Fourier Features for High-Frequency, Low-Dimensional Image and Video Representations

Ryan Diaz
Department of Computer Science and Engineering
University of Minnesota, Twin Cities
`diaz0329@umn.edu`

1 Introduction

Neural networks can be powerful function approximators in machine learning and deep learning methods, and can be trained to perform complex tasks using unstructured data. For example, Neural Radiance Fields (NeRFs) allow for complex 3D scenes to be represented as simple fully-connected networks that enable novel-view rendering [Mildenhall et al., 2020]. One major pitfall of these models, however, is their “black box" nature, making it difficult to interpret their decisions or describe their training dynamics. Kernel methods, on the other hand, are built on solid mathematical foundations with well-understood training dynamics. This project will explore the connection between neural networks and kernel methods through a specific construction known as the Neural Tangent Kernel (NTK). We will then discuss an application of NTK theory that addresses an over-smoothing behavior of neural networks by applying a nonlinear embedding of Fourier Features to the network's inputs, and investigate its effect on the task of image and video representations.

2 Related Works and Background

2.1 The Neural Tangent Kernel

NTKs were first introduced in Jacot et al. [2020] and represent a way to describe the training dynamics of a neural network trained using gradient descent. Specifically, let Ω be a fully-connected neural network with parameters θ and let $f : \mathbb{R}^m \to \mathbb{R}^n$ be the function represented by Ω. Then the Neural Tangent Kernel function for two inputs $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathbb{R}^m$ is defined as

$$\Theta_\Omega(\boldsymbol{x}_i, \boldsymbol{x}_j; \theta) = \langle \nabla_\theta f(\boldsymbol{x}_i; \theta), \nabla_\theta f(\boldsymbol{x}_j; \theta) \rangle \tag{1}$$

Although the NTK function depends on the neural network parameters θ, it has been shown that the NTK converges to a constant kernel function as the layer size of the network tends to infinity. Assuming that θ was initialized according to a standard Gaussian distribution $\mathcal{N}$, the NTK at the infinite-width limit can be written as $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \mathbb{E}_{\theta\sim\mathcal{N}}\Theta_\Omega(\boldsymbol{x}_i, \boldsymbol{x}_j; \theta)$.

How can we connect the NTK to the process of training neural networks? It turns out that gradient descent with a neural network in the infinite-width limit is equivalent to “kernel descent" with the NTK defined above. We consider an infinite-width MLP trained with an L2 loss with parameters θ initialized in a way that ensures initial outputs near 0. Using the connection to kernel descent, the exact outputs of the network after t iterations of gradient descent under these conditions given a set of input points can be written in closed form. Thus, we are able to investigate the convergence of the network's training process.

Let $\boldsymbol{K} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^T$ be the diagonalization of the kernel matrix $\boldsymbol{K}$ corresponding to the NTK, η be the learning rate, and $\hat{\boldsymbol{y}}^{(t)}$ be the output of the neural network after t training iterations for the training data with ground truth labels $\boldsymbol{y}$. Borrowing analysis from Tancik et al. [2020], the error term after t iterations of gradient descent can be approximated as

$$\boldsymbol{Q}^T(\hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}) \approx -e^{-\eta \boldsymbol{\Lambda} t}\boldsymbol{Q}^T\boldsymbol{y} \tag{2}$$

We can see that the exponential rate of decay for the error depends on the eigenvalues of the NTK matrix $\boldsymbol{K}$: larger eigenvalues correspond to a faster rate of convergence. However, NTK literature such as in Bietti and Mairal [2019] shows that these eigenvalues decay rapidly during training, leading to slow convergence to higher-frequency components of the target function. This is known as a "spectral bias", and it leads to over-smoothed outputs from simple fully-connected networks.

2.2 Fourier Feature Networks

To combat the spectral bias problem for MLPs described above, Tancik et al. [2020] proposed Fourier Feature Networks, which introduced a sinusoidal nonlinear embedding for inputs of an MLP. More explicitly, the Fourier Feature embedding function $\gamma : [0,1]^d \to \mathbb{R}^m$ is defined as

$$\gamma(\boldsymbol{v}) = [a_1\cos(2\pi\boldsymbol{b}_1^T\boldsymbol{v}), a_1\sin(2\pi\boldsymbol{b}_1^T\boldsymbol{v}), \ldots, a_m\cos(2\pi\boldsymbol{b}_m^T\boldsymbol{v}), a_m\sin(2\pi\boldsymbol{b}_m^T\boldsymbol{v})] \tag{3}$$

for some parameters $\{(a_i, \boldsymbol{b}_i)\}_{i=1}^m$. This maps the input vector to an m-dimensional hypersphere. The induced kernel is then

$$k_\gamma(\boldsymbol{v}_i, \boldsymbol{v}_j) = \gamma(\boldsymbol{v}_i)^T\gamma(\boldsymbol{v}_j) = \sum_{j=1}^m a_j^2\cos\left(2\pi\boldsymbol{b}_j^T(\boldsymbol{v}_i - \boldsymbol{v}_j)\right) \tag{4}$$

We observe that the kernel function is dependent only on the distance between the two inputs, making it a *stationary* (translation-invariant) kernel. Modifying the base MLP with the Fourier Features embedding corresponds to composing the NTK of the base MLP with the induced Fourier Features kernel, giving it the same stationary property.

A more significant effect of this kernel composition is the addition of *tunable* components to the NTK in the form of the a_i and $\boldsymbol{b}_i$ parameters in Equation 3. By setting these parameters, we can control both the convergence rate and generalization ability of our network. Experiments in the paper have shown that the best empirical results came from setting $a_i = 1$ for $1 \leq i \leq m$, and sampling the $\boldsymbol{b}_i$ parameters from a distribution rather than manual tuning. Furthermore, the convergence and generalization properties of the MLP are mainly influenced by the *variance* (or *scale*) of the distribution rather than the *shape*. Small-scale distributions converge more slowly to higher-frequency components of the target function but are able to generalize better with smoother outputs. Large-scale distributions on the other hand have faster convergence rates, but are more prone to overfitting due to high-frequency aliasing effects. We explore the effects of adjusting the sampling distribution in the following sections.

3 Methods

3.1 Low-Dimensional Image Representations

Taking inspiration from Tancik et al. [2020], we apply the Fourier Features idea to the 2D image regression problem, which seeks to represent a single 2D image as a *coordinate-based MLP* that takes in an input $(x, y) \in \mathbb{R}^2$ corresponding to pixel coordinates on the image and outputs RGB values $(R, G, B) \in \mathbb{R}^3$ at that location. Rather than explicitly use an MLP, we build the network as a series of convolutional layers with 1×1 filters, which has the same effect as an MLP over each pixel coordinate. Additionally, we add batch normalization [Ioffe and Szegedy, 2015] after each hidden layer to speed up convergence.

3.2 Low-Dimensional Video Representations

We also extend the 2D image regression task to a 2D *video* regression task by adding an additional dimension to the input $(x, y, t) \in R^3$ representing time. The time inputs are normalized to be within

$[0, 1]$. Aside from the different input dimension, the network architecture is the same as in the image regression task (along with all the previously-described implementation adjustments). In this way, we can explore the characteristics of network outputs for unseen temporal inputs, which can be seen as a form of video interpolation.

4 Experimental Results

4.1 Image Reconstruction

We perform the image regression task on an image with an MLP with no Fourier Feature embedding as well as MLPs with a Fourier Feature embedding of varying sampling distribution scales. Each network was trained over 400 iterations with a learning rate of $\eta = 0.0001$. Figure 1 shows the results of image regression for both Gaussian and Uniform sampling distributions.

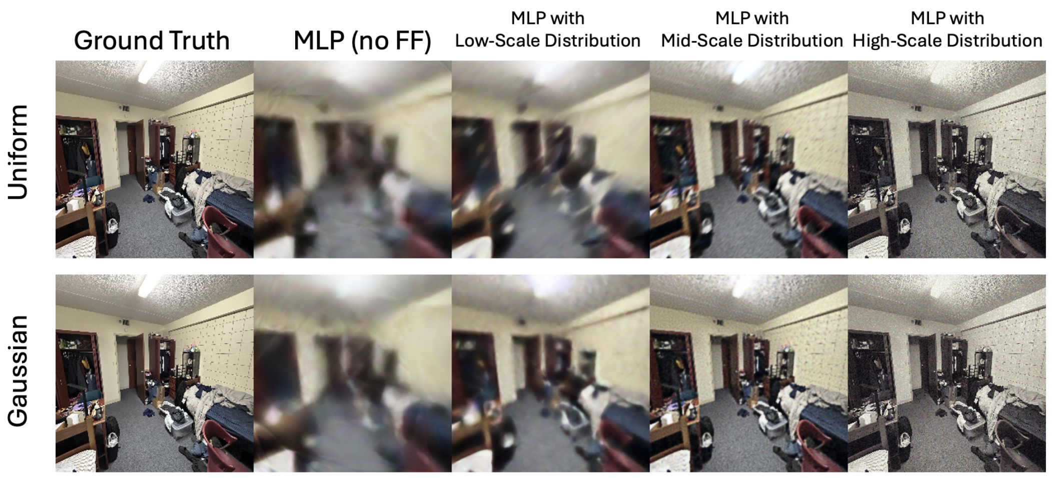
Ground Truth
MLP (no FF)
MLP with Low-Scale Distribution
MLP with Mid-Scale Distribution
MLP with High-Scale Distribution
Uniform
Gaussian

Figure 1: Results for image regression using MLP with Fourier Features of varying sampling distribution scales. Low, mid, and high-scale distributions correspond to $\sigma = 1, 10, 100$ respectively.

Looking at the image outputs, we observe little difference between the Uniform and Gaussian parameter sampling distributions, corroborating the claim that the distribution shape does not play a significant role in model behavior. We also observe the output of the standard MLP to be blurry and unfocused, which represents the MLP failing to learn the high-frequency components of the image. Across different distribution scales, we observe the clarity of the image improving as the scale increases; the PSNR values in Figure 2 quantitatively illustrate this. The convergence rates as shown in the training losses of Figure 2 also reflect predicted trends with respect to distribution scale, as higher-scale distributions tend to converge at a faster rate.

4.2 Video Interpolation

Similar to the image regression task, we perform the video regression task with an MLP with no Fourier Feature embedding as well as MLPs with a Fourier Feature embedding of varying sampling distribution scales. Additionally, we withhold a subset of the video frames during training by skipping every n frames in order to evaluate the generalization ability of the model on time steps unseen during training. Each network was trained over 1000 iterations with a learning rate of $\eta = 0.0001$.

Experimentally, the resulting outputs on individual frames that were seen during training followed many of the same trends in output image quality and training behavior across distribution scales described for the image regression task. On interpolated frames, however, we observe that MLPs with Fourier Feature embeddings of larger-scale ($\sigma = 5$) sampling distributions output noisier images on time steps that are farther away from the nearest training frame. In contrast, an MLP with no Fourier Feature embedding and MLPs with smaller-scale ($\sigma = 0.5$) sampling distributions output interpolation frames that are more similar in quality to the outputs on frames the model was trained

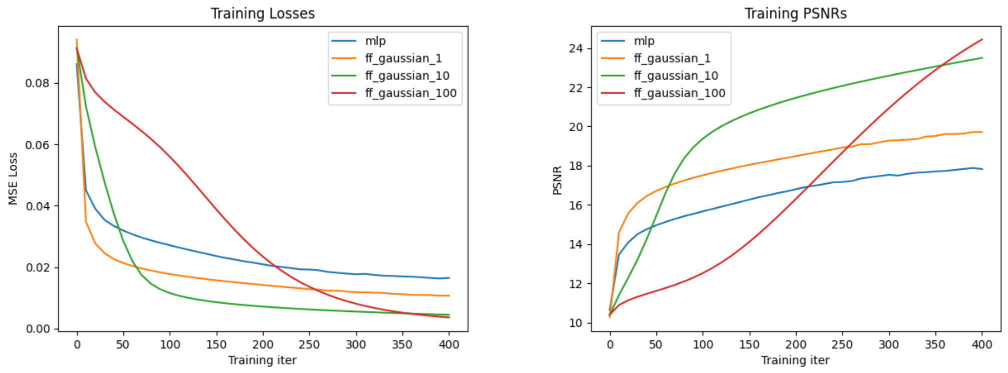

Figure 2: Training losses and PSNR values for the image regression task over 400 training iterations for Fourier Feature networks with a Gaussian sampling distribution.

on. Figure 3 illustrates this phenomenon with a training set that keeps 10% of all frames (skipping every 10 frames), with the model interpolating the other 90% of frames.

The resulting outputs of the model on interpolated frames are consistent with the predicted trends in generalization ability with respect to distribution scale. Larger-scale sampling distributions lead to an overfitting on the training frames and poor generalization to unseen frames, while smaller-scale sampling distribution retains the smooth-output behavior of the base MLP to an extent, allowing for more consistent transitions between training frames. Thus, tuning the parameters for a Fourier Feature Network requires balancing convergence rate and output image quality with robustness and generalization ability.

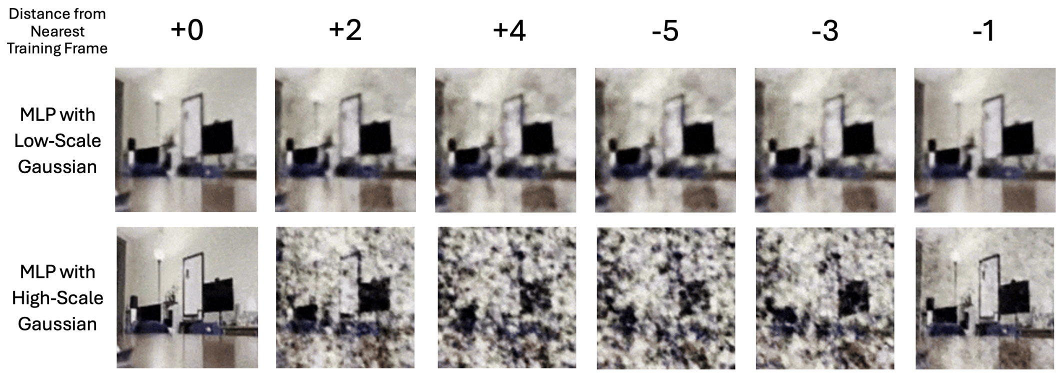
Distance from Nearest Training Frame
+0
+2
+4
-5
-3
-1
MLP with Low-Scale Gaussian
MLP with High-Scale Gaussian

Figure 3: Outputs of video regression network on interpolated (unseen during training) video frames. Distance is measured in number of frames. A negative/positive distance represents the interpolated frame being before/after the nearest training frame.

5 Conclusion

We investigate the theoretical motivations behind applying Fourier Feature embeddings by analyzing the model's associated Neural Tangent Kernel. We apply the Fourier Feature Network model to the previously-done image regression task, and extend the network's functionality into the realm of video regression. Through these two tasks, we empirically verify theoretical trends in convergence speed and generalization ability with respect to scale of the sampling distribution for the Fourier Feature embedding parameters. Evidently, Fourier Features can be very valuable in improving the training and output behavior of regression-based neural networks, but one must be careful to adjust the parameters of their model based on their desired task.

References

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

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

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.

Alberto Bietti and Julien Mairal. *On the inductive bias of neural tangent kernels*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Sergey Ioffe and Christian Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org, 2015.