Adversarial Examples for Kernel Methods

MIT M.Eng. Thesis Proposal

December 2020

Tony Wang
twang6@mit.edu

Abstract

I propose to study adversarial examples for kernel methods in order to develop kernels that are both robust and performant. I then propose to translate my findings to neural networks by leveraging neural tangent kernel theory.

1 Introduction

1.1 Why study adversarial examples?

In its 2012 debut at the ImageNet competition, AlexNet [1] set a record of 84.7% for top-5 accuracy, beating the second place model by over 10% [2]. In the years since, deep vision networks have achieved top-5 accuracies of 98.7% [3], surpassing humans who top out at around 95% [4]. Given these amazing results, one may be tempted to conclude that we have solved the problem of image classification. I claim this is not true.

Deep neural networks trained using standard methods are actually very fragile. Given an arbitrary input x, one can usually perturb x a tiny amount to $x + \epsilon$ such that the network outputs nonsense when fed $x + \epsilon$ as input [5].

As an example, consider Figure 1.1, which gives a visualization of an adversarial perturbation to a panda image. By perturbing the panda with noise that is near imperceptible to a human, a classifier is made to completely mis-classify the panda as a gibbon.

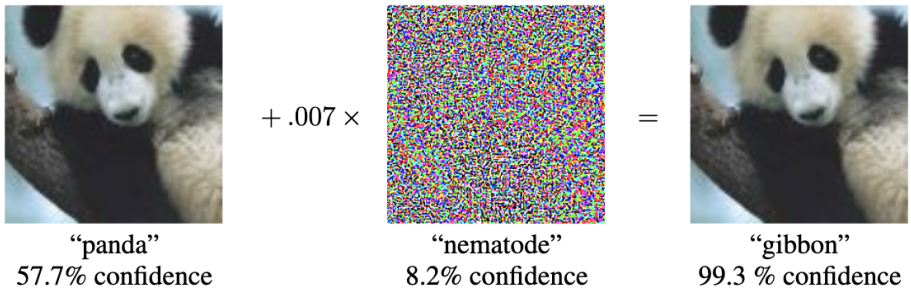

Figure 1.1: A small adversarial perturbation is enough to cause a deep network to mis-classify this panda. Figure taken from [6].

A standard ResNet-50 trained on ImageNet has 76% top-1 accuracy, which corresponds to around 93% top-5 accuracy [7]. However, if we adversarially perturb the test inputs by at most 2/255 intensity-levels per pixel, the same ResNet-50 achieves a top-1 accuracy of **0.036%** [8]. 2/255 intensity-levels per pixel is imperceptible to a human, but this is apparently enough to completely break a standard ResNet-50. Thus neural networks must solve image classification tasks in a manner that is very different from humans.

Moreover, adversarial examples are not just of theoretical interest. Researchers have performed adversarial attacks in which they placed small stickers on a road that caused a Tesla to swerve into the wrong lane [9]. The same stickers in this attack would not have impacted a human, so despite what ImageNet benchmarks might suggest, deep learning still has a ways to go before it can completely substitute humans, especially in contexts where safety and robustness are important.

I believe that understanding why adversarial examples occur can yield insight into how deep learning actually works. Additionally, since humans seem to have a decent amount of adversarial robustness, developing learning algorithms that are adversarially robust imposes a constraint upon these algorithms that forces them to behave more like humans, something that can boost both performance and interpretability. Finally, more robust models are of practical interest, since they are necessary if we wish to deploy models in safety-critical scenarios.

1.2 Why kernel methods?

Many papers have been published on adversarial examples for neural networks. However, neural networks are complicated objects, so one challenge that all of these works have to deal with is the fact that neural networks are hard to analyze. I wish to study adversarial examples for kernel methods to get around this difficulty, since kernel methods are more amenable to theoretical analysis than neural networks.

One risk of this approach is that adversarial examples for kernel methods may be very different in nature than adversarial examples for neural networks. However, preliminary results indicate that this is not the case (see Figure 3.1).

Moreover, recent theoretical advancements have shown that neural networks and kernels have close ties. I plan to leverage this *neural tangent kernel* theory to help connect my results back to the land of neural networks.

2 Related work

2.1 Adversarial examples: Attacks, defenses, and explanations

There have been hundreds of papers attacking, defending against, and explaining adversarial examples [10]. However, despite these efforts, currently no known machine learning algorithms can achieve both human-level accuracy and human-level robustness on tasks like image classification [11]. Thus adversarial robustness is still very much an open problem.

In studying adversarial examples for kernel methods, our primary attack model will be the one given in [12], which uses projected gradient descent to perform the constrained optimization problem that makes up the attack.

Additionally, whenever possible, we will try to connect research on adversarial example for neural networks to the kernel setting, focusing in particular on strong defenses [8, 13, 3] and explanations [14, 15].

2.2 Neural tangent kernel theory

It was known since the mid 1990s that wide neural networks initialized with i.i.d. weights and proper scaling (e.g., as in He [16] or Xavier [17] initialization) behave like a randomly drawn function from a Gaussian process with an architecture dependent kernel[1][18]. However, over two decades later, people now realize that these wide neural networks also *train* like a kernel method [19, 20, 21]. The architecture specific kernel that characterizes this training is known as the *neural tangent kernel*. Together, these results show that kernels capture the complete behavior of wide neural networks [22].

Our goal in this work is to leverage neural tangent kernel theory to connect our results for kernels back to neural networks. Doing so will require doing the inverse of what most neural tangent kernel theory currently does, and we comment on our strategy for doing this in Section 3.3.

3 Proposed work

I propose to study adversarial examples for kernel methods. The primary goals of this project are:

1. Develop mathematical theory that predicts the severity of kernel adversarial examples given a particular kernel and dataset.

2. Test theories of adversarial examples (my own as well as other people's) on kernel methods using synthetic and real-world data-sets. I will focus on real-world datasets commonly used for adversarial example research, e.g. MNIST, CIFAR10, ImageNet.

3. Develop kernels that are both performant (in terms of test-accuracy) and immune to adversarial examples. If this proves to be too hard, I aim to provably rule out approaches that won't work.

4. Connect results on kernel methods to results on neural networks (and vice-versa) via neural tangent kernel theory.

Though this project has been and will be undertaken by a team, I am the leader of the project and plan to make enough individual contributions such that I am able to have enough material for a M.Eng. thesis.

In what follows, I discuss some interesting preliminary results obtained by my collaborators and I this semester, and also discuss what our future plans and timeline looks like.

[1]Sometimes called the *neural network Gaussian process kernel*.

3.1 Preliminary theoretical results

On the theoretical front, we have been focusing on kernel SVM classifiers, which take the form

$$f(x) \triangleq \text{sign}(g(x)) \triangleq \text{sign}\left(b + \sum_{i=1}^{n} \alpha_i \, k(x_i, x)\right), \tag{3.1}$$

with $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ a kernel function, $x_1, \ldots x_n \in \mathbb{R}^d$ training data, and $(\alpha, b) \in \mathbb{R}^n \times \mathbb{R}$ the model parameters.

3.1.1 Approximate Margin

To quantify adversarial-robustness, we define the *true-margin* of $f(\cdot)$ at x under a norm $\|\cdot\|$ to be the size of the minimum perturbation to x (measured under $\|\cdot\|$) needed to change the output of $f(\cdot)$. We denote this true-margin as $\text{marg}_{f,\|\cdot\|}(x)$.

In general, the true-margin is difficult to compute. However, for kernel SVM classifiers (and linear classifiers more generally), we can approximate the true margin via

$$\text{marg}_{f,\|\cdot\|}(x) \approx \text{amarg}_{g,\|\cdot\|}(x) \triangleq \frac{|g(x)|}{\|\nabla g(x)\|}, \tag{3.2}$$

where $g(\cdot)$ relates to the classifier $f(\cdot)$ via the definition in Equation (3.1). We motivate $\text{amarg}_{g,\|\cdot\|}$ by noting that it is a perfect approximation when $f(\cdot)$ is linear over that data-space $\mathbb{R}^d$. Moreover, when the true-margin is sufficiently small, the approximation is also accurate since any smooth function $g(\cdot)$ behaves linearly in a sufficiently small neighborhood of x.

In the special case where we measure perturbation using the L2-norm, we have that

$$\text{amarg}_{g,\|\cdot\|_2}(x)^2 = \frac{g(x)^T g(x)}{\nabla g(x)^T \nabla g(x)} \tag{3.3}$$

$$= \frac{\alpha^T \left(K(x) K(x)^T\right) \alpha}{\alpha^T \left(\nabla K(x) \nabla K(x)^T\right) \alpha}, \tag{3.4}$$

where

$$K(x) \triangleq [k(x_1, x) \cdots k(x_n, x)]^T \in \mathbb{R}^n, \tag{3.5}$$

and α and $x_1, \ldots x_n$ are as defined in Equation (3.1).

3.2 Preliminary empirical results

On the empirical front we have primarily been benchmarking the performance of simple shallow neural networks, standard kernels, and simple neural tangent kernels on small-scale classification tasks (based off of MNIST and CIFAR10). To perform classification using kernels, we use support vector machines.

More specifically, for each of our standardly-trained classifiers f we have been benchmarking both standard measures of test loss of the form

$$\mathcal{L}_{\text{test}}(f) = \mathbb{E}_{\mathsf{x},\mathsf{y}\sim\mathcal{P}_{\text{test}}}\left[\ell(f(\mathsf{x}),\mathsf{y})\right], \tag{3.6}$$

as well as adversarial measures of test loss of the form

$$\mathcal{L}_{\text{test}}(f) = \mathbb{E}_{\mathsf{x},\mathsf{y}\sim\mathcal{P}_{\text{test}}}\left[\max_{\delta\in\mathcal{D}} \ell(f(\mathsf{x}+\delta),\mathsf{y})\right]. \tag{3.7}$$

Our primary empirical observations are twofold:

1. SVMs trained with standard kernels are vulnerable to adversarial examples in ways that are very similar to neural networks. The similarities are both quantitative (similar adversarial accuracy) as well as qualitative (similar visual appearance of adversarial examples). See Figure 3.1 for details.

2. Simple shallow neural networks, standard kernels, and simple neural tangent kernels have similar standard empirical performance. See Figure 3.2 for details.

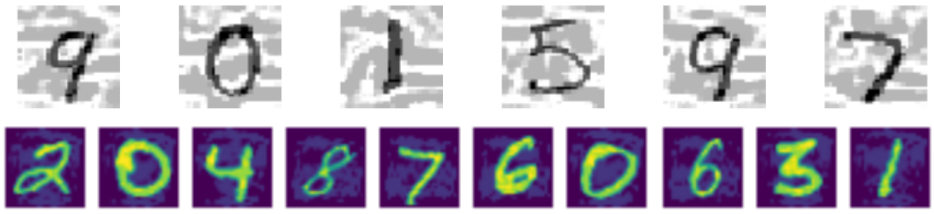

Figure 3.1: Some MNIST adversarial examples for a neural network (top, from [23]) and a kernel SVM (bottom, RBF from our experiments). These were generated to lie within a bounded L_∞ ball of the base image (top radius: 0.1/1, bottom radius: 0.15/1). We note that both the NN and kernel SVM have adversarial examples which leave the main number unchanged while making small changes to the image background. Additionally, our RBF kernel SVM only achieved 13% adversarial test-accuracy for binary classification of [0-4] vs [5-9]. This type of low adversarial accuracy is also characteristic of neural networks trained using standard methods [12, 8].

3.3 Future work

3.3.1 Immediate extensions of preliminary results

We plan on continuing the lines of exploration described in our preliminary results sections. We plane to:

1. Design and run experiments to verify the empirical validity of our preliminary theoretical results.

Architecture	MNIST	CIFAR10
Linear kernel	79.15	57.55
Poly (deg 4)	89.35	60.5
Gaussian	89	59.85
Laplace	**90.8**	61.65
FC-NTK ($d = 3$)	90.65	**61.6**
CNN-NTK ($d = 3$)	90	61
FC-NN ($d = 3, w = 20$)	88.65	59.9
CNN ($d = 3, w = 20$)	89	60.8

Figure 3.2: Empirical performance of simple shallow neural networks, standard kernels (except the linear kernel), and simple neural tangent kernels are very similar. These results are for small-scale tasks based on MNIST and CIFAR10, more specifically, binary classification (labels [0-4] vs. [5-9]) on downscaled images (7x7x1 and 8x8x3 respectively), with $n = 2000$ training datapoints. Kernel methods were trained using SVMs. Neural network-based classifiers had ReLu non-linearity, depth d, and width/channel-count w. For each row, the reported accuracy is the best across varying levels of regularization.

2. Design and run experiments to verify the empirical validity of proposed theories of adversarial examples in the kernel regime. The primary theories to test for are margin-norm scaling from [24], non-robust features from [15], and frequency-dependency as described in [14].

3. Scale up empirical evaluations to larger data-sets and deep / more complex architectures. Currently our empirical results can't differentiate between kernels and neural networks. We know that in practice, deep networks greatly outperform kernels. Eventually, we wish to transfer our analyses and results to the deep networks that people use in practice. To perform this transfer, it will be helpful to identify candidate models / architectures that are representative of the deep-learning regime.

3.3.2 Key challenges remaining

There are also some key challenges that we have yet to (but do plan to) address:

1. **How do we design more robust and performant kernels?**

   One approach would be to take inspiration from prior literature on kernel selection [25, 26] and kernel design [27, 28]. While these methods don't focus on the problem of adversarial robustness, they do take into consideration performance and noise, so I think there are useful insights to be gained here.

   Another approach would be to adapt techniques that are known to improve performance and robustness (e.g., data augmentation, adversarial training, invariance

to translation) and find ways to apply the effect of these techniques to kernels. A recent example of using this approach to improve neural tangent kernels can be found in [29].

Finally, we believe the ongoing theory development and experiments done by our team will help with designing better kernels.

2. **How do we transfer results from kernels to neural networks?**

   This is perhaps the most challenging part of the project. While neural tangent kernel theory connects neural networks to kernel methods, it does so primarily in a single direction – given some network architecture, we can analyze its infinite width limit and derive its corresponding tangent kernel. However, for this project we need to solve the inverse problem: given a performant kernel method, how can we design a neural network that approximates (or even better, improves upon) its behavior?

   To tackle this problem, our primary approach will be to first collect experimental evidence of the behavior of NTKs, and then try and build an inverse-NTK theory that matches the data. What behavioral data will we collect? One place to start would be spectral data, which has previously been used to compare simple NTKs to Laplace kernels [30].

   Finally, even neural tangent kernel theory itself only approximates the behavior of finite width neural networks. In pursuit of our goal to transfer our results to practical neural networks, it will be worth paying attention to literature that studies how NTK theory diverges from how NNs behave in practice [31, 32].

3.4 Timeline

This is primarily the timeline for my *thesis*. Research for the project will continue throughout this period, but I will probably only include results obtained up through early April 2021.

My proposed timeline is as follows:

2021-01-29 Finish designing and running the experiments described in the immediate extension list in Section 3.3.1.

Additionally, I aim to have finished going through the literature stated in Section 3.3.2 by this date.

2021-02-05 Develop a game plan for tackling the key challenges in Section 3.3.2. The next month and a half will then implement this game plan.

2021-03-15 Have an outline of my thesis completed. This outline will specify which existing theoretical and experiment results to include, and specify **specific** theoretical and experimental extensions. At this point, the goal is to have no more unknown unknowns for the thesis.

2021-04-12 Have a rough draft of my thesis ready. This will be sent to my supervisor and collaborators for content feedback. I will use the feedback to make a final pass at strengthening theoretical results and collecting experimental data.

2021-04-26 [2] Have a poster presentation prepared for the EECS Masterworks poster session.

2021-05-01 Finalize the thesis and send it to my thesis supervisor for final review. This will be followed by a final round of polish before the final thesis submission.

2021-05-11 [3] Thesis will be submitted by this date.

3.5 Mentors and collaborators

My primary advisor and supervisor for this project is Professor Gregory W. Wornell, principal investigator at the Signals, Information, and Algorithms (SIA) Lab. Additional guidance has been provided by Dr. Yuheng Bu, a postdoc also at the SIA Lab.

Additionally, I have been collaborating closely on this project with Yang Yan, a close friend of mine and undergraduate here at MIT.

Over the course of the coming IAP and the spring semester, I plan to continue working together with Yang on this project under the guidance of Dr. Bu and Professor Wornell.

References

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[2] http://www.image-net.org/challenges/LSVRC/2012/results.html.

[3] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020.

[2] This is my projection for the date of the 2021 EECS Masterworks poster session. The official date has yet to be announced, but it was last held 2019-04-25, so I assume it will also be held around this time in 2021.

[3] This is my projection of the M.Eng. thesis due date for the June 2021 degree date. The official date has yet to be announced, but the past deadline was 2020-05-12.

[4] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[5] Yandong Li, Lijun Li, Liqiang Wang, Tong Zhang, and Boqing Gong. Nattack: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks. *arXiv preprint arXiv:1905.00441*, 2019.

[6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[7] https://paperswithcode.com/sota/image-classification-on-imagenet.

[8] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *Advances in Neural Information Processing Systems*, pages 3358–3369, 2019.

[9] Evan Ackerman. Three small stickers in intersection can cause tesla autopilot to swerve into wrong lane - ieee spectrum. https://spectrum.ieee.org/cars-that-think/transportation/self-driving/three-small-stickers-on-road-can-steer-tesla-autopilot-into-oncoming-lane.

[10] Rey Reza Wiyatno, Anqi Xu, Ousmane Dia, and Archy de Berker. Adversarial examples in modern machine learning: A review. *arXiv preprint arXiv:1911.05268*, 2019.

[11] Nicholas Carlini. Are adversarial example defenses improving? https://nicholas.carlini.com/writing/2020/are-adversarial-exampe-defenses-improving.html.

[12] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[13] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 501–509, 2019.

[14] Dong Yin, Raphael Gontijo Lopes, Jon Shlens, Ekin Dogus Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. *Advances in Neural Information Processing Systems*, 32:13276–13286, 2019.

[15] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136, 2019.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[17] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[18] Radford M Neal. Priors for infinite networks. In *Bayesian Learning for Neural Networks*, pages 29–53. Springer, 1996.

[19] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks, 2020.

[20] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252. PMLR, 2019.

[21] Simon S. Du, Jason D. Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks, 2019.

[22] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in neural information processing systems*, pages 8572–8583, 2019.

[23] Aleksander Madry and Zico Kolter. Chapter 3 - adversarial examples, solving the inner maximization. http://adversarial-ml-tutorial.org/adversarial_examples/.

[24] Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. *arXiv preprint arXiv:1802.01396*, 2018.

[25] Berwin A Turlach. Bandwidth selection in kernel density estimation: A review. In *CORE and Institut de Statistique*. Citeseer, 1993.

[26] Tony Jebara. Multi-task feature and kernel selection for svms. In *Proceedings of the twenty-first international conference on Machine learning*, page 55, 2004.

[27] Koby Crammer, Joseph Keshet, and Yoram Singer. Kernel design using boosting. *Advances in neural information processing systems*, 15:553–560, 2002.

[28] Tong Zhang and Rie K Ando. Analysis of spectral kernel design based semi-supervised learning. *Advances in neural information processing systems*, 18:1601–1608, 2005.

[29] Zhiyuan Li, Ruosong Wang, Dingli Yu, Simon S Du, Wei Hu, Ruslan Salakhutdinov, and Sanjeev Arora. Enhanced convolutional neural tangent kernels. *arXiv preprint arXiv:1911.00809*, 2019.

[30] Amnon Geifman, Abhay Yadav, Yoni Kasten, Meirav Galun, David Jacobs, and Basri Ronen. On the similarity between the laplace and neural tangent kernels. *Advances in Neural Information Processing Systems*, 33, 2020.

[31] Yu Bai and Jason D Lee. Beyond linearization: On quadratic and higher-order approximation of wide neural networks. *arXiv preprint arXiv:1910.01619*, 2019.

[32] Greg Yang and Edward J Hu. Feature learning in infinite-width neural networks. *arXiv preprint arXiv:2011.14522*, 2020.