Robustness Under Fire: Supplementary Material

1 Numerical Regression Case Details

All models used for this evaluation (target and substitute models) are implemented using Scikit Learn.[1] The model parameters are chosen by grid search.

For the target model optimization, μ_t, five folds were fitted for each of the 63 candidates, totaling 315 fits. The search space used for the grid search is described by Table 1. The optimal hyperparameters for μ_t are also used for μ_{s_1}.

Table 1: Search space for parameter optimization of target model μ_t.

Hyperparameter	**Search Space**
α	$\{0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100\}$
p	]1, 20[

For the optimization of μ_{s_2}, five folds were fitted for each of the 100 candidates, totaling 500 fits. The search space used for the grid search is described by Table 2.

Table 2: Search space for parameter optimization of substitute model μ_{s_2}.

Hyperparameter	**Search Space**
C	$\{0.01, 0.1, 1, 10, 100\}$
γ	$\{0.001, 0.01, 0.1, 1, 10\}$
ϵ	$\{0.001, 0.01, 0.1, 1\}$

For the optimization of μ_{s_3}, five folds were fitted for each of the nine candidates, totaling 45 fits. The search space used for the grid search is described by Table 3. For the span of l, the search space is capped at 1, considering that the operative area of the model is $]0, 10[$.

2 Image Classification Case Details

Data transformation of images from the CIFAR-10 dataset is conducted using the settings described by Table 4.

[1] https://scikit-learn.org

Table 3: Search space for parameter optimization of substitute model μ_{s_3}.

Hyperparameter	**Search Space**
σ^2	$\{0.1, 1, 10\}$
l	$\{0.01, 0.1, 1\}$

Table 4: Data transformation used throughout the evaluation

Parameter/Setting	**Value/Description**
Data Transformation	`RandomHorizontalFlip()` `RandomCrop(32, padding=4)` `Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))`

All models used for this evaluation (target and substitute models) are pre-made, but not pre-trained, models from the PyTorch Torchvision library. Optimization is used to find the best-performing hyperparameters and optimizers for each model. The optimization is implemented with the optimization framework Optuna and uses the search space described by Table 5 [1].[2]

Table 5: Search space for Hyperparameter and property optimization.

Hyperparameter	**Search Space**
Loss Function	`nn.CrossEntropyLoss()` (used for all)
Optimizer Type	`Adam, SGD`
Epochs	50 (used for all)
Learning Rate (`lr`)	10^{-5} to 10^{-2} (log scale)
Batch Size	{16, 32, 64, 128}
Momentum	0.8 to 0.99
Weight Decay	10^{-5} to 10^{-3} (log scale)

For optimizing the models, the number of trials is set to 50 for the target model, μ_t, and 20 for the substitute models, $\mu_{s_{1-6}}$.

The substitute dataset, $\mathcal{D}_s$, is the result of querying μ_t using $\mathcal{D}_x$. This classification task is performed with an accuracy of 81.02 percent. This results in a changed distribution of image classifications, described by Table 6, where almost 19 percent of images are misclassified.

[2] https://optuna.org/

Table 6: Distribution of images in $\mathcal{D}_x$ and $\mathcal{D}_s$

Image category	$\mathcal{D}_x$	$\mathcal{D}_s$
airplanes	2.000	1,816
automobile	2,000	1,931
bird	2,000	2,061
cat	2,000	2,227
deer	2,000	2,027
dog	2,000	1,895
frog	2,000	2,107
horse	2,000	1,880
ship	2,000	2,053
truck	2,000	2,003
sum	20,000	20,000

References

[1] Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A Next-generation Hyperparameter Optimization Framework (Jul 2019). https://doi.org/10.48550/arXiv.1907.10902, `http://arxiv.org/abs/1907.10902`, arXiv:1907.10902 [cs]