SUPPLEMENTARY MATERIAL FOR OPTIMIZATION AND GENERALIZABILITY: NEW BENCHMARKING FOR STOCHASTIC ALGORITHMS

OPEN-SOURCING

The code can be found at https://anonymous.4open.science/r/ICLR-7052/

PSEUDO-CODES OF NOISE IN GRADIENT AND NOISE IN MODEL METROPOLIS ALGORITHMS

Algorithm 10 shows the pseudocode for the NiG-MpBH algorithm, and Algorithm 10 shows the pseudocode for the NiM-MpBH algorithm. The main difference in these algorithms over the Monotonic BH versions shown in the main paper is that the monotonicity requirement is replaced with a probabilistic one, the Metropolis criterion. An additional user parameter, α is utilized for these algorithms. This parameter essentially determines how high of an increase in the loss function is allowed with some probability, as shown in lines 12 and 10, respectively. Decreases in loss are always accepted, but the Metropolis criterion allows the algorithm to allow temporary increases in loss to enhance its exploration probability and so increase the likelihood that better minima will be found further in the loss landscape. Please note that lines 16-18 and 14-16, respectively, are only essential for uses of these algorithms when the lowest-loss model is desired to be attracted. These lines are not essential. One typically monitors the training loss trajectory. In our particular setup in the main paper, we sample a fixed number of lowest-loss models from an optimization trajectory.

Algorithm 10: NiG-MpBH

```
 1: **Input:** $f(\mathbf{w}), T > 0, \epsilon \cong 0, \tau > 0, \eta, \rho, \alpha$
 2: **Output:** $\mathbf{w}_{best}$
 3: $(\mathbf{w}, \Delta t) \leftarrow LclSearch(f, \mathbf{w}, \tau, \eta, \epsilon)$
 4: $t \leftarrow t + \Delta t$
 5: $\mathbf{w}_{best} \leftarrow \mathbf{w}$
 6: **while** $t \leq T$ **do**
 7:     $\mathbf{g} \leftarrow \nabla f(\mathbf{w}_t)$
 8:     $\mathbf{g} \leftarrow PerturbGradient(\mathbf{g}, \rho)$
 9:     $\mathbf{w} \leftarrow \mathbf{w} - \eta \cdot \mathbf{g}$
10:     $(\mathbf{w}_c, \Delta t) \leftarrow LclSearch(f, \mathbf{w}, \tau, \eta, \epsilon)$
11:     $\delta_f \leftarrow f(\mathbf{w}_c) - f(\mathbf{w})$
12:     **if** $\delta_f < 0$ OR $exp(-\delta_f/\alpha) > rand(0, 1)$ **then**
13:         $\mathbf{w} \leftarrow \mathbf{w}_c$
14:         $t \leftarrow t + \Delta t$
15:     **end if**
16:     **if** $f(\mathbf{w}) < f(\mathbf{w}_{best})$ **then**
17:         $\mathbf{w}_{best} \leftarrow \mathbf{w}$
18:     **end if**
19: **end while**
```

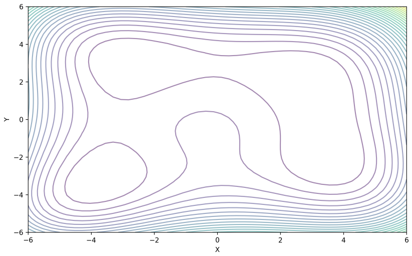
Algorithm 11: NiM-MpBH

```
Input: $f(\mathbf{w}), T > 0, \epsilon \cong 0, \tau > 0, \eta, \rho, \alpha$
Output: $\mathbf{w}_{best}$
$(\mathbf{w}, \Delta t) \leftarrow LclSearch(f, \mathbf{w}, \tau, \eta, \epsilon)$
$t \leftarrow t + \Delta t$
$\mathbf{w}_{best} \leftarrow \mathbf{w}$
while $t \leq T$ do
    $\mathbf{w} \leftarrow PerturbModel(\mathbf{w}, \rho)$
    $(\mathbf{w}_c, \Delta t) \leftarrow Lcl(f, \mathbf{w}, \tau, \eta, \epsilon)$
    $\delta_f \leftarrow f(\mathbf{w}_c) - f(\mathbf{w})$
    if $\delta_f < 0$ OR $exp(-\delta_f/\alpha) > rand(0, 1)$ then
        $\mathbf{w} \leftarrow \mathbf{w}_c$
        $t \leftarrow t + \Delta t$
    end if
    if $f(\mathbf{w}) < f(\mathbf{w}_{best})$ then
        $\mathbf{w}_{best} \leftarrow \mathbf{w}$
    end if
end while
```

Locations of Global and Local Minima of Synthetic Loss Functions

We show performance on three selected synthetic functions in the main paper, but our evaluation considers six functions: Himmelblau (shown in the main paper), Three-Hump Camel (shown in the main paper), Six-Hump Camel (shown in the main paper), Beale, Rastrigin, and Rosenbrock. Below we show the contour plots for each of these functions as well as list their global and local minima if present.

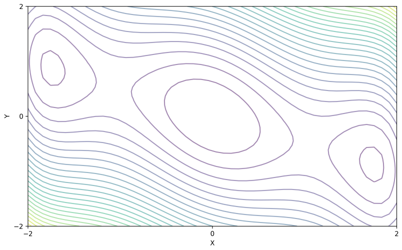
Figure 4: Himelblau



The Himmelblau function has four global minima:

1. $f(3.0, 2.0) = 0.0$
2. $f(-2.805118, 3.131312) = 0.0$
3. $f(-3.779310, -3.283186) = 0.0$
4. $f(3.584428, -1.848126) = 0.0$

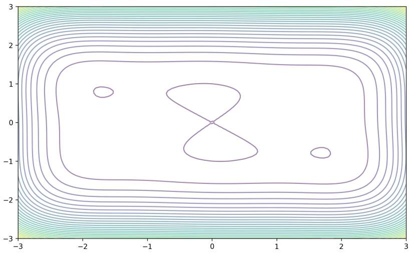
Figure 5: Three-Hump Camel



The Three-Hump camel function has a global minimum and two local minima:

1. $f(0, 0) = 0$
2. $f(1.7475, -0.8737) \approx 0.2986$
3. $f(-1.7475, 0.8737) \approx 0.2986$

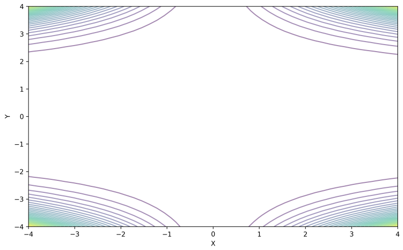
Figure 6: Six-Hump Camel



The Six-Hump Camel function has two global minima and four local minima:

1. $f(-0.0898, 0.7126) = -1.0316$
2. $f(0.0898, -0.7126) = -1.0316$
3. $f(-2.8051, -0.0312) \approx 63.848$
4. $(0.9805, 1.8367) \approx -11.5$
5. $f(1.8839, -1.5252) \approx -3.14$
6. $f(-1.8658, 1.4900) \approx -2.64$

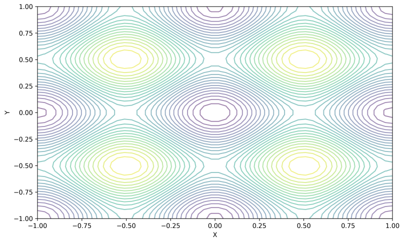
Figure 7: Beale



The Beale function has one global minimum but a very broad plateau where optimization algorithms can get stuck (and many shallow local minima):

1. $f(3, 0.5) = 0$

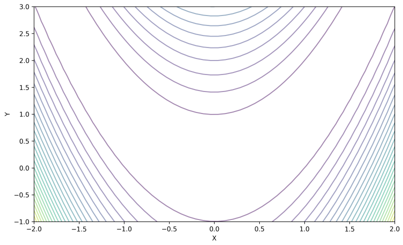
Figure 8: Rastrigin



The Rastrigin function has one global minimum and four local minima that are regularly distributed.

1. $f(-5.12, 5.12) = 529.537341$
2. $f(5.12, -5.12) = 529.537341$
3. $f(5.12, 5.12) = 529.537341$
4. $f(-5.12, -5.12) = 529.537341$
5. $f(0, 0) = 0$

Figure 9: Rosenbrock



The Rosenbrock function is a non-convex function. The global minimum is inside a long, narrow, parabolic-shaped flat valley. To find the valley is trivial, nut to converge to the global minimum, however, is difficult.

1. $f(1, 1) = 0$

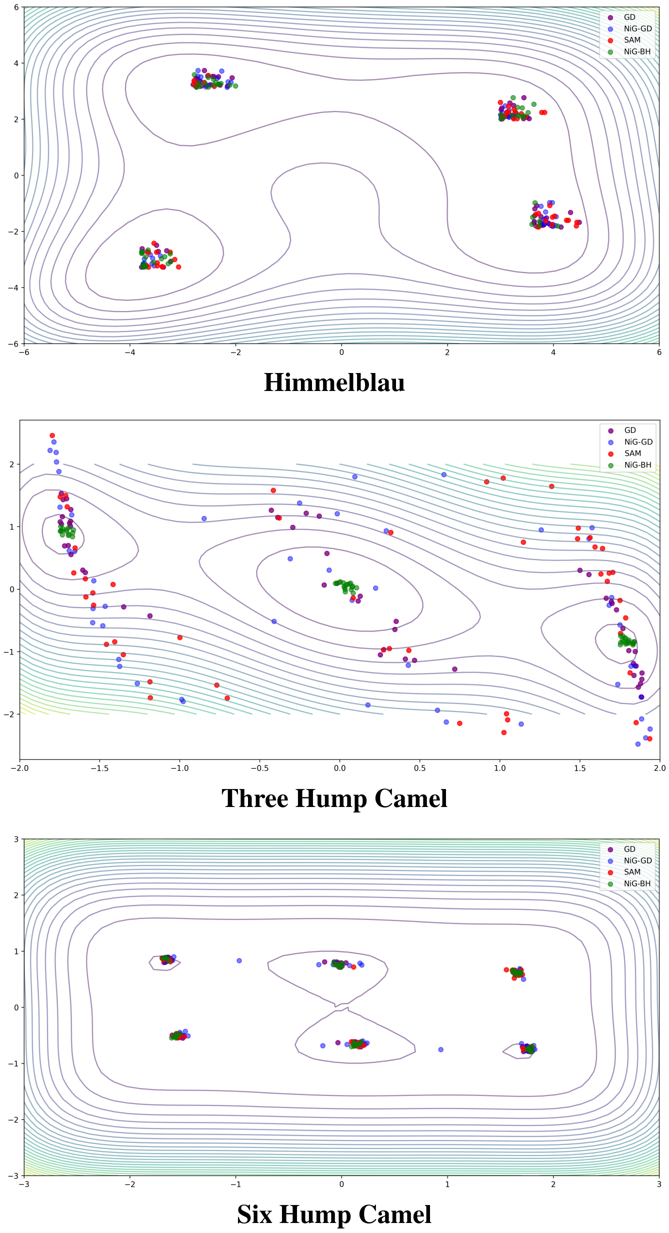
We show here the stationary distribution for Beale, Rastrigin, and Rosenbrock functions (the main paper shows for the other three functions).

Algorithms	**Beale**		**Rosenbrock**		**Rastrigin**					
	GM	**Else**	**GM**	**Else**	**GM1**	**LM1**	**LM2**	**LM3**	**LM4**	**Else**
GD	66	34	34	66	0	2.6	2.6	0	0	94.8
NiG-GD	68	32	30	70	0	5	6	0	0	89
NiM-GD	78	22	**58**	42	0	6.5	23.5	0	0	70
SAM	75	25	**55**	45	0	7	25	0	0	**68**
NiG-BH	77	23	42	58	0	7	24	0	0	69
NiM-BH	75	25	**52**	48	0	8.2	19.8	0	0	72
NiG-MBH	**78**	22	38	62	0	8	24	0	0	**68**
NiM-MBH	71	29	42	58	0	7.7	23.6	0	0	68.7
NiG-MpBH	**79**	58	42	58	0	6.3	24.0	0	0	69.7
NiM-MpBH	**80**	56	44	56	0	7.9	25.3	0	0	**66.8**

Table 5: The stationary distribution (reported in % for each entry) for the Beale, Rosenbrock, and Rastrigin function for each algorithm. The locations of the global minima (GM) and local minima (LM) for each function are listed above. The top three optimizers with the highest convergence to the global minimum on a given function are highlighted in bold font. For Rastrigin, where all optimizers have a very hard time converging to any minima, we highlight in bold font the top three optimizers that have the lowest percentage of end-points not converged to any of the minima (in the 'Else' category).

Visualization of Stationary Distributions

Figure 10 shows 50 end-points (sampled from 500) of selected algorithms on three selected synthetic functions.



Himmelblau

Three Hump Camel

Six Hump Camel

Figure 10: Stationary distribution of the optimization trajectory end-points by GD, NiG-GD (Jin et al., 2017), SAM (Foret et al., 2021), and NiG-BH. Distribution is shown for only 50 trajectories for each algorithm for a clear visual presentation.

Statistical Significance Testing

We first compare SetA to SetB in terms of test set performance and then in terms of training loss.

Comparing Test Set Performance

Mann-Whitney U Test on Base Algorithms: The Mann-Whitney-U test results on the hyperparameter-optimized algorithms are in the main paper. Table 6 reports the results on the base algorithms, with no hyperparameter optimization.

Algorithm	CIFAR10 Resnet50	CIFAR100 Resnet50	GoEmotions	TweetEval
SGD	0.2315	0.19332	0.2875	0.4621
NiG-SGD	0.2989	0.5429	0.4632	0.1654
NiM-SGD	0.6543	0.7563	0.6129	0.3219
SAM	0.0978	**0.01073**	0.1984	0.2861
NiG-BH	0.3569	**0.0285**	0.8328	0.3951
NiM-BH	0.6153	**0.0472**	0.6143	0.5178
NiG-MpBH	0.5421	0.1295	0.73256	**0.01984**
NiM-MpBH	0.6549	0.3219	0.2837	0.3542

Table 6: P-values are reported for the Mann-Whitney U test when comparing SetA to SetB for each algorithm over each of the real-world tasks. P-values less than 0.05 are highlighted in bold font.

T-Test on Hyperparameter-Optimized Algorithms

Table 7 reports results on t-tests on the hyperparameter-optimized algorithms. We test for the null hypothesis that two independent samples have identical average (expected) values. This test assumes that the populations have identical variances. With few exceptions, all p-values are under 0.05, so the null hypothesis cannot be rejected.

Algorithm	CIFAR10 Resnet50	CIFAR100 Resnet50	GoEmotions	TweetEval
SGD	0.2314	0.3156	0.7563	0.54623
NiG-SGD	0.4961	0.5753	0.72134	0.1823
NiM-SGD	**0.0421**	0.1291	0.2961	0.1962
SAM	0.7532	0.6982	0.6432	0.5391
NiG-BH	0.3612	0.18326	**0.03135**	0.1837
NiM-BH	0.6318	0.1938	0.7128	0.8723
NiG-MpBH	0.1834	0.7391	0.1935	**0.02743**
NiM-MpBH	0.13293	0.6254	0.5312	0.5193

Table 7: P-values are reported for the Mann-Whitney U test when comparing SetA to SetB for each algorithm over each of the real-world tasks. P-values less than 0.05 are highlighted in bold font.

T-test on Base Algorithms

Table 8 reports results on t-tests on the base versions of the algorithms (with no hyperparameter optimization). With few exceptions, all p-values are under 0.05, so the null hypothesis cannot be rejected.

Comparing Training Loss

Mann-Whitney U Test on Tuned Algorithms: Table 9 reports this statistical test results on comparing the loss distributions corresponding to SetA and SetB for each of the (hyperparameter-optimized) algorithms/optimizers. With few exceptions, all p-values are under 0.05, so the null hypothesis cannot be rejected; that is, there are no statistically-significant differences between SetA and SetB in terms of loss, either.

T-Test on Tuned Algorithms: Table 10 reports this statistical test results on comparing the loss distributions corresponding to SetA and SetB for each of the (hyperparameter-optimized) algo-

Algorithm	CIFAR10 Resnet50	CIFAR100 Resnet50	GoEmotions	TweetEval
SGD	0.8764	0.14019	0.2345	0.5972
NiG-SGD	0.8195	0.8423	0.8744	0.4426
NiM-SGD	0.8345	0.7425	0.34556	0.4585
SAM	0.7213	0.76894	0.6754	0.6764
NiG-BH	0.5678	**0.01245**	0.45354	0.53254
NiM-BH	0.9134	0.8325	0.3958	0.6467
NiG-MpBH	0.6753	0.34869	0.6543	0.3958
NiM-MpBH	0.7423	0.38245	0.5649	0.2867

Table 8: P-values are reported for the Mann-Whitney U test when comparing SetA to SetB for each algorithm over each of the real-world tasks. P-values less than 0.05 are highlighted in bold font.

Algorithm	CIFAR10 Resnet50	CIFAR100 Resnet50	GoEmotions	TweetEval
SGD	0.2164	**0.0021**	0.2123	0.2952
NiG-SGD	0.092	0.2385	0.0615	0.3152
NiM-SGD	0.1574	0.0612	0.1286	0.2032
SAM	**0.0001**	**0.0048**	0.3810	0.2357
NiG-BH	**0.02858**	0.1426	**0.0318**	0.5412
NiM-BH	0.3745	0.1854	**0.0325**	0.2548
NiG-MpBH	**0.0345**	**0.0238**	0.3740	0.2145
NiM-MpBH	**0.00141**	**0.0217**	0.1865	0.5402

Table 9: P-values are reported for the Mann-Whitney U test when comparing the loss distributions of SetA to SetB for each algorithm over each of the real-world tasks. P-values less than 0.05 are highlighted in bold font.

rithms/optimizers. With few exceptions, all p-values are under 0.05, so the null hypothesis cannot be rejected.

Algorithm	CIFAR10 Resnet50	CIFAR100 Resnet50	GoEmotions	TweetEval
SGD	0.5631	0.7415	0.3534	0.1983
NiG-SGD	0.6512	0.1853	**0.0916**	0.4325
NiM-SGD	0.3259	0.7122	0.3214	**0.0851**
SAM	**0.0015**	**0.0384**	0.1120	0.2352
NiG-BH	0.1523	0.2854	0.3214	**0.0254**
NiM-BH	0.2145	0.3847	**0.0978**	0.3021
NiG-MpBH	0.1854	**0.0631**	0.2654	0.5546
NiM-MpBH	0.2541	**0.0361**	0.3845	0.4153

Table 10: P-values are reported for the T test when comparing the loss distributions of SetA to SetB for each algorithm over each real-world task. P-values less than 0.05 are highlighted in bold font.

Boxplot Presentations of Distribution of Accuracy and Macro-F1 of Model Populations of Various Optimizers on Real-World Tasks

Figures 11-14 show the distribution of performance on the held-out test set of the 50 models selected by loss (to which we refer as SetA in the main manuscript) for each of the optimizers on each of the real-world tasks. On CIFAR10 (RestNet50) and CIFAR100 (ResNet50) the metric of performance is test set accuracy. On GoEmotions (BERT) and TweetEval (BERT) the metric of performance is macro-F1. Figures 15-18 do so for SetB.

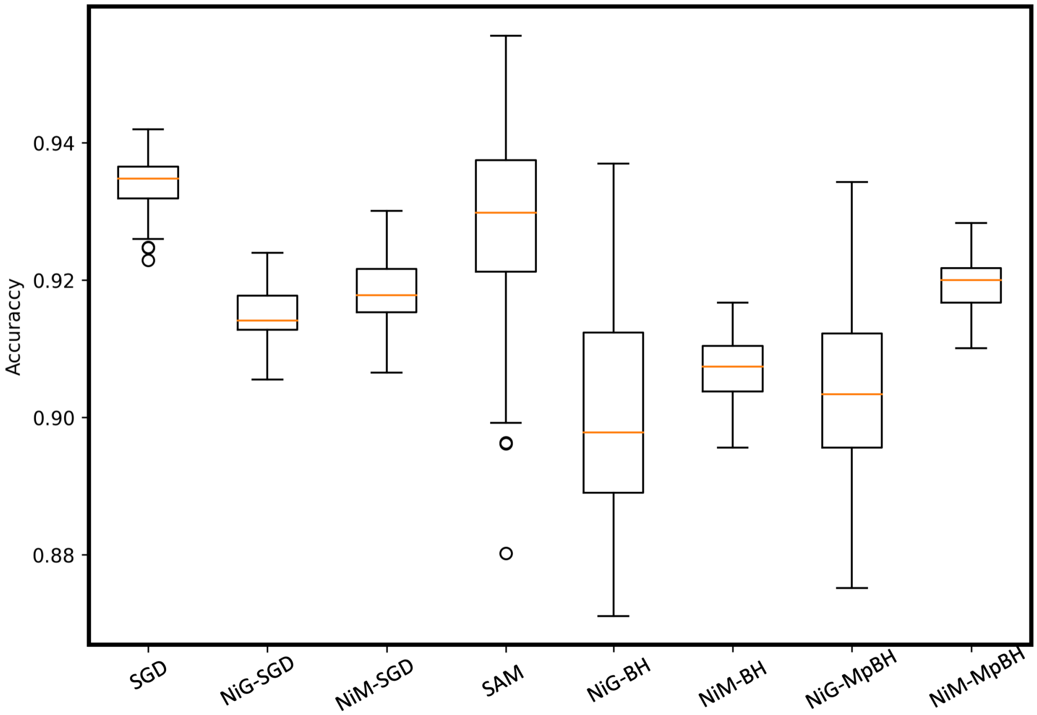


Figure 11: The distribution of test set accuracy of the 50 models extracted from each optimizer based on low-loss (to which we refer as SetA in the main paper) is shown here for each optimizer for the CIFAR10 (ResNet50) task.

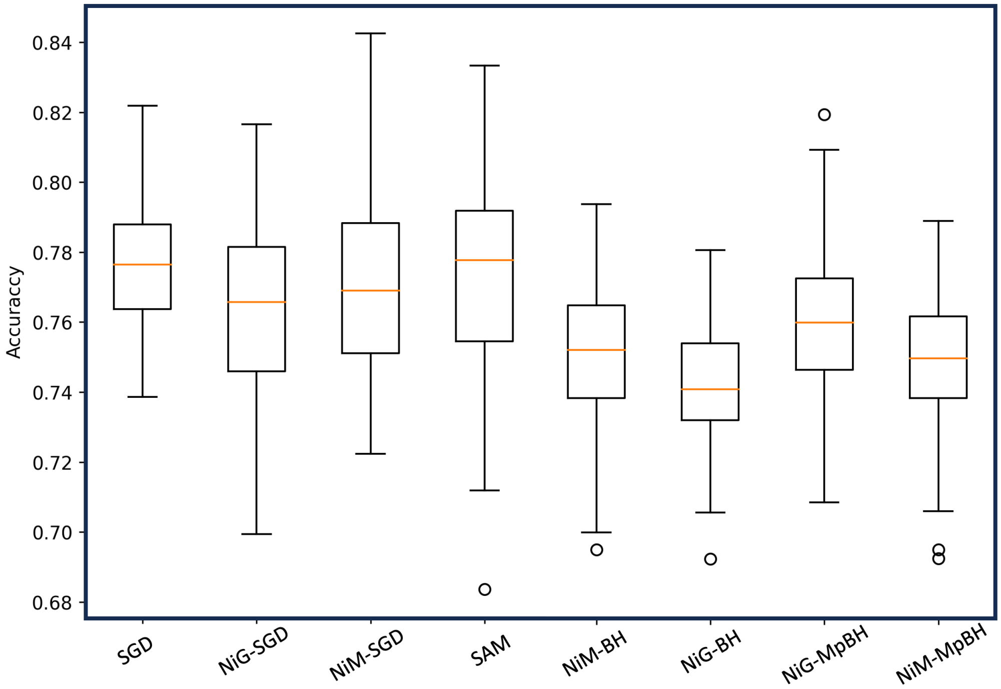


Figure 12: The distribution of test set accuracy of the 50 models extracted from each optimizer based on low-loss (to which we refer as SetA in the main paper) is shown here for each optimizer for the CIFAR100 (ResNet50) task.

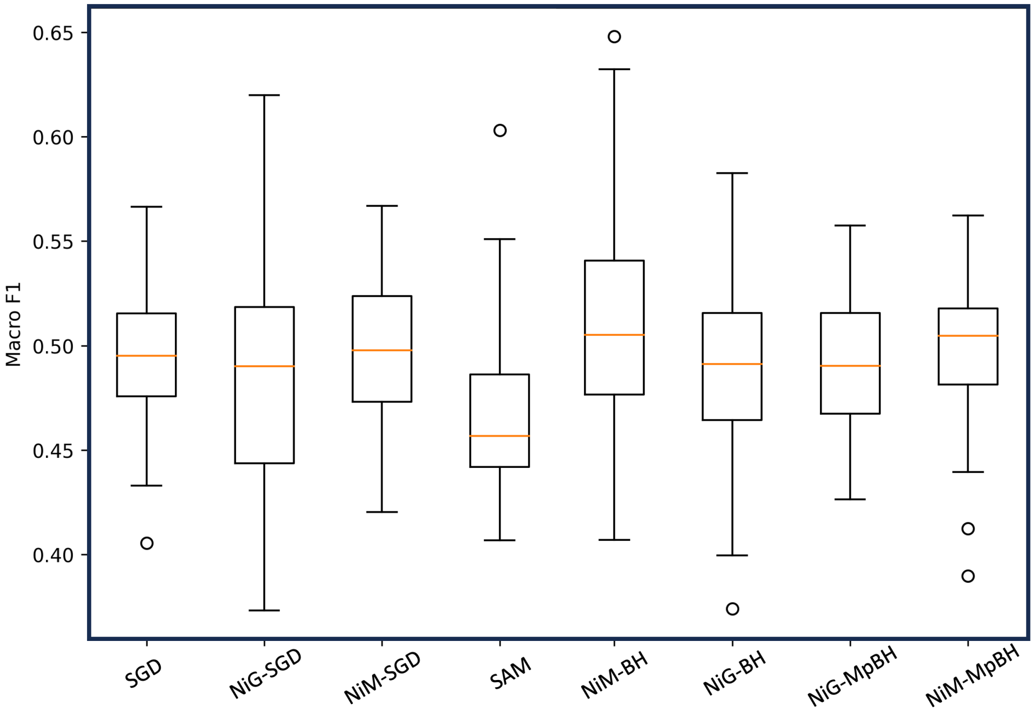


Figure 13: The distribution of macro-F1 score of the 50 models extracted from each optimizer based on low-loss (to which we refer as SetA in the main paper) is shown here for each optimizer for the GoEmotions (BERT) task.

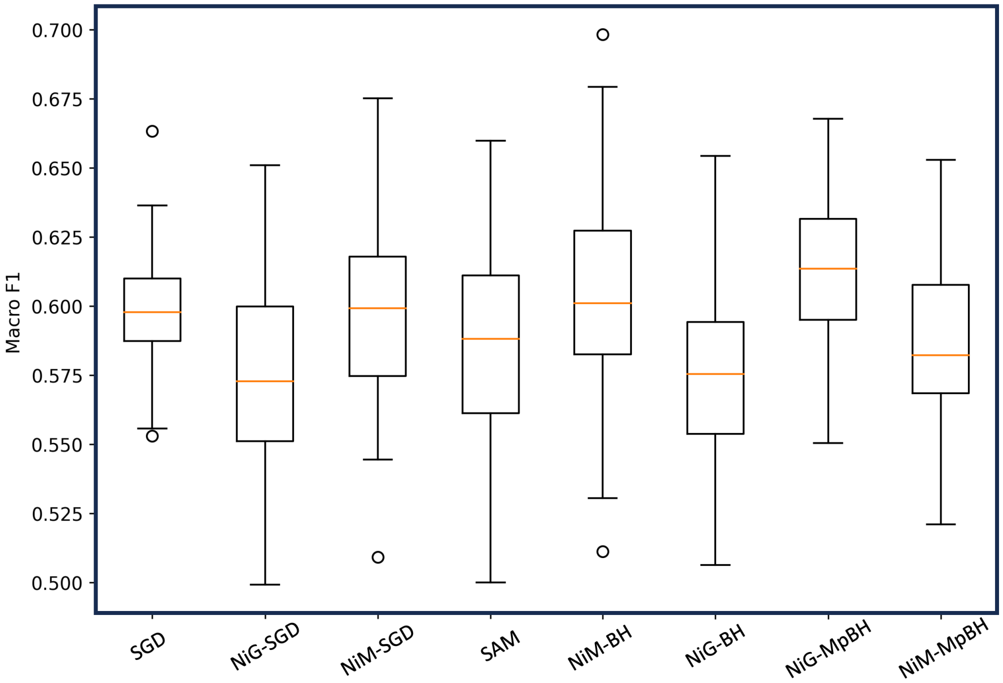


Figure 14: The distribution of macro-F1 score of the 50 models extracted from each optimizer based on low-loss (to which we refer as SetA in the main paper) is shown here for each optimizer for the TweetEval (BERT) task.

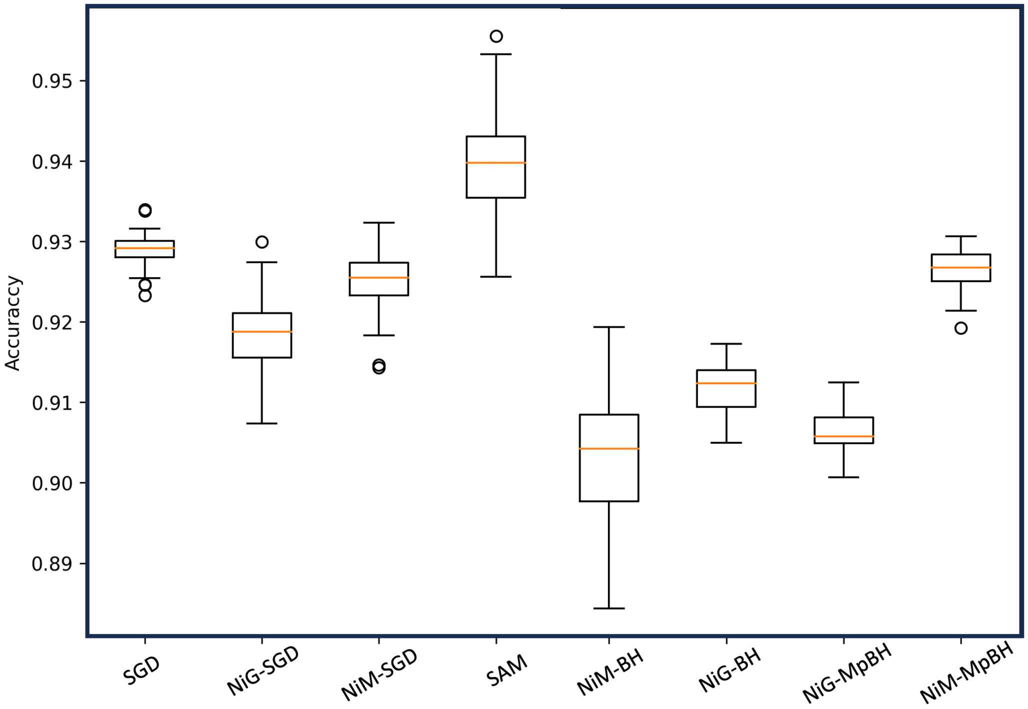


Figure 15: The distribution of test set accuracy of the 50 models extracted from each optimizer based on test set accuracy (to which we refer as SetB in the main paper) is shown here for each optimizer for the CIFAR10 (ResNet50) task.

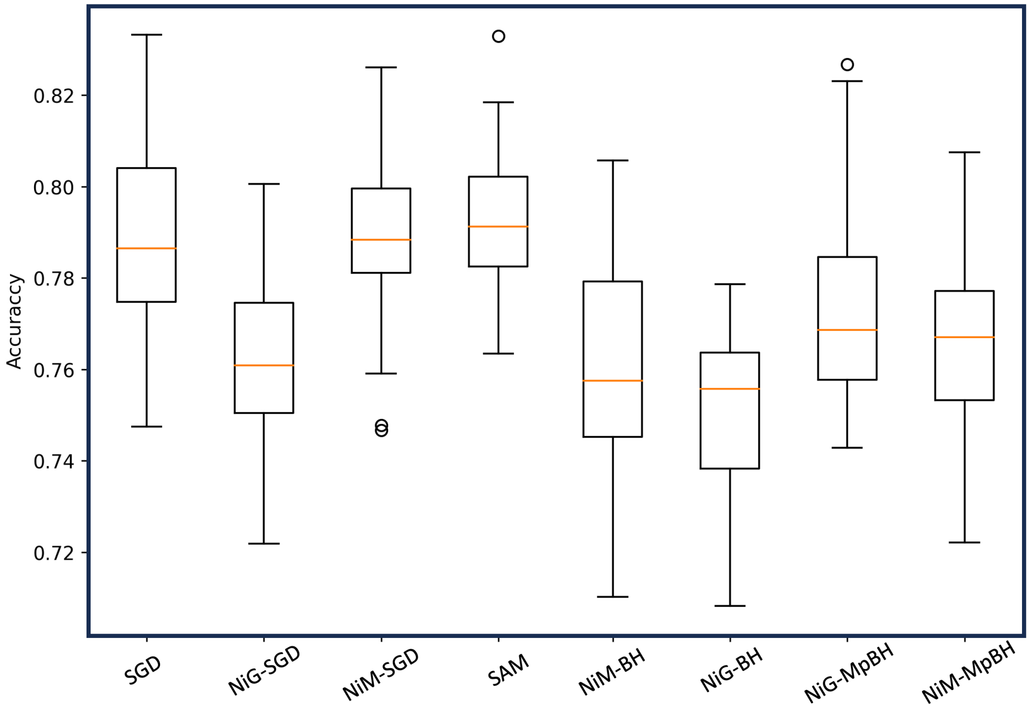


Figure 16: The distribution of test set accuracy of the 50 models extracted from each optimizer based on test set loss (to which we refer as SetB in the main paper) is shown here for each optimizer for the CIFAR100 (ResNet50) task.

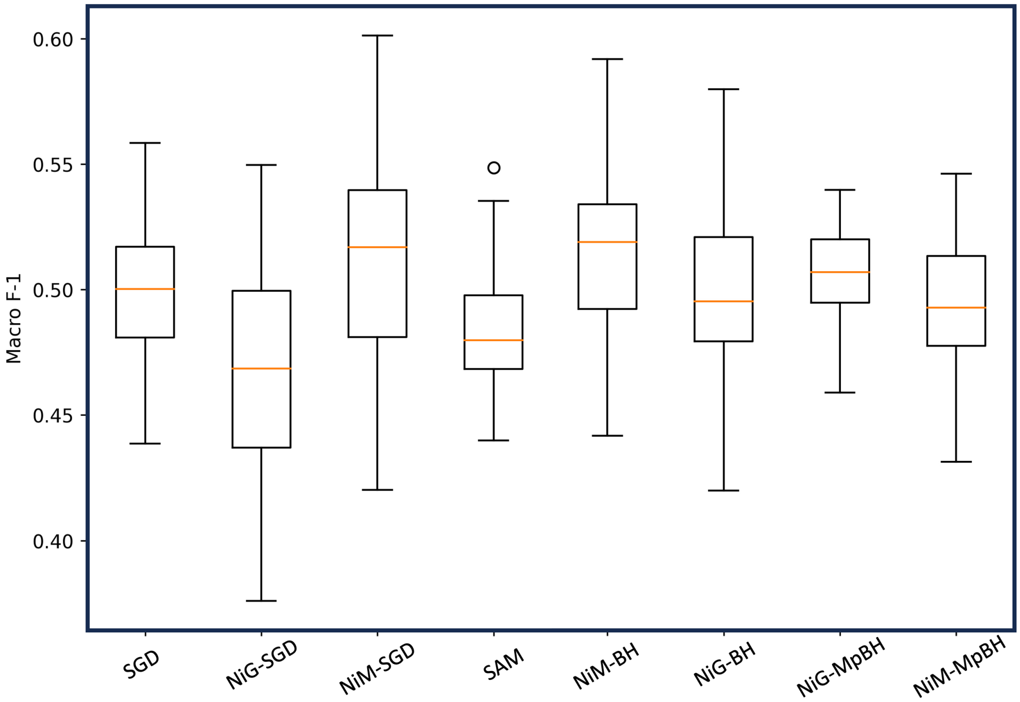


Figure 17: The distribution of test set macro-F1 of the 50 models extracted from each optimizer based on test set macro-F1 (to which we refer as SetB in the main paper) is shown here for each optimizer for the GoEmotions (BERT) task.

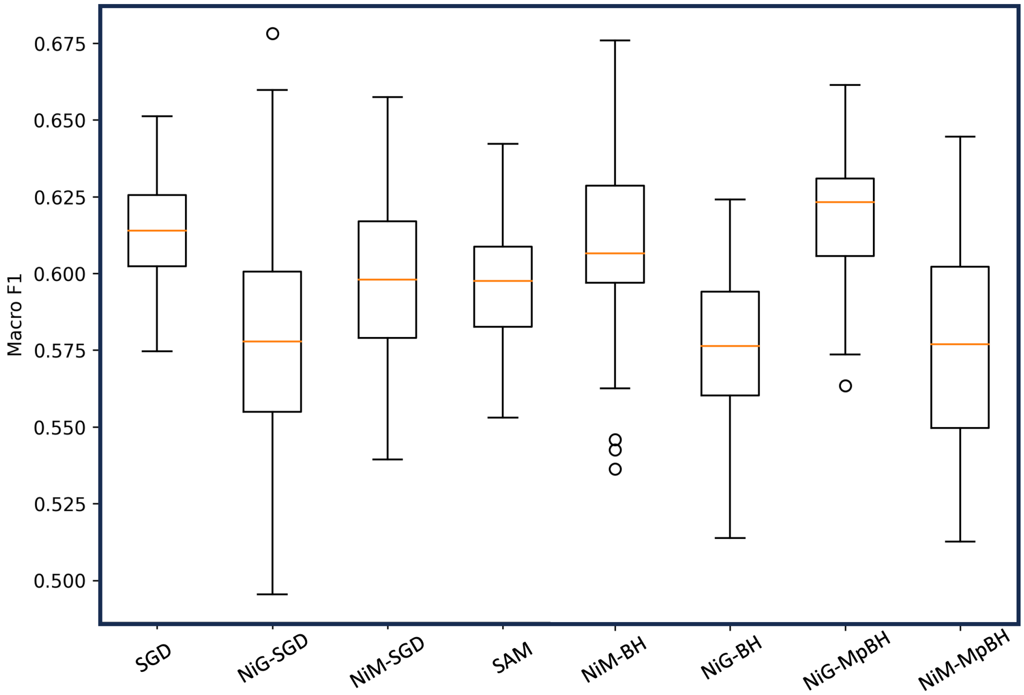


Figure 18: The distribution of test set macro-F1 of the 50 models extracted from each optimizer based on test set macro-F1 (to which we refer as SetB in the main paper) is shown here for each optimizer for the TweetEval (BERT) task.

Boxplot Presentations of Distribution of Training Loss of Model Populations of Various Optimizers on Real-World Tasks

Figures 19-22 show the distribution of performance on the training loss of the 50 models selected by loss (to which we refer as SetA in the main manuscript) for each of the optimizers on each of the real-world tasks. Figures 23-26 do so for SetB.

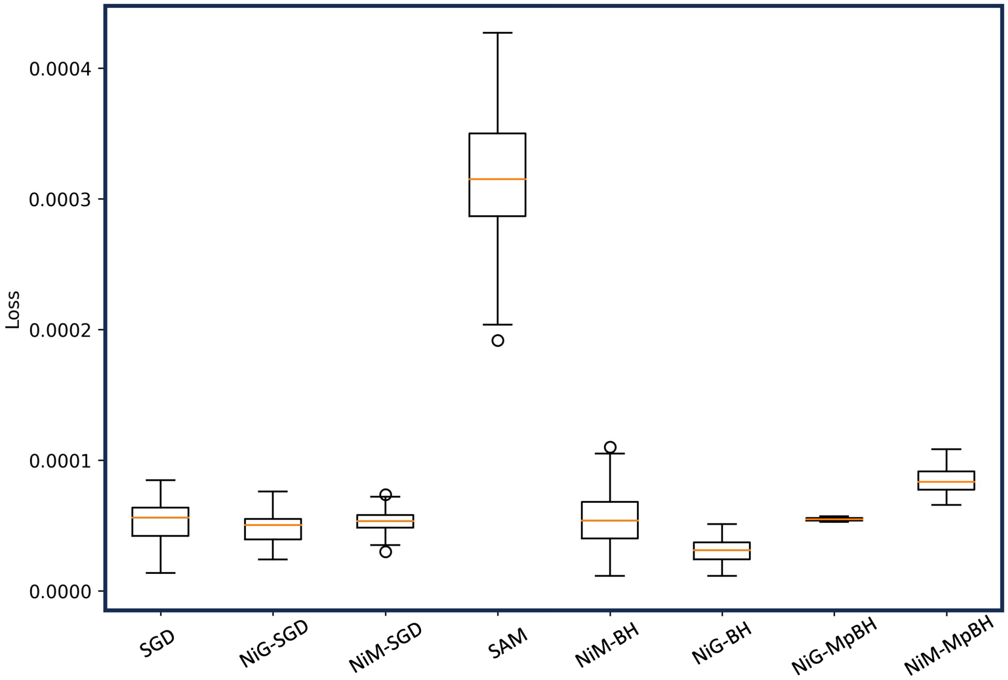


Figure 19: The distribution of training set loss of the 50 models extracted from each optimizer based on training set loss (to which we refer as SetA in the main paper) is shown here for each optimizer for the CIFAR 10 (ResNet50) task.

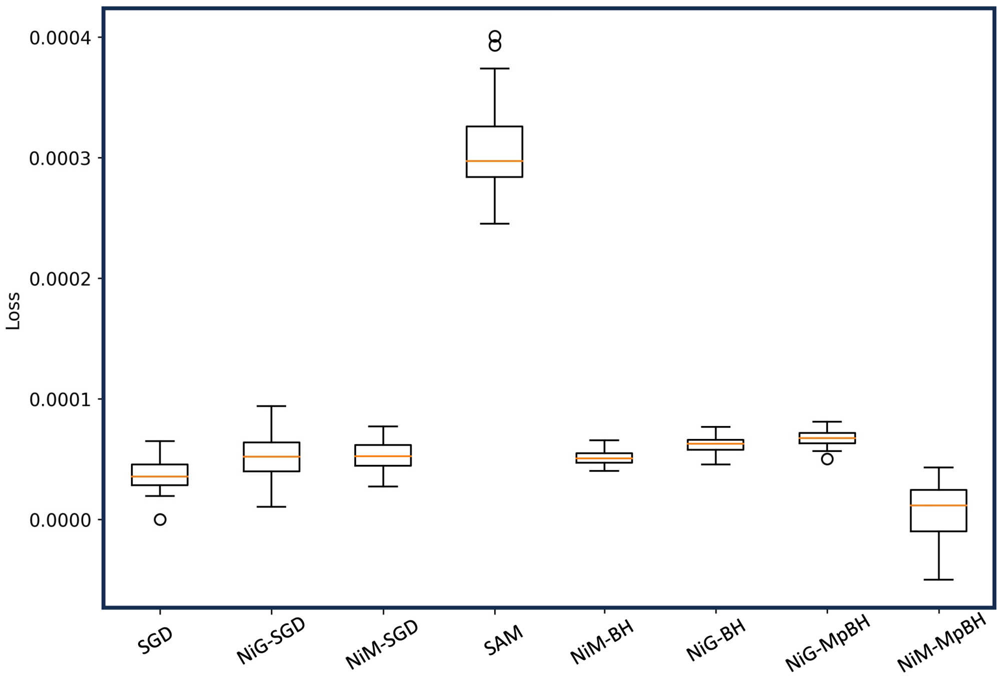


Figure 20: The distribution of training set loss of the 50 models extracted from each optimizer based on training set loss (to which we refer as SetA in the main paper) is shown here for each optimizer for the CIFAR 100 (ResNet50) task.

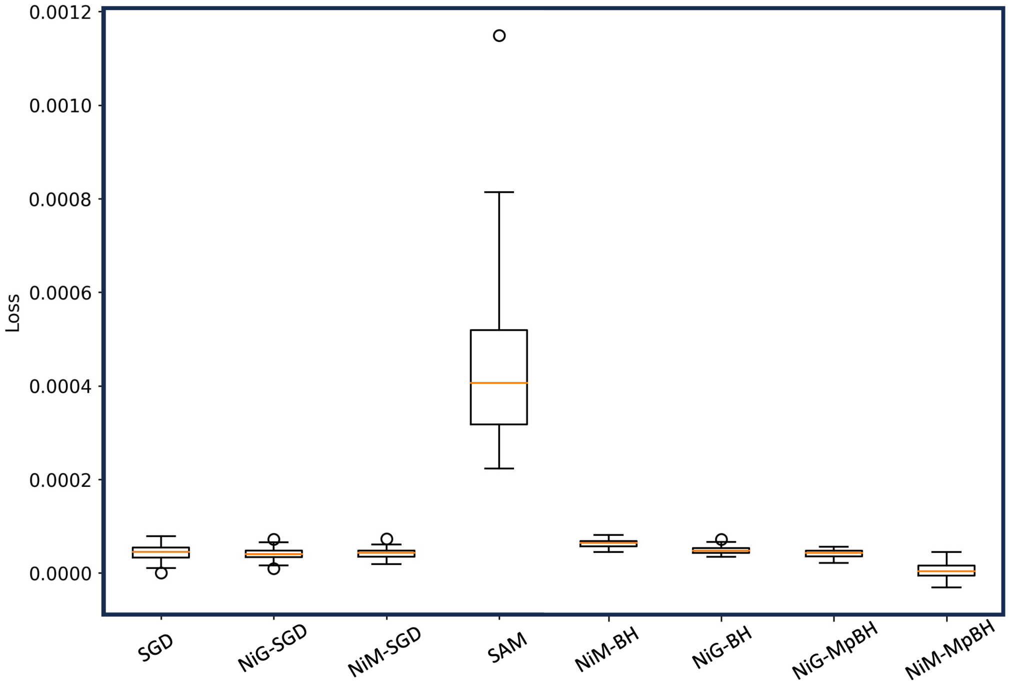


Figure 21: The distribution of training set loss of the 50 models extracted from each optimizer based on training set loss (to which we refer as SetA in the main paper) is shown here for each optimizer for the GoEmotions (BERT) task.

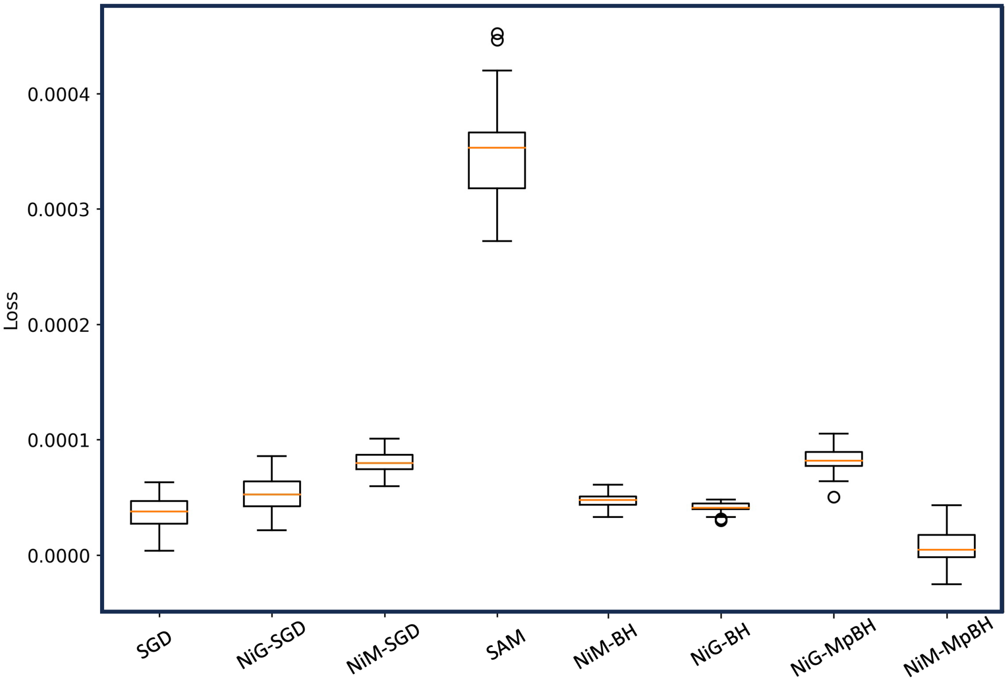


Figure 22: The distribution of training set loss of the 50 models extracted from each optimizer based on training set loss (to which we refer as SetA in the main paper) is shown here for each optimizer for the TweetEval (BERT) task.

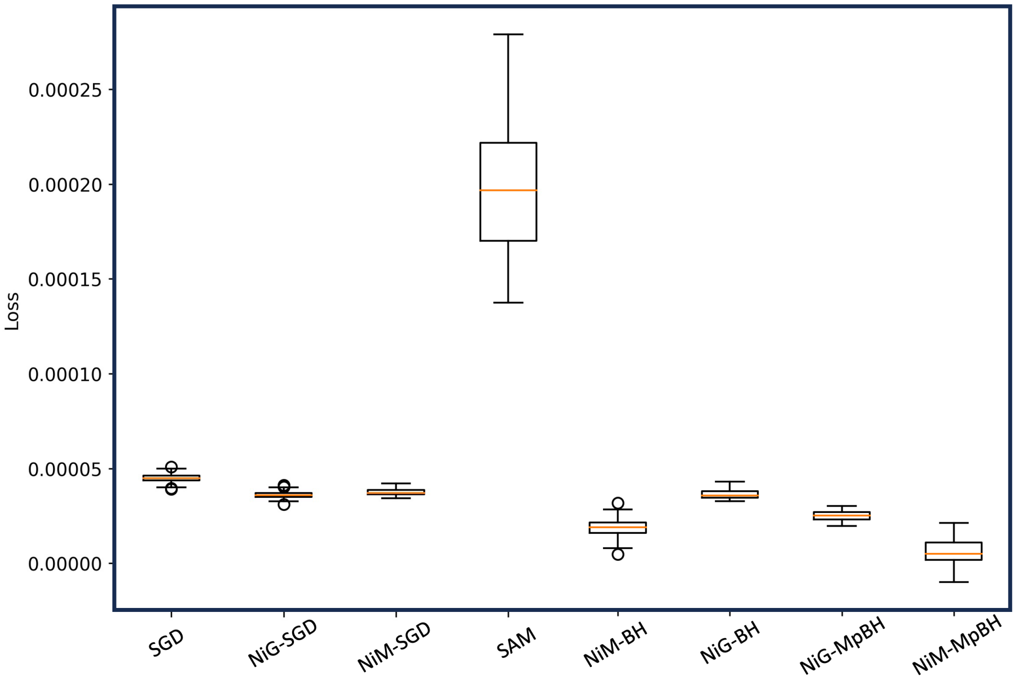


Figure 23: The distribution of training set loss of the 50 models extracted from each optimizer based on test set performance (to which we refer as SetB in the main paper) is shown here for each optimizer for the CIFAR 10 (ResNet50) task.

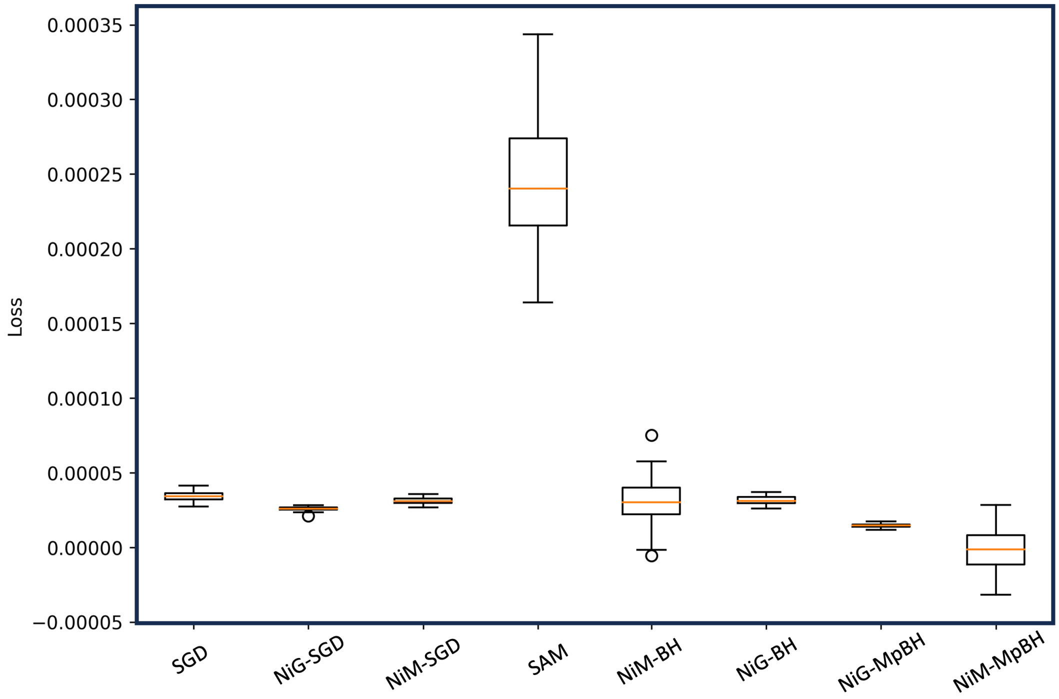


Figure 24: The distribution of training set loss of the 50 models extracted from each optimizer based on test set performance (to which we refer as SetB in the main paper) is shown here for each optimizer for the CIFAR 100 (ResNet50) task.

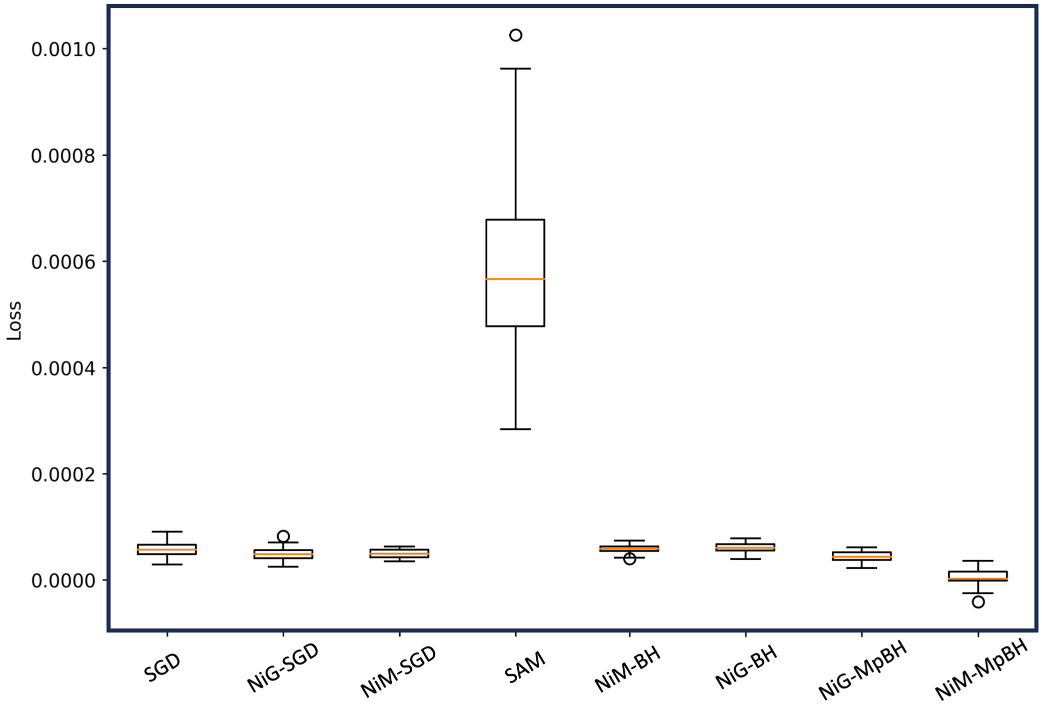


Figure 25: The distribution of training set loss of the 50 models extracted from each optimizer based on test set performance (to which we refer as SetB in the main paper) is shown here for each optimizer for the GoEmotions (Bert) task.

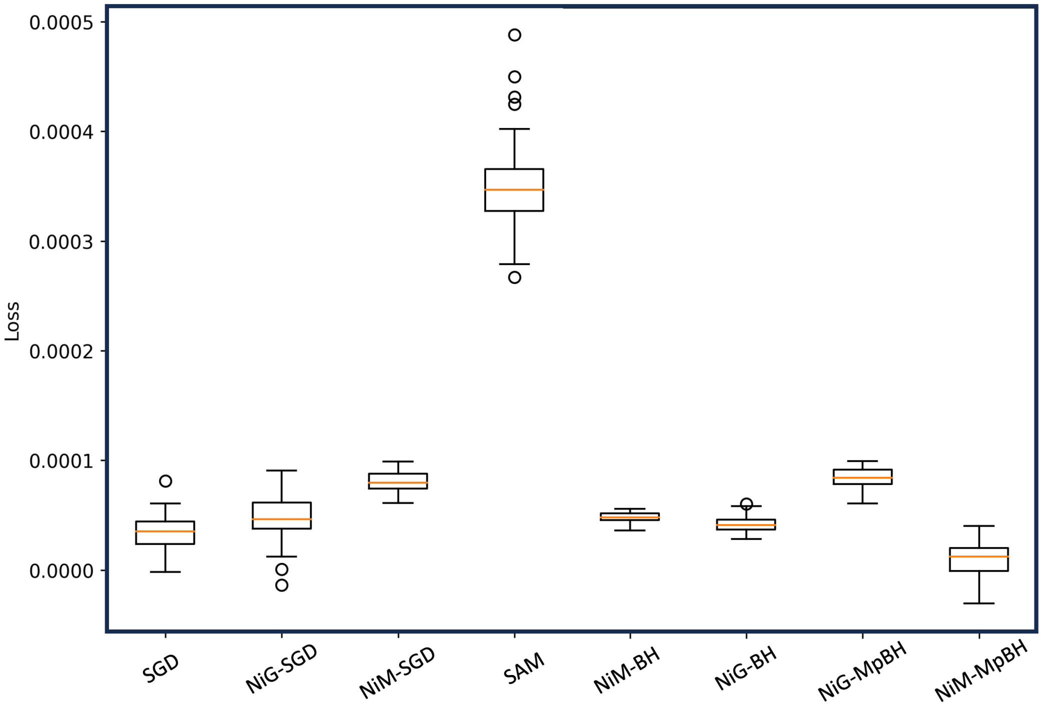


Figure 26: The distribution of training set loss of the 50 models extracted from each optimizer based on test set performance (to which we refer as SetB in the main paper) is shown here for each optimizer for the TwwetEval(Bert) task.

Expanded Analysis on Various Architectures

We expand the analysis presented in the main paper to different architectures. For a given optimizer, for each world task and for a particular model architecture, we compare SetA to SetB to relate between optimization and generalization in the statistical sense. Tables below report p-values obtained with the Mann-Whitney U test. In the main paper we report analysis with ResNet50 for CIFAR10 and CIFAR100. Here we expand to consider ResNet18, ResNet32, ResNet100, Wide-ResNet (40 X 10), and PyramidNet. In the main paper we report analysis with BERT for the NLP tasks GoEmotions and TweetEval. Here we include DistillBERT and RoBERTa. Hypothesis testing shows that the null hypothesis cannot be rejected, and so our findings are not impacted by different model architectures.

We report one table per optimizer in the interest of clarity. With few exceptions, all p-values are under 0.05, so the null hypothesis cannot be rejected. That is, the results reported in the main paper extend over model architectures, as well.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	0.2514	0.1635	-	-
ResNet 32	0.1164	0.0954	-	-
ResNet 100	0.0962	0.1535	-	-
Wide-Resnet (40 X 10)	0.1824	0.0759	-	-
PyramidNet	0.0658	0.2975	-	-
DistilBERT	-	-	0.07645	**0.03241**
RoBERTa	-	-	**0.04211**	0.07531

Table 11: Mann-Whitney U test comparing SetA to SetB for SGD over each real-world task over several model architectures. P-values < 0.05 are highlighted in bold font.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	0.2154	0.1639	-	-
ResNet 32	0.08751	0.06834	-	-
ResNet 100	0.1134	0.0861	-	-
Wide-Resnet (40 X 10)	0.2159	0.0618	-	-
PyramidNet	**0.01534**	0.0615	-	-
DistilBERT	-	-	0.2317	0.1851
RoBERTa	-	-	0.0517	**0.04531**

Table 12: Mann-Whitney U test comparing SetA to SetB for SAM over each real-world task. P-values < 0.05 are highlighted in bold font.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	0.1854	0.1125	-	-
ResNet 32	0.09645	0.1531	-	-
ResNet 100	0.2231	0.06543	-	-
Wide-Resnet (40 X 10)	**0.0314**	0.1741	-	-
PyramidNet	0.3129	0.05213	-	-
DistilBERT	-	-	0.1692	**0.0414**
RoBERTa	-	-	**0.0134**	0.0951

Table 13: Mann-Whitney U test comparing SetA to SetB for NiG-SGD over each real-world task. P-values < 0.05 are highlighted in bold font.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	0.1642	0.0851	-	-
ResNet 32	0.0613	0.3142	-	-
ResNet 100	0.0832	0.1325	-	-
Wide-Resnet (40 X 10)	0.1751	0.2162	-	-
PyramidNet	0.0835	0.1923	-	-
DistilBERT	-	-	0.9761	0.0856
RoBERTa	-	-	0.0873	0.2143

Table 14: Mann-Whitney U test comparing SetA to SetB for NiM-SGD over each real-world task. P-values < 0.05 are highlighted in bold font.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	0.1672	**0.0332**	-	-
ResNet 32	0.3511	0.0867	-	-
ResNet 100	0.1845	0.2071	-	-
Wide-Resnet (40 X 10)	0.1172	**0.0313**	-	-
PyramidNet	0.2512	0.1102	-	-
DistilBERT	-	-	0.0983	**0.02143**
RoBERTa	-	-	0.1524	**0.04531**

Table 15: Mann-Whitney U test comparing SetA to SetB for NiG-BH over each real-world task. P-values < 0.05 are highlighted in bold font.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	0.0751	0.0855	-	-
ResNet 32	0.2137	0.1980	-	-
ResNet 100	0.0631	0.1124	-	-
Wide-Resnet (40 X 10)	0.0923	0.2513	-	-
PyramidNet	0.0985	**0.0245**	-	-
DistilBERT	-	-	**0.0213**	0.2261
RoBERTa	-	-	**0.0198**	0.0678

Table 16: Mann-Whitney U test comparing SetA to SetB for NiM-BH over each real-world task. P-values < 0.05 are highlighted in bold font.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	0.0763	0.1712	-	-
ResNet 32	0.1870	0.0764	-	-
ResNet 100	0.3129	**0.0413**	-	-
Wide-Resnet (40 X 10)	0.1321	0.0571	-	-
PyramidNet	0.1439	0.0591	-	-
DistilBERT	-	-	0.2254	0.0848
RoBERTa	-	-	0.1427	0.2185

Table 17: Mann-Whitney U test comparing SetA to SetB for NiG-MpBH over each real-world task. P-values < 0.05 are highlighted in bold font.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	0.1293	0.0878	-	-
ResNet 32	0.0587	0.1859	-	-
ResNet 100	0.2391	0.1187	-	-
Wide-Resnet (40 X 10)	0.1781	0.0763	-	-
PyramidNet	0.1534	0.2871	-	-
DistilBERT	-	-	0.2143	0.0885
RoBERTa	-	-	0.0912	0.1065

Table 18: Mann-Whitney U test comparing SetA to SetB for NiM-MpBH over each real-world task. P-values < 0.05 are highlighted in bold font.

We now relate the generalization performance of SetA for each optimizer on each task on each model architecture. As in the main paper, the average accuracy (or macro-F1 for NLP tasks) and standard deviation are reported for SetA in each setting. These summary statistics are juxtaposed to the summary statistics over SetB in each setting.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	(0.930,0.012)	(0.753, 0.032)	-	-
	(0.911,0.004)	(0.762, 0.011)	-	-
ResNet 32	(0.921, 0.003)	(0.759, 0.023)	-	-
	(0.932, 0.002)	(0.761, 0.015)	-	-
ResNet 100	(0.947, 0.005)	(0.787, 0.022)	-	-
	(0.949, 0.0.012)	(0.786, 0.018)	-	-
Wide-Resnet (40 X 10)	(0.967, 0.021)	(0.813, 0.028)	-	-
	(0.971, 0.019)	(0.819, 0.017)	-	-
PyramidNet-110	(0.961, 0.005)	(0.812, 0.015)	-	-
	(0.971, 0.003)	(0.817, 0.013)	-	-
DistilBERT	-	-	(0.503, 0.053)	(0.602, 0.035)
	-	-	(0.506, 0.038)	(0.607, 0.327)
RoBERTa	-	-	(0.494, 0.033)	(0.613, 0.027)
	-	-	(0.502, 0.016)	(0.619, 0.025)

Table 19: For each architecture, we relate the average accuracy and standard deviation over SetA (top row) and SetB (bottom row) for SGD. '(,)' relates '(average, standard deviation)' over models in a set. On the NLP tasks, summary statistics are for macro-F1.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	(0.911,0.015)	(0.761, 0.025)	-	-
	(0.909,0.005)	(0.759, 0.018)	-	-
ResNet 32	(0.921, 0.018)	(0.769, 0.022)	-	-
	(0.927, 0.014)	(0.751, 0.008)	-	-
ResNet 100	(0.945, 0.005)	0.788, 0.017)	-	-
	(0.955, 0.004)	(0.791, 0.028)	-	-
Wide-Resnet (40 X 10)	0.961, 0.011)	0.8123, 0.027)	-	-
	(0.966, 0.012)	(0.823, 0.021)	-	-
PyramidNet	(0.965, 0.005)	(0.821 0.015)	-	-
	(0.971, 0.013)	(0.823, 0.011)	-	-
DistilBERT	-	-	(0.515, 0.032)	(0.611, 0.041)
	-	-	(0.521, 0.011)	(0.621, 0.012)
RoBERTa	-	-	(0.512, 0.013)	(0.621, 0.019)
	-	-	(0.525, 0.026)	(0.626, 0.031)

Table 20: For each architecture, we relate the average accuracy and standard deviation over SetA (top row) and SetB (bottom row) for SAM. '(,)' relates '(average, standard deviation)' over models in a set. On the NLP tasks, summary statistics are for macro-F1.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	(0.912,0.012)	(0.756, 0.018)	-	-
	(0.909,0.005)	(0.755, 0.011)	-	-
ResNet 32	(0.921, 0.009)	(0.761, 0.018)	-	-
	(0.925, 0.004)	(0.763, 0.022)	-	-
ResNet 100	(0.932, 0.015)	(0.781, 0.015)	-	-
	(0.945, 0.007)	(0.788, 0.021)	-	-
Wide-Resnet (40 X 10)	(0.955, 0.013)	(0.791, 0.021)	-	-
	(0.959, 0.004)	(0.795, 0.011)	-	-
PyramidNet	(0.961, 0.013)	(0.797, 0.011)	-	-
	(0.959, 0.009)	(0.787, 0.017)	-	-
DistilBERT	-	-	(0.511, 0.023)	(0.599, 0.021)
	-	-	(0.516, 0.026)	(0.601, 0.021)
RoBERTa	-	-	(0.512, 0.015)	(0.609, 0.005)
	-	-	(0.532, 0.006)	(0.611, 0.018)

Table 21: For each architecture, we relate the average accuracy and standard deviation over SetA (top row) and SetB (bottom row) for NiG-SGD. '(,)' relates '(average, standard deviation)' over models in a set. On the NLP tasks, summary statistics are for macro-F1.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	(0.891,0.011)	0.744, 0.025)	-	-
	(0.881,0.013)	(0.751, 0.012)	-	-
ResNet 32	(0.901, 0.013)	(0.754, 0.022)	-	-
	(0.905, 0.002)	(0.758, 0.023)	-	-
ResNet 100	(0.921, 0.015)	(0.768, 0.025)	-	-
	(0.918, 0.007)	(0.776, 0.015)	-	-
Wide-Resnet (40 X 10)	(0.925, 0.021)	(0.781, 0.029)	-	-
	(0.928, 0.011)	(0.788, 0.021)	-	-
PyramidNet	(0.944, 0.014)	(0.791, 0.019)	-	-
	(0.948, 0.011)	(0.798, 0.021)	-	(-
DistilBERT	-	-	(0.511, 0.013)	(0.612,0.022)
	-	-	(0.521, 0.011)	(0.611, 0.029)
RoBERTa	-	-	(0.517, 0.009)	(0.609, 0.017)
	-	-	(0.511, 0.013)	(0.613, 0.023)

Table 22: For each architecture, we relate the average accuracy and standard deviation over SetA (top row) and SetB (bottom row) for NiM-SGD. '(,)' relates '(average, standard deviation)' over models in a set. On the NLP tasks, summary statistics are for macro-F1.

Architecture	Problems			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	(0.921,0.007)	(0.761, 0.023)	-	-
	(0.924,0.006)	(0.768, 0.015)	-	-
ResNet 32	(0.919, 0.011)	(0.773, 0.015)	-	-
	(0.922, 0.010)	(0.779, 0.012)	-	-
ResNet 100	(0.941, 0.021)	0.781, 0.019)	-	-
	(0.945, 0.011)	(0.787, 0.021)	-	-
Wide-Resnet (40 X 10)	(0.959, 0.014)	(0.801, 0.023)	-	-
	(0.964, 0.024)	(0.821, 0.021)	-	-
PyramidNet	(0.961, 0.027)	(0.833, 0.022)	-	-
	(0.962, 0.013)	(0.839, 0.028)	-	(-
DistilBERT	-	-	(0.523, 0.023)	(0.622, 0.013)
	-	-	(0.541, 0.028)	(0.627, 0.013)
RoBERTa	-	-	(0.553, 0.026)	(0.63, 0.018)
	-	-	(0.552, 0.011)	(0.632, 0.010)

Table 23: For each architecture, we relate the average accuracy and standard deviation over SetA (top row) and SetB (bottom row) for NiG-BH. '(,)' relates '(average, standard deviation)' over models in a set. On the NLP tasks, summary statistics are for macro-F1.

Architecture	Problems			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	(0.922,0.012)	(0.762,0.027)	-	-
	(0.929,0.012)	(0.771, 0.024)	-	-
ResNet 32	(0.925, 0.012)	(0.769, 0.021)	-	-
	(0.928, 0.014)	(0.773, 0.021)	-	-
ResNet 100	(0.947, 0.008)	(0.785, 0.016)	-	-
	(0.951, 0.022)	(0.786, 0.011)	-	-
Wide-Resnet (40 X 10)	(0.961, 0.007)	(0.811, 0.022)	-	-
	(0.969, 0.024)	(0.823, 0.005)	-	-
PyramidNet	(0.961, 0.028)	(0.831, 0.024)	-	-
	(0.965, 0.014)	(0.835, 0.029)	-	-
DistilBERT	-	-	(0.521, 0.022)	(0.632, 0.011)
	-	-	(0.533, 0.031)	(0.635 0.011)
RoBERTa	-	-	(0.544, 0.021)	(0.638,0.022)
	-		(0.552, 0.028)	(0.639, 0.021)

Table 24: For each architecture, we relate the average accuracy and standard deviation over SetA (top row) and SetB (bottom row) for NiM-BH. '(,)' relates '(average, standard deviation)' over models in a set. On the NLP tasks, summary statistics are for macro-F1.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	(0.901,0.019) (0.911,0.017)	(0.759,0.019) (0.764, 0.003)	- -	- -
ResNet 32	(0.911, 0.010) (0.915, 0.021)	(0.771, 0.021) (0.776, 0.014)	- -	- -
ResNet 100	(0.934, 0.025) (0.936, 0.011)	(0.781, 0.021) (0.787, 0.026)	– -	- -
Wide-Resnet (40 X 10)	(0.944, 0.011) (0.949, 0.022)	(0.80, 0.028) (0.795 0.027)	- -	- -
PyramidNet	(0.954, 0.017) (0.956, 0.017)	(0.821, 0.009) (0.811, 0.021)	- -	- -
DistilBERT	- -	- -	(0.521, 0.021) (0.527, 0.031)	(0.622,0.019) (0.625, 0.027)
RoBERTa	- -	- -	(O.528, 0.014) (0.531, 0.008)	(0.625, 0.027) (0.631, 0.032)

Table 25: For each architecture, we relate the average accuracy and standard deviation over SetA (top row) and SetB (bottom row) for NiG-MpBH. '(,)' relates '(average, standard deviation)' over models in a set. On the NLP tasks, summary statistics are for macro-F1.

Architecture	**Problems**			
	CIFAR 10	CIFAR 100	GoEmotions	TweetEval
ResNet 18	(0.901,0.011) (0.905,0.010)	(0.767,0.023) (0.765, 0.019)	- -	- -
ResNet 32	(0.911, 0.021) (0.921, 0.018)	(0.767, 0.031) (0.769, 0.020)	- -	- -
ResNet 100	(0.925, 0.021) (0.922, 0.008)	(0.787, 0.031) (0.786, 0.020)	- -	- -
Wide-Resnet (40 X 10)	(0.944, 0.028) (0.949, 0.011)	(0.809,0.022) (0.809, 0.029)	- -	- -
PyramidNet	(0.957, 0.011) (0.954, 0.026)	(0.821, 0.027) (0.833, 0.024)	- -	- -
DistilBERT	- -	- -	(0.521, 0.023) (0.533, 0.011)	(0.611,0.029) (0.620, 0.021)
RoBERTa	- -	- -	(0.531, 0.026) (0.534, 0.022)	(0.622, 0.018) (0.625, 0.026)

Table 26: For each architecture, we relate the average accuracy and standard deviation over SetA (top row) and SetB (bottom row) for NiM-MpBH. '(,)' relates '(average, standard deviation)' over models in a set. On the NLP tasks, summary statistics are for macro-F1.