ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and
Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9.
Available from http://www.i6doc.com/en/.

Improving Fast Minimum-Norm Attacks with Hyperparameter Optimization

Giuseppe Floris[1*] , Raffaele Mura[1*], Luca Scionis[1*],
Giorgio Piras[1,2], Maura Pintor[1], Ambra Demontis[1] and Battista Biggio[1]

1- University of Cagliari - Department of Electrical and Electronic Engineering

2- Sapienza University of Rome - Department of Computer Engineering

Abstract. Evaluating the adversarial robustness of machine-learning models using gradient-based attacks is challenging. In this work, we show that hyperparameter optimization can improve fast minimum-norm attacks by automating the selection of the loss function, the optimizer, and the step-size scheduler, along with the corresponding hyperparameters. Our extensive evaluation involving several robust models demonstrates the improved efficacy of fast minimum-norm attacks when hyped up with hyperparameter optimization. We release our open-source code at `https://github.com/pralab/HO-FMN`.

1 Introduction

Machine learning (ML) models are susceptible to adversarial attacks [1, 13], i.e., input samples carefully perturbed to mislead the model. To evaluate adversarial robustness, many different gradient-based attacks have been proposed, whose performance is significantly affected by the choice of the loss function to optimize, the optimization algorithm, and the step-size scheduler. From a practical perspective, attacks tend to be run with a "default" configuration and set of hyperparameters that are deemed to fit most of the cases. Yet, the attack effectiveness is highly case-dependent, implying that the choice of the configuration needs to be carefully tailored to the model rather than a de-facto standard choice. In AutoAttack (AA) [4], the authors try to overcome this limitation by proposing an ensemble of parameter-free attacks, each including an internal auto-tuning process for each relevant hyperparameter. With Adaptive AutoAttack (AAA) [16], the approach is configured to run the parameter-free AA to look for a fast and good evaluation or alternatively come forward with an extensive search on a pool of attacks.

In this paper, we aim to use a smart and effective search for the best configuration that adapts the attack to the model. Hence, we propose a systematic framework for configuring the state-of-the-art, fast minimum-norm (FMN) attacks properly instead of running extensive searches on multiple attacks. To this end, we develop our framework by rethinking the choice of the loss function, optimizer, and step-size scheduler as attack hyperparameters and then using a unified hyperparameter optimization procedure.

[*] These authors contributed equally to this work

Algorithm 1: Fast Minimum-norm (FMN) Attack

```
Input  : $\boldsymbol{x}$, the input sample; $y$, the target (true) class label; $\alpha_0$, the
         initial $\boldsymbol{\delta}$-step size; $K$, the total number of iterations; $L$, the
         loss of the attack; $h$, the step size scheduler; $u$, the update
         function for the gradient.
Output: The minimum-norm adversarial example $\boldsymbol{x}^\star$.
$\boldsymbol{x}_0 \leftarrow \boldsymbol{x}$, $\epsilon_0 = 0$, $\boldsymbol{\delta}_0 \leftarrow \boldsymbol{0}$, $\boldsymbol{\delta}^\star \leftarrow \infty$, $\gamma_0 = 0.05$          ▷ initialization
for $k = 1, \dots, K$ do
    $\boldsymbol{g} \leftarrow \nabla_{\boldsymbol{\delta}} L(\boldsymbol{x} + \boldsymbol{\delta}_{k-1}, y, \boldsymbol{\theta})$                    ▷ loss gradient
    $\gamma_k \leftarrow h_\gamma(\gamma_0, k, K)$                              ▷ $\epsilon$-step size decay
    $\epsilon_k = u_\epsilon(\epsilon_{k-1}, \gamma_k, \|\boldsymbol{\delta}\|_p)$                            ▷ $\epsilon$-step
    $\alpha_k \leftarrow h(\alpha_0, k, K)$                                ▷ scheduler step
    $\boldsymbol{\delta}_k \leftarrow u(\boldsymbol{\delta}_{k-1}, \boldsymbol{g}/\|\boldsymbol{g}\|_2, \alpha_k)$                     ▷ optimizer step
    $\boldsymbol{\delta}_k \leftarrow \Pi(\boldsymbol{x}_0, \boldsymbol{\delta}_k)$          ▷ projection onto the feasible domain
return $\boldsymbol{x}^\star \leftarrow \boldsymbol{x}_0 + \texttt{best}(\boldsymbol{\delta}_0, ...\boldsymbol{\delta}_K)$              ▷ return best solution
```

2 FMN Attacks with Hyperparameter Optimization

We introduce here a modified FMN attack algorithm, referred to as HO-FMN, in which the loss function, the optimizer, and the step-size scheduler, along with their hyperparameters, are all exposed to be optimized. We then provide details on the hyperparameter optimizer considered in this work.

FMN Attacks. FMN [10] aims to find minimum-norm adversarial perturbations. The objective of the attack is to find, for a model with decision function $f(\cdot)$, the smallest perturbation $\boldsymbol{\delta}$ to add to an input sample $\boldsymbol{x}$ with true label y so that $f(\boldsymbol{x} + \boldsymbol{\delta}) \neq y$. To this end, it takes a PGD-step to optimize the perturbation by minimizing a loss function (i.e., the δ-step) within a given perturbation budget ϵ, then it adjusts ϵ to iteratively reduce the perturbation norm (i.e., the ϵ-step). In this work, we improve the δ-step in the FMN algorithm while leaving the ϵ-step unchanged (as the latter only optimizes a scalar value). In this regard, we consider different step-size schedulers (other than the *cosine annealing* [8] used in the baseline FMN) and more sophisticated optimizers (using different gradient update strategies other than SGD). The attack loss on which we optimize (logit loss [2]) is also put into question as we look for better candidates.

HO-FMN. We report in Algorithm 1 a revisited formulation of the FMN attack, in which the role of the loss function L, optimizer u, and scheduler h are better isolated. This novel formulation of the FMN attack enables us to generalize it by allowing a different selection of each component, treating each of them as a different hyperparameter or attack configuration. While the overall algorithm remains conceptually unchanged, we modify the attack loss L, the optimizer u, and the step-size scheduler h used in the δ-step. These are the elements that change from the original attack implementation and are made optimizable in our work. In practice, given a model, we exploit hyperparameter optimization to find the best combination of loss, optimizer, and scheduler along with their hyperparameter values (e.g., the initial step size, etc.).

3 Experiments

We describe below the experimental setup used in our work and then the results.
Hyperparameter Tuning. As introduced in Sect. 2, we aim to improve FMN by optimizing the choice of: (i) the loss function, selecting between the logit loss (LL) and the cross-entropy loss (CE); (ii) the optimizer, selecting between SGD (with and without Nesterov acceleration) and Adam (with and without AmsGrad); and (iii) the step-size scheduler, selecting among Cosine Annealing (CALR), Cosine Annealing with Warm Restarts (CAWR), MultiStep (MSLR), and Reduced On Plateau (RLROP). We define for optimizers and step-size schedulers a search space made by the possible hyperparameter values and sampling options. In particular, for each optimizer, we tune the initial step size, the momentum, and weight decay; for each scheduler, we tune the most important parameters such as the milestones in MSLR, the iteration parameters in CALR and CAWR and the factor in RLROP. The search space is then parsed in the context of the FMN hyperparameter optimization. The algorithms responsible for finding the best optimizer/scheduler configuration are the CFO search algorithm [15] and the ASHA scheduler [6]. We leverage the Ray Tune framework* for handling the hyperparameter optimization [7].
Dataset. We take a subset of 100 samples from the CIFAR10 test set for running our hyperparameter optimization, where for every model we analyzed each and every loss/optimizer/scheduler configuration. Upon finding a specific set of best hyperparameters, we use a separate set of 1000 samples (also taken from the CIFAR10 test set) to run the FMN attack on the models and discuss the results.
Perturbation Model. We restrict our analysis here to the ℓ_∞-norm attacks, as it is one of the most problematic cases for the baseline FMN algorithm.
Models. We consider 9 state-of-the art robust models from RobustBench [3]: *M0*, the WideResNet-70-16 in [14]; *M1*, the WideResNet-28-1 in [14]; *M2*, the WideResNet-70-16 in [5]; *M3*, the WideResNet-106-16 in [11]; *M4*, the WideResNet-28-10 in [5]; *M5*, the WideResNet-70-16 in [9]; *M6*, the ResNet-152 in [12]; *M7*, the WideResNet-28-10 in [9]; and *M8*, the ResNet-18 in [5].
Performance Metrics. For HO-FMN, we select the configuration that achieves the smallest median $||\boldsymbol{\delta}||_\infty$, following [10]. We then use it to evaluate the robust accuracy (RA) of the models. For minimum-norm attacks, RA can be evaluated over the entire range of perturbation sizes by imposing a threshold on the distances found, i.e., we can compute the full robustness-perturbation curve.
Experimental Results. We evaluate the candidate configurations by running HO-FMN on the 9 selected models. We show the resulting robust accuracy values at $\epsilon = 8/255$ in Table 1. We highlight how the LL loss consistently finds

*`https://docs.ray.io/en/latest/tune/index.html`

better perturbations than CE and that, in the case of the most unfortunate configuration, the models' robustness would have been considerably overestimated.

Table 1: Results for HO-FMN, compared with reference values for AA. We highlight in yellow HO-FMN (SGD) and in blue HO-FMN (Adam).

Optim.	Sched.	Loss	M0	M1	M2	M3	M4	M5	M6	M7	M8	Mean
AA			0.71	0.67	0.66	0.65	0.63	0.63	0.63	0.61	0.59	0.64
SGD	CALR	LL	**0.76**	0.74	0.70	0.66	0.66	0.68	0.66	0.58	0.62	0.67
Adam	RLROP	LL	**0.76**	**0.72**	**0.68**	**0.64**	**0.58**	**0.62**	**0.56**	**0.56**	**0.60**	**0.64**
Adam	CALR	LL	0.78	0.74	0.68	0.64	0.62	0.66	0.64	0.58	0.62	0.66
Adam	MSLR	LL	0.78	0.74	0.70	0.66	0.62	0.66	0.64	0.58	0.62	0.67
SGD	RLROP	LL	0.76	0.74	0.70	0.64	0.64	0.68	0.64	0.58	0.62	0.67
Adam	CAWR	LL	0.78	0.74	0.72	0.66	0.64	0.66	0.66	0.58	0.62	0.67
SGD	CALR	LL	0.78	0.74	0.70	0.66	0.64	0.68	0.66	0.58	0.62	0.67
SGD	CAWR	LL	0.78	0.74	0.70	0.66	0.66	0.68	0.66	0.58	0.62	0.68
SGD	MSLR	LL	0.78	0.74	0.74	0.66	0.66	0.68	0.66	0.58	0.62	0.68
Adam	MSLR	CE	0.90	0.90	0.86	0.76	0.76	0.86	0.70	0.86	0.88	0.83
SGD	MSLR	CE	0.96	0.94	0.88	0.84	0.82	0.92	0.80	0.82	0.80	0.86
SGD	CALR	CE	1.00	0.96	0.90	0.88	0.86	0.94	0.86	0.88	0.84	0.90
SGD	RLROP	CE	1.00	0.96	0.90	0.88	0.86	0.94	0.86	0.90	0.84	0.90
SGD	CAWR	CE	1.00	0.96	0.92	0.90	0.86	0.94	0.88	0.90	0.88	0.92
Adam	CALR	CE	1.00	1.00	0.94	0.96	0.90	0.96	0.92	0.94	0.94	0.95
Adam	CAWR	CE	1.00	1.00	0.96	0.96	0.94	0.96	0.92	0.94	0.94	0.96
Adam	RLROP	CE	1.00	1.00	0.96	0.96	0.92	0.98	0.92	0.94	0.94	0.96

From the results in Table 1 we select the configurations Adam+RLROP+LL and SGD+CALR+LL, and run the evaluation on our test set of 1000 CIFAR10 samples with the discovered best hyperparameters.[*]

We report the hyperparameters found with the first setting, which we name HO-FMN (Adam), listed for each model as (`learning_rate`, `weight_decay`, `factor`, `amsgrad`), M0: (5,534 0,025 0,327, False); M1: (8,801 0,043 0,366, False); M2: (4,073 0,019 0,286, False); M3: (9,616 0,024 0,301, False); M4: (7,078 0,019 0,260, False); M5: (7,078 0,019 0,260, False); M6: (4,194 0,020 0,235, False); M7: (9,339 0,023 0,352, True); M8: (4,073 0,019 0,286, False). We leave the other parameters of Adam+RLROP fixed: `eps`=10^{-8}, `betas`=(0.9, 0.999), `patience`=5, `threshold`=10^{-5}.

We list the hyperparameters for the second configuration, HO-FMN (SGD), as (`learning_rate`, `weight_decay`, `momentum`, `dampening`), M0: (4,453 0,917 0,010 0,085); M1: (1,523 0,880 0,041 0,037); M2: (1,222 0,916 0,010 0,089); M3: (3,837 0,924 0,001 0,114); M4: (1,013 0,926 0,014 0,122); M5: (3,141 0,936 0,010 0,136); M6: (1,222 0,943 0,010 0,058); M7: (2,562 0,911 0,010 0,071); M8: (2,124 0,922 0,010 0,104), and we keep fixed `T_max`=100, `eta_min`=0, `last_epoch`=-1.

We show the resulting robust accuracy for increasing perturbation sizes in Figure 1, compared with the value reported in RobustBench (computed only for $||\boldsymbol{\delta}|| = 8/255$). We remark that the values found are comparable with AA. However, our attack is able to compute the full robustness-perturbation curve

[*] `https://pytorch.org/docs/stable/optim.html`

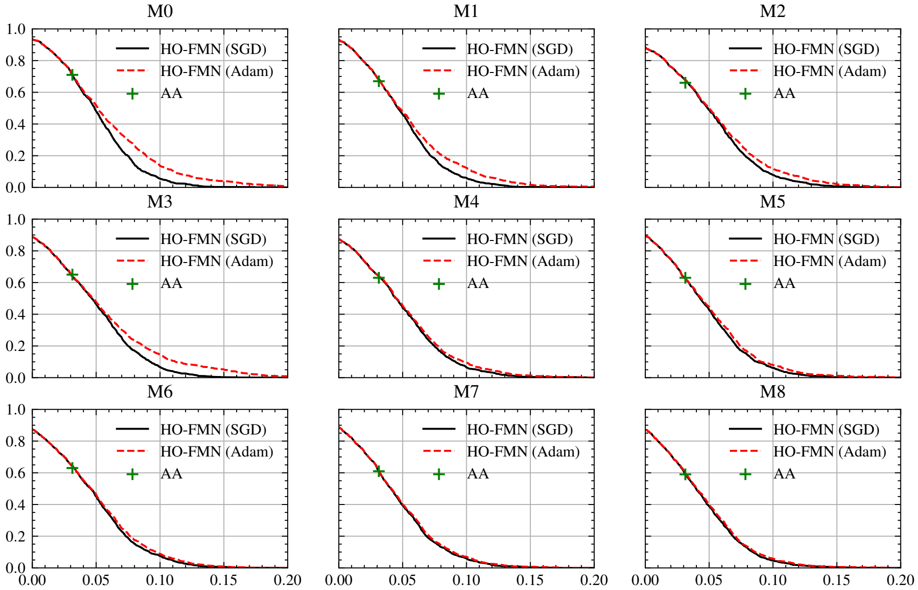

Fig. 1: Robust accuracy computed on M0-M8 with HO-FMN in the two versions Adam and SGD, and with AA at the fixed value of ϵ=8/255.

with one single optimization. Achieving the same result with AA is only possible by running AA multiple times with a range of different perturbation sizes, which would require a significant increase in computational complexity.

4 Conclusions and Future Work

In this work, we investigated the use of hyperparameter optimization to improve the performance of the FMN attack algorithm. Our findings highlight that hyperparameter optimization can improve FMN to reach competitive performance with AutoAttack while providing a more thorough adversarial robustness evaluation (i.e., computing the whole robustness-perturbation curve).

We argue that the same approach can be combined with other attacks, different threat models, or more configurations. In future work, we will extend our analysis beyond the ℓ_∞-norm FMN attack, considering ℓ_0, ℓ_1, and ℓ_2 norms.

We remark that adding more hyperparameters to tune would make the search space bigger, resulting in a longer optimization time. To this end, we will also try to develop sound heuristics to make hyperparameter tuning faster, designing faster exploration phases in the initial steps of the FMN optimization process.

Acknowledgements

We would like to express our gratitude to Sarthak Gupta for his preliminary experiments that contributed to the realization of this project. This work has been carried out while G. Piras was enrolled in the Italian National Doctorate on AI run by the Sapienza University of Rome in collaboration with the University of Cagliari. It has been supported by the PRIN 2017 project RexLearn (grant no. 2017TWNMH2), funded by the Italian Ministry of Education, University and Research; by BMK, BMDW, and the Province of Upper Austria in the frame of the COMET Programme managed by FFG in the COMET Module S3AI; and by Fondazione di Sardegna under the project "TrustML: Towards Machine Learning that Humans Can Trust", with CUP: F73C22001320007.

References

[1] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, and F. Roli. Evasion attacks against machine learning at test time. In *ECML PKDD*, 2013.

[2] N. Carlini and D. A. Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy*, pages 39–57. IEEE Computer Society, 2017.

[3] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein. Robustbench: a standardized adversarial robustness benchmark. *arXiv:2010.09670*, 2020.

[4] F. Croce and M. Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

[5] S. Gowal, S. Rebuffi, O. Wiles, F. Stimberg, D. A. Calian, and T. A. Mann. Improving robustness using generated data. In *NeurIPS*, 2021.

[6] L. Li, K. G. Jamieson, A. Rostamizadeh, E. Gonina, J. Ben-tzur, M. Hardt, B. Recht, and A. Talwalkar. A system for massively parallel hyperparameter tuning. In *MLSys*, 2020.

[7] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez, and I. Stoica. Tune: A research platform for distributed model selection and training. *arXiv:1807.05118*, 2018.

[8] I. Loshchilov and F. Hutter. SGDR: SGD with warm restarts. In *ICLR*, 2017.

[9] T. Pang, M. Lin, X. Yang, J. Zhu, and S. Yan. Robustness and accuracy could be reconcilable by (proper) definition. In *ICML*, 2022.

[10] M. Pintor, F. Roli, W. Brendel, and B. Biggio. Fast minimum-norm adversarial attacks through adaptive norm constraints. In *NeurIPS*, 2021.

[11] S. Rebuffi, S. Gowal, D. A. Calian, F. Stimberg, O. Wiles, and T. A. Mann. Fixing data augmentation to improve adversarial robustness. *CoRR*, abs/2103.01946, 2021.

[12] V. Sehwag, S. Mahloujifar, T. Handina, S. Dai, C. Xiang, M. Chiang, and P. Mittal. Robust learning meets generative models: Can proxy distributions improve adversarial robustness? In *ICLR*. OpenReview.net, 2022.

[13] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

[14] Z. Wang, T. Pang, C. Du, M. Lin, W. Liu, and S. Yan. Better diffusion models further improve adversarial training. *CoRR*, abs/2302.04638, 2023.

[15] Q. Wu, C. Wang, and S. Huang. Frugal optimization for cost-related hyperparameters. In *Thirty-Fifth AAAI Conf. Artificial Intelligence*, pages 10347–10354. AAAI Press, 2021.

[16] C. Yao, P. Bielik, P. Tsankov, and M. T. Vechev. Automated discovery of adaptive attacks on adversarial defenses. In *NeurIPS*, pages 26858–26870, 2021.