Kernel methods and Deep Learning II

Johannes Pitz

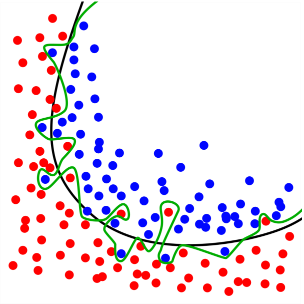
Technische Universität München

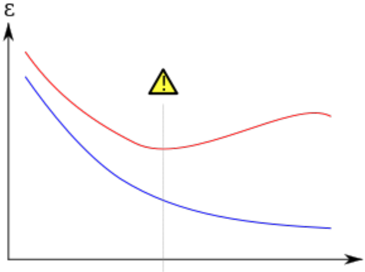
johannes.pitz@tum.de

April 24, 2020

Overview

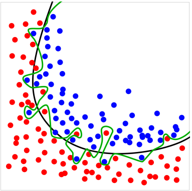
Overfitting

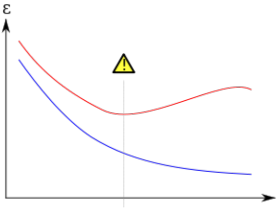




en.wikipedia.org/wiki/Overfitting

Overfitting





- Expect drop in test performance
- However, highly over parameterized setting
  $\rightarrow$ Overfit on training data, still get good generalization
- Recently a lot of attention in deep neural networks

Recent Work

- Understanding deep learning requires rethinking generalization. Zhang et al.(2016).
  - Trained state-of-the-art CNN's on random labels

  $\rightarrow$ Model family/regularization techniques don't explain the small difference between training and test performance.

- Theory of Deep Learning III: explaining the non-overfitting puzzle. Poggio et al. (2017).
  - Extend the properties of SGD for linear networks to deep non linear networks

    (1. $\exists$ optimal early stopping, 2. classification: min norm $\Longrightarrow$ maximum margin)

This paper

"To Understand Deep Learning We Need to Understand Kernel Learning"
Mikhail Belkin, Siyuan Ma, Soumik Mandal.

- Kernel Learning == a two-layer neural network
- Check if depth is needed

Reproducing Kernel Hilbert Space $\mathscr{H}$

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 RKHS $\mathscr{H}$ of functions on $\mathbb{R}^d$.

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

$$f^* = \underset{f \in \mathscr{H}, f(\mathbf{x}_i) = y_i}{\arg\min} \|f\|_{\mathscr{H}}$$

Where $\|f\|^2_{\mathscr{H}} = \langle \boldsymbol{\alpha}, \mathbf{K}\boldsymbol{\alpha} \rangle = \sum_{ij} \alpha_i \mathbf{K}_{ij} \alpha_j$,
and $\mathbf{K}_{ij} = K(x_i, x_j)$

Interpolation

By the classical representer theorem:

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

$$\text{where } \boldsymbol{\alpha}^* = \mathbf{K}^{-1}\mathbf{y}$$

Note:

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

Solving linear System is close to $O(N^3)$

Stochastic Gradient Descent

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

Minimization:

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

Use EigenPro Optimizer to speed up training

Kernel Layer

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
indexed-feat- (InputLayer)   (None, 785)               0
_________________________________________________________________
lambda_1 (Lambda)            (None, 784)               0
_________________________________________________________________
kernel_embedding_2 (KernelEm (None, 10000)             7840000
_________________________________________________________________
trainable (Dense)            (None, 10)                100000
=================================================================
Total params: 7,940,000
Trainable params: 100,000
Non-trainable params: 7,840,000
_________________________________________________________________
```

Kernels

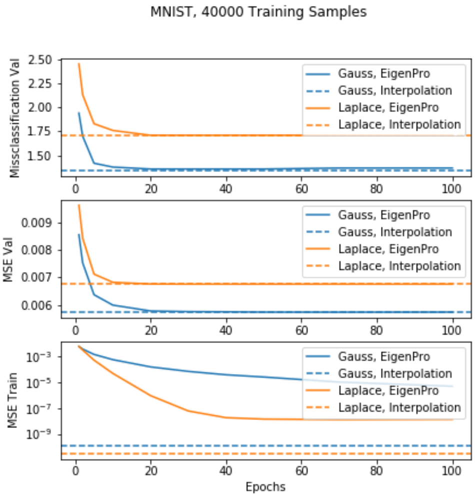
Gauss (smooth):

$$K(x,y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$$

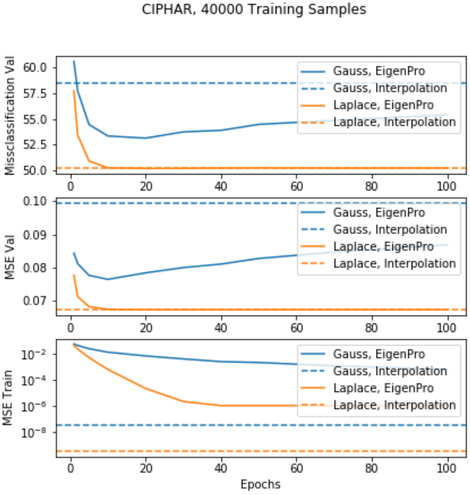
Laplace (non-smooth):

$$K(x,y) = \exp\left(-\frac{\|x-y\|}{\sigma}\right)$$

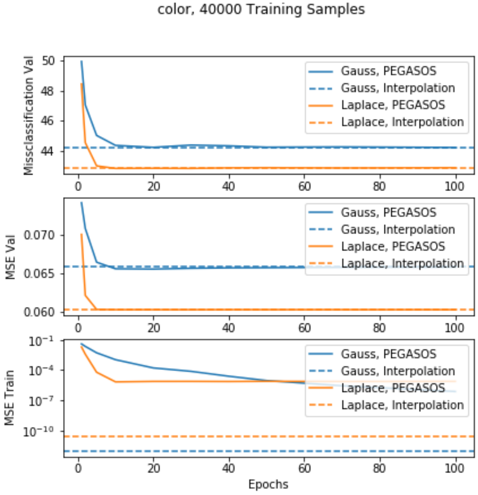
Reproduced MNIST



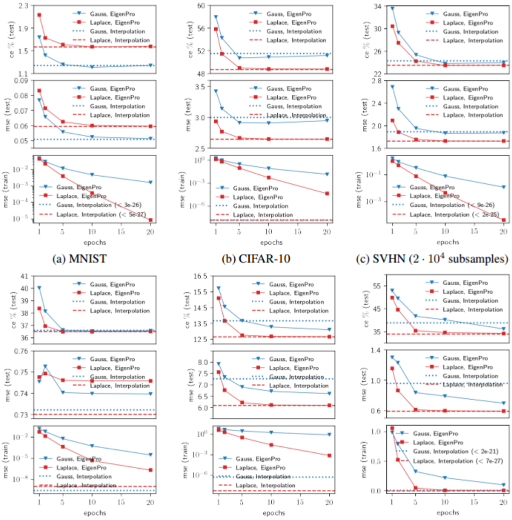
Reproduced CIFAR-10 grey-scale



Reproduced CIFAR-10 color



Belkin et al.



(a) MNIST (b) CIFAR-10 (c) SVHN ($2 \cdot 10^4$ subsamples)

(d) TIMIT ($5 \cdot 10^4$ subsamples) (e) HINT-S ($2 \cdot 10^4$ subsamples) (f) 20 Newsgroups

Recap

- Overfit on Training Data, but still reasonable generalization
  - $\rightarrow$ Apparently works also in shallow networks
  - $\rightarrow$ Not inherent to depth of deep neural nets
- Even more surprising:
  - $\rightarrow$ Authors show that any known generalization bound goes to infinity assuming some label noise

Theoretical Bound

$\Omega \subset \mathbb{R}^d$, $\mathscr{H}$ is RKHS

Probability measure P on $\Omega \times \{-1, 1\}$
$\{(\mathbf{x}_i, y_i), i = 1, ..., n\}$ sampled from P
is a labeled dataset, with non zero label noise.

$B_R = \{f \in \mathscr{H}, \|f\|_{\mathscr{H}} < R\} \subset \mathscr{H}$

$h \in \mathscr{H}$ t-overfits the data $\iff$ it achieves zero classification error, and $\forall_i y_i h(\mathbf{x}_i) > t > 0$

Interpolation requires $\forall_i y_i = h(\mathbf{x}_i)$

Theoretical Bound

Theorem

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

$$\|h\|_{\mathscr{H}} > A\mathrm{e}^{Bn^{\frac{1}{d}}}$$

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

The results also apply to other classes of smooth kernels

Theoretical Bound - Proof

$l(f(\mathbf{x}), y) = \max(t - yf(\mathbf{x}), 0)$
hinge loss with margin t

$V_\gamma(B_R)$ be fat shattering dimension of the function space B_R with the parameter γ.

By Anthony, Bartlett in *Neural network learning: Theoretical foundation*
$\exists C_1, C_2 > 0$ such that with high probability $\forall_{f \in B_R}$:

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

Theoretical Bound - Proof

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

Since y is not a deterministic function of x (label noise) $\mathbb{E}_P[l(f(\mathbf{x}), y)] > 0$

Now fix $\gamma > 0$ such that $C_1\gamma < \mathbb{E}_P[l(f(\mathbf{x}), y)]$

Suppose $h \in B_R$ t-overfits the data. Then by construction $\frac{1}{n}\sum_i l(h(\mathbf{x}_i), y_i) = 0$

Theoretical Bound - Proof

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

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

By Belkin in *Approximation beats concentration?*
$V_\gamma(B_R) < O(\log^d(\frac{R}{\gamma}))$

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

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

Theoretical Bound - Consequences

Most available bounds are of the form:

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

where $C_1, C_2, \alpha, \beta \geq 0$

$\|h\|_{\mathscr{H}} > A\mathrm{e}^{Bn^{\frac{1}{d}}}$ yields trivial bound

Probability Distribution

Proposition

Let P be a multiclass probability distribution on $\Omega \times \{1, ..., k\}$. Let P_ϵ be the same distribution, with a ϵ fraction of the labels flipped at random with equal probability for all labels.

1. The Bayes Optimal Classifier c^* for P_ϵ is the same the Bayes Optimal Classifier for P
2. The error rate is

$$P_\epsilon(c^*(x) \neq y) = \epsilon \frac{k-1}{k} + (1-\epsilon) P(c^*(x) \neq y)$$

Synthetic datasets

$$y \in \{0, 1\}$$

$$x_2 \sim \mathcal{U}(-1, 1), ..., x_{50} \sim \mathcal{U}(-1, 1)$$

Synthetic 1, Separable classes:

$$x_1 \in \begin{cases} \mathcal{N}(0, 1), & \text{if } y = 1 \\ \mathcal{N}(10, 1), & \text{otherwise} \end{cases}$$

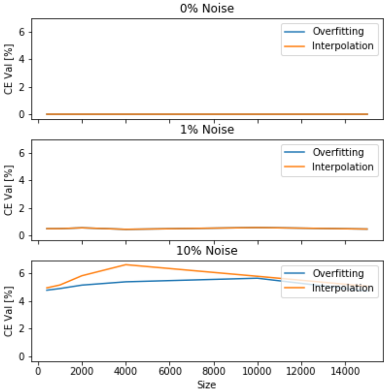
Synthetic 2, Non-Separable classes:

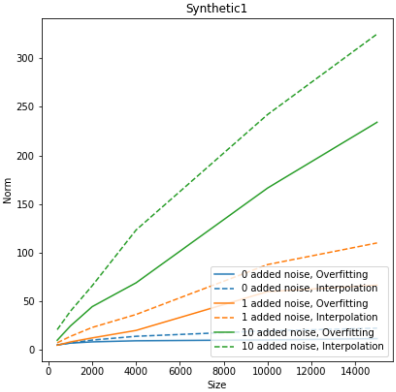
$$x_1 \in \begin{cases} \mathcal{N}(0, 1), & \text{if } y = 1 \\ \mathcal{N}(2, 1), & \text{otherwise} \end{cases}$$

Bayes Optimal Error close to 0%, around 15.9% respectively

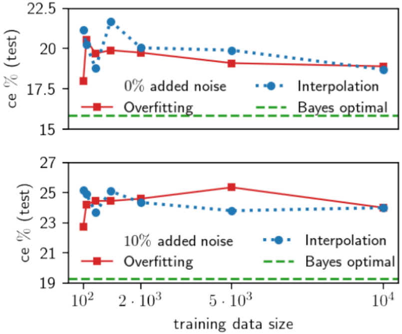
$y \in \{-1, 1\}$ in the original paper

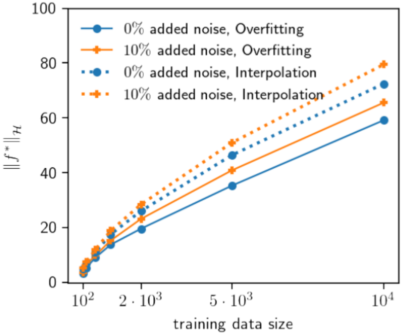
Reproduced Synthetic 1



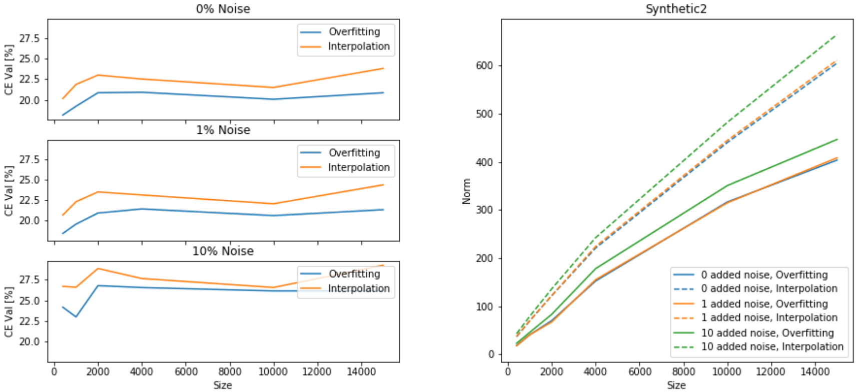


Synthetic 2

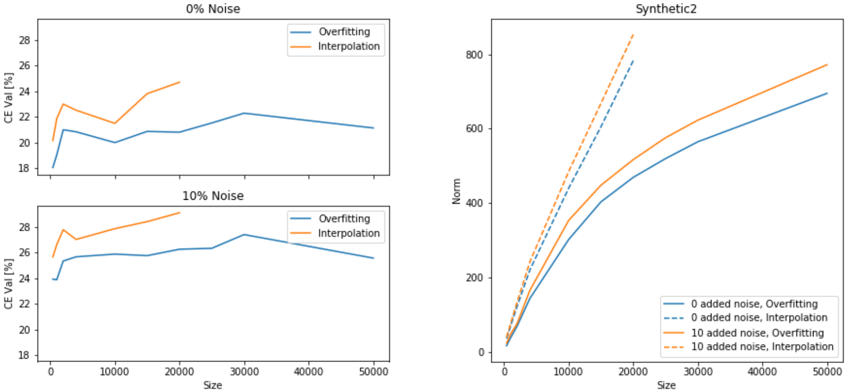

22.5
20
17.5
15
ce % (test)
0% added noise
Overfitting
Interpolation
Bayes optimal
27
25
23
21
19
ce % (test)
10% added noise
Overfitting
Interpolation
Bayes optimal
10^2
$2 \cdot 10^3$
$5 \cdot 10^3$
10^4
training data size


100
80
60
40
20
0
$\|f^*\|_{\mathcal{H}}$
0% added noise, Overfitting
10% added noise, Overfitting
0% added noise, Interpolation
10% added noise, Interpolation
10^2
$2 \cdot 10^3$
$5 \cdot 10^3$
10^4
training data size

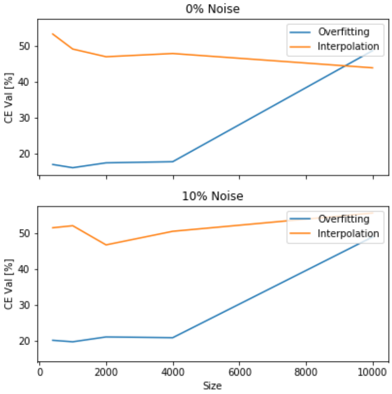
Reproduced Synthetic 2

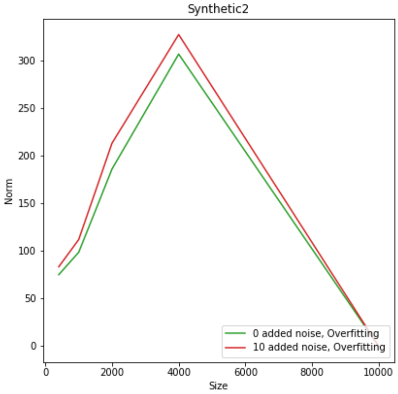


Extended Synthetic 2

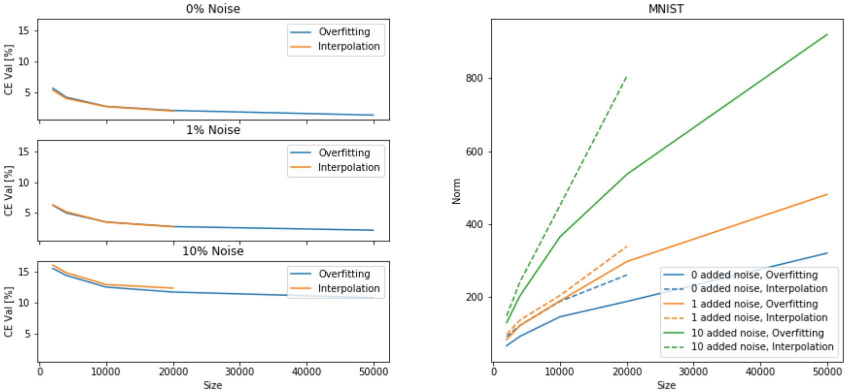


d = 4; Synthetic 2

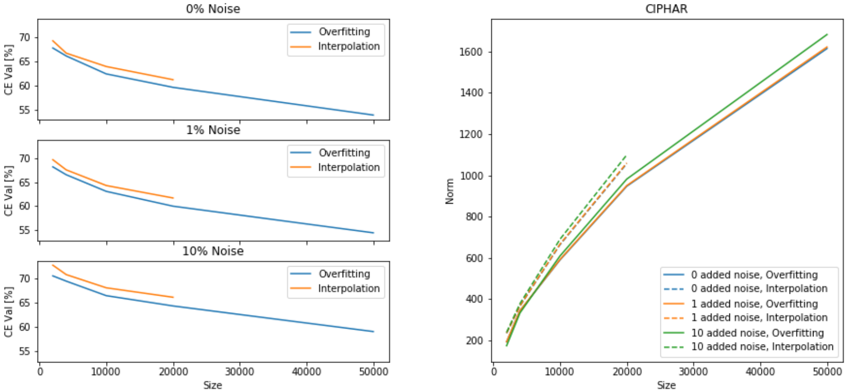




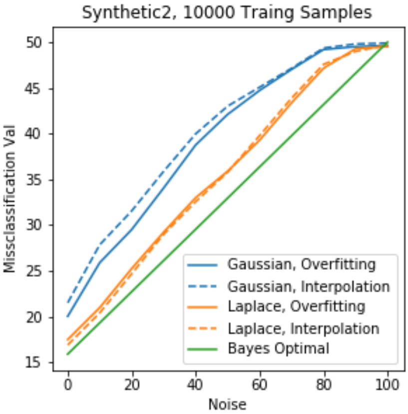
Reproduced MNIST

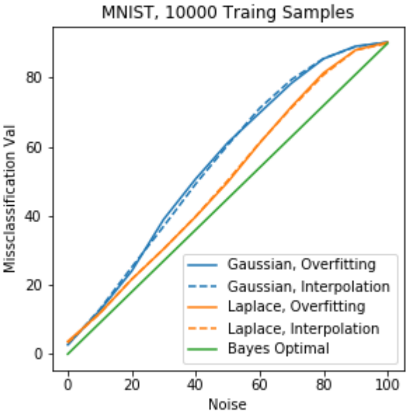


Tried also CIFAR-10

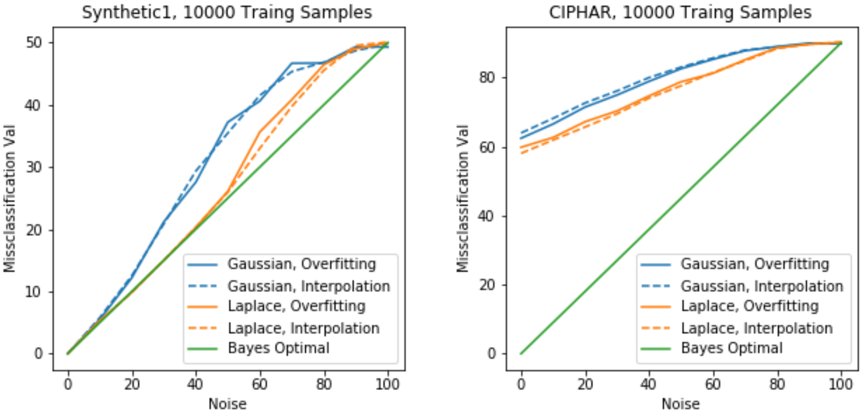


Reproduced


Synthetic2, 10000 Traing Samples
Missclassification Val
50
45
40
35
30
25
20
15
0
20
40
60
80
100
Noise
Gaussian, Overfitting
Gaussian, Interpolation
Laplace, Overfitting
Laplace, Interpolation
Bayes Optimal


MNIST, 10000 Traing Samples
Missclassification Val
80
60
40
20
0
0
20
40
60
80
100
Noise
Gaussian, Overfitting
Gaussian, Interpolation
Laplace, Overfitting
Laplace, Interpolation
Bayes Optimal

Tried also


Synthetic1, 10000 Traing Samples
Missclassification Val
Noise
0
10
20
30
40
50
0
20
40
60
80
100
Gaussian, Overfitting
Gaussian, Interpolation
Laplace, Overfitting
Laplace, Interpolation
Bayes Optimal
CIPHAR, 10000 Traing Samples
Missclassification Val
Noise
0
20
40
60
80
0
20
40
60
80
100
Gaussian, Overfitting
Gaussian, Interpolation
Laplace, Overfitting
Laplace, Interpolation
Bayes Optimal

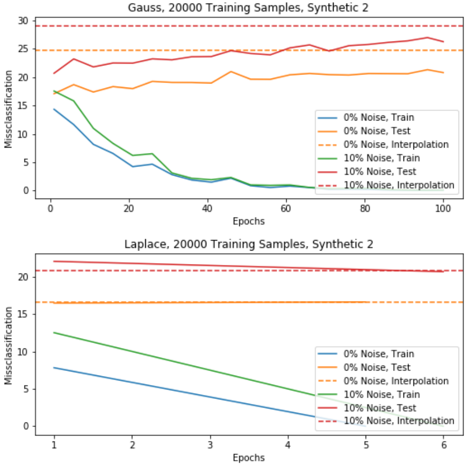
Recap

- Norm increases with training data size
- Still yields reasonable generalization on test data
- Even when noise is close to 1

(Consistently) better than any known bound can guarantee

$\Rightarrow$ Need to find new bounds!

Early Stopping would have helped



Computational Reach

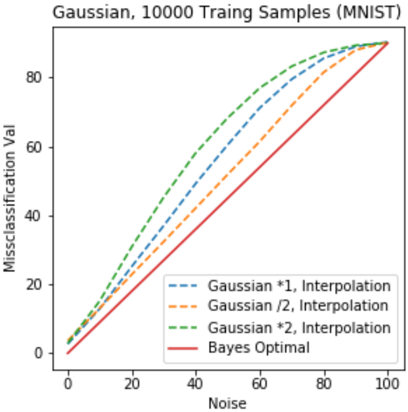
Gaussian Kernel:

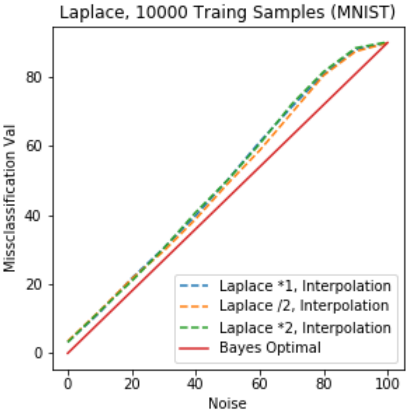
Label	MNIST	CIFAR-10	Synthetic 1	Synthetic 2
Original	8	$>$300	1	205
Random	$>$300	$>$300	106	269

Laplaceian Kernel:

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

Bandwidth


Gaussian, 10000 Traing Samples (MNIST)
Misclassification Val
Noise
0
20
40
60
80
100
Gaussian *1, Interpolation
Gaussian /2, Interpolation
Gaussian *2, Interpolation
Bayes Optimal


Laplace, 10000 Traing Samples (MNIST)
Misclassification Val
Noise
0
20
40
60
80
100
Laplace *1, Interpolation
Laplace /2, Interpolation
Laplace *2, Interpolation
Bayes Optimal

Conclusion

Practical Considerations:

- Laplace Kernel ($\sim$ relu) often outperforms Gaussian
- Model scales with Training Data size

We have seen:

- Kernel Learning $==$ shallow Neural Networks
- Interesting theoretical results
- Surprisings experimental results

$\rightarrow$ Hope for new bounds