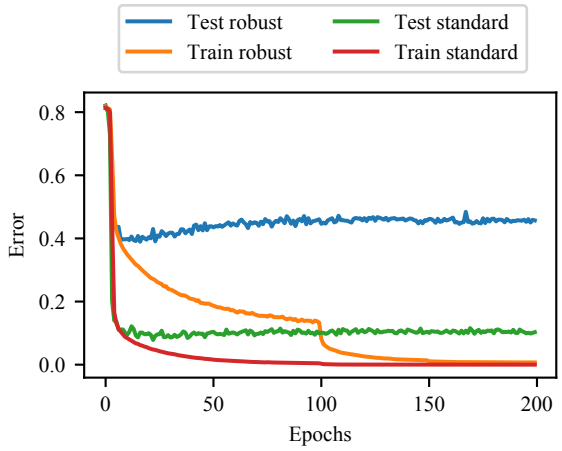

Figure 9. Learning curves for training an SVHN classifier which is adversarially robust to ℓ_∞ perturbations of radius 8/255. Note that robust overfitting occurs before the learning rate has decayed, likely due to the lower initial learning rate.

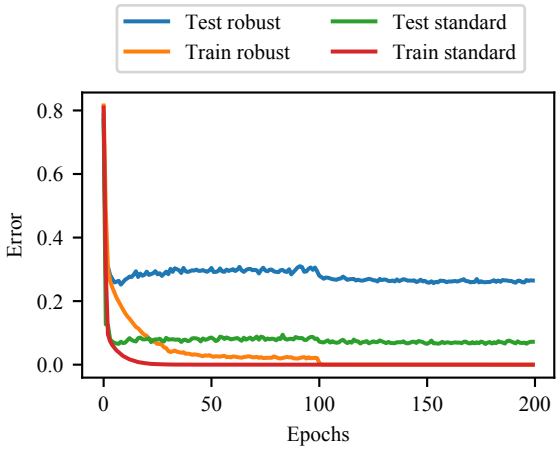

Figure 10. Learning curves for training an SVHN classifier which is adversarially robust to ℓ_2 perturbations of radius 128/255. Robust overfitting occurs early here as well, with robust test error increasing after the 9th epoch.

A. Full set of results for Table 1

In this section, we extend Table 1 to additionally include standard error and results from different adversarial training schemes (FGSM and TRADES), as shown in Table 3. The final error is an average over the final 5 epochs of when the model has converged, along with the standard deviation. The best error is the lowest test error of all model checkpoints during training. For convenience we also show the difference in the final model's error and the best model's error, which indicates the amount of degradation incurred by robust overfitting.

The remainder of this section contains the experimental details for reproducing these experiments, as well as the learning curves for each experiment as visual evidence of robust overfitting. We default to using pre-activation ResNet18s for our experiments, with the exception of Wide ResNets with width factor 10 for ℓ_∞ adversaries on CIFAR-10 (for a proper comparison to what is reported for TRADES), and ResNet50s for ImageNet. For CIFAR-10 and CIFAR-100, we train with the SGD optimizer using a batch size of 128, a step-wise learning rate decay set initially at 0.1 and divided by 10 at epochs 100 and 150, and weight decay $5 \cdot 10^{-4}$. For SVHN, we use the same parameters except with a starting learning rate of 0.01 instead. For ImageNet, we use the same learning configuration used to train the pretrained models and simply run them for longer epochs and lower learning rates using the publicly released repository available at `https://github.com/madrylab/robustness`.

ℓ_∞ adversary We consider the ℓ_∞ threat model with radius 8/255, with the PGD adversary taking 10 steps of size 2/255 on all datasets except for ImageNet. For ImageNet, we fine-tune the pretrained model from `https://github.com/madrylab/robustness` (Engstrom et al., 2019) and continue training with the exact same parameters with a learning rate of 0.001, which uses an adversary with 5 steps of size 0.9/255 within a ball of radius 4/255.

ℓ_2 adversary We consider the ℓ_2 threat model with radius 128/255, with the PGD adversary taking 10 steps of size 15/255 on all datasets except for ImageNet. For Imagenet, we fine-tune the pretrained model from `https://github.com/madrylab/robustness` (Engstrom et al., 2019) and continue training with the exact same parameters with a learning rate of 0.001, which uses an adversary with 7 steps of size 0.5 within a ball of radius 3.

A.1. SVHN experiments

Figures 9 and 10 contain the convergence plots for the PGD-based adversarial training experiments on SVHN for ℓ_∞ and ℓ_2 perturbations respectively. We find that robust overfitting occurs even earlier on this dataset, before the initial learning rate decay, indicating that the learning rate threshold at which robust overfitting begins to occur has already been passed. The best checkpoint for ℓ_∞ achieves 39.0% robust error, which is a 6.6% improvement over the 45.6% robust error achieved at the end of training.

A.2. CIFAR-100 experiments

Figures 11 and 12 contain the convergence plots for the PGD-based adversarial training experiments on CIFAR-100 for ℓ_∞ and ℓ_2 perturbations respectively. We find that ro-

Table 3. Performance of adversarially robust training over a variety of datasets, adversarial training algorithms, and perturbation threat models, where the best error refers to the lowest robust test error achieved during training and the final error is an average of the robust test error over the last 5 epochs. We observe robust overfitting to occur across all experiments.

				Robust Test Error (%)			Standard Test Error (%)		
Dataset	Adversary	Norm	Radius	Final	Best	Diff	Final	Best	Diff
SVHN	PGD	ℓ_∞	8/255	45.6 ± 0.40	39.0	6.6	10.0 ± 0.15	10.2	−0.2
		ℓ_2	128/255	26.4 ± 0.27	25.2	1.2	7.0 ± 0.23	7.2	−0.2
CIFAR-10	PGD	ℓ_∞	8/255	51.4 ± 0.41	43.2	8.2	13.4 ± 0.19	13.9	−0.5
		ℓ_2	128/255	31.1 ± 0.46	28.4	2.7	11.0 ± 0.08	11.3	−0.3
	FGSM	ℓ_∞	8/255	59.8 ± 0.09	53.7	6.1	12.4 ± 0.21	13.6	−1.2
		ℓ_2	128/255	31.6 ± 0.18	29.2	2.4	9.9 ± 0.16	10.5	−0.6
	TRADES	ℓ_∞	8/255	50.6 ± 0.31	45.0	5.6	14.97 ± 0.24	15.9	−0.9
		ℓ_2	128/255	58.2 ± 0.66	53.6	4.6	33.9 ± 0.95	15.7	18.2
CIFAR-100	PGD	ℓ_∞	8/255	78.6 ± 0.39	71.9	6.7	45.9 ± 0.23	47.3	−1.4
		ℓ_2	128/255	62.5 ± 0.09	56.8	5.7	39.9 ± 0.22	37.5	2.4
ImageNet	PGD	ℓ_∞	4/255	85.5 ± 8.87	62.7	22.8	50.5 ± 14.32	37.0	13.5
		ℓ_2	76/255	94.8 ± 1.16	63.0	31.8	63.2 ± 6.80	40.1	23.1

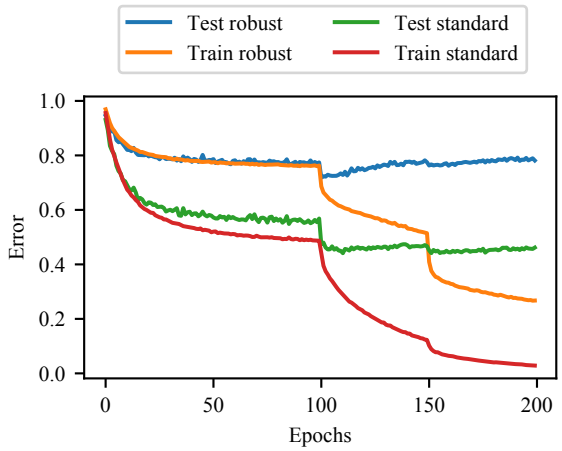

Figure 11. Learning curves showing robust overfitting on CIFAR-100 for the ℓ_∞ perturbation model.

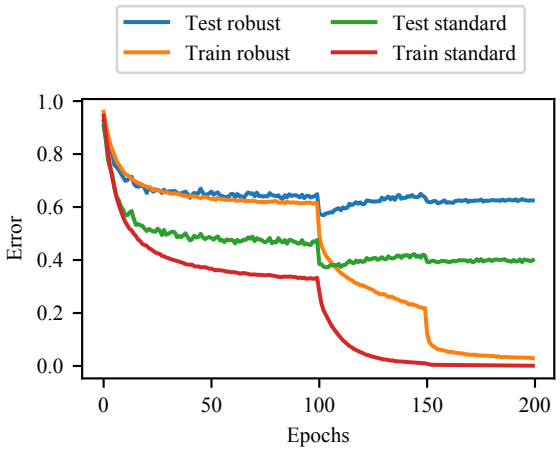

Figure 12. Learning curves showing robust overfitting on CIFAR-100 for the ℓ_2 perturbation model.

bust overfitting on this dataset reflects the CIFAR-10 case, occurring after the initial learning rate decay. Note that in this case, both the robust test accuracy and the standard test accuracy are degraded from robust overfitting. The best checkpoint for ℓ_∞ achieves 71.9% robust error, which is a 6.7% improvement over the 78.6% robust error achieved at the end of training.

A.3. ImageNet experiments

Figure 13 contains the convergence plots for our continuation of PGD-based adversarial training experiments on ImageNet for ℓ_∞ and ℓ_2 perturbations respectively. Thanks to logs provided by the authors (Engstrom et al., 2019), we know the pretrained ℓ_2 robust ImageNet model had already been trained for 100 epochs at learning rate 0.1 followed by at least 10 epochs at learning rate 0.01, and so we continue training from there and further decay the learning rate at the 150th epoch to 0.001. Logs could not be found for the pretrained ℓ_∞ model, and so it is unclear how long it was trained and under what schedule, however the pretrained model checkpoint indicated that the model had been trained for at least one epochs at learning rate 0.001, so we continue training from this point on.

The ℓ_∞ pre-trained model appeared to have not yet converged for the checkpointed learning rate, and so further training without any form of learning rate decay was able to gradually deteriorate the performance of the model. The

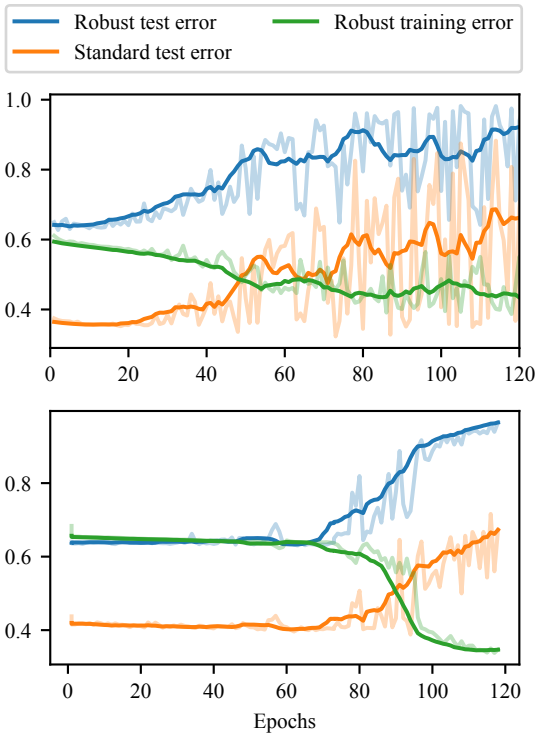

Figure 13. Continuation of training released pre-trained ImageNet models for ℓ_∞ (top) and ℓ_2 (bottom). The number of epochs indicate the number of additional epochs the pre-trained models were trained for.

ℓ_2 pre-trained model seemed to have already converged at the checkpointed learning rate, and so we do not see any significant changes in performance until after decaying the learning rate down to 0.001.

Note that the learning curves here are smoothed by taking an average over a consecutive 10 epoch window, as the actual curves are quite noisy in comparison to other datasets. This noise is reflected in Table 3, where ImageNet has the greatest variation in final error rates (both robust and standard). Training the models further can in fact improve the performance of the pretrained model slightly at specific checkpoints (e.g. from 66.4% initial robust test error down to 62.7% robust test error at the best checkpoint for ℓ_∞), however eventually the ImageNet models suffer greatly from robust overfitting, with an average increase of 22.8% robust error for the ℓ_∞ model and 31.8% robust error for the ℓ_2 model.

A.4. CIFAR-10 experiments

For CIFAR-10, in addition to the standard PGD training algorithm, we also consider the FGSM adversarial training algorithm (Wong et al., 2020) and TRADES (Zhang et al., 2019b). The convergence curves showing that robust overfitting still occurs for these two algorithms in both the ℓ_∞ and ℓ_2 setting are shown in Figures 14 and 15 for FGSM and Figures 16 and 17 for TRADES.

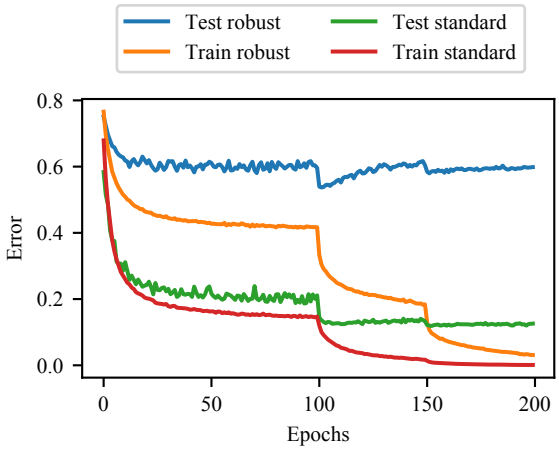

Figure 14. Learning curves showing robust overfitting from training with an FGSM adversary on CIFAR-10 for the ℓ_∞ perturbation model.

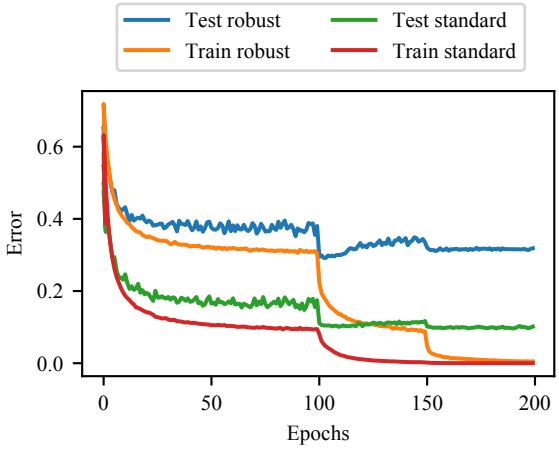

Figure 15. Learning curves showing robust overfitting from training with an FGSM adversary on CIFAR-10 for the ℓ_2 perturbation model.

FGSM adversarial training For FGSM adversarial training, we use the random initialization described by Wong et al. (2020). However, we find that when training until convergence using the piecewise decay learning rate schedule, the recommended step size of $\alpha = 10/255$ for ℓ_∞ training eventually results in catastrophic overfitting. We resort to reducing the step size of the ℓ_∞ FGSM adversary to 7/255 to avoid catastrophic overfitting, but still see robust overfitting.

We also note that Wong et al. (2020) use a cyclic learning rate schedule to further boost the speed of convergence,

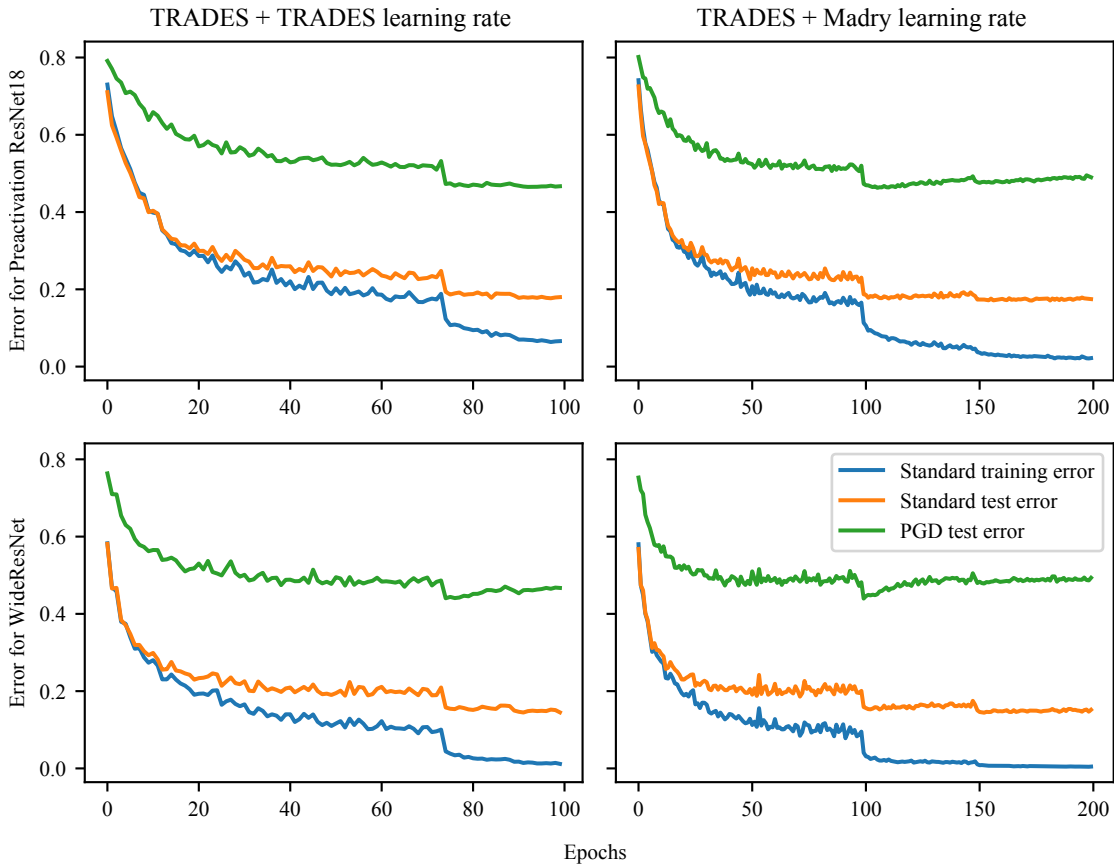

Figure 16. Learning curves when running TRADES for robustness to ℓ_∞ perturbations of radius 8/255 on combinations of learning rates and architectures for CIFAR10.

which differs from the piecewise decay schedule we discuss in this paper. If we run FGSM adversarial training in a more similar fashion to Wong et al. (2020) with the cyclic learning rate and fewer epochs, we find that this can sidestep the robust overfitting phenomenon and converge directly to the best checkpoint at the end of training. However, this requires a careful selection of the number of epochs: too few epochs and the final model underperforms, whereas too many epochs and we observe robust overfitting. In our setting, we find that training against an FGSM adversary for 50 epochs using a cyclic learning rate with a maximum learning rate of 0.2 allows us to recover a final robust test error of 53.22%, similar to the best checkpoint of FGSM adversarial training with piecewise decay and 200 epochs which achieved 53.7% robust test error in Table 3.

Relation of robust overfitting to catastrophic overfitting Previous work studying the effectiveness of an FGSM adversary for robust training noted that it is necessary to prevent "catastrophic overfitting" in order for FGSM training to be successful, which can be avoided by evaluating a PGD adversary on a training minibatch (Wong et al., 2020). Here we note that this is a distinct and separate behavior from robust overfitting: while catastrophic overfitting is a product of a model overfitting to a weaker adversary and can be detected by a stronger adversary on the training set, robust overfitting is a degradation of robust test set performance under the *same* adversary used during training which *cannot be detected on the training set*. Indeed, even successful FGSM adversarial training can suffer from robust overfitting when given enough epochs without catastrophically overfitting, as shown in Figure 14, suggesting that this is related to the generalization properties of adversarially robust training rather than the strength of the adversary.

TRADES For TRADES we use the publicly released implementation of both the defense and attack available at `https://github.com/yaodongyu/TRADES` to remove the potential for any confounding factors resulting from differences in implementation. We consider two possible options for learning rate schedules: the default schedule used by TRADES which decays at 75 and 90 epochs and runs for 100 epochs total (denoted TRADES learning

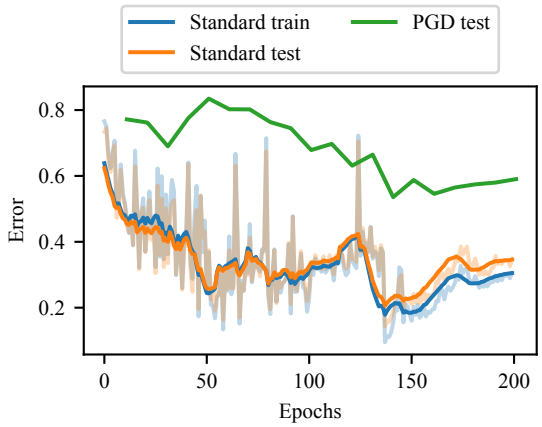

Figure 17. Learning curves when running TRADES for robustness to ℓ_2 perturbations of radius 128/255 for CIFAR10.

rate),[10] and the standard learning rate schedule used by Madry et al. (2017) for PGD adversarial training, which decays at 100 epochs and 150 epochs. We additionally explore both the pre-activation ResNet18 architecture that we use extensively in this paper, as well as the Wide ResNet architecture which TRADES uses. The corresponding learning curves for each combination of learning rate and model can be found in Figure 16 for ℓ_∞.

We note that in three of the four cases, we see a clear instance of robust overfitting. Only the default learning rate schedule used by TRADES on the smaller, pre-activation ResNet18 model doesn't indicate any degradation in robust test set performance. This is likely due the shortened learning rate schedule which implicitly early stops combined with the regularization induced by a smaller architecture having less representational power. The results here are consistent with our earlier findings on the impact of architecture size, where the Wide ResNet architecture achieves better performance than the ResNet18. The shortened TRADES learning rate schedule does not show the full extent of robust overfitting, as the models have not yet converged, whereas the Madry learning rate does (and also achieves a slightly better best checkpoint).

Figure 17 shows a corresponding curve for ℓ_2 robustness using TRADES for the pre-activation ResNet18 model with the Madry learning rate, which was the optimal combination from ℓ_∞ training. Note that the TRADES repository does not provide default training parameters or a PGD adversary for ℓ_2 training on CIFAR-10 nor could we find any such description in the corresponding paper, and so we used our attack parameters which were successful for PGD-based adversarial training (10 steps of size 15/255).

[10]This is the learning rate schedule described in the paper by Zhang et al. (2019c). Note that this differs slightly from the implementation in the TRADES repository, which uses the same schedule but only trains for 76 epochs, which is one more epoch after decaying. In our reproduction of the TRADES experiment, the checkpoint after the initial learning rate decay ends up with the best test performance over all 100 epochs.

B. Experiments for various learning rate schedules

In this section, we explore the effect of the learning rate schedule with greater detail on the CIFAR10 dataset with a pre-activation ResNet18. Our search begins with a sweep over a range of different potential schedules which are commonly used in deep learning. Following this, we tune the best learning rate schedule to investigate its effect on the prevalence of robust overfitting.

B.1. Different types of schedules

We consider the following types of learning rates for our setting.

1. **Piecewise decay:** This is a fairly common learning rate used in deep learning, which decays the learning rate by a constant factor at fixed epochs. We begin with a learning rate of 0.1 and decay it by a factor of 10 at the 100th and 150th epochs, for 200 total epochs.

2. **Multiple decay:** This is a more gradual version of the piecewise decay schedule, with a piecewise constant schedule which reduces the learning rate at a linear rate in order to make the drop in learning rate less drastic. Specifically, the learning rate begins at 0.1 and is reduced by 0.01 every 50 epochs over 500 total epochs, eventually reaching a learning rate of 0.01 in the last 50 epochs.

3. **Linear decay:** This schedule does a linear interpolation of the drop from 0.1 to 0.01, resulting in a piecewise linear schedule. The learning rate is trained at 0.1 for the first 100 epochs, then linearly reduced down to 0.01 over the next 50 epochs, and further trained at 0.01 for the last 50 epochs for a total of 200 epochs.

4. **Cyclic:** This schedule grows linearly from 0 to to some maximum learning rate λ, and then is reduced linearly back to 0 over training as proposed by Smith (2017). We adopt the version from Wong et al. (2020) which already computed the maximum learning rate for the CIFAR10 setting on the same architecture which peaks 2/5 of the way through training at a learning rate of 0.2 over 200 epochs.

5. **Cosine:** This schedule reduces the learning rate using the cosine function to interpolate from 0.1 to 0 over 200 epochs. This type of schedule was used by Carmon

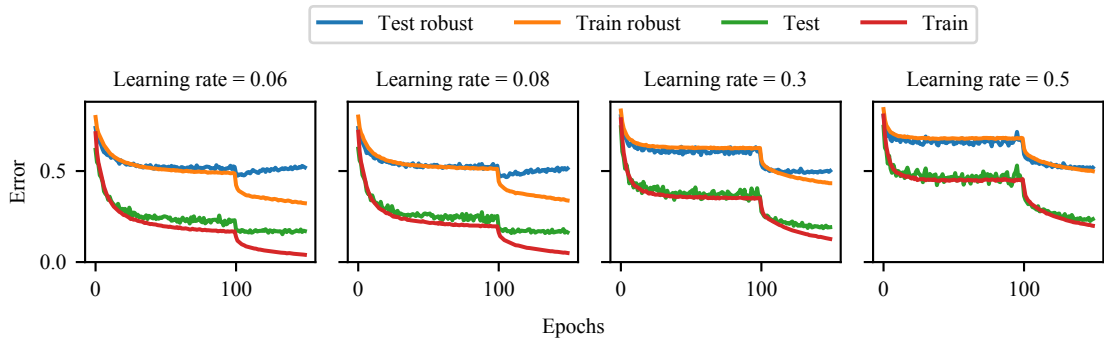

Figure 18. Learning curves for a piecewise decay schedule with a modified starting learning rate.

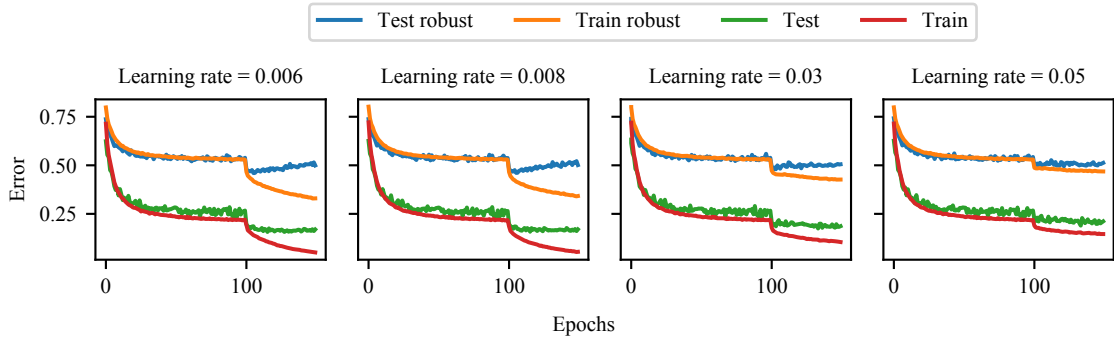

Figure 19. Learning curves for a piecewise decay schedule with a modified ending learning rate.

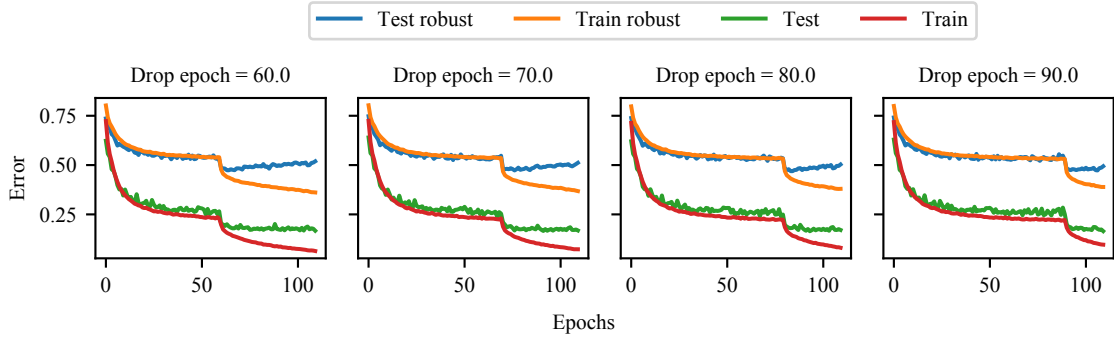

Figure 20. Learning curves for a piecewise decay schedule with a modified epoch at which the decay takes effect.

Table 4. Tuning experiments using stochastic gradient descent to optimize the best robust test error obtained from the piecewise decay schedule for a pre-activation ResNet18 on CIFAR-10.

DECAY EPOCH	START LR	END LR	BEST ROB ERR
100	0.1	0.01	46.7%
60	0.1	0.01	47.4%
70			47.3%
80			46.9%
90			47.3%
100	0.06	0.01	47.4%
	0.08		46.7%
	0.3		48.7%
	0.5		51.0%
100	0.1	0.006	46.0%
		0.008	46.1%
		0.03	47.8%
		0.05	49.3%

et al. (2019) when leveraging semi-supervised data augmentation to improve adversarial robustness.

Note that the piecewise decay schedule is the primary learning rate schedule used in this paper. All of these approaches beyond the standard piecewise decay schedule dampen the initial drop in robust test error experienced by the piecewise decay schedule. As a result, the best checkpoints of these alternatives end up with worse performance than the best checkpoint of the piecewise decay schedule, since all of the learning rates eventually start increasing in robust test error due to robust overfitting after the initial drop. Robust overfitting appears to be ubiquitous across different schedules, as most approaches achieve their best checkpoint well before training has converged.

The cyclic learning rate is the exception here, which has two phases corresponding to when the learning rate is growing and shrinking, with the best checkpoint occurring near the end of the second phase. In both phases, the robust performance begins to improve, but then robust overfitting eventually occurs and keeps the model from improving any further. We found that stretching the cyclic learning rate over a longer number of epochs (e.g. 300) results in a similar learning curve but with worse robust test error for both the best checkpoint and the final converged model.

B.2. Tuning the piecewise decay schedule

Since the piecewise decay schedule appeared to be the most effective method for finding a model with the best robust performance, we investigate whether this schedule can be potentially tuned to improve the robust performance of the best checkpoint even further. The discrete piecewise decay schedule has three possible parameters: the starting learning rate, the ending learning rate, and the epoch at which the decay takes effect. We omit the last 50 epochs of the final decay, since the bulk of the impact from robust overfitting occurs shortly after the first decay in this setting.

While tuning the starting learning rate and the decay epoch largely results in either similar or worse performance, we find that adjusting the learning rate used after the decay epoch can actually slightly improve the robust performance of the best checkpoint by 0.5%, as seen in Table 4. Note that robust overfitting still occurs in these tuned learning rate schedules as seen in Figures 18, 19, and 20, which show the learning curves for each one of the models shown in Table 4.

C. Double descent: exploring architecture sizes

For architecture size experiments, we use a Wide ResNet architecture (Zagoruyko & Komodakis, 2016) with depth 28 and varying widths to control the size of the network. For each width tested, we plot the standard and robust performance from the best checkpoint and final model in Figure 21. Learning curves for each width can be found in Figure 22. All models were trained with the same training parameters described in Section 4. Mean and standard deviation of the final model was taken over the last 5 epochs.

From both the generalization curves and the individual convergence plots, we see that no matter how large the architecture is, the checkpoint which achieves the lowest robust test error always has higher training robust error than the final model at convergence. We also find that both the final model at the end of convergence as well as the best checkpoint found during training all benefit from the increase in architecture size. Consequently, we find that robust overfitting and double descent can occur at the same time, despite having seemingly opposite effects on the notion of overfitting.

In contrast to the standard setting, we observe that the double descent occurs well before robust interpolation of the training data at a width factor of 5, after which the robust test set performance of the final model continues to improve with even larger architecture sizes. The network with width factor 20, the largest that we could run on our hardware, achieves 48.8% robust test error at the end of training and 41.8% robust test error at the best checkpoint. This marks a further improvement over the more typical choice of width factor 10 which achieves 51.4% robust test error at the end of training and 43.2% robust test error at the best checkpoint.

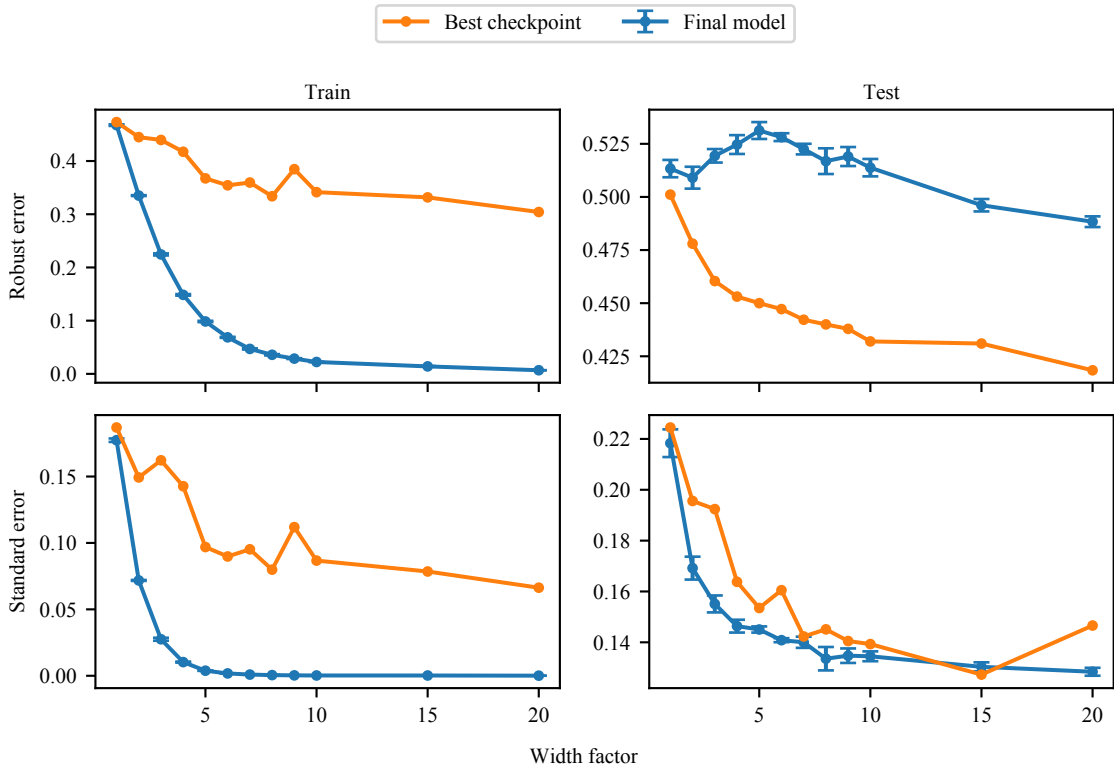

Figure 21. Standard and robust performance on the train and test set across Wide ResNets with varying width factors.

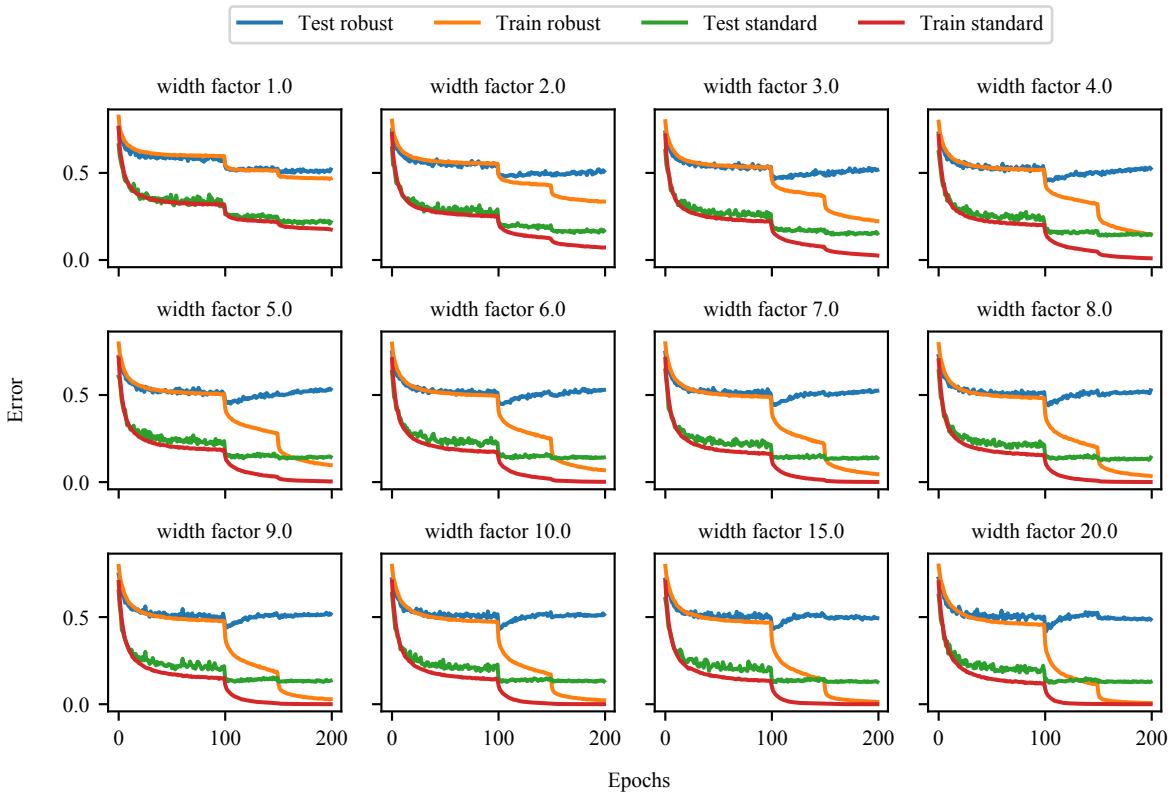

Figure 22. Learning curves for training Wide ResNets with different width factors.

Table 5. Performance of adversarially robust training over a variety of regularization techniques for PGD-based adversarial training on CIFAR-10 for ℓ_∞ with radius $8/255$.

	Robust Test Error (%)			Standard Test Error (%)		
Regularization method	Final	Best	Diff	Final	Best	Diff
Early stopping w/ val	**46.9**	46.7	**0.2**	18.2	18.2	0.0
ℓ_1 regularization	53.0 ± 0.39	48.6	4.4	15.9 ± 0.13	15.4	0.5
ℓ_2 regularizaiton	51.4 ± 0.73	46.4	8.8	15.7 ± 0.21	14.9	0.8
Cutout	48.8 ± 0.79	46.7	2.1	16.8 ± 0.21	16.4	0.4
Mixup	49.1 ± 1.32	46.3	2.8	23.3 ± 3.04	19.0	4.3
Semi-supervised	47.1	40.2	6.9	23.0 ± 3.82	17.2	5.8

D. Preventing overfitting

D.1. Experimental setup

For the experiments in preventing overfitting, we use a PGD adversary with random initialization and 10 steps of step size 2/255. This is a slightly stronger adversary than considered in Madry et al. (2017) by using 3 additional steps, and we found the attack to be more effective than the adversary implemented by TRADES, achieving approximately 1% more PGD error than the TRADES adversary. However, our goal here is to explore the prevention of robust overfitting, and so it is not necessary to have strongest possible adversarial attack, and so for our purposes this adversary is good enough (and is known to be reasonably strong in the ℓ_∞ setting). For training, we use the same parameters as used for the CIFAR-10 experiments in Appendix A.4 (batch size, learning rate, weight decay, number of epochs). We primarily use the pre-activation ResNet18 since it is already sufficient for exhibiting the robust overfitting behavior.

D.2. Full set of results for Table 2

In this section, we present the expanded version of Table 2 to include standard test error metrics. The final robust and standard errors are an average of over the final 5 epochs of training when the model has converged, from which the standard deviation is also computed. The one exception is validation-based early stopping, where the final error is taken from the checkpoint chosen by the validation set, and consequently does not have a standard deviation. The best robust error is the lowest test robust error of all checkpoints through training, and the best standard error is the corresponding standard error which comes from this same checkpoint. For convenience we also show the difference in the final model's error and the best model's error, which indicates the amount of degradation incurred by robust overfitting.

D.3. Explicit regularization

In this section, we extend the plots depicting the robust and standard error over various regularization hyperparameters to also show the performance on the training set. We also show the learning curves for models trained with explicit regularization to show the extent of robust overfitting on various hyperparameter choices.

ℓ_1 regularization Figure 23 shows the training and testing performance of models using various degrees of ℓ_1 regularization. We performed a search over regularization parameters $\lambda = \{5 \cdot 10^{-6}, 5 \cdot 10^{-5}, 5 \cdot 10^{-4}, 5 \cdot 10^{-3}\}$, and found that both the final checkpoint and the best checkpoint have an optimal regularization parameter of $5 \cdot 10^{-5}$. Note that we only see robust overfitting at smaller amounts of regularization, since the larger amounts of regularization actually regularize the model to the point where the performance is being severely hurt.

Figure 24 shows the corresponding learning curves for these four models. We see clear robust overfitting for the smaller two options in λ, and find no overfitting but highly regularized models for the larger two options, to the extent that there is no generalization gap and the training and testing curves actually appear to match.

ℓ_2 regularization Figure 25 shows the training and testing performance of models using various degrees of ℓ_2 regularization. We performed a search over regularization parameters $\lambda = \{5 \cdot 10^k\}$ for $k \in \{-4, -3, -2, -1, 0\}$ as well as $\lambda = 0.01$. Note that $5 \cdot 10^{-4}$ is a fairly widely used value for weight decay in deep learning. We find that only the smallest choices for λ result in robust overfitting (e.g. $\lambda \leq 0.1$. However, inspecting the corresponding learning curves in Figure 26 reveals that the larger choices for λ have a similar behavior to the larger forms of ℓ_1 regularization, and end up with highly regularized models whose test performance perfectly matches the training performance at the cost of converging to a worse final robust test error.

D.4. Data augmentation

In this section, we present additional details for the data augmentation approaches for preventing overfitting, namely cutout, mixup, and semi-supervised data.

Cutout To analyze the effect of cutout on generalization, we range the cutout hyperparameter of patch length from 2 to 20. Figure 27 shows the training and testing performance of models using varying choices of patch lengths. Additionally, for each hyperparameter choice, we plot the resulting learning curves in Figure 28.

We find the optimal length of cutout patches to be 14, which on it's own is not quite as good as vanilla early stopping, but when combined with early stopping merely matches the performance of vanilla early stopping. In all cases, we observe robust overfitting to steadily degrade the robust test performance throughout training, with less of an effect as we increase the cutout patch length.

Mixup When training using mixup, we vary the hyperparameter α from 0.2 to 2.0. The training and testing performance of models using varying degrees of mixup can be found in Figure 29. The resulting learning curves for each choice of α can be found in Figure 30.

For mixup, we find an optimal parameter value of $\alpha = 1.4$. Similar to cutout, when combined with early stopping, it can only attain similar performance to vanilla early stopping, and otherwise converges to a worse model. However, although the learning curves for mixup training are significantly noisier than other methods, we do observe the robust test error to steadily decrease over training, indicating that mixup does stop robust overfitting to some degree (but does not obtain significantly better performance).

E. Semi-supervised approaches

For semi-supervised training, we use a batch size of 128 with equal parts labeled CIFAR-10 data and pseudo-labeled TinyImages data, as recommended by Carmon et al. (2019). Each epoch of training is now equivalent in computation to two epochs of standard adversarial training. Note that the pre-activation ResNet18 is a smaller architecture than used by Carmon et al. (2019), and so in our reproduction, the best checkpoint which achieves 40.2% error is about 2% higher than 38.5%, which is what Carmon et al. (2019) can achieve with a Wide ResNet. Note that in the typical adversarially robust setting without additional semi-supervised data, a Wide ResNet can achieve about 3.5% lower error than a pre-activation ResNet18.

We observe that the semi-supervised approach does not exhibit severe robust overfitting, as the smoothed learning curves tend to be somewhat relatively flat and don't show significant increases in robust test error. However, relative to the base setting of using only the original dataset, the robust test performance is extremely variable, with a range spanning almost 10% robust error even when training error is relatively flat and has converged. As a result, it is critical to still use the best checkpoint even without robust overfitting, in order to avoid the fluctuations in test performance induced by the augmented training data.

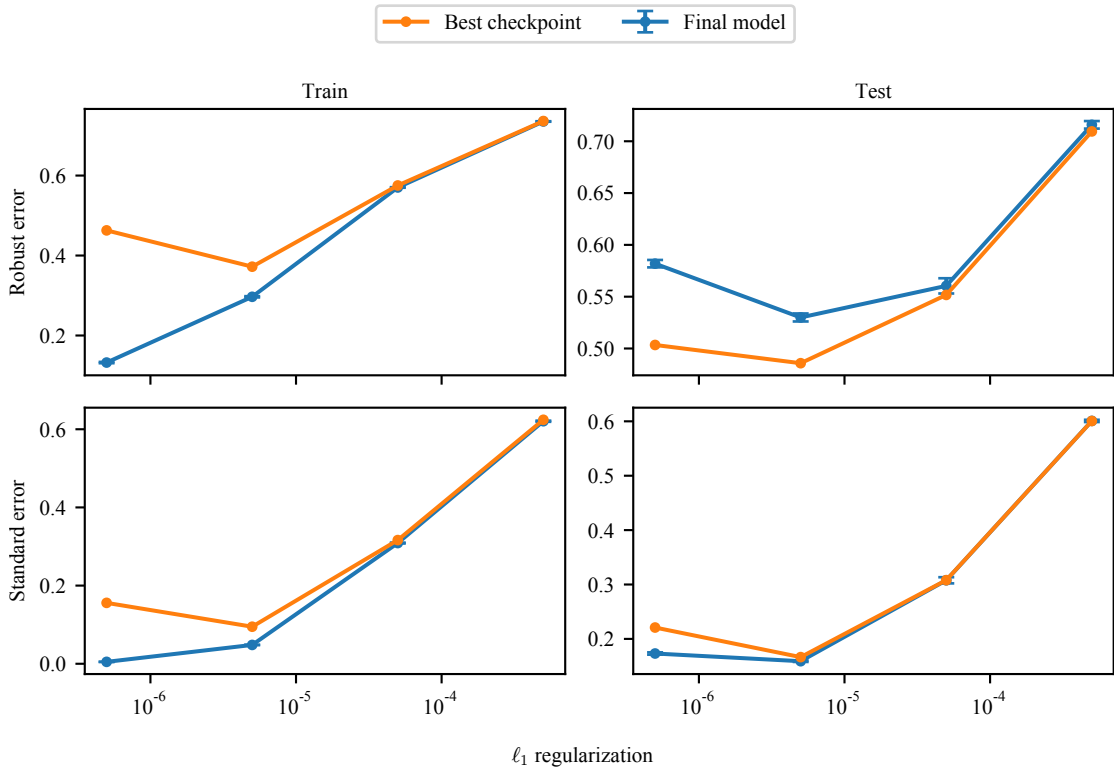

Figure 23. Standard and robust performance on the train and test set using varying degrees of ℓ_1 regularization.

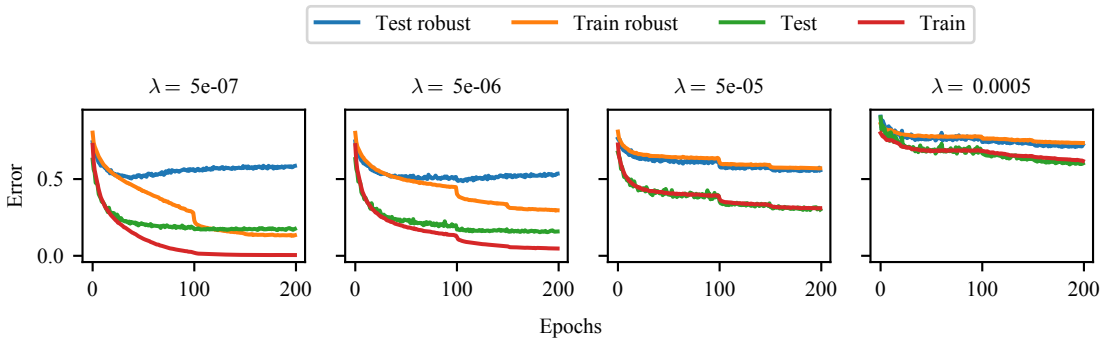

Figure 24. Learning curves for adversarial training using ℓ_1 regularization.

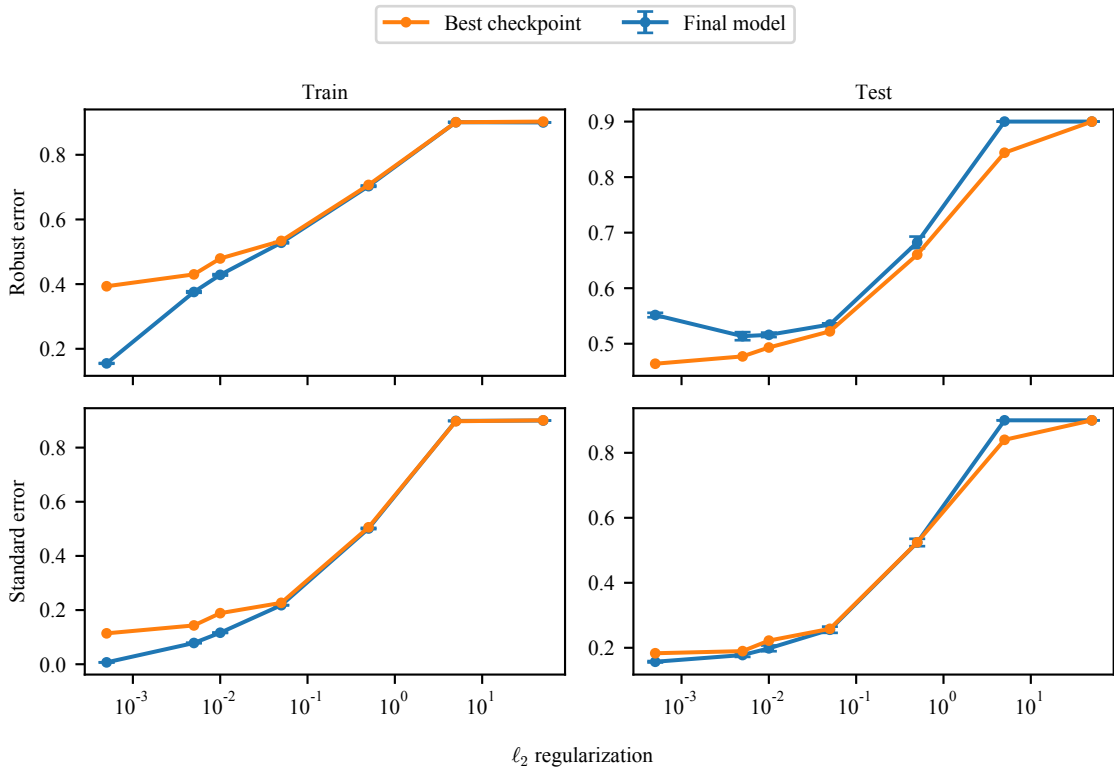

Figure 25. Standard and robust performance on the train and test set using varying degrees of ℓ_2 regularization.

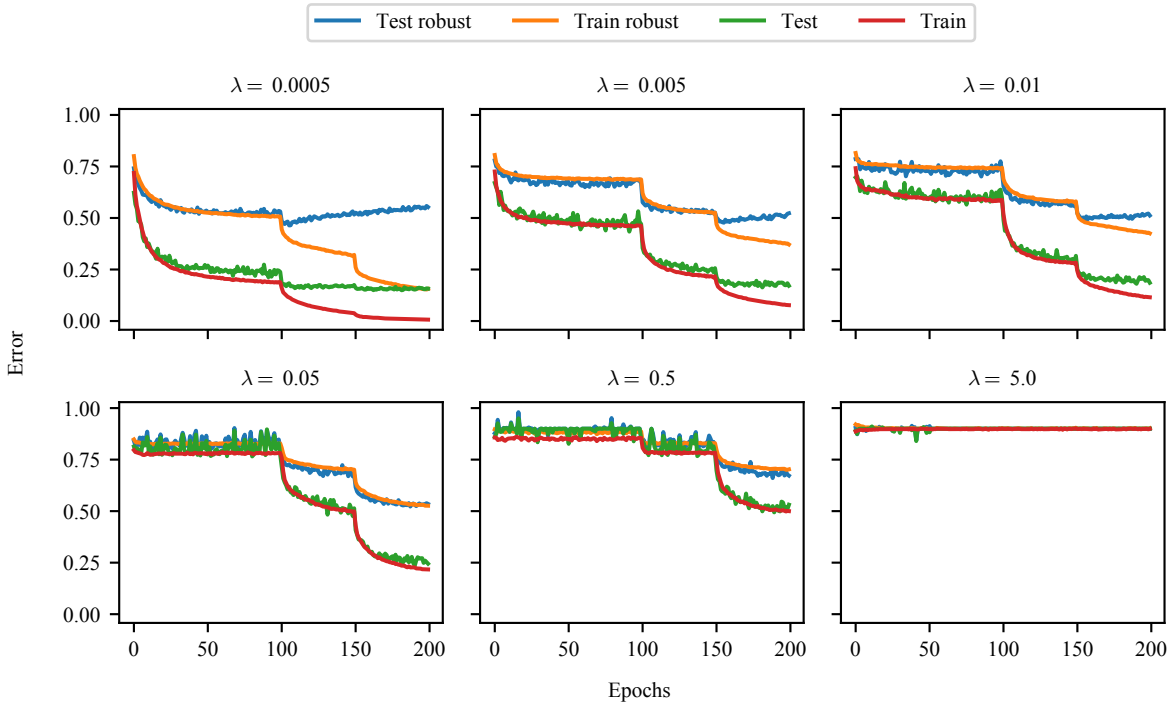

Figure 26. Learning curves for adversarial training using ℓ_2 regularization.

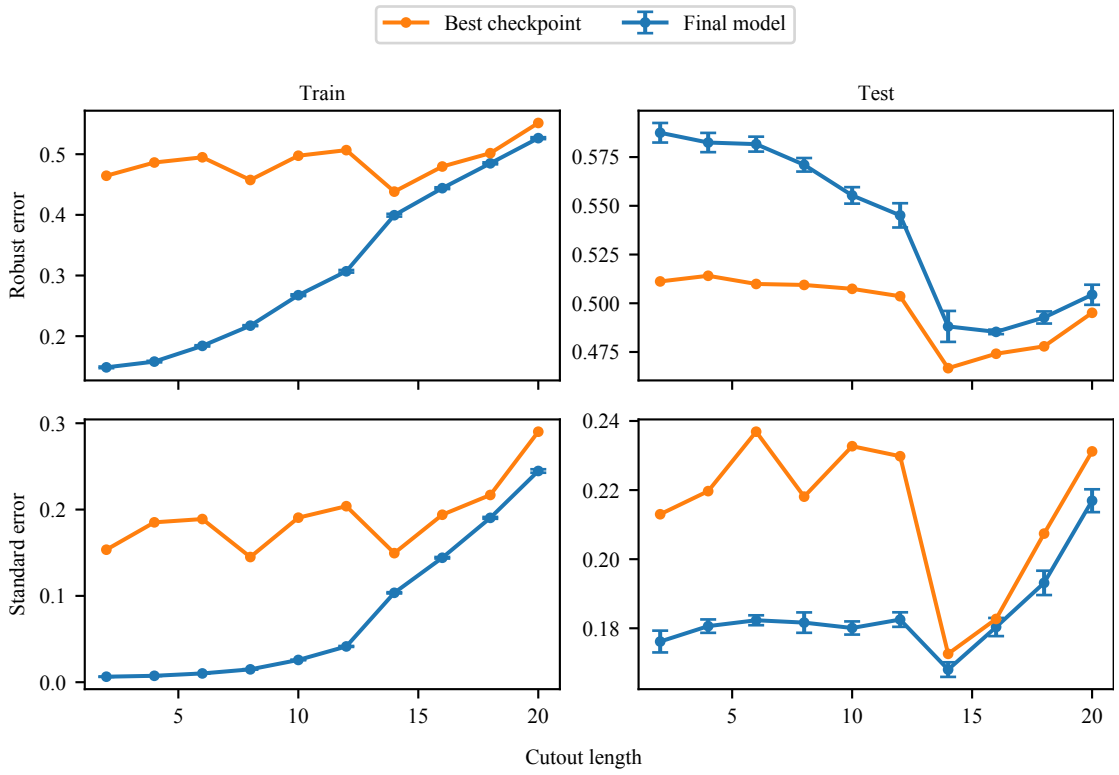

Figure 27. Standard and robust performance on the train and test set for varying cutout patch lengths.

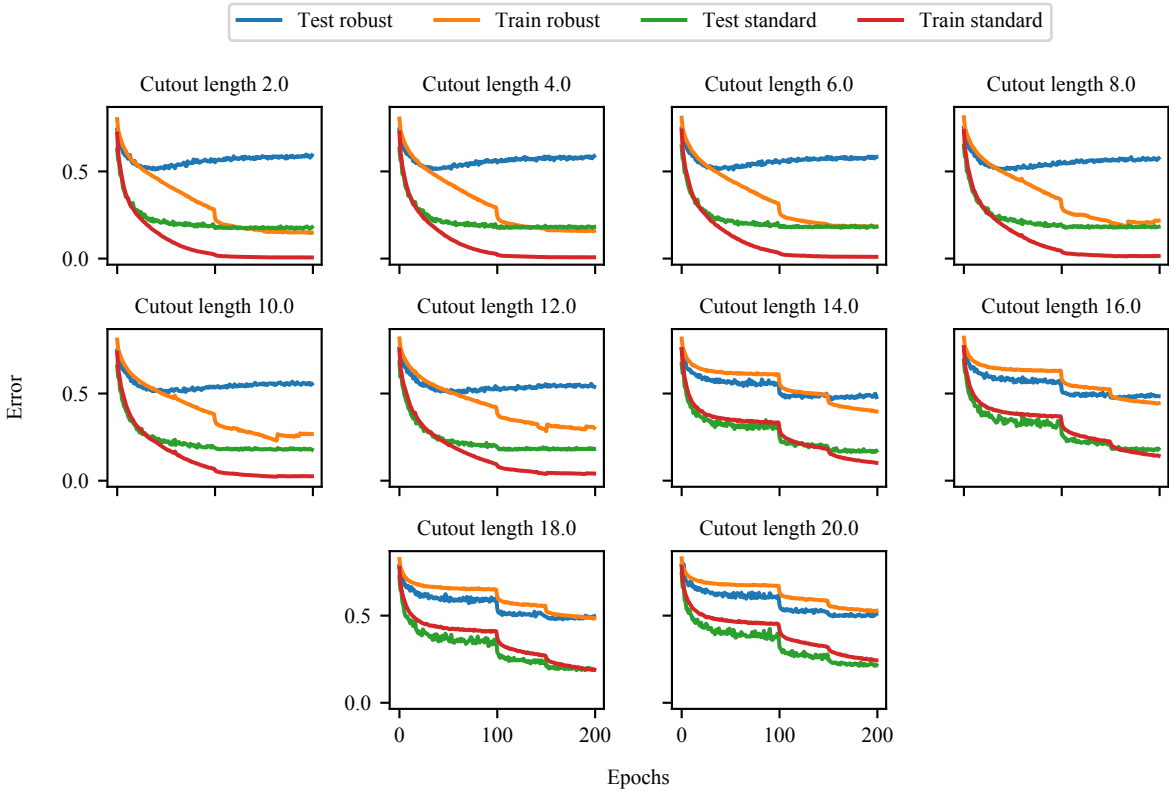

Figure 28. Learning curves for adversarial training using cutout data augmentation with different cutout patch lengths.

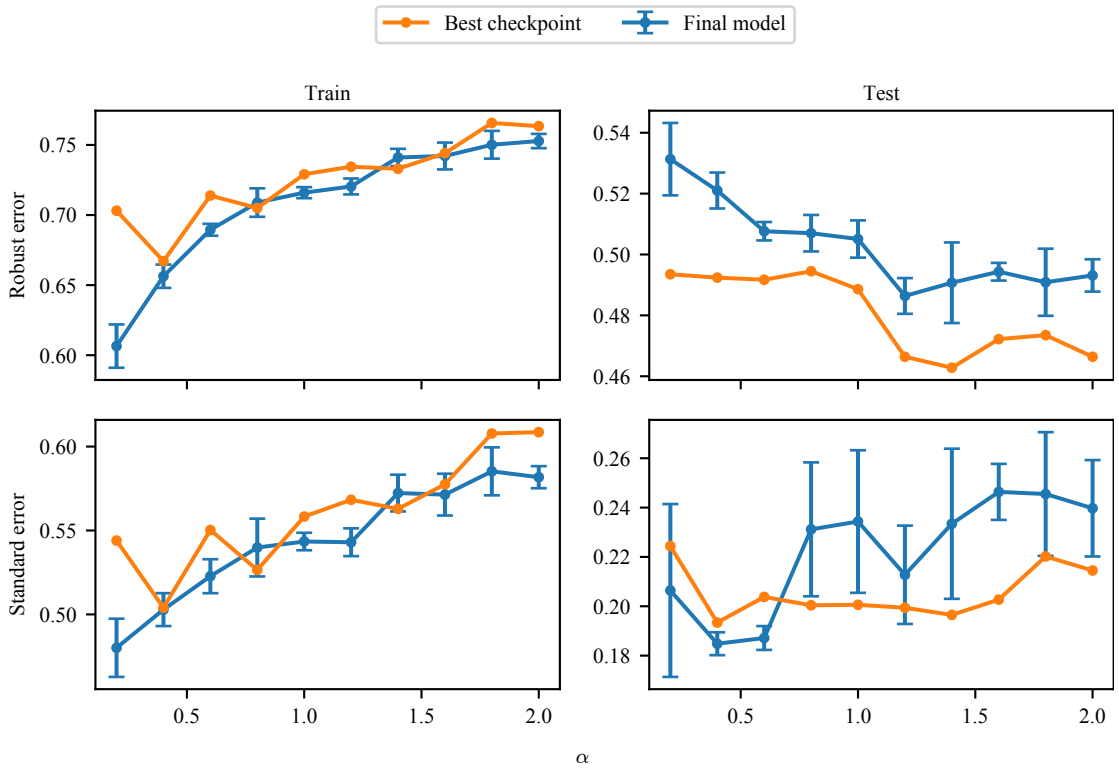

Figure 29. Standard and robust performance on the train and test set for varying degrees of mixup.

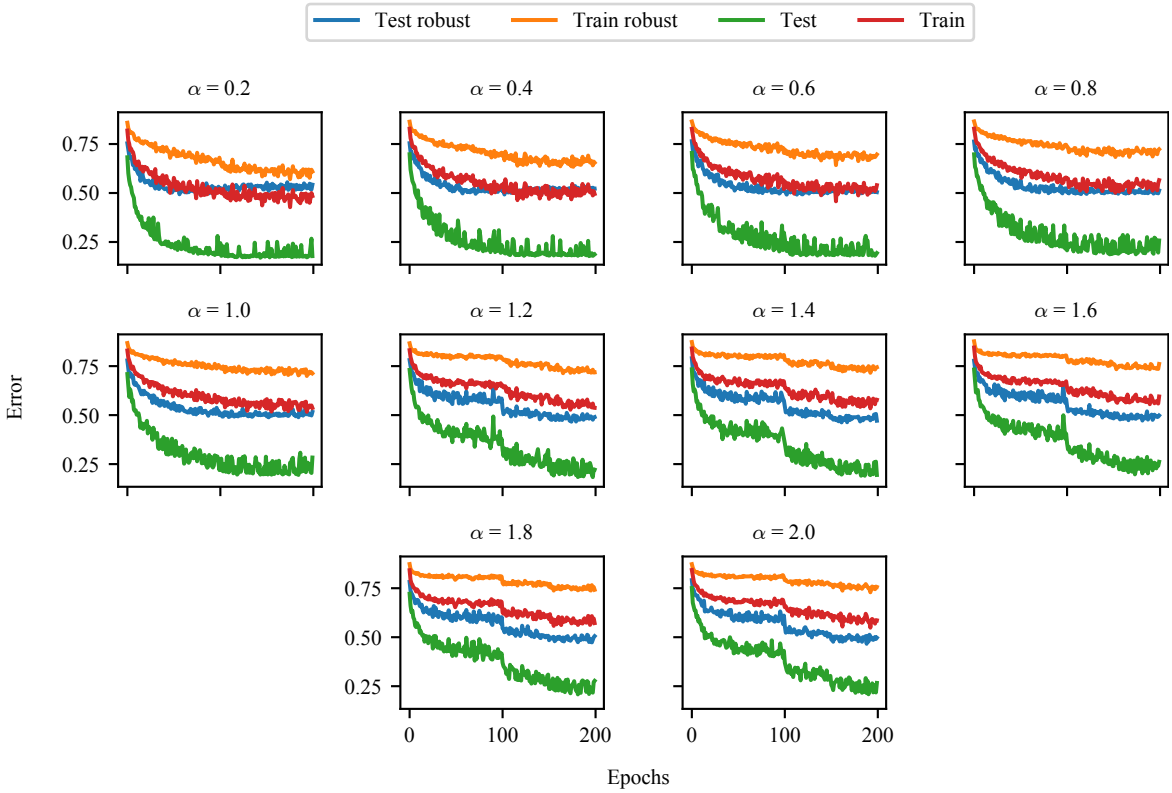

Figure 30. Learning curves for adversarial training using mixup with different choices of hyperparameter α.