of Different Data Augmentations on the Identification of Class Relevant Model Parameters

The purpose of applying any data augmentation (not used during training) in our approach is to study the gradient updates when the model performs back-propagation over slightly different versions of the training data of a class and use this information to identify the highly relevant parameters of the model with respect to that class. We have performed experiments using various data augmentation techniques (grayscale, vertical flip, rotation, random affine augmentations) and have provided these results in Fig-3. We chose the same restricted class of CIFAR-100 and use the ResNet-56 network for all the experiments. The results in Fig. 3 indicate that for all the compared data augmentations approaches, pruning/zeroing out the high relevance parameters obtained using our approach

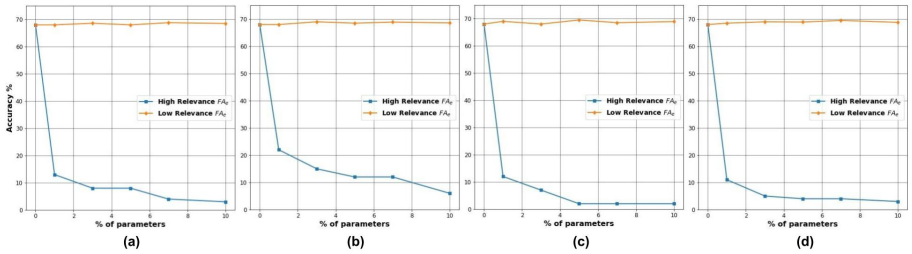


Fig. 3: Ablation to validate our approach for identifying restricted class relevant model parameters using different augmentation techniques w.r.t. the same randomly chosen restricted class of CIFAR-100. We use the ResNet-56 network for these experiments. The data augmentation techniques used are (a) grayscale augmentation, (b) vertical flip augmentation, (c) rotation augmentation, (d) random affine augmentation. In each case, the figure shows the model performance when the low relevance and high relevance parameters obtained using our approach are zeroed out

results in a huge drop in the forgetting accuracy of the excluded class. Further, zeroing out the low relevance parameters has a minor impact on the forgetting accuracy of the excluded class. Therefore, the data augmentation techniques are almost equally effective in our approach for finding the relevant parameters with respect to any restricted class.

5.7 Ablation Experiments on the Restricted Class Relevant Parameters

We perform ablation experiments with ERwP to check if only 25% and 50% of the restricted class relevant parameters of each layer identified using our proposed procedure can be used for ERwP. We run each of these experiments for the same number of epochs for the CIFAR-100 dataset and ResNet-56 network. However, we observed that the final $\texttt{FPA}_\texttt{e}$ falls from 68.65% to 60.35% and 53.7%, respectively, for 25% and 50% of restricted class relevant parameters of each layer as compared to 47.84% when using all the restricted class relevant parameters per layer identified using our approach. The good performance of our approach is more evident in light of the performance of the FDR model that achieves a $\texttt{FPA}_\texttt{e}$ accuracy of 45.40%. We provide this result as a reference to demonstrate that the 47.84% $\texttt{FPA}_\texttt{e}$ accuracy is due to the generalization power of the model and not due to the restricted classes information in the model. This shows that our approach effectively identifies the class-relevant parameters of the model for a given class.

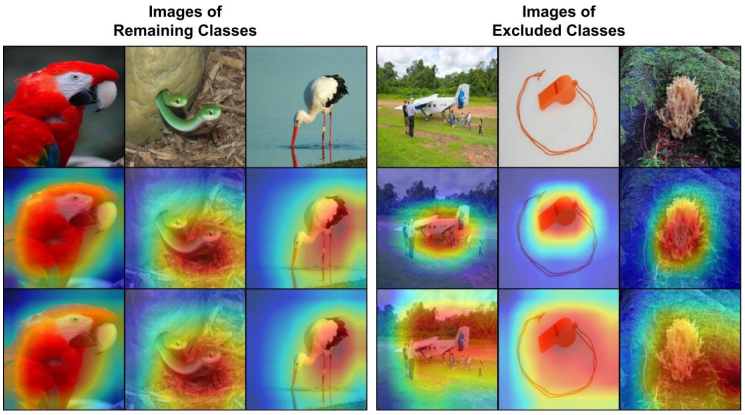


Fig. 4: Class activation maps of ImageNet images from the excluded and non-excluded classes, for the original ResNet-50 (second row) and ResNet-50 after applying our proposed ERwP approach (third row). First row depicts the real images

5.8 Effect of Using the Proposed ERwP Approach When the Entire Dataset is Available

We perform ablation experiments to demonstrate the performance of our proposed ERwP approach when the entire training data is available. We perform these experiments on the CIFAR-100 dataset using ResNet-20 and ResNet-56. We observe experimentally that for both the ResNet-20 and ResNet-56 experiments using ERwP, the forgetting accuracy $\texttt{FA}_\texttt{e}$ accuracy is 0% and the constraint accuracy $\texttt{CA}_\texttt{ne}$ matches that of the original model. Further, the gap between the forgetting prototype accuracy $\texttt{FPA}_\texttt{e}$ of ERwP and the FDR model reduces from 3.86% (for limited data) to 2.79% for ResNet-20. Similarly, the gap reduces from 2.44% (for limited data) to 1.65% for ResNet-56. However, ERwP requires only 2-3 epochs of optimization ($\sim$100-150$\times$ faster than the FDR model) for achieving this performance when trained on the entire dataset. This makes it significantly faster than any approach that trains on the entire dataset.

5.9 Qualitative Analysis

In order to analyze the effect of removing the excluded class information from the model using our proposed ERwP approach, we study the class activation map visualizations [6] of the model before and after applying ERwP. We observe in Fig. 4 that for the images from the excluded classes, the model's region of attention gets scattered after applying ERwP, unlike the images from the remaining classes.

References

1. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778 (2016)
2. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
3. McMahan, B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.y.: Communication-Efficient Learning of Deep Networks from Decentralized Data. In: Singh, A., Zhu, J. (eds.) Proceedings of the 20th International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 54, pp. 1273–1282. PMLR (20–22 Apr 2017), https://proceedings.mlr.press/v54/mcmahan17a.html
4. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch (2017)
5. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**(3), 211–252 (2015)
6. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 618–626 (2017)
7. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset (2011)