A Additional Experiments

A.1 Estimating Test Robust Accuracy

Instead of estimating the robust generalization gap, one might expect the analysis on the relationship between test robust accuracy and the measures. In this regard, we investigate the correlation between the measures and the test robust accuracy $1 - \mathcal{E}(\boldsymbol{w}; \epsilon, \mathcal{D})$ on the test dataset $\mathcal{D}$ instead of the robust generalization gap $g(\boldsymbol{w})$.

Table 8: Numerical results of ψ_k between each measure and *test robust accuracy*, along with its corresponding standard deviation.

	Model Architecture	Training Methods	Steps	Optimizer	Batch-size	Aug	Extra-data	Early Stopping	Total τ
num_params (7)	0.14±0.44	-	-	-	-	-	-	-	0.02
path_norm (8)	0.22±0.57	0.36±0.41	0.33±0.94	0.08±0.72	0.22±0.50	0.20±0.70	0.17±0.73	-0.43±0.51	0.02
log_prod_of_spec (9)	-0.22±0.43	0.06±0.42	0.05±1.00	0.04±0.73	0.03±0.49	0.07±0.73	0.07±0.73	-0.16±0.68	-0.15
log_prod_of_fro (10)	0.12±0.41	0.36±0.39	0.64±0.77	0.30±0.70	0.44±0.41	0.39±0.66	0.40±0.65	0.08±0.66	0.07
euclid_init_norm (11)	0.16±0.45	0.28±0.31	-0.14±0.99	-0.01±0.73	-0.14±0.48	-0.13±0.69	-0.10±0.68	-0.23±0.62	0.10
average_ce (12)	-0.09±0.62	0.03±0.53	0.08±1.00	0.17±0.73	-0.13±0.57	-0.01±0.77	0.01±0.76	0.55±0.47	0.14
inverse_margin (13)	0.24±0.56	0.44±0.36	0.48±0.88	0.44±0.60	0.18±0.50	0.28±0.69	0.24±0.68	0.71±0.37	0.44
prob_margin (14)	0.05±0.62	-0.03±0.54	-0.09±1.00	-0.16±0.74	0.11±0.57	-0.01±0.77	-0.03±0.75	-0.56±0.46	-0.14
boundary_thickness (15)	0.07±0.60	0.24±0.49	0.26±0.97	0.29±0.71	0.06±0.53	0.17±0.75	0.21±0.73	0.62±0.43	0.32
kl_divergence (16)	-0.55±0.49	-0.57±0.36	-0.94±0.35	-0.65±0.48	-0.46±0.40	-0.37±0.60	-0.45±0.61	-0.63±0.47	-0.45
local_lip (17)	-0.46±0.49	-0.44±0.44	-0.70±0.72	-0.58±0.56	-0.32±0.44	-0.28±0.65	-0.33±0.69	-0.64±0.49	-0.40
pacbayes_flat (18)	0.31±0.52	0.21±0.44	0.31±0.95	0.17±0.77	0.25±0.51	0.28±0.68	0.26±0.67	-0.17±0.67	0.15
estimated_sharpness (19)	0.09±0.60	0.25±0.39	0.36±0.93	0.16±0.75	0.21±0.50	0.30±0.68	0.32±0.67	0.20±0.66	0.18
estimated_inv_sharpness (20)	0.24±0.56	0.39±0.33	0.56±0.83	0.29±0.74	0.29±0.50	0.38±0.65	0.41±0.64	0.51±0.58	0.33
average_flat (21)	-0.44±0.50	-0.60±0.41	-0.82±0.57	-0.64±0.51	-0.40±0.42	-0.30±0.62	-0.35±0.63	-0.66±0.46	-0.42
x_grad_norm (22)	-0.40±0.48	-0.34±0.39	-0.74±0.68	-0.49±0.59	-0.37±0.41	-0.26±0.65	-0.30±0.65	-0.09±0.76	-0.17
w_grad_norm (23)	0.20±0.52	0.42±0.35	0.66±0.75	0.33±0.75	0.29±0.50	0.42±0.66	0.42±0.65	0.60±0.54	0.37
average_ce(PGD) (12)	-0.70±0.41	-0.67±0.19	-0.89±0.46	-0.60±0.48	-0.61±0.35	-0.51±0.58	-0.50±0.60	-0.57±0.51	-0.58
inverse_margin(PGD) (13)	0.58±0.47	0.64±0.30	0.86±0.50	0.69±0.45	0.51±0.43	0.49±0.62	0.59±0.56	0.78±0.33	0.61
prob_margin(PGD) (14)	0.74±0.43	0.73±0.18	0.92±0.38	0.66±0.43	0.60±0.39	0.54±0.58	0.52±0.60	0.58±0.50	0.60
pacbayes_flat(PGD) (18)	0.62±0.44	0.59±0.24	0.81±0.58	0.60±0.56	0.52±0.45	0.60±0.48	0.72±0.38	0.11±0.77	0.61
estimated_sharpness(PGD) (19)	-0.17±0.65	-0.28±0.54	-0.25±0.97	-0.22±0.75	-0.06±0.57	-0.01±0.74	-0.01±0.72	-0.49±0.48	-0.22
estimated_inv_sharpness(PGD) (20)	-0.20±0.65	-0.22±0.56	-0.22±0.98	-0.19±0.75	-0.05±0.56	0.01±0.74	0.04±0.71	-0.48±0.48	-0.21
x_grad_norm(PGD) (22)	-0.37±0.51	-0.45±0.35	-0.77±0.64	-0.55±0.59	-0.32±0.44	-0.21±0.64	-0.29±0.66	-0.63±0.45	-0.38
w_grad_norm(PGD) (23)	-0.30±0.61	-0.28±0.50	-0.28±0.96	-0.26±0.73	-0.09±0.57	-0.05±0.72	-0.05±0.73	-0.47±0.50	-0.28

Table 9: Numerical results of π_k between each measure and *test robust accuracy*, along with its corresponding standard deviation.

	Model Architecture	Training Methods	Steps	Optimizer	Batch-size	Aug	Extra-data	Early Stopping	Total τ
num_params (7)	-	0.04±0.04	0.05±0.12	0.03±0.01	0.02±0.03	0.02±0.00	0.03±0.02	0.02±0.02	0.02
path_norm (8)	0.01±0.12	-0.20±0.19	-0.33±0.14	0.04±0.00	0.02±0.03	0.04±0.01	0.02±0.05	0.27±0.18	0.02
log_prod_of_spec (9)	-0.06±0.12	-0.08±0.10	-0.00±0.09	-0.15±0.06	-0.14±0.05	-0.15±0.01	-0.15±0.03	-0.16±0.04	-0.15
log_prod_of_fro (10)	0.22±0.08	0.06±0.05	-0.00±0.14	0.07±0.01	0.07±0.01	0.07±0.02	0.06±0.01	0.12±0.05	0.07
euclid_init_norm (11)	0.08±0.02	-0.10±0.16	-0.12±0.18	0.11±0.00	0.11±0.04	0.11±0.04	0.12±0.01	0.16±0.14	0.10
average_ce (12)	0.16±0.06	0.26±0.21	0.32±0.21	0.13±0.02	0.16±0.02	0.15±0.02	0.15±0.03	-0.08±0.40	0.14
inverse_margin (13)	0.45±0.08	0.47±0.13	0.43±0.14	0.44±0.02	0.44±0.08	0.44±0.04	0.44±0.03	0.34±0.02	0.44
prob_margin (14)	-0.16±0.05	-0.27±0.21	-0.32±0.22	-0.13±0.02	-0.15±0.02	-0.15±0.01	-0.15±0.03	0.08±0.41	-0.14
boundary_thickness (15)	0.33±0.02	0.36±0.22	0.41±0.22	0.30±0.04	0.33±0.02	0.32±0.02	0.32±0.03	0.19±0.38	0.32
kl_divergence (16)	-0.44±0.01	-0.47±0.15	-0.29±0.36	-0.45±0.01	-0.45±0.03	-0.48±0.04	-0.46±0.08	-0.40±0.20	-0.45
local_lip (17)	-0.39±0.03	-0.45±0.15	-0.40±0.27	-0.38±0.06	-0.40±0.03	-0.43±0.01	-0.40±0.03	-0.31±0.31	-0.40
pacbayes_flat (18)	0.16±0.04	0.07±0.30	-0.08±0.37	0.16±0.06	0.14±0.06	0.14±0.08	0.14±0.10	0.26±0.16	0.15
estimated_sharpness (19)	0.19±0.05	0.18±0.11	0.05±0.28	0.19±0.10	0.17±0.04	0.17±0.00	0.16±0.03	0.20±0.11	0.18
estimated_inv_sharpness (20)	0.35±0.06	0.35±0.20	0.20±0.15	0.35±0.05	0.33±0.03	0.33±0.00	0.32±0.01	0.28±0.07	0.33
average_flat (21)	-0.42±0.02	-0.42±0.17	-0.31±0.38	-0.40±0.07	-0.42±0.02	-0.45±0.02	-0.43±0.05	-0.36±0.24	-0.42
x_grad_norm (22)	-0.17±0.02	-0.13±0.11	0.04±0.22	-0.17±0.01	-0.17±0.06	-0.20±0.16	-0.19±0.18	-0.20±0.05	-0.17
w_grad_norm (23)	0.41±0.08	0.39±0.22	0.24±0.08	0.38±0.06	0.38±0.05	0.36±0.02	0.37±0.01	0.29±0.01	0.37
average_ce(PGD) (12)	-0.56±0.06	-0.52±0.18	-0.41±0.43	-0.59±0.05	-0.57±0.01	-0.59±0.09	-0.58±0.08	-0.61±0.07	-0.58
inverse_margin(PGD) (13)	0.61±0.01	0.64±0.04	0.52±0.16	0.61±0.04	0.61±0.01	0.63±0.01	0.60±0.02	0.56±0.15	0.61
prob_margin(PGD) (14)	0.58±0.06	0.54±0.20	0.45±0.44	0.61±0.05	0.59±0.02	0.61±0.09	0.60±0.09	0.64±0.05	0.60
pacbayes_flat(PGD) (18)	0.60±0.02	0.29±0.57	0.08±0.54	0.61±0.04	0.61±0.03	0.61±0.05	0.58±0.07	0.62±0.10	0.61
estimated_sharpness(PGD) (19)	-0.20±0.01	-0.21±0.23	-0.22±0.40	-0.20±0.08	-0.22±0.04	-0.24±0.02	-0.25±0.01	-0.15±0.32	-0.22
estimated_inv_sharpness(PGD) (20)	-0.18±0.04	-0.23±0.24	-0.24±0.41	-0.19±0.06	-0.21±0.05	-0.23±0.04	-0.24±0.00	-0.13±0.34	-0.21
x_grad_norm(PGD) (22)	-0.39±0.00	-0.41±0.13	-0.29±0.34	-0.37±0.04	-0.39±0.03	-0.41±0.01	-0.39±0.05	-0.27±0.16	-0.38
w_grad_norm(PGD) (23)	-0.25±0.03	-0.28±0.24	-0.30±0.40	-0.27±0.07	-0.28±0.02	-0.31±0.01	-0.30±0.03	-0.21±0.28	-0.28

Tables 8 and 9 present the values of ψ_k and π_k for each measure. Fig. 6 illustrates the difference in total τ when the robust generalization gap and the test robust accuracy are used as the target variable for correlation analysis. Certain measures exhibit stronger rank correlations with the test robust accuracy than the robust generalization gap. `w_grad_norm`, `inverse_margin`, and `inverse_margin(PGD)` exhibit the total τ values exceeding 0.4 with respect to the test robust accuracy, whereas they show

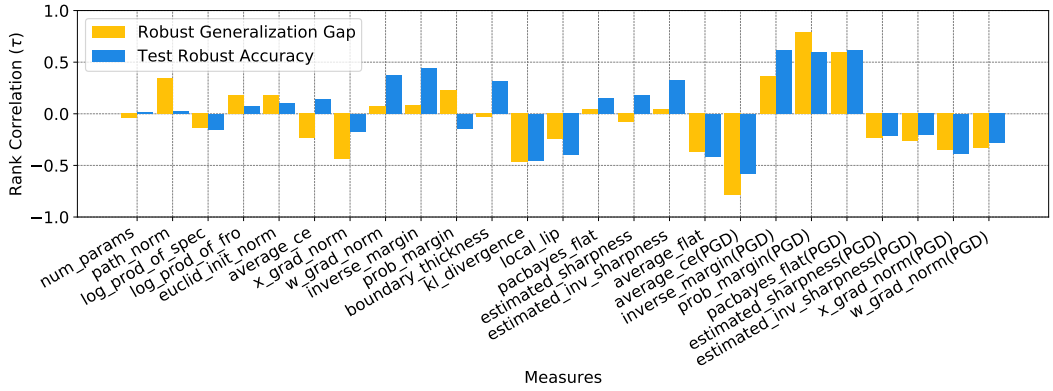


Figure 6: Comparison of the total τ when the robust generalization $g(\boldsymbol{w})$ (yellow) and the test robust accuracy $1 - \mathcal{E}(\boldsymbol{w}; \epsilon, \mathcal{D})$ (blue) are used as the target variables for correlation analysis.

near-zero total τ values for the robust generalization gap. Similarly, `local_lip` and `average_flat` show stronger correlations. Although we observe some different behavior of measures, we find that estimating the test robust accuracy can be more challenging. When we perform a linear regression analysis, predicting the test robust accuracy yields poor R^2 values. To ease comparison, we refer the readers detailed analysis to Table 15 in Appendix A.3.

A.2 Focusing on Adversarially Robust models

In the main paper, we use all models regardless of their robust accuracy, to ensure the generality of our analyses. However, as we discussed in Table 4, adversarially robust and non-robust models may exhibit different behaviors with some measures. For instance, as shown in Fig. 7, non-robust models have extremely low values of `pacbayes_flat(PGD)` less 5. In constrast, robust models show higher values of `pacbayes_flat(PGD)` over 5. Thus, investigating only robust models might potentially reveal hidden behaviors of measures in predicting the robust generalization gap.

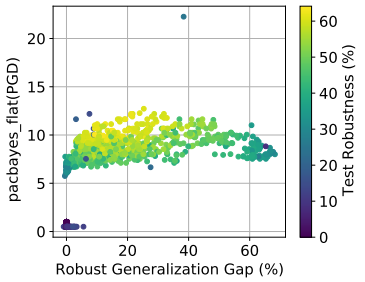


Figure 7: Scatter plot of `pacbayes_flat(PGD)`. Adversarially robust models (bright colors) and non-robust models (darker colors) show distinct range of values for `pacbayes_flat(PGD)`.

In Tables 10 and 11, we summarize ψ_k and π_k with conditioned on `average_ce(PGD)` ≤ 1.5. To ease the comparison between the previous results, we also illustrate the total τ of those in Fig. 8. First, overall π_k and ψ_k of `path_norm` increases, and the total τ increases 0.35 to 0.63. As shown in Fig. 9a, the log of `path_norm` shows almost linear relationships between the robust generalization gap. Similarly, overall π_k and ψ_k of `prob_margin`, and the total τ decreased -0.23 to -0.62. Similar to `prob_margin`, the margin-based measures, i.e., `inverse_margin` and `average_ce`, show more negative correlations. Thus, similar to probability margins on adversarial examples, maximizing the margins on benign examples may harm to the robust generalization with high probability. Lastly, `boundary_thickness` shows strong correlation to the robust generalization gap for `average_ce(PGD)` ≤ 1.5. As shown in Fig. 9c, high `boundary_thickness` shows low robust generalization gap. Thus, we can conclude that `average_ce(PGD)` is also an effective condition for `boundary_thickness` as well as the training method.

Notably, some flatness-based measures have a weaker correlation, `average_flat` (-0.36 to -0.15), `estimated_sharpness(PGD)` (-0.22 to -0.02), and `estimated_inv_sharpness(PGD)` (-0.25 to-0.03). Their overall values of ψ_k and π_k in Tables 10 and 11 are also close to 0, which supports our claim that flatness measures cannot serve as reliable indicators of correlation in the adversarial training framework.

Table 10: Numerical results of ψ_k for each measure, along with its corresponding standard deviation. Conditioned on `average_ce(PGD)` ≤ 1.5.

	Model Architecture	Training Methods	Steps	Optimizer	Batch-size	Aug	Extra-data	Early Stopping	Total τ
num_params (7)	0.23±0.64	-	-	-	-	-	-	-	0.04
path_norm (8)	0.43±0.54	0.45±0.50	0.54±0.84	0.77±0.46	0.32±0.52	0.71±0.49	0.71±0.51	0.88±0.30	0.63
log_prod_of_spec (9)	-0.14±0.56	0.25±0.57	0.21±0.98	0.44±0.73	0.34±0.50	0.16±0.79	0.29±0.78	0.55±0.62	-0.09
log_prod_of_fro (10)	-0.14±0.58	0.56±0.54	0.55±0.84	0.66±0.59	0.68±0.39	0.36±0.73	0.45±0.73	0.84±0.38	0.16
euclid_init_norm (11)	0.23±0.62	-0.02±0.56	-0.04±1.00	0.39±0.72	-0.16±0.55	0.54±0.60	0.53±0.62	-0.40±0.73	0.13
average_ce (12)	-0.50±0.53	-0.49±0.46	-0.28±0.96	-0.76±0.39	-0.74±0.34	-0.68±0.49	-0.74±0.41	-0.71±0.53	-0.62
inverse_margin (13)	-0.12±0.70	-0.28±0.61	0.23±0.97	-0.33±0.77	-0.28±0.65	-0.35±0.75	-0.33±0.77	-0.45±0.67	-0.48
prob_margin (14)	0.50±0.53	0.45±0.46	0.29±0.96	0.76±0.38	0.74±0.34	0.70±0.48	0.75±0.41	0.71±0.53	0.63
boundary_thickness (15)	-0.38±0.57	-0.25±0.50	-0.08±1.00	-0.59±0.55	-0.62±0.41	-0.57±0.58	-0.65±0.50	-0.58±0.62	-0.50
kl_divergence (16)	-0.36±0.60	-0.03±0.55	-0.52±0.85	-0.05±0.80	-0.39±0.57	-0.70±0.46	-0.36±0.73	-0.53±0.68	-0.31
local_lip (17)	-0.16±0.66	0.15±0.52	-0.37±0.93	0.24±0.76	-0.06±0.59	-0.46±0.68	0.11±0.79	0.11±0.79	0.04
pacbayes_flat (18)	0.32±0.65	0.10±0.54	0.30±0.95	0.10±0.83	0.15±0.54	-0.35±0.71	-0.31±0.75	0.47±0.72	0.00
estimated_sharpness (19)	-0.02±0.67	-0.12±0.55	0.26±0.97	0.16±0.82	-0.02±0.61	-0.41±0.68	-0.29±0.78	0.13±0.81	-0.26
estimated_inv_sharpness (20)	-0.06±0.66	-0.14±0.56	0.32±0.95	0.19±0.82	0.02±0.63	-0.41±0.68	-0.28±0.77	0.03±0.81	-0.29
average_flat (21)	-0.12±0.56	0.12±0.60	-0.48±0.88	0.14±0.79	-0.09±0.48	-0.63±0.52	-0.27±0.76	-0.22±0.74	-0.15
x_grad_norm (22)	-0.39±0.63	-0.22±0.59	-0.52±0.85	-0.52±0.68	-0.65±0.44	-0.80±0.36	-0.75±0.47	-0.70±0.59	-0.63
w_grad_norm (23)	-0.13±0.65	-0.14±0.56	0.41±0.91	0.03±0.84	-0.14±0.60	-0.45±0.65	-0.34±0.74	-0.03±0.79	-0.34
average_ce(PGD) (12)	-0.64±0.55	-0.74±0.41	-0.70±0.71	-0.87±0.28	-0.86±0.26	-0.80±0.37	-0.84±0.34	-0.81±0.45	-0.76
inverse_margin(PGD) (13)	0.57±0.48	-0.13±0.61	0.55±0.84	0.13±0.82	0.39±0.63	0.28±0.81	0.26±0.81	0.10±0.80	-0.01
prob_margin(PGD) (14)	0.61±0.58	0.57±0.42	0.66±0.75	0.88±0.27	0.85±0.27	0.79±0.37	0.84±0.34	0.79±0.50	0.76
pacbayes_flat(PGD) (18)	0.45±0.62	0.29±0.55	0.63±0.78	0.33±0.78	0.45±0.48	0.16±0.79	0.00±0.83	0.65±0.59	0.26
estimated_sharpness(PGD) (19)	0.10±0.66	0.03±0.56	0.12±0.99	0.40±0.71	0.17±0.61	-0.21±0.76	-0.18±0.79	0.29±0.76	0.02
estimated_inv_sharpness(PGD) (20)	-0.03±0.66	-0.01±0.57	0.16±0.99	0.39±0.73	0.17±0.59	-0.31±0.74	-0.20±0.79	0.20±0.78	-0.03
x_grad_norm(PGD) (22)	-0.08±0.64	0.09±0.61	-0.48±0.88	0.11±0.79	-0.31±0.55	-0.61±0.53	-0.40±0.72	-0.32±0.75	-0.19
w_grad_norm(PGD) (23)	-0.09±0.67	-0.13±0.57	0.06±1.00	0.25±0.78	0.07±0.60	-0.27±0.76	-0.17±0.80	0.08±0.79	-0.12

Table 11: Numerical results of π_k for each measure, along with its corresponding standard deviation. Conditioned on `average_ce(PGD)` ≤ 1.5. Total τ is same as in Table 10.

	Model Architecture	Training Methods	Steps	Optimizer	Batch-size	Aug	Extra-data	Early Stopping	Total τ
num_params (7)	-	0.03±0.04	-0.01±0.09	0.03±0.01	0.05±0.03	0.05±0.07	0.04±0.02	0.04±0.00	0.04
path_norm (8)	0.64±0.03	0.65±0.03	0.62±0.02	0.64±0.05	0.68±0.00	0.65±0.03	0.63±0.01	0.52±0.01	0.63
log_prod_of_spec (9)	0.19±0.25	-0.09±0.01	-0.06±0.00	-0.08±0.02	-0.07±0.02	-0.09±0.02	-0.09±0.00	-0.08±0.06	-0.09
log_prod_of_fro (10)	0.47±0.04	0.16±0.05	0.17±0.06	0.16±0.01	0.15±0.04	0.19±0.02	0.17±0.01	0.11±0.01	0.16
euclid_init_norm (11)	0.21±0.07	0.15±0.07	0.10±0.11	0.11±0.01	0.16±0.03	0.11±0.03	0.11±0.02	0.17±0.06	0.13
average_ce (12)	-0.63±0.04	-0.70±0.05	-0.65±0.03	-0.61±0.05	-0.62±0.03	-0.66±0.06	-0.62±0.05	-0.56±0.15	-0.62
inverse_margin (13)	-0.48±0.09	-0.44±0.22	-0.47±0.12	-0.45±0.10	-0.46±0.12	-0.47±0.12	-0.47±0.06	-0.42±0.28	-0.48
prob_margin (14)	0.63±0.03	0.71±0.04	0.65±0.03	0.61±0.04	0.62±0.03	0.66±0.06	0.63±0.05	0.57±0.16	0.63
boundary_thickness (15)	-0.51±0.03	-0.61±0.03	-0.57±0.01	-0.49±0.07	-0.50±0.04	-0.53±0.07	-0.49±0.09	-0.45±0.21	-0.50
kl_divergence (16)	-0.31±0.07	-0.40±0.14	-0.13±0.18	-0.33±0.03	-0.31±0.11	-0.25±0.18	-0.30±0.08	-0.27±0.13	-0.31
local_lip (17)	0.03±0.12	0.01±0.23	0.17±0.08	0.01±0.04	0.03±0.08	0.13±0.03	0.04±0.03	0.02±0.03	0.04
pacbayes_flat (18)	-0.03±0.19	-0.01±0.06	-0.05±0.07	0.02±0.01	-0.00±0.08	0.11±0.07	0.07±0.06	-0.16±0.18	0.00
estimated_sharpness (19)	-0.26±0.02	-0.24±0.13	-0.37±0.01	-0.25±0.05	-0.26±0.10	-0.15±0.16	-0.22±0.12	-0.32±0.27	-0.26
estimated_inv_sharpness (20)	-0.28±0.02	-0.28±0.14	-0.42±0.01	-0.29±0.04	-0.29±0.06	-0.18±0.15	-0.26±0.11	-0.32±0.27	-0.29
average_flat (21)	-0.17±0.05	-0.22±0.13	0.02±0.18	-0.20±0.09	-0.16±0.05	-0.04±0.19	-0.14±0.04	-0.17±0.01	-0.15
x_grad_norm (22)	-0.64±0.04	-0.69±0.13	-0.55±0.08	-0.65±0.04	-0.62±0.08	-0.58±0.12	-0.62±0.04	-0.57±0.14	-0.63
w_grad_norm (23)	-0.34±0.03	-0.34±0.11	-0.48±0.01	-0.33±0.05	-0.33±0.08	-0.26±0.12	-0.31±0.09	-0.35±0.27	-0.34
average_ce(PGD) (12)	-0.77±0.03	-0.77±0.01	-0.73±0.03	-0.76±0.02	-0.77±0.01	-0.80±0.02	-0.77±0.00	-0.72±0.07	-0.76
inverse_margin(PGD) (13)	-0.02±0.03	0.13±0.11	-0.14±0.07	0.03±0.11	-0.02±0.05	-0.01±0.15	0.02±0.22	-0.02±0.31	-0.01
prob_margin(PGD) (14)	0.77±0.03	0.77±0.01	0.73±0.03	0.76±0.01	0.77±0.02	0.80±0.01	0.77±0.00	0.72±0.08	0.76
pacbayes_flat(PGD) (18)	0.26±0.08	0.25±0.07	0.25±0.15	0.29±0.03	0.26±0.07	0.34±0.03	0.36±0.04	0.08±0.20	0.26
estimated_sharpness(PGD) (19)	0.01±0.04	0.03±0.09	0.06±0.19	-0.01±0.01	0.01±0.13	0.14±0.10	0.07±0.11	-0.09±0.17	0.02
estimated_inv_sharpness(PGD) (20)	-0.02±0.05	-0.02±0.08	0.00±0.20	-0.06±0.04	-0.04±0.08	0.10±0.10	0.02±0.10	-0.14±0.20	-0.03
x_grad_norm(PGD) (22)	-0.20±0.04	-0.24±0.06	0.04±0.28	-0.25±0.10	-0.19±0.13	-0.12±0.18	-0.18±0.02	-0.17±0.08	-0.19
w_grad_norm(PGD) (23)	-0.12±0.04	-0.08±0.08	-0.06±0.21	-0.13±0.01	-0.11±0.11	-0.01±0.10	-0.08±0.08	-0.19±0.18	-0.12

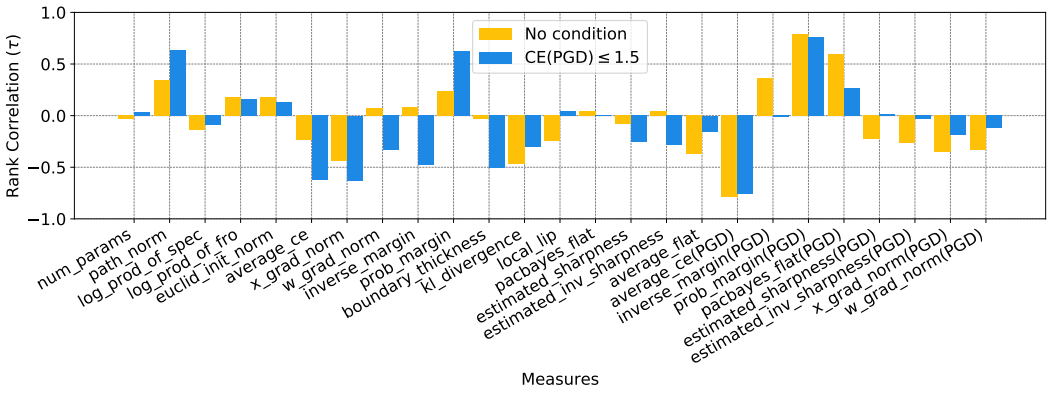
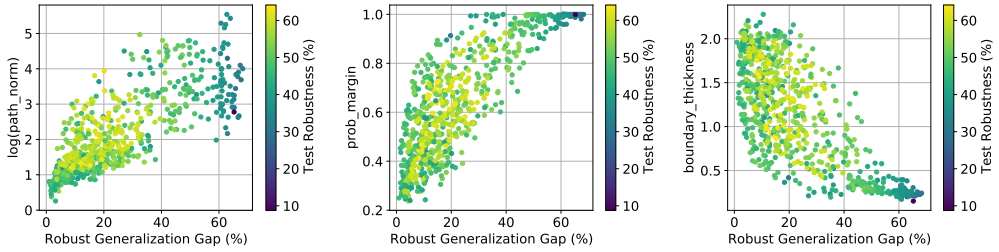


Figure 8: Comparison of total τ between no condition and `average_ce(PGD)` ≤ 1.5. Following measures show strong correlation on the condition: `path_norm`, `average_ce`, `x_grad_norm`, `inverse_margin`, `prob_margin`, and `boundary_thickness`.



(a) log(`path_norm`) (b) `prob_margin` (c) `boundary_thickness`

Figure 9: Scatter plot of the measures showing the most increased the total τ when conditioned on `average_ce(PGD)` ≤ 1.5.

Table 12: Regression analysis in Fig. 10 for the target variable, the robust generalization gap $g(\boldsymbol{w})$. We summarize their R^2 for each measure.

Measures	R^2	Measures	R^2
num_params (7)	0.87	estimated_sharpness (19)	0.88
path_norm (8)	0.87	estimated_inv_sharpness (20)	0.89
log_prod_of_spec (9)	0.88	average_flat (21)	0.87
log_prod_of_fro (10)	0.88	x_grad_norm (22)	**0.91**
euclid_init_norm (11)	0.87	w_grad_norm (23)	0.88
average_ce (12)	0.86	inverse_margin(PGD) (13)	0.86
inverse_margin (13)	0.86	prob_margin(PGD) (14)	**0.91**
prob_margin (14)	0.87	pacbayes_flat(PGD) (18)	0.89
boundary_thickness (15)	0.86	estimated_sharpness(PGD) (19)	0.87
kl_divergence (16)	0.87	estimated_inv_sharpness(PGD) (20)	0.86
local_lip (17)	0.87	x_grad_norm(PGD) (22)	0.88
pacbayes_flat (18)	0.90	w_grad_norm(PGD) (23)	0.87

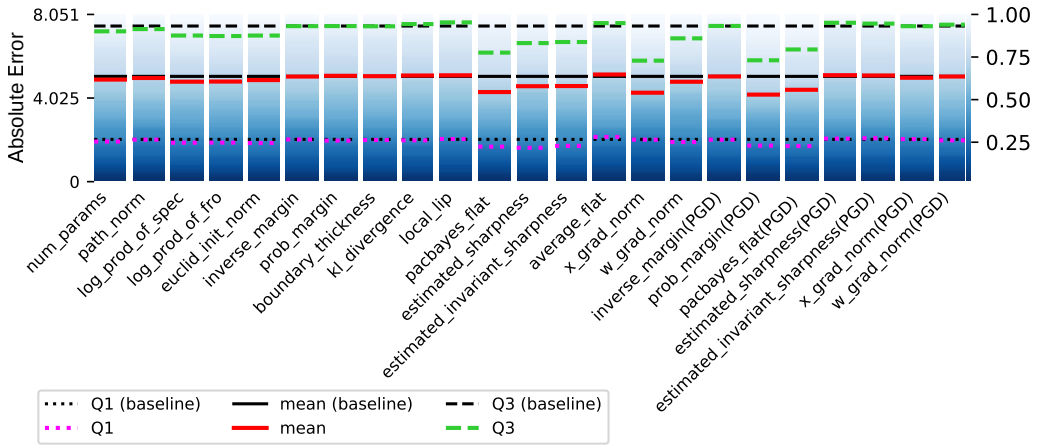


Figure 10: Cumulative distribution of absolute errors for the robust generalization gap estimation is shown. A simple linear regression model is constructed to estimate the robust generalization gap, using `average_ce(PGD)` as the baseline measure, and each measure considered as an additional independent variable. The mean (solid line), Q1 (dotted line), and Q3 (dashed line) are plotted. A lower absolute error indicates that the measure is more effective in estimating the robust generalization gap.

A.3 Robust Measures with Regression Analysis

In the seminar work of [17], the concept of *an affine oracle* was proposed, which utilizes linear regression to assess the performance of measures. Motivated by this experiment, we conduct a simple linear regression analysis. Specifically, for each measure, we calculate the optimal coefficients and bias to predict the robust generalization gap $g(\boldsymbol{w})$. Considering that `average_ce(PGD)` consistently exhibits the highest correlation across all settings, we use it as a baseline for the regression, i.e., $\beta_1 \times$`average_ce(PGD)`$+\beta_0$. We then consider each measure as an independent variable, resulting in a new form of measure $\beta_1 \times$`average_ce(PGD)`$+\beta_2 \times$`measure`$+\beta_0$. To ensure high predictability, we perform this regression analysis with robust models with `average_ce(PGD)` ≤ 1.5.

In Table 12, we calculate the coefficient of determination (R^2) for the regression analysis to evaluate the extent to which each measure explained the generalization gap. Consistent with the findings in the main paper, `x_grad_norm` and `prob_margin(PGD)` exhibit the highest R^2 values.

Additionally, to examine the distributional information of each measure, we plot the cumulative distribution of absolute errors for each model with the mean (solid line) and the interval denoted by the first quartile (Q1, dotted line) and the third quartile (Q3, dashed line). The results are illustrated in Figure 10. We observe that only a few measures, such as `x_grad_norm` and `prob_margin(PGD)`, provided meaningful information about the generalization gap, while the effects of other measures are

Table 13: Total τ of generated measures from the regression analysis. Both new measures show improved performance in predicting the robust generalization gap.

Generated Measures	Total τ
$-514.11\times$`x_grad_norm`$-23.87\times$`average_ce(PGD)`$+57.32$	0.81 (+0.18)
$81.17\times$`prob_margin(PGD)`$+13.52\times$`average_ce(PGD)`-20.60	0.78 (+0.02)

Table 14: Total τ of generated predictors from the regression analysis with forward selection.

#Measures	Selected Measures	5-fold τ (Avg.$\pm$Std.)
1	`average_ce(PGD)`	0.7229±0.1301
2	`x_grad_norm`, `average_ce(PGD)`	0.7683±0.1040
3	`x_grad_norm`, `average_ce(PGD)`, `pacbayes_mag_flat(PGD)`	0.8145±0.0728
4	`x_grad_norm`, `average_ce(PGD)`, `x_grad_norm(PGD)`, `pacbayes_mag_flat(PGD)`	0.8219±0.0730
All	-	0.8165±0.0868

Table 15: Regression analysis for the target variable, the test robust accuracy $1-\mathcal{E}(\boldsymbol{w};\epsilon,\mathcal{D})$. We summarize their R^2 for each measure.

Measures	R^2	Measures	R^2
`num_params` (7)	0.10	`estimated_sharpness` (19)	0.33
`path_norm` (8)	0.07	`estimated_inv_sharpness` (20)	0.41
`log_prod_of_spec` (9)	0.14	`average_flat` (21)	0.12
`log_prod_of_fro` (10)	0.14	`x_grad_norm` (22)	0.09
`euclid_init_norm` (11)	0.12	`w_grad_norm` (23)	0.38
`average_ce` (12)	0.12	`inverse_margin(PGD)` (13)	0.04
`inverse_margin` (13)	0.05	`prob_margin(PGD)` (14)	0.24
`prob_margin` (14)	0.20	`pacbayes_flat(PGD)` (18)	0.48
`boundary_thickness` (15)	0.20	`estimated_sharpness(PGD)` (19)	0.04
`kl_divergence` (16)	0.13	`estimated_inv_sharpness(PGD)` (20)	0.05
`local_lip` (17)	0.16	`x_grad_norm(PGD)` (22)	0.16
`pacbayes_flat` (18)	0.40	`w_grad_norm(PGD)` (23)	0.05

negligible. Specifically, `x_grad_norm` and `prob_margin(PGD)` achieve the lowest mean absolute error across all trained models.

In Table 13, we evaluate the effectiveness of predictors generated from the regression analysis. The predictor using `x_grad_norm` achieves an exceptionally strong correlation of 0.81, an increase of 0.18 compared to the previous value in Table 10. The predictor using `prob_margin(PGD)` also exhibits improved performance. These results suggest the potential to effectively predict the robust generalization gap by combining existing measures.

To push further, we conduct a 5-fold evaluation strategy with a linear regression model to predict the robust generalization gap. Specifically, we use the forward selection to identify the most effective set of measures. In Table 14, we present the results of our 5-fold evaluation, reporting the average τ along with its standard deviation. `average_ce(PGD)` is selected as a prominent predictor, followed by the selection of `x_grad_norm`. Furthermore, our exploration identifies `pacbayes_mag_flat(PGD)` and `x_grad_norm(PGD)` as additional effective measures, resulting in higher average τ compared to using the entire feature set.

In Table 15, we conduct a linear regression analysis to predict the test robust accuracy, rather than the robust generalization gap. Compared to Table 12, overall values of R^2 for each measure are lower. This implies that directly predicting the test robust accuracy might be more difficult than the robust generalization gap.

A.4 Robust Generalization Gap with AutoAttack [11]

In the main paper, we estimate the robust generalization gap using PGD10. While we acknowledge the potential benefits of stronger attacks, such as AutoAttack [11], we mainly use PGD10 due to the following reasons. Firstly, the computational cost of AutoAttack is substantial. Our experimental design involves training models across diverse adversarial settings and requires adversarial examples for both training and test datasets to estimate the robust generalization gap. AutoAttack takes 10 min per batch for WRN-34-10 on our resources. Since we have 1300 models, we need at least 1 year to obtain all adversarial examples even with 6 GPUs. Secondly, the prevalent usage of PGD among various methods. AT, TRADES, and MART use PGD as a baseline during training and further adopt early-stopping by using PGD on training or validation sets. Lastly, some measures, namely `boundary_thickness` and `local_lip`, rely on PGD adversarial examples for their calculation. As these robustness measures are often computed using PGD, the choice to use PGD for evaluation contributes to consistency across our experiments.

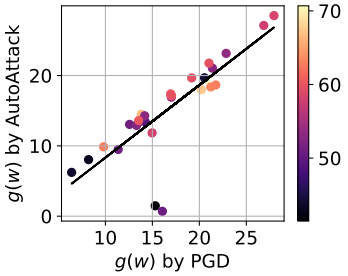
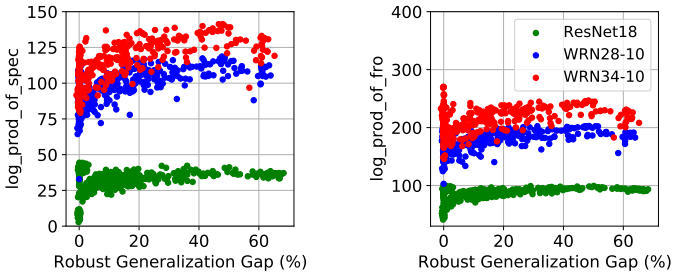


Figure 11: Scatter plot of the robust generalization gap calculated by PGD and AutoAttack. The color of each dot implies the test robust accuracy on CIFAR-10.

However, we here highlight the potential benefits of utilizing AutoAttack in future work. In Fig. 11, we calculate the gap $g(\boldsymbol{w})$ using AutoAttack for 30 models on CIFAR-10 in RobustBench [12]. While the gaps calculated using PGD10 and AutoAttack exhibit an almost linear relationship, there are a few exceptions (2 out of 30): 'Ding2020MMA' [14] and 'Sitawarin2020Improving' [55]. We leave this question open for further exploration.

A.5 Importance of Model Architecture in Norm-based Measures



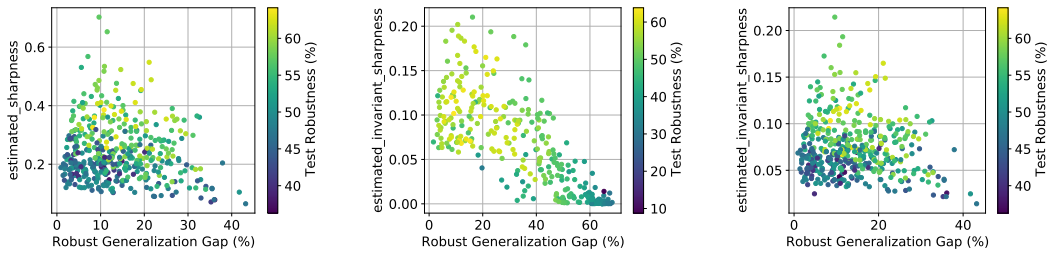
(a) `log_prod_of_spec` (b) `log_prod_of_fro`

Figure 12: Distributions of norm-based measures for model architectures. For each model architecture, the range of measure extremely varies.

In Fig. 12, we plot `log_prod_of_spec` and `log_prod_of_fro` for each model architecture. We can observe that the range of each measure extremely varies with respect to the used model architecture. When the model architecture is fixed, they show some correlation with the robust generalization gap as described in the main paper.

A.6 Early Stopping and Estimated Sharpness

In the main paper, we discussed that `estimated_sharpness` exhibits a significant negative correlation when early stopping is not used. Fig. 13 shows the importance of early-stopping for estimated sharpness measures. Compared to Figures 4 and 13b, when early stopping is employed, the correlation approaches zero as shown in Figures 13a and 13c. This results supports the observation of prior study [56] that the importance of early stopping in the analysis of flatness.



(a) `estimated_sharpness` w/ early-stopping

(b) `estimated_inv_sharpness` w/o early-stopping

(c) `estimated_inv_sharpness` w/ early-stopping

Figure 13: Scatter plot of the estimated sharpness measures for the use of early-stopping. The same condition, `average_ce(PGD)` < 2, used as Fig. 4.

B Measures

In this section, we introduce the concept of each category of measures in the main paper, then explain the details of each measure and their mathematical definitions. Here, we denote $\mathbf{W}_i$ as the weight tensor of i-th layer, following [17]. Given the number of layers d, the whole trainable parameters are denoted as $\boldsymbol{w} = \text{vec}(\mathbf{W}_1, \mathbf{W}_2, \cdots, \mathbf{W}_d)$.

Weight-norm. Based on some theoretical frameworks such as PAC-Bayes [43, 47, 38], weight norm-based measures are expected to be correlated with generalization performance. Among them, the product of Frobenius norm [46] (`log_prod_of_fro`), the product of spectral norm [7] (`log_prod_of_spec`), and path norm (`path_norm`) have been considered as important measures [27, 17]. Furthermore, the distance to the converged weight from the initial weight (`euclid_init_norm`) is also used to estimate the generalization gap [27]. Liu et al. [40] also argues that this distance can be used to judge the difficulty of optimization in adversarial training.

▷ `num_params`.

$$\sum_{i=1}^{d} k_i c_{i-1}(c_i + 1), \tag{7}$$

where c_i is the number of channels and k_i is the kernel size at layer i. In the experiments, we calculated `num_params` by adding the number of parameters of all convolutional and linear layers. `num_params` is a fixed value when a model architecture is given.

▷ `path_norm`.

$$\sum_{i} f_{\boldsymbol{w}^2}(\mathbf{1})[i], \tag{8}$$

where $\boldsymbol{w}^2$ is the element-wise square operation and $f(\cdot)[i]$ is the i-th logit output of the network. By setting all input variables as 1, this measure captures geometric properties of optimization under scale-invariant characteristics.

▷ `log_prod_of_spec`.

$$\log\left(\prod_{i=1}^{d} \|\mathbf{W}_i\|_2^2\right), \tag{9}$$

where $\|\cdot\|_2$ is a matrix L_2-norm, i.e., the largest singular value of each layer.

▷ `log_prod_of_spec`.

$$\log\left(\prod_{i=1}^{d} \|\mathbf{W}_i\|_F^2\right), \tag{10}$$

where $\|\cdot\|_F$ is a Frobenius norm, i.e., the square root of the sum of the squares of the weight matrix.

▷ `euclid_init_norm`.

$$\frac{1}{d}\sum_{i=1}^{d} \|\boldsymbol{w}_i - \boldsymbol{w}_i^0\|_2, \tag{11}$$

where $\boldsymbol{w}_i = \text{vec}(\mathbf{W}_i)$ and the initial weight of i-th layer $\boldsymbol{w}_i^0$.

Margin. Margins are also actively researched measures to estimate the generalization gap [26]. For instance, the 10-th percentile of the margin values in the output space on the training set (`inverse_margin`) is often used to measure the generalization bound for neural networks [50, 27]. In terms of adversarial robustness, most of attack methods and defense methods utilize the probability margins (`prob_margin`). Yang et al. [67] further proposed a new measure, called boundary thickness (`boundary_thickness`), that is a generalized version of margin and argued that it is highly correlated to the robust generalization gap.

▷ `average_ce`.

$$\mathbb{E}_{\boldsymbol{x},y}\left[\mathcal{L}_{ce}(f(\boldsymbol{x},\boldsymbol{w}),y)\right], \tag{12}$$

where $\mathcal{L}_{ce}$ is the cross-entropy loss.

▷ `inverse_margin`.

$$1/\gamma^2, \tag{13}$$

where γ is 10th-percentile of $\{\sigma(f(\boldsymbol{x},\boldsymbol{w}))_y - \max_{i\neq y}\sigma(f(\boldsymbol{x},\boldsymbol{w}))_i\}$ for all $\boldsymbol{x}, y$, with the sigmoid function $\sigma(\cdot)$.

▷ `prob_margin`.

$$\mathbb{E}_{\boldsymbol{x},y}\left[\sigma(f(\boldsymbol{x},\boldsymbol{w}))_y - \max_{i\neq y}\sigma(f(\boldsymbol{x},\boldsymbol{w}))_i\right]. \tag{14}$$

▷ `boundary_thickness`.

$$\mathbb{E}_{\boldsymbol{x}}\left[\|\boldsymbol{x}-\boldsymbol{x}^*\|_2\int_0^1 \mathbb{1}\{a < g(\boldsymbol{x},\boldsymbol{x}^*,\lambda) < b\}d\lambda \,\middle|\, \arg\max_i \sigma(f(\boldsymbol{x},\boldsymbol{w}))_i \neq \arg\max_i \sigma(f(\boldsymbol{x}^*,\boldsymbol{w}))_i\right], \tag{15}$$

where $\mathbb{1}\{\cdot\}$ is an indicator function, $g(\boldsymbol{x},\boldsymbol{x}^*,\lambda) = \sigma(f(\lambda\boldsymbol{x}+(1-\lambda)\boldsymbol{x}^*,\boldsymbol{w}))_{\hat{y}} - \sigma(f(\lambda\boldsymbol{x}+(1-\lambda)\boldsymbol{x}^*,\boldsymbol{w}))_{\hat{y}^*}$, $\hat{y} = \arg\max_i \sigma(f(\boldsymbol{x},\boldsymbol{w}))_i$, $\hat{y}^* = \arg\max_i \sigma(f(\boldsymbol{x}^*,\boldsymbol{w}))_i$ and a, b are the hyper-parameters that controls the sensitivity of the boundary thickness. Following [67], we find $\boldsymbol{x}^*$ by using PGD10 with L_2-norm, $\epsilon = 1$, $\alpha = 0.2$, then $a = 0$, $b = 0.75$, and batch size 128. A higher value of `boundary_thickness` implies a larger margin in the output space.

Smoothness. Based on prior works [6, 10, 28], a line of work has focused on the smoothness for achieving adversarial robustness in adversarial training. Most simply, the KL divergence between benign and adversarial logits (`kl_divergence`) of TRADES [71] can be regarded as a smoothness regularization due to its logit pairing. Yang et al. [66] investigated the theoretical benefit of local Lipschitzness (`local_lip`) and demonstrated that its value estimated on the test dataset correlates with the robust generalization gap.

▷ `kl_divergence`.

$$\mathbb{E}_{\boldsymbol{x}}\left[\max_{\|\boldsymbol{x}-\boldsymbol{x}*\|\leq\epsilon} \mathtt{KL}(f(\boldsymbol{x},\boldsymbol{w}), f(\boldsymbol{x}^*,\boldsymbol{w}))\right], \tag{16}$$

where `KL` is KL-divergence and the maximization is conducted by PGD10 with the step-size $2/255$. A lower value of `kl_divergence` implies that a model outputs similar outputs for both benign example and adversarial example.

▷ `local_lip`.

$$\mathbb{E}_{\boldsymbol{x}}\left[\max_{\|\boldsymbol{x}-\boldsymbol{x}*\|\leq\epsilon} \frac{\|f(\boldsymbol{x},\boldsymbol{w}) - f(\boldsymbol{x}^*,\boldsymbol{w})\|_1}{\|\boldsymbol{x}-\boldsymbol{x}^*\|_\infty}\right], \tag{17}$$

where the maximization is conducted by PGD10 with the step-size $2/255$. This is the empirical version of the local Lipschitzness, resulting a lower value of `local_lip` implies a smoother model.

Flatness. Flatness is a recently focused measure in the generalization domain [43, 30]. Recent works argue that flatter minima yield better generalization performance than sharper minima. The common strategy to achieve flatness is to minimize the estimated sharpness [18, 73], which is the difference between the current loss and the maximum loss for a given neighborhood (`estimated_sharpness`). Kwon et al. [36] investigated the scale-invariant sharpness (`estimated_inv_sharpness`). Note that other diverse concept of estimated sharpness is actively researched in recent works [33, 5, 34]. Adversarial weight perturbation (AWP) [64] also has dramatically improved adversarial robustness by minimizing the loss of perturbed weight. Recently, Stutz et al. [56] has demonstrated that their

proposed measure, average flatness (`average_flat`), is effective for estimating robust generalization gap.

▷ `pacbayes_flat`. Based on PAC-Bayesian framework [43], Jiang et al. [27] proposed a simplified version of PAC-Bayesian bounds as follows:

$$1/\sigma' \tag{18}$$

where σ' is the largest value such that $\mathbb{E}_{\boldsymbol{u}}[\mathcal{E}(\boldsymbol{w}+\boldsymbol{u};\{\boldsymbol{x},y\}) - \mathcal{E}(\boldsymbol{w};\{\boldsymbol{x},y\})] < 0.1$. Here, $u_j \sim \mathcal{N}(0,\sigma')$ and $\mathcal{E}(\boldsymbol{w};\{\boldsymbol{x},y\})$ denotes the error on the given set $\{\boldsymbol{x},y\}$. Thus, a higher value of `pacbayes_flat` implies flatter optimum in terms of error landscape.

▷ `estimated_sharpness`.

$$\mathbb{E}_{\boldsymbol{x},y}\left[\max_{\|\boldsymbol{v}\|_2\leq\rho} \mathcal{L}_{ce}(f(\boldsymbol{x},\boldsymbol{w}+\boldsymbol{v}),y) - \mathcal{L}_{ce}(f(\boldsymbol{x},\boldsymbol{w}),y)\right]. \tag{19}$$

Following [73], we calculate the estimated sharpness with a single-step ascent with $\rho = 0.1$. A higher value of `estimated_sharpness` implies a sharper optimum in terms of loss landscape.

▷ `estimated_inv_sharpness`.

$$\mathbb{E}_{\boldsymbol{x},y}\left[\max_{\|T_{\boldsymbol{w}}^{-1}\boldsymbol{v}\|_2\leq\rho} \mathcal{L}_{ce}(f(\boldsymbol{x},\boldsymbol{w}+\boldsymbol{v}),y) - \mathcal{L}_{ce}(f(\boldsymbol{x},\boldsymbol{w}),y)\right], \tag{20}$$

where $T_{\boldsymbol{w}}$ is $\|\boldsymbol{w}\|$, i.e., element-wise adaptive sharpness, and $\mathcal{L}_{ce}$ is the cross-entropy loss. Similar to the estimated sharpness, we calculate the estimated invariant sharpness with a single-step ascent with $\rho = 0.1$ [36]. A higher value of `estimated_inv_sharpness` implies a sharper optimum in terms of loss landscape.

▷ `average_flat`.

$$\mathbb{E}_{\boldsymbol{x},y}\left[\mathbb{E}_{\boldsymbol{v}\in\mathcal{B}_\rho(\boldsymbol{w})}\left[\max_{\|\boldsymbol{x}-\boldsymbol{x}^*\|\leq\epsilon} \mathcal{L}_{ce}(f(\boldsymbol{x}^*,\boldsymbol{w}+\boldsymbol{v}),y)\right] - \max_{\|\boldsymbol{x}-\boldsymbol{x}^*\|\leq\epsilon} \mathcal{L}_{ce}(f(\boldsymbol{x}^*,\boldsymbol{w}),y)\right], \tag{21}$$

where $\mathcal{B}_\rho(\boldsymbol{w}) = \{\boldsymbol{w}+\boldsymbol{v} \big| \|\boldsymbol{v}_i\|_2 \leq \rho\|\boldsymbol{w}_i\|_2 \forall \text{ layers } i\}$. We use PGD10 with $\epsilon = 8/255$ for both inner maximizations. Following [56], we take 10 random weight perturbations with $\rho = 0.5$. A higher value of `average_flat` implies a sharper optimum in terms of adversarial loss landscape.

Gradient-norm. Gradient-norm with respect to input or weight is also a consistently researched area in terms of generalization. Recently, Zhao et al. [72] also demonstrated that regularizing the gradient norm of weights (`w_grad_norm`) can achieve sufficient improvement on several tasks. Additionally, there are a few works that emphasize the importance of regularizing the gradient norm of weights [53, 44]. The gradient norm of inputs (`x_grad_norm`) also can have underlying correlation between the robust generalization performance. Prior works utilized the input gradient for analyzing adversarially trained models [4] and generating adversarial examples [35, 16].

▷ `x_grad_norm`.

$$\mathbb{E}_{\boldsymbol{x},y}\left[\|\nabla_{\boldsymbol{x}}\mathcal{L}_{ce}(f(\boldsymbol{x},\boldsymbol{w}),y)\|_2\right]. \tag{22}$$

▷ `w_grad_norm`.

$$\mathbb{E}_{\boldsymbol{x},y}\left[\|\nabla_{\boldsymbol{w}}\mathcal{L}_{ce}(f(\boldsymbol{x},\boldsymbol{w}),y)\|_2\right]. \tag{23}$$

Comment on batch normalization fusion Here, we provide comments on some further discussed things when estimating the above measures. In previous studies [27, 17], it has been observed that considering batch normalization (batch-norm) layers can have an impact on common generalization measures, such as sharpness [15]. To address this issue, the batch-norm layers and other moving statistics were fused with the preceding convolution layers before calculating the values of generalization measures. Thus, when estimating {`num_params`, `path_norm`, `log_prod_of_spec`, `log_prod_of_fro`, `pacbayes_flat`}, we apply batch-norm fusion to all ResNet blocks. However, for certain blocks, such as pre-activation ResNets, where the batch-norm layer is placed at the beginning, the fusion cannot be directly applied. To ensure consistency, we add an identity convolutional layer in front of all batch-norm layers that do not have the preceding convolution layer. While there are various batch-norm fusion (or batch-norm folding) techniques, including those related to

generalization [27, 17, 5], quantization [69, 37, 49], and memory optimization domains [8, 19, 2], there is no precise solution to address this problem in the context of various model structures (ResNets, PreActResNets, ViT, etc.) and activation functions (SiLU, LeakyReLU, etc.), which we leave as a topic for future work.

C Training Details

In Section 3, 1,344 models were trained using the CIFAR-10 dataset with $\epsilon = 8/255$. We here provide the detailed training settings. We followed the common settings used in [42, 71, 48, 21].

Given the higher Rademacher complexity [68] and larger sample complexity [54] of adversarial training, data augmentation [21] and the utilization of extra data [9] can significantly improve the adversarial robustness. Therefore, we also considered the impact of augmentation technique, including *RandomCrop* with padding 4 and *RandomHorizontalFlip*, as well as the use of additional data collected by Carmon et al. [9].

Regarding model architectures, we employed three different models: ResNet18 [23], WRN28-10 [70], and WRN34-10 [70]. These models have been widely adopted and serve as benchmarks for evaluating the stability and performance of adversarial training methods. It is worth noting that the majority of models trained on CIFAR-10 in RobustBench [12] consist of WRN28-10 (15) and WRN34-10 (14) among the 63 available models.

For training methods, we considered four different approaches: Standard, AT [42], TRADES [71], and MART [61]. Notably, AT, TRADES, and MART have been shown to outperform other variations by incorporating various training tricks [48] and integrating recent techniques [9, 64]. For all methods, we generated adversarial examples using projected gradient descent (PGD) [42]. During training, a single-step approximation of the inner maximization in Eq. (1) can lead to faster adversarial training, but may suffer from catastrophic overfitting [63, 31]. On the other hand, a large number of steps leads to stable robustness, but requires heavy computational costs. Therefore, we considered both 1 and 10 steps for each adversarial training method.

In terms of optimization, we used SGD with momentum 0.9 and weight decay of 5×10^{-4}, and a step-wise learning rate decay was performed at epochs 100 and 150 with a decay rate of 0.1. In all the experiments, we trained the models for 200 epochs. As highlighted by Pang et al. [48], the batch size used during adversarial training has been found to affect its performance. Thus, we varied the batch size among {32, 64, 128, 256}. Additionally, we also considered adversarial weight perturbation (AWP) [64], which can improve the robust generalization performance of models. AWP belongs to the class of sharpness-aware minimization methods [18, 36]. This can be formalized as follows:

$$\min_{\boldsymbol{w}} \max_{\|\boldsymbol{x}^{adv}-\boldsymbol{x}\|\leq\epsilon, \boldsymbol{v}\in\mathcal{B}_\rho(\boldsymbol{w})} \mathcal{L}(f(\boldsymbol{x}^{adv}, \boldsymbol{w}+\boldsymbol{v}), y), \tag{24}$$

where $\mathcal{B}_\rho(\boldsymbol{w}) = \{\boldsymbol{w}+\boldsymbol{v} \big| \|\boldsymbol{v}_i\|_2 \leq \rho\|\boldsymbol{w}_i\|_2 \; \forall i\text{-th layer}\}$. As described by Wu et al. [64], $\boldsymbol{x}^{adv}$ is calculated based on the non-perturbed model $f(\boldsymbol{w})$, and a single step of maximization with respect to $\boldsymbol{v}$ is sufficient to improve robustness. We used the best-performing value of $\rho = 5 \times 10^{-3}$ from [64].