Neural Kernels without Tangents: Appendix

June 27, 2020

Appendix A Nonparametric prediction with kernels

We proceed with kernel classification as follows. Let C be the total number of classes. Let $\{x_1 ... x_N\}$ be N training examples in d dimensions. Let $\{y_1 ... y_N\}$ be N one-hot encoded training labels. We use v_{im} to denote the m^{th} entry of the vector v_i. For a choice of kernel function $k(x, y)$, loss function $\mathcal{L}$, and regularization value λ, we solve the following optimization problem:

$$\underset{\alpha}{\text{minimize}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\big(\sum_{j=1}^{N} \boldsymbol{K}_{j} . \alpha, y_i \big) + \lambda \operatorname{Tr}(\alpha^T K \alpha) \tag{1}$$

where $\boldsymbol{K}$ is the matrix of kernel evaluations on the data: $K_{ij} = k(x_i, x_j)$. The prediction for an example x_{test} is:

$$\underset{c}{\operatorname{argmax}} \sum_{i=1}^{N} \alpha_{ic} k(x_{test}, x_i) \tag{2}$$

If not otherwise specificed, we use the squared error loss, $\mathcal{L}(\hat{y}, y) = \|\hat{y} - y\|^2$, for our experiments. In this case, α in (1) is given by

$$\alpha = (\boldsymbol{K} + \lambda \boldsymbol{I})^{-1} \boldsymbol{Y}$$

where $\boldsymbol{Y}$ is the $N \times C$ matrix of all one-hot-encodings of the labels. We allow for the value of $\lambda = 0$ in our experiments, and oftentimes this value produces the lowest test error.

Appendix B LOO Tilting and ZCA Augmentation

Two additional techniques were used for the Cifar-10 experiments for an additional 0.5% performance improvement in test accuracy.

B.1 ZCA Augmentation

As mentioned in Section **??**, incorporating augmentation directly is difficult for kernel methods. To capture a small fraction of the benefit of the augmentation in the preprocessing method itself, we first augment the data 20 times using the random augment method proposed in Cubuk et al. [2].

We then learn the ZCA preprocessing matrix by computing the eigendecomposition of the augmented training data as described in [4]. We then use just the portion of the preprocessed data matrix corresponding to the regular unaugmented training data (or corresponding to the unaugmented training data and its horizontal flips) to compute the kernel matrix.

We find this technique offered a small performance boost of around 0.2% across CIFAR-10 and CIFAR-10.

B.2 Leave-One-Out Tilting

We additionally found a minor improvement in prediction by averaging the predictions from the true labels and from the labels imputed by leave-one-out prediction. With $\boldsymbol{K}$ and $\boldsymbol{Y}$ defined as they are in section A, let $\boldsymbol{Q}$ be $(\boldsymbol{K} + \lambda \boldsymbol{I})^{-1}$ and $\boldsymbol{\alpha}$ be $(\boldsymbol{K} + \lambda \boldsymbol{I})^{-1}\boldsymbol{Y}$. $\boldsymbol{\alpha}$ are the coefficients for standard ridge regression.

Let $\boldsymbol{Y_{loo}}$ be a $N \times C$ matrix of leave-one-out predictions. Here, the ith row of $\boldsymbol{Y}_{loo}$ is the output of ridge regression where we predict example i using every element in the entire training set *except* example i. For kernel ridge regression, we can actually compute the leave-one-out prediction matrix in closed form:

$$Y_{loo_{ic}} = Y_{ic} - \frac{Q_{ii}}{\alpha_{ic}}$$

Our titled prediction uses an affine combination of the true labels $\boldsymbol{Y}$ and the imputed leave-one-out predictions $\boldsymbol{Y}_{loo}$:

$$\boldsymbol{\alpha}_{loo} = (\boldsymbol{K} + \lambda \boldsymbol{I})^{-1}(\boldsymbol{Y} - t\boldsymbol{Y}_{loo})$$

Where t is chosen to maximize test accuracy on CIFAR-10. We empirically find the optimal value of t to be 0.3. Though we do not yet have a theoretical justification for this method, we found that this solution never reduced test error, and always performed well on the test set CIFAR-10.1. For our best model, we found this technique offered a modest performance boost of around 0.3% on both CIFAR-10 and CIFAR-10.1. We leave an analysis of the efficacy of this technique to future work.

Appendix C Supplementary proof

For completeness, we present the proof details for section 3.3 using results from [3].

We aim to prove the following equality:

$$\mathbb{E}\Big[\sum_{c=1}^{D_4} \Psi(\boldsymbol{U})[i,j,k,c]\Psi(\boldsymbol{U})[\ell,m,n,c]\Big] = k_{relu}\Big(c_w\big(k_0(\boldsymbol{U})\big)\Big)[i,j,k,\ell,m,n] \tag{3}$$

where Ψ, $\boldsymbol{U}$ and k_0 are as defined in the main text.

Daniely et al. [3] (Section 4.2) presents the concepts of dual activation and kernel:

$$\hat{\sigma}(\rho) = \mathbb{E}_{(X,Y)\sim N_\rho}[\sigma(X)\sigma(Y)]$$

where we denote by N_ρ the multivariate Gaussian distribution with mean 0 and covariance matrix $\begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$.

In our case, we have the *relu* activation $\sigma(\cdot) = \max(x, 0)$, whose dual activation function takes the form $\hat{\sigma}(\rho) = \frac{\sqrt{1-\rho^2}+\rho(\pi-\cos^{-1}(\rho))}{\pi}$ [3].

Now we show how the above result translates to equation **??**. Recall that for a convolutional layer followed by a ReLU layer, we have tensors $\boldsymbol{U}$ with shape $N \times D_1 \times D_2 \times D_3$, and $\boldsymbol{W}$ with shape $(2w+1) \times (2w+1) \times D_3 \times D_4$. To ease the notation, denote $\boldsymbol{Z}$ as the tensor $\boldsymbol{W} * \boldsymbol{U}$ with shape $N \times D_1 \times D_2 \times D_4$. We begin with the LHS of equation **??**:

$$\begin{aligned}&\mathbb{E}\Big[\sum_{c=1}^{D_4} \Psi(\boldsymbol{U})[i,j,k,c]\Psi(\boldsymbol{U})[\ell,m,n,c]\Big] \\ =&\mathbb{E}\Big[\sigma\Big(\sqrt{D_4}(\boldsymbol{Z}[i,j,k,1])\Big)\sigma\Big(\sqrt{D_4}(\boldsymbol{Z}[l,m,n,1])\Big)\Big]\end{aligned}$$

Let $X = \sqrt{D_4}(\boldsymbol{Z}[i,j,k,1])$, $Y = \sqrt{D_4}(\boldsymbol{Z}[l,m,n,1])$, and choose entries of W to be independent and identically distributed Gaussian random variables with mean 0 and variance $\frac{1}{D_4}$. With normalization on every patch, we have that (X, Y) follow the multivariate Gaussian distribution with mean 0 and covariance matrix $\begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$, where

$$\rho = \sum_{\delta=-w}^{w} \sum_{\Delta=-w}^{w} \sum_{d=1}^{D_3} U[i, j+\delta, k+\Delta, d]U[l, m+\delta, n+\Delta, d]$$

Following Daniely et al. [3], we have:

$$\begin{aligned}&\mathbb{E}\Big[\sum_{c=1}^{D_4} \Psi(\boldsymbol{U})[i,j,k,c]\Psi(\boldsymbol{U})[\ell,m,n,c]\Big] \\ =&\frac{\sqrt{1-\rho^2}+\rho(\pi-\cos^{-1}(\rho))}{\pi}\end{aligned} \tag{4}$$

Now we show that the RHS of equation **??** is indeed $\frac{\sqrt{1-\rho^2}+\rho(\pi-\cos^{-1}(\rho))}{\pi}$.

As defined in Subsection **??**,

$$k_0(\boldsymbol{U})[i,j,k,l,m,n] = \sum_{d=1}^{D_3} U[i,j,k,d]U[l,m,n,d]$$

.

Let $\boldsymbol{C}$ be the tensor $c_w\Big(k_0(\boldsymbol{U})\Big)$,

$$\begin{aligned} &C[i,j,k,l,m,n] \\ &= \sum_{\delta=-w}^{w} \sum_{\Delta=-w}^{w} \sum_{d=1}^{D_3} U[i,j\pm\delta,k\pm\Delta,d]U[l,m\pm\delta,n\pm\Delta,d] \end{aligned}$$

With normalization for every patch, $\sqrt{C[i,j,k,i,j,k]}=1$, and

$$\begin{aligned} & k_{relu}\Big(c_w\Big(k_0(\boldsymbol{U})\Big)\Big) \\ = \; & \frac{1}{\pi}\Big(\sqrt{1-\rho^2}+\rho(\pi-\cos^{-1}(\rho))\Big) \end{aligned} \tag{5}$$

Combining (4) and (5) completes the proof.

Appendix D Neural Network Parameters

The parameters used to train neural networks for the experiments in this paper are as follows:

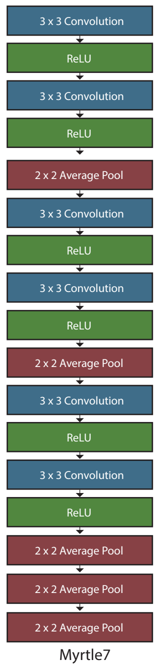
For Myrtle5 on MNIST, we used a width of 1,024 filters for all layers and trained for 20 epochs using MSE loss and Adam as the optimizer with a learning rate of 0.001, without weight decay, and without any form of data preprocessing. For the Myrtle5 Kernel on MNIST we used a regularization value (λ) of 1e-4.

For CIFAR-10, all experiments are trained using MSE loss and SGD with Nesterov momentum, setting weight decay to 0.0005, momentum to 0.9, and minibatch size to 128. All experiments using Myrtle5 used 1,024 filters for all layers and trained for 60 epochs at half-precision with an initial learning rate of 0.1, which is decayed by 0.1 at 15, 30, and 45 epochs. Myrtle7, Myrtle10 without augmentation, and Myrtle10 with flips used 1,024 filters and are trained for 200 epochs with an initial learning rate of 0.05, which is decayed by 0.1 at 80, 120, and 160 epochs. Myrtle10 with flips, cutout, and random crops used 2,048 filters and is trained for 400 epochs with an initial learning rate of 0.1, which is decayed by 0.1 at 80, 160, 240, and 320 epochs. For all CIFAR-10 kernel experiments we used a regularization value (λ) of 0.

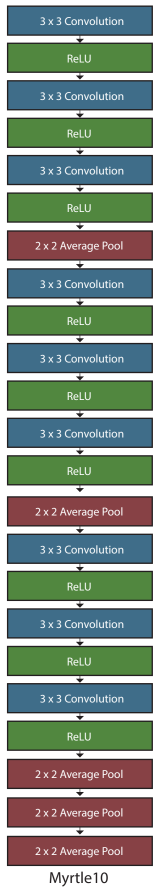
For CIFAR-100, all experiments use a width of 2,048 filters for all layers and are trained for 200 epochs using cross entropy loss and SGD with Nesterov momentum, setting weight decay to 0.0005, momentum to 0.9, and minibatch size to 128. The learning rate is decayed by 0.2 at 60, 120, and 160 epochs. The initial learning rate is set to 0.1 for both experiments with batch normalization, 0.05 for Myrtle10 CNN with flips, and 0.01 for Myrtle10 CNN without augmentation. For all CIFAR-100 kernel experiments we used a regularization value (λ) of 1e-4.

Appendix E Neural Network Architectures

In Figure 1 we illustrate the two “deeper" Myrtle architectures used. The architectures are similar to the 5 layer variant illustrated in main text, except with more convolution and nonlinearity layers.



(a)



(b)

Figure 1: a) 7 layer b) 10 layer variants of the Myrtle architectures

Appendix F UCI experiments details

In this section, we provide supplementary details for the experiments on the UCI datasets.

Table 2 provides the accuracy values (with Clopper-Pearson confidence intervals) for both the NTK and the tuned Gaussian kernel on each of the 90 small UCI datasets. For the Gaussian kernel we swept the kernel bandwidth γ value from $\nu * 2^{-19}$ to $\nu * 2^{20}$ in log space, where ν was chosen heuristically to be the *median* ℓ_2 distance between data points. We swept the SVM parameter C from 2^{-19} to 2^{20} in log space.

For the NTK we tuned the kernel exactly using the protocol from [1].

Classifier	Friedman Rank	Average Accuracy (%)	P90 (%)	P95 (%)	PMA (%)
SVM NTK (tuned on 4 eval folds)	14.3	83.2 ± 13.5	96.7	83.3	97.3 ± 3.8
SVM Gaussian (tuned on 4 eval folds)	11.6	83.4 ± 13.4	95.6	83.3	97.5 ± 3.7
SVM NTK (tuned on 1 eval fold)	29.0	82.0 ± 14.0	88.9	72.2	95.6 ± 5.2
RF (tuned on 1 eval fold)	32.4	81.6 ± 13.8	85.6	65.6	95.1 ± 5.3
SVM Gaussian (tuned on 1 eval fold)	35.1	81.0 ± 15.0	85.6	72.2	94.4 ± 8.2
SVM Polynomial (tuned on 1 eval fold)	36.6	78.2 ± 20.2	81.6	63.2	94.3 ± 6.0

Table 1: Results on 90 UCI datasets. Friedman rank reports the average (over datasets) of the accuracy ranking (among classifiers). Average accuracy is reported ± standard deviation. P90/P95 denotes the percentage of datasets on which a classifier achieves at least 90%/95% of the maximum accuracy. PMA denotes the average (over datasets) percentage of the maximum accuracy ± standard deviation.

Appendix G Subsampled CIFAR-10 experiments details

Subsets of CIFAR-10 were selected uniformly at random without replacement, and each experiment was repeated in 20 independent trials (over which we report standard deviations). This procedure, and the sizes of training sets we consider, match the setup in [1]. Table 4 compares the performance (on all 10,000 test examples) of the CNTK from [1] with that of our Myrtle5 kernel and CNN (with and without batch normalization), our Myrtle10-Gaussian kernel, and a baseline linear classifier. Each linear model used a regularization parameter λ tuned on a log scale between 10^{-4} and 10^{6}. The optimal values for λ from top to bottom were: $10^2, 10^{-2}, 10^2, 10^3, 10^3, 10^3, 10^5, 10^4$. The Myrtle10-Gaussian kernel is our highest-performing unaugmented kernel on the full CIFAR-10 dataset; here we confirm that it retains high performance in the small-data regime.

References

[1] Sanjeev Arora, Simon S. Du, Zhiyuan Li, Ruslan Salakhutdinov, Ruosong Wang, and Dingli Yu. Harnessing the power of infinitely wide deep nets on small-data tasks. In *International Conference on Learning Representations*, 2020.

[2] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical data augmentation with no separate search. *arXiv preprint arXiv:1909.13719*, 2019.

[3] Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances in Neural Information Processing Systems*, 2016.

[4] Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron C. Courville, and Yoshua Bengio. Maxout networks. In *International Conference on Machine Learning*, 2013.

Table 2: Results on 90 UCI datasets, with Clopper-Pearson confidence intervals.

Dataset	NTK Accuracy	Gaussian Kernel Accuracy
abalone	66.04 [63.97, 68.08]	66.71 [64.65, 68.74]
acute-inflammation	100.0 [94.04, 100.0]	100.0 [94.04, 100.0]
acute-nephritis	100.0 [94.04, 100.0]	100.0 [94.04, 100.0]
arrhythmia	72.35 [66.02, 78.07]	72.79 [66.49, 78.48]
balance-scale	99.04 [97.22, 99.8]	99.52 [97.96, 99.97]
balloons	100.0 [63.06, 100.0]	100.0 [63.06, 100.0]
bank	89.93 [88.62, 91.14]	90.04 [88.74, 91.25]
blood	80.48 [76.1, 84.38]	79.01 [74.53, 83.03]
breast-cancer	75.7 [67.8, 82.5]	75.35 [67.42, 82.19]
breast-cancer-wisc	97.86 [95.73, 99.1]	97.57 [95.36, 98.91]
breast-cancer-wisc-diag	97.54 [94.99, 99.0]	97.54 [94.99, 99.0]
breast-cancer-wisc-prog	84.18 [75.43, 90.77]	83.16 [74.26, 89.97]
breast-tissue	75.0 [61.05, 85.97]	74.04 [60.01, 85.2]
car	98.84 [97.88, 99.44]	99.31 [98.49, 99.74]
cardiotocography-10clases	86.53 [84.33, 88.53]	85.55 [83.29, 87.61]
cardiotocography-3clases	93.6 [91.95, 94.99]	92.98 [91.28, 94.45]
chess-krvkp	99.25 [98.69, 99.61]	99.22 [98.65, 99.59]
congressional-voting	63.3 [56.53, 69.71]	63.3 [56.53, 69.71]
conn-bench-sonar-mines-rocks	86.54 [78.45, 92.44]	87.98 [80.14, 93.54]
contrac	54.89 [51.21, 58.53]	55.71 [52.03, 59.33]
credit-approval	87.94 [84.02, 91.18]	87.65 [83.7, 90.93]
cylinder-bands	80.66 [75.29, 85.32]	81.05 [75.71, 85.67]
dermatology	98.08 [94.86, 99.53]	98.35 [95.26, 99.66]
echocardiogram	86.36 [75.69, 93.57]	86.36 [75.69, 93.57]
ecoli	87.5 [81.53, 92.09]	88.1 [82.21, 92.57]
energy-y1	96.22 [93.8, 97.9]	96.48 [94.12, 98.09]
energy-y2	89.97 [86.52, 92.79]	91.41 [88.14, 94.01]
fertility	89.0 [76.93, 96.08]	89.0 [76.93, 96.08]
flags	51.56 [41.14, 61.89]	54.69 [44.2, 64.88]
glass	70.75 [61.13, 79.19]	72.17 [62.62, 80.44]
haberman-survival	73.68 [65.93, 80.49]	75.66 [68.04, 82.25]
heart-cleveland	59.87 [51.62, 67.73]	60.2 [51.95, 68.04]
heart-hungarian	87.67 [81.22, 92.53]	86.99 [80.43, 91.98]
heart-switzerland	49.19 [36.26, 62.21]	50.0 [37.02, 62.98]
heart-va	40.0 [30.33, 50.28]	41.5 [31.73, 51.79]
hepatitis	83.97 [73.93, 91.31]	86.54 [76.92, 93.21]
ilpd-indian-liver	74.32 [68.9, 79.23]	73.29 [67.82, 78.27]
ionosphere	95.74 [91.61, 98.2]	95.74 [91.61, 98.2]
iris	98.65 [92.7, 99.97]	98.65 [92.7, 99.97]
led-display	74.6 [70.55, 78.36]	75.3 [71.28, 79.02]
lenses	87.5 [56.38, 99.09]	87.5 [56.38, 99.09]
libras	86.67 [80.81, 91.27]	85.83 [79.87, 90.57]
low-res-spect	93.8 [90.19, 96.38]	94.55 [91.1, 96.95]
lung-cancer	65.62 [38.34, 86.94]	62.5 [35.43, 84.8]
lymphography	89.19 [79.8, 95.22]	89.19 [79.8, 95.22]
mammographic	82.29 [78.58, 85.6]	83.75 [80.14, 86.94]
molec-biol-promoter	91.35 [80.2, 97.35]	91.35 [80.2, 97.35]
molec-biol-splice	87.61 [85.89, 89.19]	87.7 [85.99, 89.28]
musk-1	90.97 [86.58, 94.29]	91.39 [87.07, 94.62]

Table 3: Results on 90 UCI datasets, with Clopper-Pearson confidence intervals: continued.

Dataset	NTK Accuracy	Gaussian Kernel Accuracy
oocytes-merluccius-nucleus-4d	84.22 [80.76, 87.27]	85.78 [82.45, 88.7]
oocytes-merluccius-states-2f	93.63 [91.14, 95.59]	94.12 [91.71, 96.0]
oocytes-trisopterus-nucleus-2f	86.62 [83.15, 89.61]	87.28 [83.87, 90.2]
oocytes-trisopterus-states-5b	94.41 [91.88, 96.33]	95.18 [92.79, 96.95]
ozone	97.2 [96.14, 98.04]	97.36 [96.32, 98.17]
parkinsons	93.37 [86.49, 97.4]	93.37 [86.49, 97.4]
pima	76.69 [72.14, 80.83]	77.34 [72.82, 81.44]
pittsburg-bridges-MATERIAL	92.31 [81.46, 97.86]	94.23 [84.05, 98.79]
pittsburg-bridges-REL-L	75.0 [61.05, 85.97]	75.0 [61.05, 85.97]
pittsburg-bridges-SPAN	73.91 [58.87, 85.73]	73.91 [58.87, 85.73]
pittsburg-bridges-T-OR-D	89.0 [76.93, 96.08]	90.0 [78.19, 96.67]
pittsburg-bridges-TYPE	72.12 [57.95, 83.65]	68.27 [53.89, 80.48]
planning	71.67 [61.19, 80.67]	73.89 [63.56, 82.58]
plant-margin	84.38 [81.67, 86.82]	84.5 [81.8, 86.94]
plant-shape	74.0 [70.81, 77.01]	73.0 [69.78, 76.05]
plant-texture	85.44 [82.8, 87.81]	85.25 [82.6, 87.64]
post-operative	72.73 [57.21, 85.04]	72.73 [57.21, 85.04]
primary-tumor	53.96 [46.02, 61.76]	55.49 [47.54, 63.24]
seeds	96.15 [90.44, 98.94]	96.15 [90.44, 98.94]
semeion	95.73 [94.08, 97.02]	95.85 [94.23, 97.13]
spambase	94.93 [93.96, 95.79]	94.02 [92.97, 94.96]
statlog-australian-credit	68.31 [63.11, 73.2]	68.31 [63.11, 73.2]
statlog-german-credit	78.4 [74.53, 81.93]	78.7 [74.85, 82.21]
statlog-heart	88.06 [81.33, 93.02]	89.18 [82.65, 93.88]
statlog-image	98.09 [97.13, 98.8]	97.66 [96.61, 98.45]
statlog-vehicle	84.72 [80.92, 88.01]	85.31 [81.56, 88.55]
steel-plates	78.04 [75.3, 80.61]	77.53 [74.77, 80.12]
synthetic-control	99.5 [97.88, 99.96]	99.67 [98.16, 99.99]
teaching	60.53 [48.65, 71.56]	61.18 [49.31, 72.16]
tic-tac-toe	99.79 [98.84, 99.99]	100.0 [99.23, 100.0]
titanic	78.95 [76.42, 81.33]	78.95 [76.42, 81.33]
trains	87.5 [28.38, 99.99]	87.5 [28.38, 99.99]
vertebral-column-2clases	87.34 [81.03, 92.15]	87.66 [81.41, 92.41]
vertebral-column-3clases	84.74 [78.07, 90.02]	85.06 [78.44, 90.29]
waveform	87.16 [85.79, 88.45]	86.96 [85.58, 88.26]
waveform-noise	86.8 [85.41, 88.1]	86.8 [85.41, 88.1]
wine	98.86 [93.83, 99.97]	98.3 [92.91, 99.88]
wine-quality-red	67.38 [64.0, 70.62]	65.31 [61.9, 68.61]
wine-quality-white	67.59 [65.69, 69.44]	66.14 [64.22, 68.01]
yeast	61.05 [57.44, 64.58]	61.32 [57.71, 64.84]
zoo	100.0 [92.89, 100.0]	99.0 [91.03, 100.0]

Training Size	CNTK	Myrtle5 CNN	Myrtle5 CNN + BN	Myrtle5 Kernel	Myrtle10-G Kernel	Linear
10	15.33 ± 2.43	11.29 ± 1.48	19.60 ± 3.32	17.22 ± 2.95	19.15 ± 1.94	12.94 ± 0.74
20	18.79 ± 2.13	11.83 ± 1.34	22.82 ± 2.56	22.16 ± 1.69	21.65 ± 2.97	13.54 ± 0.69
40	21.34 ± 1.91	12.16 ± 2.20	27.53 ± 1.61	26.74 ± 1.56	27.20 ± 1.90	14.66 ± 0.60
80	25.48 ± 1.91	18.96 ± 2.04	33.58 ± 1.22	32.56 ± 1.12	34.22 ± 1.08	15.54 ± 0.61
160	30.48 ± 1.17	20.36 ± 1.68	39.96 ± 1.44	38.61 ± 1.06	41.89 ± 1.34	17.15 ± 0.64
320	36.57 ± 0.88	34.79 ± 4.60	46.96 ± 1.29	46.03 ± 0.82	50.06 ± 1.06	19.18 ± 0.71
640	42.63 ± 0.68	43.36 ± 3.80	56.03 ± 0.80	53.45 ± 0.80	57.60 ± 0.48	22.30 ± 0.57
1280	48.86 ± 0.68	53.27 ± 1.55	61.94 ± 0.74	60.46 ± 0.58	64.40 ± 0.48	25.64 ± 0.61

Table 4: Accuracy results (%) on random subsets of CIFAR-10, with standard deviations over 20 resamplings. Myrtle10-G denotes the Myrtle10-Gaussian kernel, our best-performing kernel on the full CIFAR-10 dataset which retains its high performance in the small-data regime. The shallower Myrtle5 CNN trained with batch normalization has similar performance to the corresponding compositional kernel, both of which outperform the CNTK and the Myrtle5 CNN trained without batch normalization.