
COMPARING SPECTRAL BIAS AND ROBUSTNESS FOR TWO-LAYER NEURAL NETWORKS: SGD VS ADAPTIVE RANDOM FOURIER FEATURES

Aku Kammonen^{*1}, Lisi Liang^{†2}, Anamika Pandey^{‡2}, and Raúl Tempone^{§1,2}

¹KAUST, Saudi Arabia
²RWTH Aachen, Germany

ABSTRACT

We present experimental results highlighting two key differences resulting from the choice of training algorithm for two-layer neural networks. The spectral bias of neural networks is well known, while the spectral bias dependence on the choice of training algorithm is less studied. Our experiments demonstrate that an adaptive random Fourier features algorithm (ARFF) can yield a spectral bias closer to zero compared to the stochastic gradient descent optimizer (SGD). Additionally, we train two identically structured classifiers, employing SGD and ARFF, to the same accuracy levels and empirically assess their robustness against adversarial noise attacks.

Keywords machine learning · spectral bias · adversarial attack · random features · neural network · SGD · MNIST · Metropolis sampling

1 Introduction

Consider the data $\{x_n, y_n\} \in \mathbb{R}^d \times \mathbb{R}$, $n = 1, \dots, N$ i.i.d. sampled from an unknown probability distribution with probability density function $\tilde{\rho}(x, y)$. We assume that there exists an unknown underlying function $f: \mathbb{R}^d \rightarrow \mathbb{R}$ such that $f(x_n) = y_n + \epsilon_n$, $n = 1, \dots, N$ where $\mathbb{E}[\epsilon_n] = 0$ and $\mathbb{E}[\epsilon_n^2] < \infty$. We address the supervised learning problem of fitting a two-layer neural network $\beta(\mathbf{x}) = \sum_{k=1}^K \hat{\beta}_k e^{i\omega_k \cdot \mathbf{x}}$ to the data $\{x_n, y_n\}_{n=1}^N$ by numerically approximating a solution to the minimization problem

$$\min_{(\omega, \hat{\beta}) \in \mathbb{R}^{Kd} \times \mathbb{C}^K} \left\{ \mathbb{E}_{\tilde{\rho}} [|y - \beta(x)|^2] + \lambda \sum_{k=1}^K |\hat{\beta}_k|^2 \right\} \quad (1)$$

where $\lambda \in \mathbb{R}$ is a Tikhonov regularization parameter. Note that, in practice, the data density $\tilde{\rho}$ is unknown, therefore we consider the empirical minimization problem instead. Considering ω_k as i.i.d. samples from an unknown distribution with probability density function $p(\omega_k)$ and by following [3] an $\mathcal{O}(1/K)$ error bound to Problem (1) is derived in [13]. However, in the work [13], only ω_k is treated as a random variable, while $\hat{\beta}_k$ is determined for a sampled ω_k without accounting for any inherent randomness in $\hat{\beta}_k$. Further, the constant in the $\mathcal{O}(1/K)$ error bound is shown to be minimized by sampling the weights ω_k from $p_*(\omega) := |\hat{f}(\omega)| / \|\hat{f}\|_{L^1(\mathbb{R}^d)}$ where \hat{f} denotes the Fourier transform of f . Given only data $\{x_n, y_n\}_{n=1}^N$ means that \hat{f} typically is not accessible. The goal of the adaptive random Fourier features with Metropolis sampling algorithm (ARFF), presented in and denoted as Algorithm 1 in [13], is to approximately sample ω from p_* , given only data. Applications of the ARFF algorithm include wind field reconstruction, see [15], and pre-training of deep residual neural networks, see [14].

*akujaakko.kammonen@kaust.edu.sa

†lisi.liang@rwth-aachen.de

‡pandey@uq.rwth-aachen.de

§raul.tempone@kaust.edu.sa

For simplicity in our numerical experiments, we employ the cosine activation function. This choice necessitates the introduction of a bias term b_k for each $k = 1, \dots, K$ in the argument of the cosine function. The purpose of this bias is to offset the contribution of the sinusoidal component in the Fourier representation of the target function, f .

For the ARFF training, the associated least squares problem, in the amplitudes $\hat{\beta}$ for sampled frequencies ω_k and bias b_k , $k = 1, \dots, K$ is $\min_{\hat{\beta} \in \mathbb{R}^K} (N^{-1} \|\mathbf{S}\hat{\beta} - \mathbf{y}\|^2 + \lambda \|\hat{\beta}\|^2)$, where $\mathbf{S} \in \mathbb{R}^{N \times K}$ is a matrix with elements $S_{n,k} = \cos(\omega_k \cdot \mathbf{x}_n + b_k)$, $n = 1, \dots, N$, $k = 1, \dots, K$ and $\mathbf{y} = (y_1, \dots, y_N) \in \mathbb{R}^N$. The SGD training follows the standard approach and is adapted from the TensorFlow implementation.

Rahimi and Recht first introduce random Fourier features in [21] as a computationally fast and accurate way to approximate the kernels of Support Vector Machines, [29, 25]. In [27], they show that by manually adjusting the sampling distribution, Fourier features neural networks can learn high-frequency components and thus overcome the spectral bias. Our approach is to adaptively sample the frequencies with ARFF and quantify the spectral bias with the definition introduced in [16].

While several innovative adversarial attack methods [11, 28, 2], and efficient defense mechanisms [18, 1, 7] exist, we demonstrate how just changing the training algorithm can affect the robustness of a simple adversarial attack.

Our contributions in summary

- We show, experimentally, how training a two-layer neural network with adaptive random Fourier features [13] can overcome the spectral bias, as defined in [16], compared to training with stochastic gradient descent.
- We show, experimentally, how two sets of identically structured neural networks with the same classification rate on the MNIST dataset have different degrees of robustness to a simple additive adversarial attack, depending on if we train them with adaptive random Fourier features [13] or with stochastic gradient descent.

2 Spectral bias

Numerous prior investigations [20, 12, 32, 19, 5], encompassing both empirical and theoretical approaches have delved into the spectral bias of neural networks: the inclination to learn low-frequency content. Kiessling et al. introduced a computable definition of spectral bias, as documented in [16], which shares similarities with the approaches outlined in [33] and [34].

In this work, we employ the spectral bias definition provided by Kiessling et al. in [16] to assess and contrast the spectral bias between two-layer neural networks trained by ARFF and SGD.

The spectral bias of a neural network β is defined as $SB = (\mathcal{E}_{high} - \mathcal{E}_{low}) / (\mathcal{E}_{high} + \mathcal{E}_{low})$, in [16], where $\mathcal{E}_{high} = \frac{\int_{\Omega_{high}} |\hat{r}_\rho(\omega)|^2 d\omega}{(2\pi)^d \text{Var}(f(x))}$ and $\mathcal{E}_{low} = \frac{\int_{\Omega_{low}} |\hat{r}_\rho(\omega)|^2 d\omega}{(2\pi)^d \text{Var}(f(x))}$, respectively, represent the error in the high- and low-frequency domain. Here $\text{Var}(f(x))$ denotes the variance of the unknown target function f and is approximated by the given dataset by using Monte Carlo integration, $\hat{r}_\rho(\omega)$ denotes the frequency spectrum of the function $r_\rho(x) = \sqrt{\rho(x)}(r(x) - \mathbb{E}_\rho[r(x)])$ where $r(x) = f(x) - \beta(x)$ is the residual, and $\rho(x)$ denotes the density of the input data x .

The symbols Ω_{high} and Ω_{low} denote the partitioned frequency domain, delineated by a designated cutoff frequency ω_0 , that is, $\Omega_{high} = \{\omega \in \mathbb{R}^d : |\omega|_\infty > \omega_0\}$, and $\Omega_{low} = \{\omega \in \mathbb{R}^d : |\omega|_\infty \leq \omega_0\}$. The cutoff frequency ω_0 is specified to equalize the contributions of Ω_{low} and Ω_{high} to the total variance, expressed as $\int_{\Omega_{low}} |\hat{f}_\rho(\omega)|^2 d\omega = \int_{\Omega_{high}} |\hat{f}_\rho(\omega)|^2 d\omega$, where \hat{f}_ρ denotes the Fourier transform of the function $f_\rho(x) = \sqrt{\rho(x)}(f(x) - \mathbb{E}_\rho[f(x)])$. The integral equality to obtain the cutoff frequencies and, consequently, the spectral bias are in practice approximated by Methods 1 and 2 given in [16].

If the neural network performs equally well for low and high-frequency content, the spectral bias (SB) tends to approach zero. In such cases, the neural network will be characterized as spectrally unbiased.

Numerical comparison of spectral bias in trained neural networks using SGD and ARFF training algorithms

For this experiment, we consider the target function $f : \mathbb{R} \rightarrow \mathbb{R}$

$$f(x) = e^{-\frac{x^2}{2}} Si\left(\frac{x}{a}\right), \quad \text{where } Si(x) = \int_0^x \frac{\sin(t)}{t} dt \text{ with } a = 10^{-2}.$$

We train a two-layer neural network $\beta(x; \omega, \hat{\beta}) = \sum_{k=1}^K \hat{\beta}_k \cos(\omega_k \cdot x + b_k)$ to approximate the target function f , where K is the number of nodes in the neural network, $\omega_k \in \mathbb{R}$, $x \in \mathbb{R}$ and $\hat{\beta} \in \mathbb{R}$. While keeping the architecture the same, we train the neural network with SGD and ARFF. We compute the spectral bias of the trained neural network

after getting the desired accuracy in the approximation of the target function. We use Method 1, described in [16], for this experiment.

We generate training, validation, and test datasets as follows: sample N x -points from $\mathcal{N}(0, 1)$, evaluate the target function to obtain $f(x)$, and normalize all three datasets by subtracting the mean and dividing by the standard deviation of the training dataset $\{x_n, f(x_n)\}_{n=1}^N$. The error is then computed as the mean squared error for each dataset.

Both training algorithms, SGD and ARFF, are executed for $M = 10^4$ epochs. In the case of ARFF, we employ the following parameters: Tikhonov regularization $\lambda = 0.05$, proposal width $\delta = 2$ in the sampling algorithm, and the exponent $\gamma = 3d - 2$ in the acceptance-rejection ratio, where $d = 1$. For SGD we utilize a batch size of 32 and a learning rate of 0.0002.

In Figure 1, we delineate a comparison of the spectral bias exhibited by a neural network comprising $K = 1024$ neurons in the hidden layer and trained by two distinct training algorithms, SGD and ARFF.

The comparison reveals that the spectral bias of the neural network trained with SGD remains close to unity. In contrast, the ARFF training algorithm enables the neural network to achieve a state of spectral unbiasedness.

We have extended this test to include varying widths in a two-layer neural network and obtained similar results.

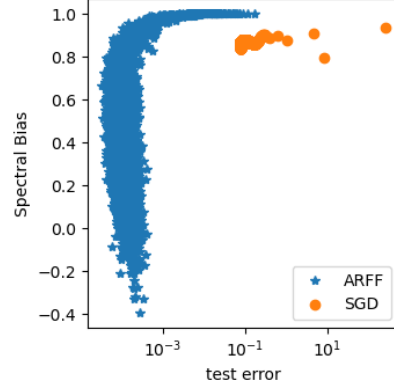


Figure 1: Spectral bias comparison, computed after each epoch, of a neural network trained with SGD and ARFF by using Method 1 from [16].

3 Robustness to an additive adversarial attack

This experiment compares the robustness of two-layer neural networks trained with ARFF and SGD against an additive adversarial black box attack.

Datasets

We consider the MNIST dataset that consists of 70,000 handwritten digit images of size 28×28 and its corresponding labels $y \in \{0, 1, \dots, 9\}$. The task is to classify the images between the digits 0 to 9.

We partition the datasets into training, validation, and test data with the corresponding ratio of 7 : 2 : 1.

Neural network structure and training

We train ten two-layer neural networks $\beta^i(\mathbf{x}; \boldsymbol{\omega}^i, \hat{\boldsymbol{\beta}}^i, b^i) = \sum_{k=1}^K \hat{\beta}_k^i \cos(\boldsymbol{\omega}_k^i \cdot \mathbf{x} + b^i)$, $i = 0, \dots, 9$ with $K = 2^{10}$ nodes each, with both ARFF and SGD where $\boldsymbol{\omega}_k^i \in \mathbb{R}^d$, $\mathbf{x} \in \mathbb{R}^d$, $b^i \in \mathbb{R}$, $\hat{\beta}_k^i \in \mathbb{R}$, and $d = 784$ for the MNIST dataset.

We train each neural network to classify one number each. E.g. $\beta^2(\mathbf{x}; \boldsymbol{\omega}^2, \hat{\boldsymbol{\beta}}^2, b^2)$ is trained to tell if a handwritten digit is the number 2 or not. For SGD we use one least squares loss function for each neural network.

The training data $\{(\mathbf{x}_n; (y_n^0, y_n^1, \dots, y_n^9))\}_{n=1}^N$ consist of vectorized handwritten digit images $\mathbf{x}_n \in \mathbb{R}^{784}$ with corresponding vector labels $(y_n^0, y_n^1, \dots, y_n^9)$. Each vector label has one component equal to one and the others equal to zero. The index i of the component y_n^i equal to 1 is the number that the handwritten digit \mathbf{x}_n represents.

In the following experiments, we run both the training algorithms, SGD and ARFF, for $M = 100$ epochs each. For ARFF, the parameters are set to $\lambda = 0.05$, $\delta = 0.02$, $\gamma = 3d - 2$, with $d = 784$. For SGD, we use a batch size of 32 and a learning rate of 0.002.

Classification and computation of the accuracy

A handwritten test digit $\mathbf{x} \in \mathbb{R}^{784}$ is classified as the number $\arg \max_i \{\beta^i(\mathbf{x}; \boldsymbol{\omega}^i, \hat{\boldsymbol{\beta}}^i, b^i)\}$ where $\{\boldsymbol{\omega}_k^i, \hat{\beta}_k^i, b_k^i\}_{k=1}^K, i = 0, \dots, 9$ are the trained weights and biases resulting from either ARFF or SGD.

We evaluate the performance of the artificial neural networks by the classification accuracy, defined as $accuracy = \frac{N_{\text{correctly classified}}}{N_{\text{total}}}$, where N_{total} represents the total number of test images, and $N_{\text{correctly classified}}$ denotes the count of accurately classified handwritten test images out of N_{total} .

Random noise attack

We apply the simple black box attack of adding random normal distributed noise to $n_{pixel} \in [0, d]$ randomly chosen pixels of each N_{test} images in the test dataset to get the noisy test data $\mathbf{x}_{noise}^n = \mathbf{x}_{test}^n + \boldsymbol{\delta}^n \odot \mathbf{r}^n$, $\mathbf{r}^n \sim N(\mathbf{0}, \sigma^2 \mathbf{I})$, $n = 1, \dots, N_{test}$ where $\mathbf{I} \in \mathbb{R}^{d \times d}$ is the identity matrix, $\sigma \in \mathbb{R}$, and \odot denotes component-wise multiplication. Here the component δ_i of $\boldsymbol{\delta}$ satisfies $\delta_i \in \{0, 1\}$ and the number of the elements which are equal to one is n_{pixel} , i.e. $\sum_{i=1}^d \delta_i = n_{pixel}$.

Experiment 1: We present the classification rates on the noisy test images, where $n_{pixel} = 50$, for different values of σ on the left in Figure 2 for both SGD and ARFF. Even though both training algorithms have trained the neural networks to the same classification rate, the accuracy of the neural networks trained by SGD decreases faster when σ increases compared to the accuracy of the neural networks trained by ARFF.

Experiment 2: We present the classification rates on the noisy test images, where $n_{pixel} = d = 784$, for different values of σ in the middle in Figure 2. Contrary to the results in Experiment 1, we see a faster decrease in the accuracy when σ increases for ARFF than for SGD.

Experiment 3: As in Experiment 2 we use $n_{pixel} = d = 784$. The difference is that we also add noise, from $\mathcal{N}(0, \sigma^2)$, to all 784 pixels of each image in the validation dataset and use the best classification rate on the validation data as stopping criterion for the training. We present the results on the right in Figure 2. The classification rate for SGD is qualitatively the same as in Experiment 2, but now the classification rate for ARFF is better than for SGD. The optimal stopping epoch differs for each value of σ , though.

Remark: For comparison, we run Experiments 1, 2, and 3 analogously on the CIFAR-10 dataset as well. The results are qualitatively the same. I.e., ARFF shows better results for Experiments 1 and 3, while SGD gives better results for Experiment 2.

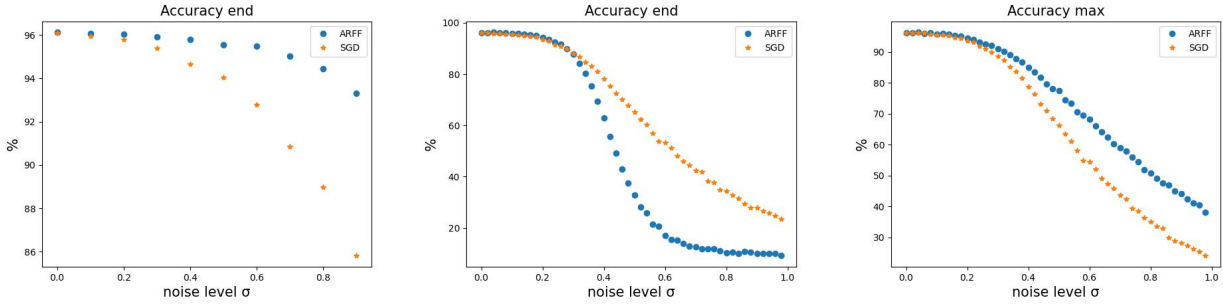


Figure 2: **Left, Experiment 1:** The figure shows the accuracy on the MNIST test data after a sparse black box attack. **Middle, Experiment 2:** Accuracy on the MNIST test data after a black box attack for different noise levels. **Right, Experiment 3:** Accuracy on the MNIST test dataset after a black box noise attack for classifiers tuned, with the help of noisy validation data, to withstand noise attacks. In Experiment 3 the stopping criteria depends on the noise level σ .

4 Conclusion

Our experimental findings underscore the effectiveness of ARFF-trained neural networks in efficiently capturing high-frequency content within the context of function reconstruction. We have also demonstrated the heightened robustness of ARFF-trained models when subjected to sparse black-box noise attacks at varying levels. While the same superiority is not consistently observed in the face of full black-box noise attacks, where noise is added to every pixel of the input image, we establish that the ARFF-trained neural network can maintain its robustness and outperform SGD by strategically leveraging noisy validation datasets and implementing early stopping. This insight provides practitioners with a viable strategy for prioritizing neural network robustness. Moreover, this study paves the way for future investigations, urging a more comprehensive exploration of ARFF training under sophisticated and well-structured attacks, the Square Attack [2], which we eagerly anticipate exploring in subsequent research endeavors.

5 Acknowledgement

We express our gratitude to Prof. Anders Szepeszy and Dr. Erik von Schwerin for reading the manuscript and providing their valuable feedback. This collaborative work, integral to Ms. Lisi Liang’s master’s thesis, is submitted as a component of her master’s degree fulfillment at RWTH Aachen University.

We also acknowledge the financial support by the KAUST Office of Sponsored Research (OSR) under Award numbers URF/1/2281 – 01 – 01, URF/1/2584 – 01 – 01 in the KAUST Competitive Research Grants Program Round 8 and the Alexander von Humboldt Foundation.

References

- [1] Martín Abadi, Andy Chu, Ian J. Goodfellow, H. B. McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.
- [2] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *European conference on computer vision*, pages 484–501. Springer, 2020.
- [3] Andrew R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.
- [4] Chris M Bishop. Neural networks and their applications. *Review of scientific instruments*, 65(6):1803–1832, 1994.
- [5] Yuan Cao, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu. Towards understanding the spectral bias of deep learning. *arXiv preprint arXiv:1912.01198*, 2019.
- [6] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. Ieee, 2017.
- [7] Bryant Chen, Wilka Carvalho, Nathalie Baracaldo, Heiko Ludwig, Ben Edwards, Taesung Lee, Ian Molloy, and B. Srivastava. Detecting backdoor attacks on deep neural networks by activation clustering. *ArXiv*, abs/1811.03728, 2018.
- [8] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.
- [9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- [10] Kevin Gurney. *An introduction to neural networks*. CRC press, 1997.
- [11] Hatem Hajri, Manon Cesaire, Lucas Schott, Sylvain Lamprier, and Patrick Gallinari. Neural adversarial attacks with random noises. *International Journal on Artificial Intelligence Tools*, 2022.
- [12] Qingguo Hong, Qinyan Tan, Jonathan W. Siegel, and Jinchao Xu. On the activation function dependence of the spectral bias of neural networks. *ArXiv*, abs/2208.04924, 2022.
- [13] Aku Kammonen, Jonas Kiessling, Petr Plecháč, Mattias Sandberg, and Anders Szepeszy. Adaptive random fourier features with metropolis sampling. *Foundations of Data Science*, 2(3):309–332, 2020.
- [14] Aku Kammonen, Jonas Kiessling, Petr Plecháč, Mattias Sandberg, Anders Szepeszy, and Raúl Tempone. Smaller generalization error derived for a deep residual neural network compared with shallow networks. *IMA Journal of Numerical Analysis*, 43(5):2585–2632, 09 2022.
- [15] Jonas Kiessling, Emanuel Ström, and Raúl Tempone. Wind field reconstruction with adaptive random fourier features. *Proc. R. Soc. A.*, 477, 2021.
- [16] Jonas Kiessling and Filip Thor. A computable definition of the spectral bias. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7168–7175, 2022.
- [17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [18] Tian Yu Liu, Yu Yang, and Baharan Mirzasoleiman. Friendly noise against adversarial noise: A powerful defense against data poisoning attack. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 11947–11959. Curran Associates, Inc., 2022.

- [19] Nasim Rahaman, Devansh Arpit, Aristide Baratin, Felix Dräxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron C. Courville. On the spectral bias of deep neural networks. *ArXiv*, abs/1806.08734, 2018.
- [20] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pages 5301–5310. PMLR, 2019.
- [21] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.
- [22] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.
- [23] Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. *Advances in neural information processing systems*, 30, 2017.
- [24] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- [25] Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer Publishing Company, Incorporated, 1st edition, 2008.
- [26] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- [27] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020.
- [28] Yi-Ting Tsai, Min-Chu Yang, and Han-Yu Chen. Adversarial attack on sentiment classification. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 233–240, 2019.
- [29] Vapnik Vladimir Naumovich. *Statistical learning theory*. Adaptive and learning systems for signal processing, communications, and control. Wiley, New York, 1998.
- [30] E Weinan, Chao Ma, and Lei Wu. A comparative analysis of optimization and generalization properties of two-layer neural network and random feature models under gradient descent dynamics. *Sci. China Math*, 2020.
- [31] E Weinan, Chao Ma, and Lei Wu. The barron space and the flow-induced function spaces for neural network models. *Constructive Approximation*, 55(1):369–406, February 2022.
- [32] Yongtao Wu, Zhenyu Zhu, Fanghui Liu, Grigorios G. Chrysos, and Volkan Cevher. Extrapolation and spectral bias of neural nets with hadamard product: a polynomial net study. *ArXiv*, abs/2209.07736, 2022.
- [33] Zhi-Qin John Xu, Yaoyu Zhang, Tao Luo, Yanyang Xiao, and Zheng Ma. Frequency principle: Fourier analysis sheds light on deep neural networks. *arXiv preprint arXiv:1901.06523*, 2019.
- [34] Zhi-Qin John Xu, Yaoyu Zhang, and Yanyang Xiao. Training behavior of deep neural network in frequency domain. In *Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, December 12–15, 2019, Proceedings, Part I 26*, pages 264–274. Springer, 2019.