Appendix: A Probabilistic Framework for Lifelong Test-Time Adaptation

Dhanajit Brahma
Indian Institute of Technology Kanpur
dhanajit@cse.iitk.ac.in

Piyush Rai
Indian Institute of Technology Kanpur
piyush@cse.iitk.ac.in

In this Appendix, we provide further details about benchmark datasets, computational resources, training details, evaluation metrics and additional experimental analysis.

1. Benchmark Datasets

CIFAR10C and CIFAR100C datasets are corrupted versions of the standard CIFAR10 and CIFAR100 [7] datasets, respectively. ImageNetC [5] and ImageNet3DCC [6] are both corrupted versions of the standard ImageNet [4] dataset.

Both CIFAR10C and CIFAR100C datasets contain 10,000 images per corruption type, making 150,000 images in total for each dataset. For ImageNetC dataset, there are 50,000 images for each corruption type. CIFAR10C, CIFAR100C, and ImageNet-C datasets consists of 15 diverse corruption types with 4 extra corruption types for validation. Every corruption has five different levels of severity. The various corruptions, along with a brief description, are as follows: i. Gaussian noise: often occurs in low light; ii. Shot noise: electronic noise due to discrete nature of light; iii. Impulse noise: color equivalent of salt-and-pepper noise, and it may be due to bit errors; iv. Defocus blur: caused when a photograph is taken out of focus; v. Frosted Glass Blur: due to an image through a frosted glass window; vi. Motion blur: occurs when a camera is rapidly moving; vii. Zoom blur: happens when a camera quickly approaches an object; viii. Snow: visually an obscuring kind of precipitation; ix. Frost arises when ice crystals adhere to windows; x. Fog: Objects are cloaked in fog, which is rendered using the diamond-square algorithm; xi. Brightness: varying brightness with the sunshine intensity; xii. Contrast: depends on the lighting conditions and color of the photographed item; xiii. Elastic transformations: Small image areas are stretched or contracted via elastic transformations; xiv. Pixelation: happens when a low-resolution picture is upsampled; xv. JPEG: lossy image compression that results in compression artifacts.

Recently proposed by [6], the ImageNet 3D Common Corruptions (ImageNet3DCC) dataset exploits the geometry of the scene in transformations, resulting in more realistic corruptions. The Imagenet3DCC dataset has 50,000 images for each corruption type. It consists of 12 different types of corruption, each with 5 severity levels. The corruptions are as follows: i. Near focus: changing the focus area to the near portion of the scene at random; ii. Far focus: randomly change the focus to the scene's far part; iii. Bit error: caused by an imperfect video transmission channel; iv. Color quantization: decreases the RGB image's bit depth; v. Flash: caused by positioning a light source near the camera; vi. Fog 3D: generated by using a standard optical model for fog; vii. H.265 ABR: codec H.265 for compression with Average Bit Rate control mode; viii. H.265 CRF: codec H.265 for compression with Constant Rate Factor (CRF) control mode; ix. ISO noise: noise using a Poisson-Gaussian distribution; x. Low light: simulate low-light imaging setting by lowering the pixel intensities and adding Poisson-Gaussian distributed noise; xi. XY-motion blur: the main camera is moving along the image XY-plane; xii. Z-motion blur: the main camera is moving along the image Z-axis.

These datasets are developed to serve as benchmarks for measuring robustness of classification models.

2. Computational Resource Details

All the experiments were run locally on a GPU server with Nvidia Titan RTX GPUs with 24 GB memory, Intel(R) Xeon(R) Silver 4110 CPU with 128 GB RAM, and Ubuntu 18.04.6 LTS OS. PETAL is implemented using PyTorch version 1.10.0. More details about the other relevant libraries are provided in the source code.

For CIFAR-10 experiments, further training the pre-trained WideResNet-28 model from RobustBench benchmark [3] using the source domain training data takes approximately 20 minutes. For CIFAR-100 experiments, we further train the pre-trained ResNeXt-29 [10] model from RobustBench benchmark [3] using the source domain training data for approximately 15 minutes. For ImageNet experiments, training the pre-trained ResNet50 model from RobustBench benchmark [3] further using the source domain training data takes approximately 5 hours.

For the CIFAR10-to-CIFAR10C adaptation experiment on the highest severity level of 5 using our approach

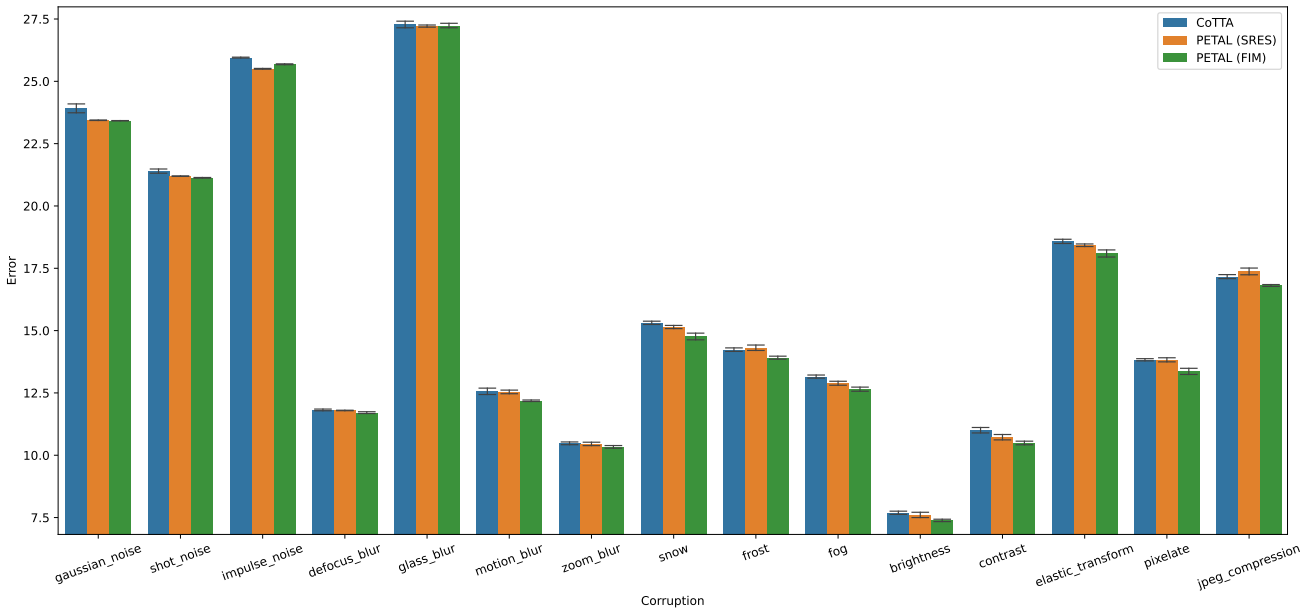

Figure 1. CIFAR10-to-CIFAR10C Results for the corruption order as depicted in the figure with the corruption of severity level 5. The error rate is averaged over 5 runs. The standard deviation is depicted in the bar plot. PETAL (FIM) performs better in most of the settings.

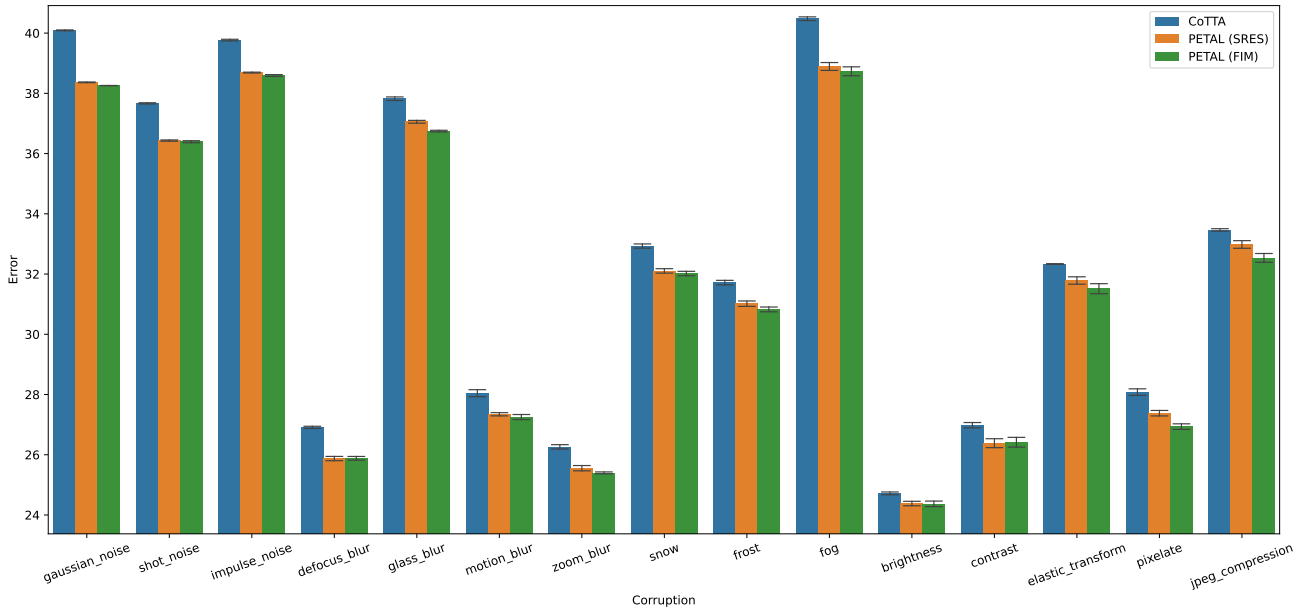

Figure 2. CIFAR100-to-CIFAR100C Results for the corruption order as depicted in the figure with the corruption of severity level 5. The error rate is averaged over 5 runs. The standard deviation is depicted in the bar plot. PETAL (FIM) performs better in most of the settings.

PETAL, the total time taken for adapting the pre-trained WideResNet-28 model on 15 corruption types takes approximately 3 hours. In CIFAR10-to-CIFAR10C gradual experiment, since the severity level changes gradually before the change in corruption type, the total number of different corruption-severity level pairs is 131. Further, the experiment is conducted for 10 random orders of corruption sequences and so the total time taken is about 6 days and 19 hours.

CIFAR100-to-CIFAR100C adaptation experiment using pre-trained ResNeXt-29 for the highest severity level of 5 takes approximately 1 hour and 10 minutes.

For ImageNet-to-ImageNetC adaptation experiment on the highest severity level of 5, the total time taken for

adapting the pre-trained ResNet50 model on 15 corruption types repeated for 10 diverse corruption sequences is about 2 hours and 30 minutes in total. The ImageNet-to-ImageNet3DCC adaptation experiment on the highest severity level of 5 for 12 corruptions took a total time of about 2 hours and 15 minutes.

3. Training Details

3.1. Training on Source Domain data

To obtain the approximate posterior using the source domain training data, we use SWAG-D [8] posterior. This requires further training of the pre-trained models on the source domain training data for a few more epochs.

For CIFAR-10 experiments, the pre-trained WideResNet-28 model from RobustBench benchmark [3] is used, which is already trained using CIFAR-10 training data. We further train it using SGD with momentum for 5 epochs with a learning rate of 8e-4, collecting iterates for SWAG-D [8] once in each epoch. Similarly, for CIFAR-100 experiments, we use the pre-trained ResNeXt-29 [10] model from RobustBench benchmark [3] by training it further using SGD with momentum for 5 more epochs with a learning rate of 8e-4. For ImageNet experiments, we further train the pre-trained standard ResNet50 model from RobustBench benchmark [3] for 2 more epochs and collect 4 iterates per epoch with a learning rate of 1e-4 for obtaining SWAG-D approximate posterior.

3.2. Adaptation on Target Domain Test data

For the online lifelong test-time adaptation (TTA), we use the same hyperparameters mentioned in CoTTA [9] using a learning rate of 0.001 with Adam optimizer. Based on [2] and [9], we apply random augmentations that include color jitter, random affine, Gaussian blur, random horizontal flip, and Gaussian noise. For stochastic restore probability, we use the same value used by CoTTA, i.e., 0.01 for CIFAR10-to-CIFAR10C/CIFAR100-to-CIFAR100C and 0.001 for ImageNet-to-ImageNetC experiments. We tune the FIM based parameter restoration quantile value δ using the extra four validation corruptions and use 0.03 for CIFAR10-to-CIFAR10C/CIFAR100-to-CIFAR100C, and use 0.003 for ImageNet-to-ImageNetC and ImageNet-to-ImageNet3DCC.

To adapt the model, we optimize the following training objective from Equation 7:

$$\max_{\theta} \log q(\theta) - \frac{\bar{\lambda}}{M} \sum_{m=1}^{M} \mathrm{H}^{\mathrm{xe}}(y', y|\bar{\mathbf{x}}) \tag{11}$$

Here, $q(\theta)$ is the approximate posterior density obtained using SWAG-D [8]. $\bar{\lambda}$ is a hyperparameter that determines the importance of the cross-entropy minimization term relative to the posterior term. For the implementation, putting $\alpha = 1/\bar{\lambda}$, we rewrite the training objective as follows:

$$\max_{\theta} \alpha \log q(\theta) - \frac{1}{M} \sum_{m=1}^{M} \mathrm{H}^{\mathrm{xe}}(y', y|\bar{\mathbf{x}}) \tag{12}$$

For all the experiments, we tune the α hyperparameter using the corresponding extra four validation corruptions and use one of the values from {1e-6, 1e-7, 1e-9, 1e-10, 5e-10, 1e-11, 5e-11, 1e-12} for which the average error is the lowest. We follow CoTTA [9] for setting the hyperparameters: augmentation threshold for confident predictions τ and exponential moving average smoothing factor π = 0.999. CoTTA [9] discusses choice of τ in detail in their supplementary.

Gradually changing corruptions: In this setting, the severity of corruption changes gradually before the change in corruption type. For example, if A_s, B_s, and C_s are three different corruptions of severity level s, then there are 23 corruption-severity pairs in total. They arrive as follows:

$$A_5 \rightarrow A_4 \rightarrow A_3 \rightarrow A_2 \rightarrow A_1 \xrightarrow[corruption]{change} B_1 \rightarrow B_2 \rightarrow B_3 \rightarrow B_4 \rightarrow B_5 \rightarrow B_4 \rightarrow B_3 \rightarrow B_2 \rightarrow B_1 \xrightarrow[corruption]{change} C_1 \rightarrow C_2 \rightarrow C_3 \rightarrow C_4 \rightarrow C_5 \rightarrow C_4 \rightarrow C_3 \rightarrow C_2 \rightarrow C_1$$

When the corruption type changes, the severity level is 1, and thus, the domain shift is gradual. In addition, distribution shifts within each corruption type are also gradual.

4. Evaluation Metrics

Let y'_n be the predicted probability vector and y_n be the true label ($y_{ni} = 1$ if i is the true class label, else $y_{ni} = 0$) for input x_n. Let the number of examples be N and the number of classes be D.

4.1. Error

The average error rate is defined as:

$$\text{Error} = \frac{1}{N} \sum_{n=1}^{N} \mathbb{I}(y'_n \neq y_n)$$

where $\mathbb{I}()$ is the indicator function.

4.2. Brier Score

The average Brier score [1] is defined as:

$$\text{Brier score} = \frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{D} (y'_{ni} - y_{ni})^2$$

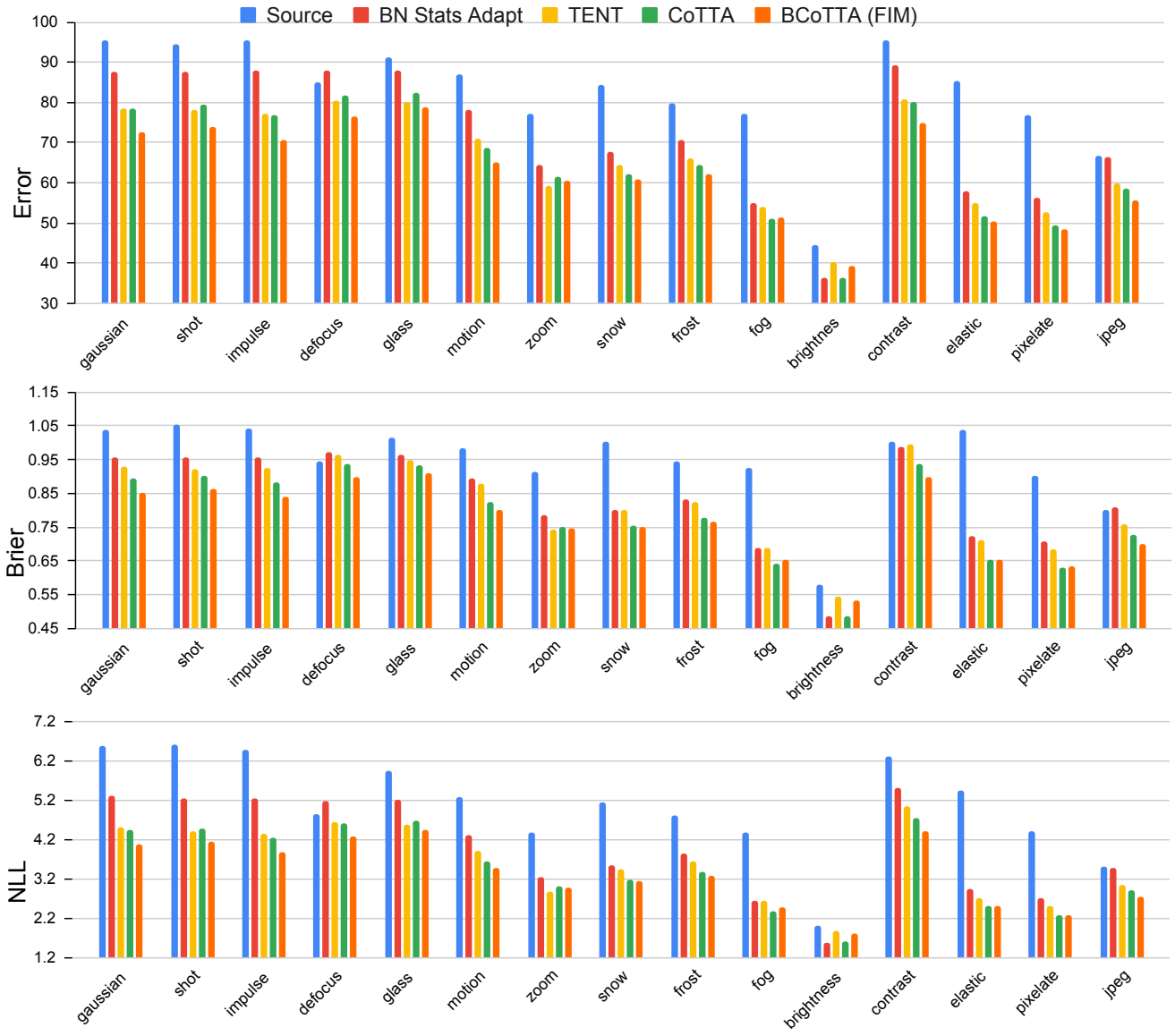

Figure 3. ImageNet-to-ImageNetC results averaged over 10 different corruption sequences with corruption of severity level 5. PETAL (FIM) outperforms other approaches for most of the corruptions.

4.3. Negative Log-Likelihood

The average negative log-likelihood (NLL) is defined as:

$$\text{NLL} = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{D}(y_{ni}\log y'_{ni})$$

5. Additional Experimental Results

5.1. CIFAR10-to-CIFAR10C Results

Figure 1 shows the results of CIFAR10-to-CIFAR10C experiments in which the corruption types arrive in the depicted order. The standard deviations are depicted in the figure. PETAL performs better in most of the settings. Moreover, PETAL (FIM) performs better than PETAL (SRES) demonstrating the effectiveness of data-driven FIM based parameter resetting.

5.2. CIFAR100-to-CIFAR100C Results

The CIFAR100-to-CIFAR100C adaptation results are shown in Figure 2 in which the corruption types arrive in the depicted order. The standard deviations are depicted in the figure. We observe that PETAL outperforms most of the approaches.

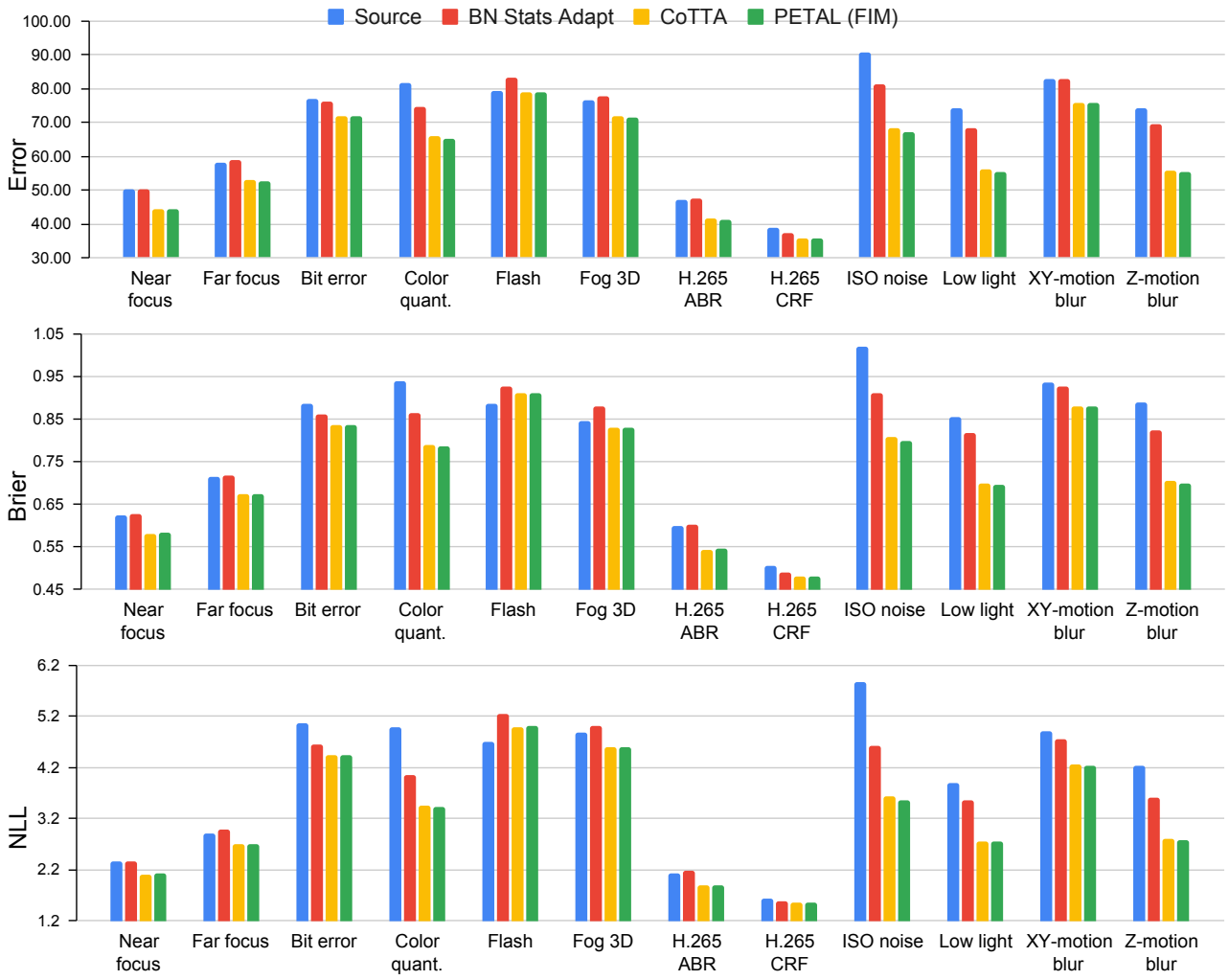

Figure 4. ImageNet-to-ImageNet3DCC results averaged over 10 different corruption sequences with corruption of severity level 5. PETAL (FIM) outperforms other approaches for most of the corruptions.

5.3. ImageNet-to-ImageNetC Results

For ImageNet-to-ImageNetC experiments, we provide the error rate, Brier score, and NLL in Figure 3. The experiments are conducted for 10 diverse corruption sequences. The numbers in Figure 3 are obtained by averaging over the 10 different orders of corruption sequences. PETAL (FIM) outperforms CoTTA and PETAL (SRES) for most of the corruptions in terms of average error rate, average Brier score, and average NLL.

5.4. ImageNet-to-ImageNet3DCC Results

We report the results for ImageNet-to-ImageNet3DCC experiments in Fig. 4. The numbers are averaged over 10 random corruption orders. Our approach PETAL performs better than other approaches for most of the corruption types.

Results demonstrate that our probabilistic approach, in which the source model's posterior leads to a regularizer and the data-driven reset helps make an informed choice of weights to reset/keep, significantly outperforms a number of baselines, including CoTTA, which is specifically designed for continual test time adaptation.

5.5. Experiments with Similar Resetting Rate

We compare the performance of PETAL (FIM) and PETAL (SRes) with a similar resetting rate of 0.03. For CIFAR10-to-CIFAR10C, the average error rate over 5 runs for PETAL (SRes) with 0.03 restoring rate is 16.51 ± 0.02, whereas that of PETAL (FIM) is 15.95 ± 0.04. This indicates that FIM enables more parameters to be restored than stochastic restore (SRes).

5.6. Effectiveness of Various Components

In Table 1 and Table 2, the better performance of PETAL (FIM) than PETAL (S-Res) highlights the contribution of FIM for improved performance over the stochastic restore. For CIFAR10-to-CIFAR10C, the average error rate for PETAL **with no parameter restore** over 5 runs is 16.37±0.03, which is worse than having a parameter restore. In addition, the contribution of the regularizer term (caused by the source model's posterior) is clear from the better performance of PETAL over CoTTA [9], given that CoTTA is a special case of PETAL with no regularizer term.

5.7. Limitation and Future Scope

A limitation of our proposed approach is that we use an approximate posterior obtained using SWAG-diagonal, and we can experiment with better approximations for the posterior in future work. Further, we can explore the problem setting of lifelong TTA where we have a small number of examples per batch.

References

[1] Glenn W Brier et al. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950. 3

[2] Gilad Cohen and Raja Giryes. Katana: Simple post-training robustness using test time augmentations. *arXiv preprint arXiv:2109.08191*, 2021. 3

[3] Francesco Croce, Maksym Andriushchenko, Vikash Sehwag, Edoardo Debenedetti, Nicolas Flammarion, Mung Chiang, Prateek Mittal, and Matthias Hein. Robustbench: a standardized adversarial robustness benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 1, 3

[4] 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*, pages 248–255. IEEE, 2009. 1

[5] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019. 1

[6] Oğuzhan Fatih Kar, Teresa Yeo, Andrei Atanov, and Amir Zamir. 3d common corruptions and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18963–18974, 2022. 1

[7] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 1

[8] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in Neural Information Processing Systems*, 32, 2019. 3

[9] Qin Wang, Olga Fink, Luc Van Gool, and Dengxin Dai. Continual test-time domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 3, 6

[10] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 1, 3