Kernel methods and Deep Learning II

Johannes Pitz

Seminar: Optimization and Generalization in Deep Learning

Abstract

In deep neural networks often models which were trained until they reach zero training error with little to no regularization perform better on unseen test sets than the same model trained with early stopping and/or explicit regularization. This effect is contradicting the general wisdom taught in machine learning classes, that overfitting will lead to decreasing test performance. It is not well understood, why these generalization properties can be observed.

This report will show that the mentioned effect is not unique to deep neural networks, but can also be observed in kernel methods. We will show that kernel methods can be viewed as two-layer neural networks. Therefore it might be advantageous to closely investigate the much simpler kernel methods before trying to understand more complex deep architectures.

1 Introduction

Closely following the work of Beklin et al. [4] we will investigate the generalization properties of kernel methods and compare our results to recent work in deep neural networks.

- First we show experimentally that overfitted or interpolated kernel classifiers generalize well onto unseen test sets.
- Then we look at the proof given by Belkin et al. showing that currently known bounds are unlikely to explain this behaviour.
- From our final experiment we conclude that more analysis regarding the correlation of training data size, function norm, and generalization properties is required.

2 Related Work

Except from the mainly analyzed work by Belkin et al. [4] most research regarding the surprising generalization properties of overfitted classifiers is done in deep neural networks. For example Poggio et al. [7] showed that properties of stochastic gradient descent (SGD) for linear networks also hold for deep non linear networks. Firstly, SGD enforces a implicit form of regularization and converges to the minimum norm solution, and secondly for classification tasks this minimum norm solution is also the maximum margin solution. Therefore it yields good generalization results on low noise datasets. Bartlett et al. [2] showed that SGD selects predictors whose complexity scales with the difficulty of the learning task, and conclude that the good generalization is a property of the optimization algorithm. Zhang et al. [9] successfully trained state-of-the-art convolutions neural networks (CNN) on randomly labeled datasets close to zero classification error, and found that choosing the right model family or regularization techniques doesn't explain the small difference between training and test performance on correctly labeled datasets.

3 Approach

3.1 Experiment 1

At first we will train kernel classifiers on MNIST and CIFAR-10 using two different approaches. One will be called *interpolated* and the other *overfitted.*

Let $K(\mathbf{x},\mathbf{z}) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be positive definite. Given data $\{(\mathbf{x}_i, y_i), i = 1, ..., n\}, \mathbf{x}_i \in \mathbb{R}^d, y_i \in \mathbb{R}$, then there exists a Reproducing Kernel Hilbert Space (RKHS) $\mathscr{H}$ of functions f on

$\mathbb{R}^d$. Moreover let $\mathbf{K}$ be the kernel matrix with $\mathbf{K}_{ij} = K(x_i, x_j)$. Then we are looking for the function f^* which correctly classifies all training samples and minimizes the RKHS norm.

$$f(\cdot) = \sum_{i=1}^{n} \alpha_i K(x_i, \cdot) \tag{1}$$

$$f^* = \underset{f \in \mathscr{H}, f(\mathbf{x}_i)=y_i}{\arg\min} \|f\|_{\mathscr{H}}, \text{ where } \|f\|^2_{\mathscr{H}} = \langle \boldsymbol{\alpha}, \mathbf{K}\boldsymbol{\alpha} \rangle = \sum_{ij} \alpha_i \mathbf{K}_{ij} \alpha_j \tag{2}$$

Interpolated: By the classical representer theorem we get an explicit form to solve for the optimal α_i.

$$\boldsymbol{\alpha}^* = \mathbf{K}^{-1}\mathbf{y} \tag{3}$$

Note when we plug in the training data X into the function f^* we can see that indeed all samples are classified correctly.

$$\begin{aligned} f^*(\mathbf{X}) &= \mathbf{K}\boldsymbol{\alpha}^* \\ &= \mathbf{K}\mathbf{K}^{-1}\mathbf{y} \\ f^*(\mathbf{x}_i) &= y_i \end{aligned} \tag{4}$$

Overfitted: To avoid having to solve the linear system which is prohibitive for larger datasets we can also pick any non-negative and strictly convex loss function l, such that $l(y, y) = 0$, and solve the unconstrained optimization problem.

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha} \in \mathbb{R}}{\arg\min} \sum_{j=1}^{N} l(\sum_{i=1}^{N} \alpha_i K(x_i, x_j), y_i) \tag{5}$$

Since f^* minimizes $\sum_{j=1}^{N} l(f(x_i), y_i)$, this allows us to use gradient descent methods, and therefore shows the resemblance to a neural network. Basically, the weights between the input and the first layer are fixed (all the features of all training samples). This first layer has N neurons, where N is the number of training samples. The α_i are then the trainable weights connecting the first layer with the output layer.

3.2 Experiment 2

For this experiment we will additionally use two synthetic datasets, and we will add label noise to both synthetic and real datasets.

Adding noise: We will randomly flip the true label of a ϵ-fraction of the training and test samples to any of the possible labels with equal probability. Note that the Bayes Optimal Classifier remains the same on this new dataset and that the error rate scales linearly in ϵ between the error rate on the original dataset and random guessing (confer to Proportion 1 in Belkin et al. [4] for more details).

Synthetic dataset: The synthetic dataset has features $x \in \mathbb{R}^{50}$ and labels $y \in \{0, 1\}$. While $x_2 \sim \mathcal{U}(-1, 1), ..., x_{50} \sim \mathcal{U}(-1, 1)$ are drawn from the same uniform distribution, x_1 is drawn from a normal distribution depending on the class label.

$$x_1 \in \begin{cases} \mathcal{N}(0, 1), & \text{if } y = 1 \\ \mathcal{N}(2, 1), & \text{otherwise} \end{cases} \tag{6}$$

Note that the error of the Bayes Optimal Classifier is around 15.9% for this dataset (the classes are not separable).

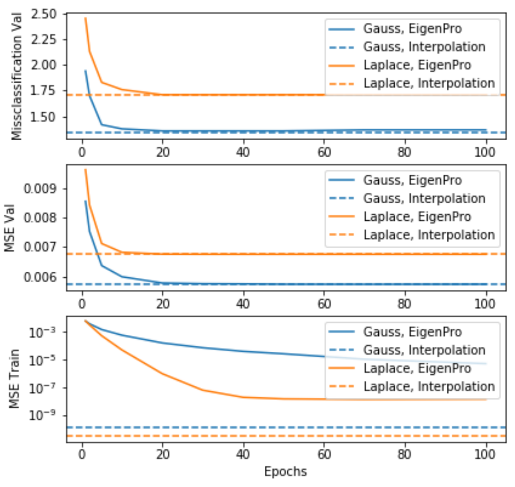
3.3 General

Throughout this report we will use two different kernels, the smooth Gaussain kernel $K(x, y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$ and the non-smooth Laplacian kernel $K(x, y) = \exp\left(-\frac{\|x-y\|}{\sigma}\right)$. Most of the overfitted examples are trained using the iterative EigenPro-SGD method by Ma and Belkin [6]. EigenPro is a preconditioned gradient descent iteration especially designed for fast convergence in kernel learning. In one occasion we used the Pegasos optimizer [8] because the EigenPro optimizer consistently converged towards random guessing on the training data instead of overfitting it. Both optimizer are available in the Keras Deep Learning library. The code for all experiments can be found on github (link).

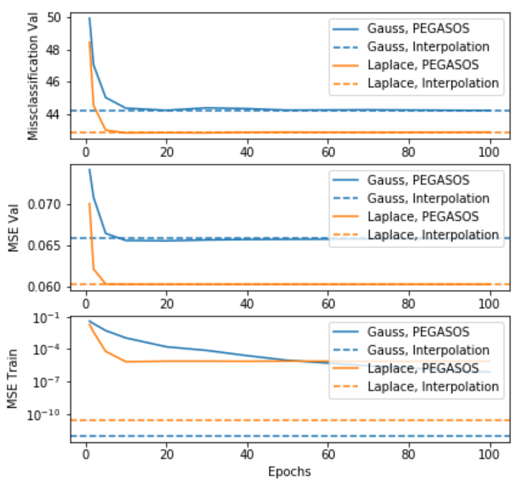
4 Results

4.1 Experiment 1

Unlike on MNIST, we weren't immediately successfully to reproduce the results of Belkin et al. on CIFAR-10, however using all features (i.e. not converting the pictures to grey scale) we achieved the desired results. Looking at Figure 1 one can now see that for the *overfitted* classifiers early stopping doesn't offer any benefit on the test error even though they are heavily (over)fitting to the training data. Moreover even the interpolated classifier achieves the same test error.



(a) MNIST



(b) CIFAR-10

Figure 1: Comparison between overfitted and interpolated classifiers.

4.2 Theory

Surprisingly we can show that any existing generalization bounds are unlikely to explain the behaviour seen in the first experiment. Before stating the theorem (Theorem 1 in Belin et al. [4]) we will go over some definitions.

Let $\Omega \subset \mathbb{R}^d$, and $\mathscr{H}$ the RKHS corresponding to a given kernel function $K(\mathbf{x}, \mathbf{z}) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Let P be a probability measure on $\Omega \times \{-1, 1\}$, and $\{(\mathbf{x}_i, y_i), i = 1, ..., n\}$ a dataset sampled from P. For the theorem to hold there must be non zero label noise, i.e. y needs to be a non-deterministic function of x. And finally we say $h \in \mathscr{H}$ t-overfits the data, if and only if, it achieves zero classification error, and $\forall_i y_i h(\mathbf{x}_i) > t > 0$. This margin condition needs to be added because we can always scale a solution by some constant allowing us to control the RKHS norm.

Theorem 1. *Any $h \in \mathscr{H}$ corresponding to a Gaussian kernel that t-overfits the data satisfies with high probability*

$$\|h\|_{\mathscr{H}} > Ae^{Bn^{\frac{1}{d}}} \tag{7}$$

for some constants $A, B > 0$ depending on t.

Proof. Let $B_R = \{f \in \mathscr{H}, \|f\|_{\mathscr{H}} < R\} \subset \mathscr{H}$ be the ball of functions which have a RKHS norm less than or equal to R, let $l(f(\mathbf{x}), y) = \max(t - yf(\mathbf{x}), 0)$ be the hinge loss with margin t, and let $V_\gamma(B_R)$ be the fat shattering dimension of the function space B_R with the parameter γ.

Then by Anthony and Bartlett [1] $\exists C_1, C_2 > 0$ such that with high probability $\forall_{f \in B_R}$:

$$\left| \frac{1}{n} \sum_i l(f(\mathbf{x}_i), y_i) - \mathbb{E}_P[l(f(\mathbf{x}), y)] \right| \leq C_1 \gamma + C_2 \sqrt{\frac{V_\gamma(B_R)}{n}} \tag{8}$$

Since y is not a deterministic function of x, $\mathbb{E}_P[l(f(\mathbf{x}), y)] > 0$. Now we fix $\gamma > 0$ such that $C_1\gamma < \mathbb{E}_P[l(f(\mathbf{x}), y)]$. And suppose $h \in B_R$ t-overfits the data, then by construction $\frac{1}{n}\sum_i l(h(\mathbf{x}_i), y_i) = 0$.

$$0 < \mathbb{E}_P[l(f(\mathbf{x}), y)] - C_1\gamma < C_2 \sqrt{\frac{V_\gamma(B_R)}{n}} \tag{9}$$

$$\frac{n}{C_2^2}(\mathbb{E}_P[l(f(\mathbf{x}), y)] - C_1\gamma)^2 < V_\gamma(B_R) \tag{10}$$

Then by Belkin [3] $V_\gamma(B_R) < O(\log^d(\frac{R}{\gamma}))$, meaning that:

$$\frac{n}{C_2^2}(\mathbb{E}_P[l(f(\mathbf{x}), y)] - C_1\gamma)^2 < \log^d(\frac{R}{\gamma}) \tag{11}$$

$$A e^{Bn^{\frac{1}{d}}} < R \text{ where } A, B > 0 \tag{12}$$

□

Considering that that most generalization bounds are of the form

$$|\underbrace{\tfrac{1}{n}\textstyle\sum_i l(f(\mathbf{x}_i), y_i)}_{\text{Train Loss}} - \underbrace{\mathbb{E}_P[l(f(\mathbf{x}), y)]}_{\text{Expected Test Loss}}| \leq C_1 + C_2 \frac{\|f\|^\alpha_{\mathscr{H}}}{n^\beta} \tag{13}$$

with $C_1, C_2, \alpha, \beta \geq 0$, we may conclude that for large enough n these bounds are trivial.

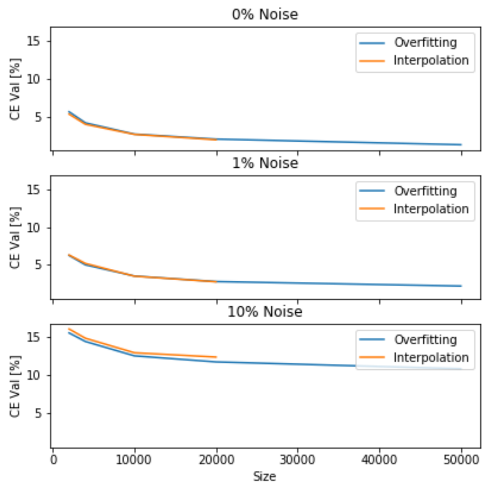
4.3 Experiment 2

Since we assumed non-zero label noise in Theorem 1 we investigate how the classification error of *overfitted* and *interpolated* classifiers behaves with added noise. We use the Gaussian Kernel (applicable to the theorem) and different sized subsamples of MNIST, CIPHAR-10 and the Synthetic dataset described in Section 3. The *overfitted* classifiers are trained until they reach zero classification error or 100 iterations, after which the training error was less than 1 percent in all cases.

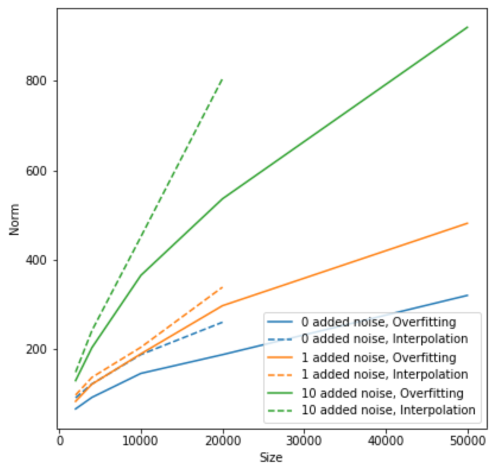
In Figure 2 and 3 we can see that even after adding a considerable amount of noise the classification error decreases with training data size. On the MNIST dataset the classifier actually achieves close to optimal performance. This might not be surprising initially since more data often yields better performance in machine learning tasks. However, the RKHS norm increases quickly, as predicted by Theorem 1. Which again leaves us with no good explanation for the generalization properties.

Now for the first time somewhat conflicting with the results of Belkin et al. [4] in Figure 4 we can see that the norm seems not to scale exponentially with the training data size. We encountered this behaviour by simply extending the training sample size of the same experiment conducted by Belkin et al. Looking back at Theorem 1 this indicates that due to the d^{th} root the exponential term does not yet dominate the bound itself. Note that the classifiers here are around 5% off the Bayes Optimal performance, but clearly learned something useful.

In Figure 5 we ran the same experiment, but with only 4 input features instead of 50. Interestingly we can now see that once the norm increases exponentially the classifiers performance drops to random guessing. This could potentially imply that we can explain the generalization using existing bounds, contradicting the claims made by Belkin et al. However, it would take much more analysis before we should draw any conclusions from this single non-generalizing example. Presumably they picked $d = 50$ deliberately, so we assume they are aware that using less dimensions makes it harder to produce the desired generalization properties, but still believe researchers should seek for new bounds independent of the norm.

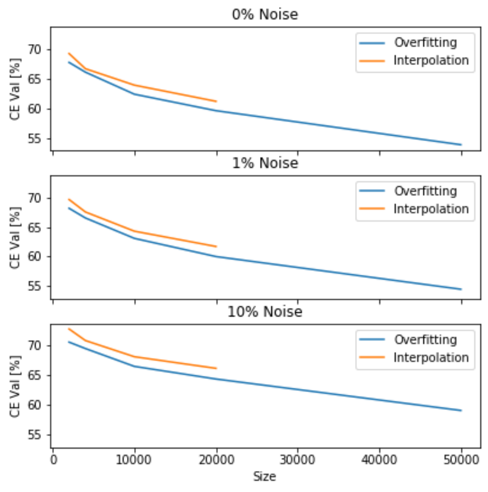


(a) Classification Error

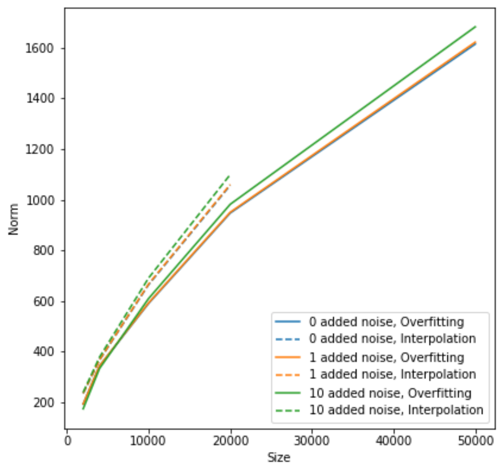


(b) Norm

Figure 2: MNIST

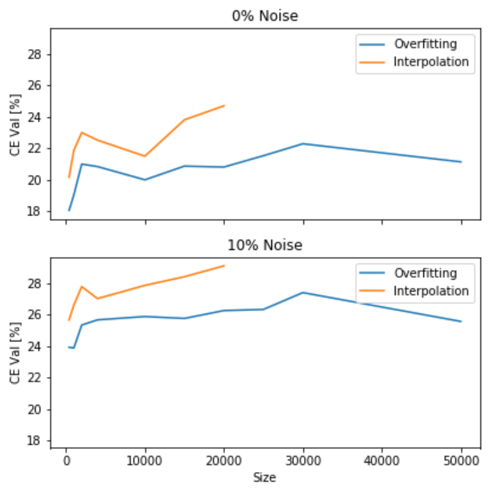


(a) Classification Error

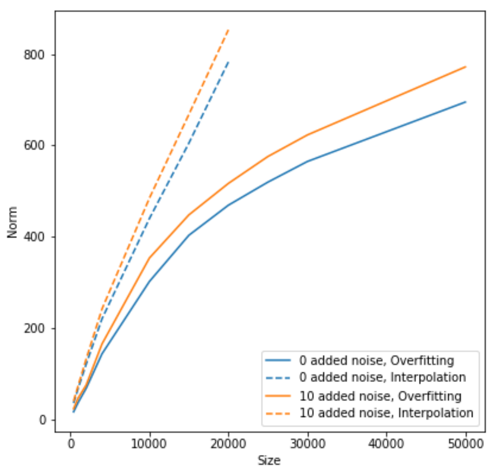


(b) Norm

Figure 3: CIFAR-10

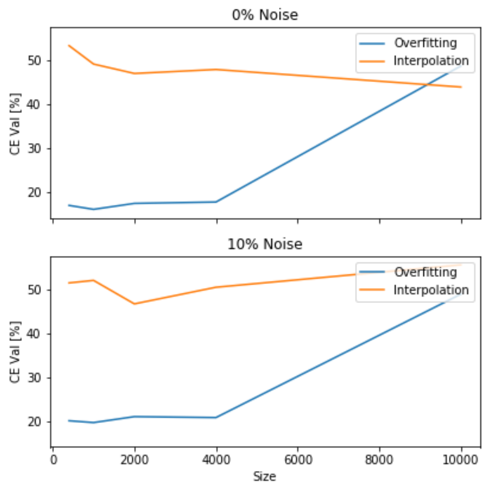


(a) Classification Error

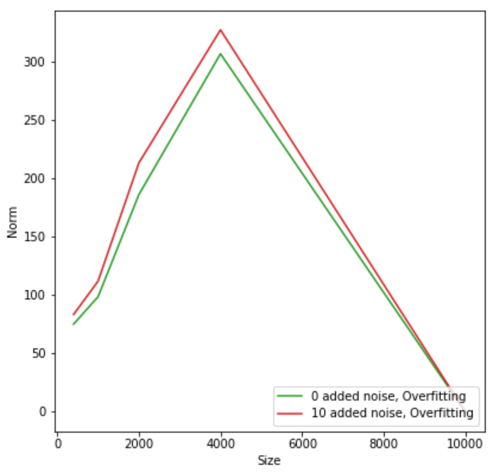


(b) Norm

Figure 4: Synthetic, d=50



(a) Classification Error



(b) Norm

Figure 5: Synthetic, d=4
(Note: the norm at 10000 samples is NaN not zero)

5 Conclusion

The first experiment clearly shows similar generalization behaviour between the modern highly overparameterized deep neural networks and classical kernel methods, therefore it might be very valuable to investigate kernel methods further as suggested by Belkin et al. [4], and apply potential findings to more complex deep networks.

The theoretical result clearly shows that generalization bounds depending on the norm are not useful for very large datasets. However, it remains unclear whether there would still be good generalization if we trained on datasets large enough to force the exponential component of the lower bound on the function norm to dominate the other constants. Further experimental work with larger datasets and more computational resources, or more realistically, with lower dimensional input features could clear up these questions. Moreover, it might even give us more intuition about where to focus the search for new generalization bounds independent of the norm (as suggested by Belkin et al.).

Saying that we should also not ignore the possibility that the generalization depends on the optimization algorithm as indicated by Poggio et al. [7] and Bartlett et al. [2], or on properties of the dataset. Properties which are shared among real datasets, but are not easily expressible and more importantly might not be found in dully created synthetic datasets. I.e. Zhang et al. [9] argue that we need a precise formal measure under which complex models are simple, but we could also extend that claim to datasets.

it depends on all three: data, model, and optimizer.

Finally it remains to say we should consider all of them and also kernel methods.

References

[1] Martin Anthony and Peter L. Bartlett. *Neural Network Learning - Theoretical Foundations.* Cambridge University Press, 2002.

[2] Peter L. Bartlett, Dylan J. Foster, and Matus J. Telgarsky. Spectrally-normalized margin bounds for neural networks. In Guyon et al. [5], pages 6241–6250.

[3] Mikhail Belkin. Approximation beats concentration? an approximation view on inference with smooth radial kernels. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet, editors, *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018.*, volume 75 of *Proceedings of Machine Learning Research*, pages 1348–1361. PMLR, 2018.

[4] Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 540–548. PMLR, 2018.

[5] Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors. *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, 2017.

[6] Siyuan Ma and Mikhail Belkin. Diving into the shallows: a computational perspective on large-scale shallow learning. In Guyon et al. [5], pages 3781–3790.

[7] Tomaso A. Poggio, Kenji Kawaguchi, Qianli Liao, Brando Miranda, Lorenzo Rosasco, Xavier Boix, Jack Hidary, and Hrushikesh Mhaskar. Theory of deep learning III: explaining the non-overfitting puzzle. *CoRR*, abs/1801.00173, 2018.

[8] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: primal estimated sub-gradient solver for SVM. *Math. Program.*, 127(1):3–30, 2011.

[9] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Appendices

Here we list some more practical findings, which are not immediately relevant to the main story of this report:

A Early Stopping would have helped

During the second experiment we trained the classifiers until (almost) zero classification error since that is required for Theorem 1 to hold. However it is very interesting that in Figure 6 we can actually see the classifier using the Gaussian Kernel drops in performance after only a few iterations. Recall, that this is the different to our results on MNIST and CIFAR-10.

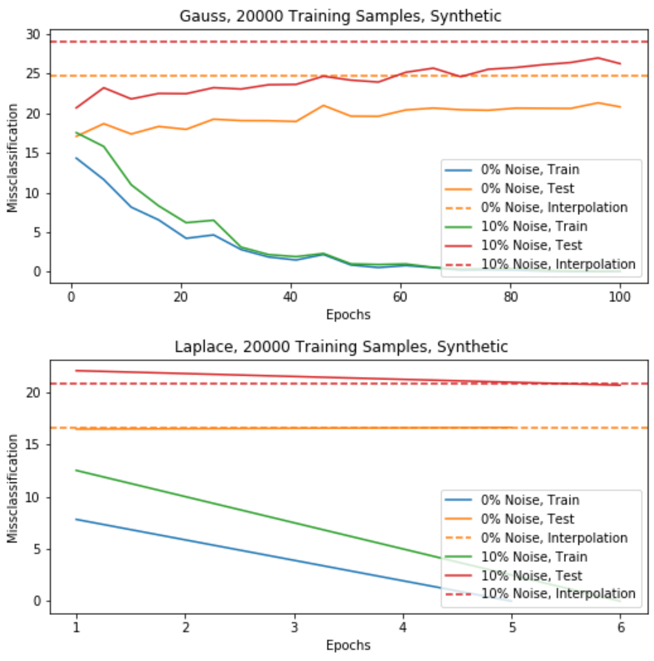


Figure 6: Classification error over epochs on Synthetic dataset.

B Computational Reach

In Figure 6 the Laplacian Kernel strongly outperforms the Gaussian in terms of test error, and that follows a general trend visible in all our experiments. Also very interesting in that regard is the high computational reach of the Laplacian Kernel (confer to Figure 7), which is similar to the findings of Zhang et al. [9] using neural networks with ReLU units. Therefore the Laplacian Kernel is not only better in terms of performance in our experiments, but also vastly superior in terms of computational effort.

Label	MNIST	CIFAR-10	Synthetic 1	Synthetic 2
Original	8	>300	1	205
Random	>300	>300	106	269

(a) Gaussian Kernel

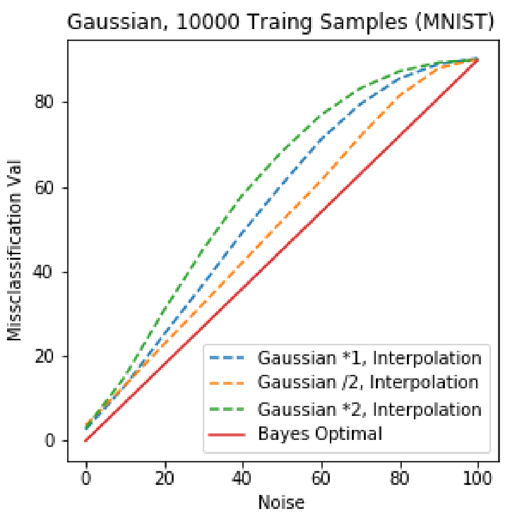
Label	MNIST	CIFAR-10	Synthetic 1	Synthetic 2
Original	3	5	1	5
Random	7	5	4	4

(b) Laplacian Kernel

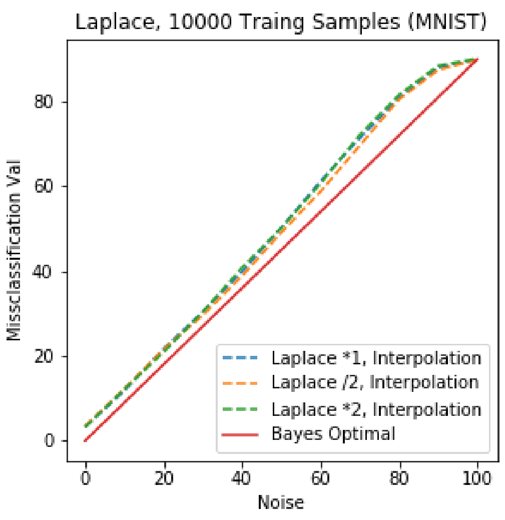
Figure 7: Number of Epochs until training reached zero classification error.

C Bandwidth

Another interesting experiment can be seen in Figure 8. Here we can see that smaller bandwidth (i.e. less smoothness in case of the Gaussian Kernel) results in better performance for higher levels of noise. This gives us the intuition that maybe the possibility for sharper ”cut outs” helps generalizing on noisy datasets.



(a) Gauss



(b) Laplace

Figure 8: Comparison between gauss, laplace.