Identifying good directions to escape the NTK regime and efficiently learn low-degree plus sparse polynomials

Eshaan Nichani [1], Yu Bai [2], and Jason D. Lee [1]

NeurIPS 2022

[1] Princeton University, [2] Salesforce Research

arxiv.org/abs/2206.03688

Background

Background

Optimization | **Generalization**

Background

	Optimization	Generalization
Neural Tangent Kernel (NTK) Theory		

Background

	Optimization	Generalization
Neural Tangent Kernel (NTK) Theory	✓	

Background

	Optimization	**Generalization**
Neural Tangent Kernel (NTK) Theory	✓	

- Couple to convex problem

Background

	Optimization	Generalization
Neural Tangent Kernel (NTK) Theory	✓	✗

- Couple to convex problem

Background

	Optimization	**Generalization**
Neural Tangent Kernel (NTK) Theory	✓	✗
	• Couple to convex problem	• Neural nets outperform their corresponding NTK in practice [1]. • d^p sample complexity lower bound for learning degree p polynomial [2].

Background

	Optimization	Generalization
Neural Tangent Kernel (NTK) Theory	✓	✗
	• Couple to convex problem	• Neural nets outperform their corresponding NTK in practice [1]. • d^p sample complexity lower bound for learning degree p polynomial [2].

Training is **lazy** — neurons move small distance and interpolate training data.

Background

	Optimization	Generalization
Neural Tangent Kernel (NTK) Theory	✓	✗
	• Couple to convex problem	• Neural nets outperform their corresponding NTK in practice [1]. • d^p sample complexity lower bound for learning degree p polynomial [2].

Training is **lazy** — neurons move small distance and interpolate training data.

Q: Can we encourage each neuron to move further and escape the NTK regime?
Does this allow us to break NTK sample complexity lower bounds?

Preliminaries

Preliminaries

- Dataset $\{(x_i, y_i)\}_{i \in [n]}$ of n samples.

Preliminaries

- Dataset $\{(x_i, y_i)\}_{i \in [n]}$ of n samples.
  - x_i drawn iid from d-dimensional sphere of radius $\sqrt{d}$.

Preliminaries

- Dataset $\{(x_i, y_i)\}_{i \in [n]}$ of n samples.
  - x_i drawn iid from d-dimensional sphere of radius $\sqrt{d}$.
  - $y_i = f^*(x_i)$ for unknown target $f^* : \mathbb{R}^d \to \mathbb{R}$.

Preliminaries

- Dataset $\{(x_i, y_i)\}_{i\in[n]}$ of n samples.
  - x_i drawn iid from d-dimensional sphere of radius $\sqrt{d}$.
  - $y_i = f^*(x_i)$ for unknown target $f^* : \mathbb{R}^d \to \mathbb{R}$.
- 2-layer neural network of width m: $f(x; \mathbf{W}) = m^{-1/2} a^T \sigma(\mathbf{W}x)$, where $a \in \mathbb{R}^m, \mathbf{W} \in \mathbb{R}^{m\times d}$.

Preliminaries

- Dataset $\{(x_i, y_i)\}_{i\in[n]}$ of n samples.
    - x_i drawn iid from d-dimensional sphere of radius $\sqrt{d}$.
    - $y_i = f^*(x_i)$ for unknown target $f^* : \mathbb{R}^d \to \mathbb{R}$.
- 2-layer neural network of width m: $f(x; \mathbf{W}) = m^{-1/2} a^T \sigma(\mathbf{W}x)$, where $a \in \mathbb{R}^m, \mathbf{W} \in \mathbb{R}^{m\times d}$.
- Recall that the NTK is the linearization of network at initialization:

$$f(x, \mathbf{W}) \approx f(x, \mathbf{W}_0) + \langle \mathbf{W} - \mathbf{W}_0, \nabla_{\mathbf{W}} f(x, \mathbf{W}_0) \rangle$$

Preliminaries

- Dataset $\{(x_i, y_i)\}_{i \in [n]}$ of n samples.
  - x_i drawn iid from d-dimensional sphere of radius $\sqrt{d}$.
  - $y_i = f^*(x_i)$ for unknown target $f^* : \mathbb{R}^d \to \mathbb{R}$.
- 2-layer neural network of width m: $f(x; \mathbf{W}) = m^{-1/2} a^T \sigma(\mathbf{W}x)$, where $a \in \mathbb{R}^m, \mathbf{W} \in \mathbb{R}^{m \times d}$.
- Recall that the NTK is the linearization of network at initialization:

$$f(x, \mathbf{W}) \approx f(x, \mathbf{W}_0) + \langle \mathbf{W} - \mathbf{W}_0, \underbrace{\nabla_{\mathbf{W}} f(x, \mathbf{W}_0)}_{\text{Feature map } \phi : \mathbb{R}^d \to \mathbb{R}^{md}.} \rangle$$

Why does the NTK overfit?

Why does the NTK overfit?

- Kernel regression theory: parameters first move in large eigenvalue directions of kernel matrix to learn signal, then move in small eigenvalue directions to overfit noise.

Why does the NTK overfit?

- Kernel regression theory: parameters first move in large eigenvalue directions of kernel matrix to learn signal, then move in small eigenvalue directions to overfit noise.
- Prior work [3] shows with $n \gtrsim d^k$ samples, large eigenvalue directions express degree $\leq k$ polynomials.

Why does the NTK overfit?

- Kernel regression theory: parameters first move in large eigenvalue directions of kernel matrix to learn signal, then move in small eigenvalue directions to overfit noise.
- Prior work [3] shows with $n \gtrsim d^k$ samples, large eigenvalue directions express degree $\leq k$ polynomials.
- We would like to move in large eigenvalue directions to learn the low degree signal, but not move in small eigenvalue directions (and hence move in null space instead).

Why does the NTK overfit?

- Kernel regression theory: parameters first move in large eigenvalue directions of kernel matrix to learn signal, then move in small eigenvalue directions to overfit noise.
- Prior work [3] shows with $n \gtrsim d^k$ samples, large eigenvalue directions express degree $\leq k$ polynomials.
- We would like to move in large eigenvalue directions to learn the low degree signal, but not move in small eigenvalue directions (and hence move in null space instead).

Goal #1: Regularize to prevent movement in small eigenvalue directions.

Generalizing to Test Predictions

Generalizing to Test Predictions

Size of test predictions governed by NTK feature covariance matrix:

$$\boldsymbol{\Sigma} := \mathbb{E}[\phi(x)\phi(x)^T]$$

Generalizing to Test Predictions

Size of test predictions governed by NTK feature covariance matrix:

$$\mathbf{\Sigma} := \mathbb{E}[\phi(x)\phi(x)^T]$$

Lemma (informal): Eigenspectrum of $\mathbf{\Sigma}$ can be partitioned into 3 groups:

Generalizing to Test Predictions

Size of test predictions governed by NTK feature covariance matrix:

$$\mathbf{\Sigma} := \mathbb{E}[\phi(x)\phi(x)^T]$$

Lemma (informal): Eigenspectrum of $\mathbf{\Sigma}$ can be partitioned into 3 groups:

$$\mathbf{\Sigma} = \begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2 & \mathbf{Q}_3\end{bmatrix} \begin{bmatrix}\mathbf{\Lambda}_1 & \cdot & \cdot \\ \cdot & \mathbf{\Lambda}_2 & \cdot \\ \cdot & \cdot & \mathbf{\Lambda}_3\end{bmatrix} \begin{bmatrix}\mathbf{Q}_1^T \\ \mathbf{Q}_2^T \\ \mathbf{Q}_3^T\end{bmatrix}$$

Generalizing to Test Predictions

Size of test predictions governed by NTK feature covariance matrix:

$$\mathbf{\Sigma} := \mathbb{E}[\phi(x)\phi(x)^T]$$

Lemma (informal): Eigenspectrum of $\mathbf{\Sigma}$ can be partitioned into 3 groups:

$$\mathbf{\Sigma} = \begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2 & \mathbf{Q}_3\end{bmatrix} \begin{bmatrix}\mathbf{\Lambda}_1 & \cdot & \cdot \\ \cdot & \mathbf{\Lambda}_2 & \cdot \\ \cdot & \cdot & \mathbf{\Lambda}_3\end{bmatrix} \begin{bmatrix}\mathbf{Q}_1^T \\ \mathbf{Q}_2^T \\ \mathbf{Q}_3^T\end{bmatrix}$$

→ Large, express degree $\leq k$ polynomials.

Generalizing to Test Predictions

Size of test predictions governed by NTK feature covariance matrix:

$$\mathbf{\Sigma} := \mathbb{E}[\phi(x)\phi(x)^T]$$

Lemma (informal): Eigenspectrum of $\mathbf{\Sigma}$ can be partitioned into 3 groups:

$$\mathbf{\Sigma} = \begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2 & \mathbf{Q}_3\end{bmatrix} \begin{bmatrix}\mathbf{\Lambda}_1 & \cdot & \cdot \\ \cdot & \mathbf{\Lambda}_2 & \cdot \\ \cdot & \cdot & \mathbf{\Lambda}_3\end{bmatrix} \begin{bmatrix}\mathbf{Q}_1^T \\ \mathbf{Q}_2^T \\ \mathbf{Q}_3^T\end{bmatrix}$$

→ Large, express degree $\leq k$ polynomials.

→ Medium, blow up test predictions (bad)

Generalizing to Test Predictions

Size of test predictions governed by NTK feature covariance matrix:

$$\mathbf{\Sigma} := \mathbb{E}[\phi(x)\phi(x)^T]$$

Lemma (informal): Eigenspectrum of $\mathbf{\Sigma}$ can be partitioned into 3 groups:

$$\mathbf{\Sigma} = \begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2 & \mathbf{Q}_3\end{bmatrix} \begin{bmatrix}\mathbf{\Lambda}_1 & \cdot & \cdot \\ \cdot & \mathbf{\Lambda}_2 & \cdot \\ \cdot & \cdot & \mathbf{\Lambda}_3\end{bmatrix} \begin{bmatrix}\mathbf{Q}_1^T \\ \mathbf{Q}_2^T \\ \mathbf{Q}_3^T\end{bmatrix}$$

- $\mathbf{\Lambda}_1$ → Large, express degree $\leq k$ polynomials.
- $\mathbf{\Lambda}_2$ → Medium, blow up test predictions (bad)
- $\mathbf{\Lambda}_3$ → Small, large movement minimally effects predictions (good)

Generalizing to Test Predictions

Size of test predictions governed by NTK feature covariance matrix:

$$\boldsymbol{\Sigma} := \mathbb{E}[\phi(x)\phi(x)^T]$$

Lemma (informal): Eigenspectrum of $\boldsymbol{\Sigma}$ can be partitioned into 3 groups:

$$\boldsymbol{\Sigma} = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 & \mathbf{Q}_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{\Lambda}_1 & \cdot & \cdot \\ \cdot & \boldsymbol{\Lambda}_2 & \cdot \\ \cdot & \cdot & \boldsymbol{\Lambda}_3 \end{bmatrix} \begin{bmatrix} \mathbf{Q}_1^T \\ \mathbf{Q}_2^T \\ \mathbf{Q}_3^T \end{bmatrix}$$

Eigenvalue gap

- $\mathbf{Q}_1^T$ → Large, express degree $\leq k$ polynomials.
- $\mathbf{Q}_2^T$ → Medium, blow up test predictions (bad)
- $\mathbf{Q}_3^T$ → Small, large movement minimally effects predictions (good)

Generalizing to Test Predictions

Size of test predictions governed by NTK feature covariance matrix:

$$\mathbf{\Sigma} := \mathbb{E}[\phi(x)\phi(x)^T]$$

Lemma (informal): Eigenspectrum of $\mathbf{\Sigma}$ can be partitioned into 3 groups:

$$\mathbf{\Sigma} = \begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2 & \mathbf{Q}_3\end{bmatrix} \begin{bmatrix}\mathbf{\Lambda}_1 & \cdot & \cdot \\ \cdot & \mathbf{\Lambda}_2 & \cdot \\ \cdot & \cdot & \mathbf{\Lambda}_3\end{bmatrix} \begin{bmatrix}\mathbf{Q}_1^T \\ \mathbf{Q}_2^T \\ \mathbf{Q}_3^T\end{bmatrix}$$

Eigenvalue gap

- $\mathbf{Q}_1^T$ → Large, express degree $\leq k$ polynomials.
- $\mathbf{Q}_2^T$ → Medium, blow up test predictions (bad)
- $\mathbf{Q}_3^T$ → Small, large movement minimally effects predictions (good)

Goal #2: Move in $\mathbf{Q}_3$ directions, but minimally in $\mathbf{Q}_2$ directions.

The Quadratic NTK

The Quadratic NTK

- When $||\mathbf{W} - \mathbf{W}_0||_F$ is large, we can no longer linearize. Instead, require *second-order* Taylor expansion:

The Quadratic NTK

- When $\|\mathbf{W} - \mathbf{W}_0\|_F$ is large, we can no longer linearize. Instead, require *second-order* Taylor expansion:

$$f(x; \mathbf{W}) \approx \underbrace{\langle \mathbf{W} - \mathbf{W}_0, \nabla_{\mathbf{W}} f(x, \mathbf{W}_0) \rangle}_{\text{NTK}} + \frac{1}{2} \underbrace{(\mathbf{W} - \mathbf{W}_0)^T \nabla^2_{\mathbf{W}} f(x; \mathbf{W}_0)(\mathbf{W} - \mathbf{W}_0)}_{\text{QuadNTK}}$$

The Quadratic NTK

- When $\|\mathbf{W} - \mathbf{W}_0\|_F$ is large, we can no longer linearize. Instead, require *second-order* Taylor expansion:

$$f(x; \mathbf{W}) \approx \underbrace{\langle \mathbf{W} - \mathbf{W}_0, \nabla_{\mathbf{W}} f(x, \mathbf{W}_0) \rangle}_{\text{NTK}} + \frac{1}{2} \underbrace{(\mathbf{W} - \mathbf{W}_0)^T \nabla^2_{\mathbf{W}} f(x; \mathbf{W}_0)(\mathbf{W} - \mathbf{W}_0)}_{\text{QuadNTK}}$$

- Pros: [4] shows QuadNTK learns low-rank polynomials with improved sample complexity.

The Quadratic NTK

- When $\|\mathbf{W} - \mathbf{W}_0\|_F$ is large, we can no longer linearize. Instead, require *second-order* Taylor expansion:

$$f(x; \mathbf{W}) \approx \underbrace{\langle \mathbf{W} - \mathbf{W}_0, \nabla_{\mathbf{W}} f(x, \mathbf{W}_0) \rangle}_{\text{NTK}} + \frac{1}{2} \underbrace{(\mathbf{W} - \mathbf{W}_0)^T \nabla^2_{\mathbf{W}} f(x; \mathbf{W}_0)(\mathbf{W} - \mathbf{W}_0)}_{\text{QuadNTK}}$$

- Pros: [4] shows QuadNTK learns low-rank polynomials with improved sample complexity.

- Cons: [4] uses randomness to “delete” linear term. QuadNTK has poor sample complexity for learning arbitrary (dense) polynomials.

The Quadratic NTK

- When $\|\mathbf{W} - \mathbf{W}_0\|_F$ is large, we can no longer linearize. Instead, require *second-order* Taylor expansion:

$$f(x; \mathbf{W}) \approx \underbrace{\langle \mathbf{W} - \mathbf{W}_0, \nabla_{\mathbf{W}} f(x, \mathbf{W}_0) \rangle}_{\text{NTK}} + \frac{1}{2} \underbrace{(\mathbf{W} - \mathbf{W}_0)^T \nabla^2_{\mathbf{W}} f(x; \mathbf{W}_0)(\mathbf{W} - \mathbf{W}_0)}_{\text{QuadNTK}}$$

- Pros: [4] shows QuadNTK learns low-rank polynomials with improved sample complexity.

- Cons: [4] uses randomness to "delete" linear term. QuadNTK has poor sample complexity for learning arbitrary (dense) polynomials.

NTK is minimax optimal for **dense polynomials**, QuadNTK can learn **sparse polynomials**.
Question: Can we jointly use **both** terms to learn a larger class of functions?

Main Theorem

Main Theorem

Low-rank plus Sparse Signal: $f^*(x) = f_{\leq k}(x) + f_{sp}(x)$,

- $f_{\leq k}$ is dense degree k polynomial (low-degree term)
- $f_{sp}(x) = \sum_{j=1}^{R} (\beta_j^T x)^{k+1}$ (sparse term)

Main Theorem

Low-rank plus Sparse Signal: $f^*(x) = f_{\leq k}(x) + f_{sp}(x)$,

- $f_{\leq k}$ is dense degree k polynomial (low-degree term)
- $f_{sp}(x) = \sum_{j=1}^{R} (\beta_j^T x)^{k+1}$ (sparse term)

Algorithm: Perturbed (noisy) GD on regularized loss $L_\lambda(\mathbf{W})$. Regularizer is chosen specifically to encourage movement in "good" directions and prevent movement in "bad" directions.

Main Theorem

Low-rank plus Sparse Signal: $f^*(x) = f_{\leq k}(x) + f_{sp}(x)$,

- $f_{\leq k}$ is dense degree k polynomial (low-degree term)
- $f_{sp}(x) = \sum_{j=1}^{R} (\beta_j^T x)^{k+1}$ (sparse term)

Algorithm: Perturbed (noisy) GD on regularized loss $L_\lambda(\mathbf{W})$. Regularizer is chosen specifically to encourage movement in “good” directions and prevent movement in “bad” directions.

Theorem (informal): With $n \gtrsim d^k$ samples and width $m = \text{poly}(d)$, perturbed GD on the regularized loss converges to a minimizer with small test loss.

Main Theorem

Low-rank plus Sparse Signal: $f^*(x) = f_{\leq k}(x) + f_{sp}(x)$,

- $f_{\leq k}$ is dense degree k polynomial (low-degree term)
- $f_{sp}(x) = \sum_{j=1}^{R} (\beta_j^T x)^{k+1}$ (sparse term)

Algorithm: Perturbed (noisy) GD on regularized loss $L_\lambda(\mathbf{W})$. Regularizer is chosen specifically to encourage movement in "good" directions and prevent movement in "bad" directions.

Theorem (informal): With $n \gtrsim d^k$ samples and width $m = \text{poly}(d)$, perturbed GD on the regularized loss converges to a minimizer with small test loss.

- NTK or QuadNTK alone require $n \geq \Omega(d^{k+1})$ samples to learn f^*.

Main Theorem

Low-rank plus Sparse Signal: $f^*(x) = f_{\leq k}(x) + f_{sp}(x)$,

- $f_{\leq k}$ is dense degree k polynomial (low-degree term)
- $f_{sp}(x) = \sum_{j=1}^{R} (\beta_j^T x)^{k+1}$ (sparse term)

Algorithm: Perturbed (noisy) GD on regularized loss $L_\lambda(\mathbf{W})$. Regularizer is chosen specifically to encourage movement in "good" directions and prevent movement in "bad" directions.

Theorem (informal): With $n \gtrsim d^k$ samples and width $m = \text{poly}(d)$, perturbed GD on the regularized loss converges to a minimizer with small test loss.

- NTK or QuadNTK alone require $n \geq \Omega(d^{k+1})$ samples to learn f^*.
- We obtain better sample complexity than either NTK or QuadNTK on their own $\implies$ **best of both worlds!**

Proof Sketch

Proof Sketch

1. Expressivity

There exist network weights $\mathbf{W}^*$ with small regularized training loss.

Proof Sketch

1. Expressivity

There exist network weights $\mathbf{W}^*$ with small regularized training loss.

2. Landscape

All second order stationary points of the regularized training loss are global minima.

Proof Sketch

1. Expressivity

There exist network weights $\mathbf{W}^*$ with small regularized training loss.

2. Landscape

All second order stationary points of the regularized training loss are global minima.

3. Optimization

GD will find a second order stationary point in poly time.

Proof Sketch

1. Expressivity

There exist network weights $\mathbf{W}^*$ with small regularized training loss.

2. Landscape

All second order stationary points of the regularized training loss are global minima.

3. Optimization

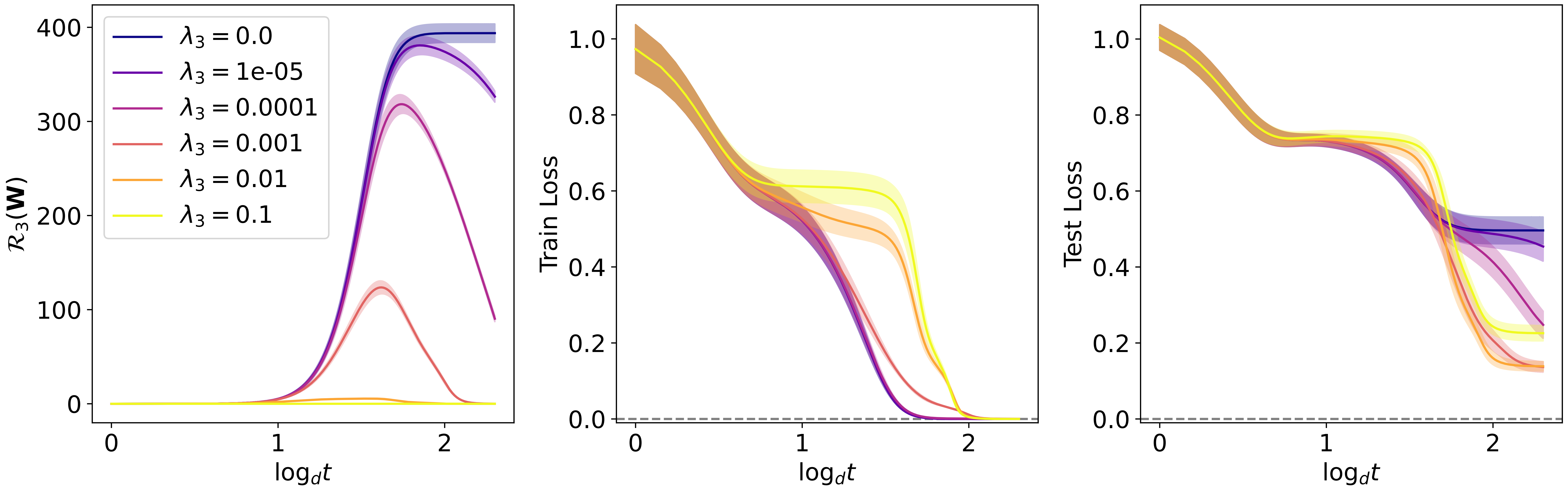
GD will find a second order stationary point in poly time.

4. Generalization

Small regularized training loss implies small population loss.

Experiments

$f_L + f_Q$ trained on a degree 2 signal with $d^{1.5}$ samples



Conclusion

Conclusion

- Feature covariance matrix has 3 sets of directions: Top directions fit degree $\leq k$ signal, middle directions are “bad” (NTK overfits or test predictions blow up), bottom directions are “good” and allow for large movement.

Conclusion

- Feature covariance matrix has 3 sets of directions: Top directions fit degree $\leq k$ signal, middle directions are “bad” (NTK overfits or test predictions blow up), bottom directions are “good” and allow for large movement.

- Regularization encourages movement in good directions.

Conclusion

- Feature covariance matrix has 3 sets of directions: Top directions fit degree $\leq k$ signal, middle directions are "bad" (NTK overfits or test predictions blow up), bottom directions are "good" and allow for large movement.
- Regularization encourages movement in good directions.
- QuadNTK and NTK can jointly fit low-degree plus sparse signal.

Conclusion

- Feature covariance matrix has 3 sets of directions: Top directions fit degree $\leq k$ signal, middle directions are “bad” (NTK overfits or test predictions blow up), bottom directions are “good” and allow for large movement.
- Regularization encourages movement in good directions.
- QuadNTK and NTK can jointly fit low-degree plus sparse signal.
- End-to-end convergence and generalization guarantee with provable sample complexity improvement.

Conclusion

- Feature covariance matrix has 3 sets of directions: Top directions fit degree $\leq k$ signal, middle directions are “bad” (NTK overfits or test predictions blow up), bottom directions are “good” and allow for large movement.
- Regularization encourages movement in good directions.
- QuadNTK and NTK can jointly fit low-degree plus sparse signal.
- End-to-end convergence and generalization guarantee with provable sample complexity improvement.
- **Future Directions:**

Conclusion

- Feature covariance matrix has 3 sets of directions: Top directions fit degree $\leq k$ signal, middle directions are “bad” (NTK overfits or test predictions blow up), bottom directions are “good” and allow for large movement.
- Regularization encourages movement in good directions.
- QuadNTK and NTK can jointly fit low-degree plus sparse signal.
- End-to-end convergence and generalization guarantee with provable sample complexity improvement.
- **Future Directions:**
  - How to leverage higher-order terms in Taylor expansion?

Conclusion

- Feature covariance matrix has 3 sets of directions: Top directions fit degree $\leq k$ signal, middle directions are “bad” (NTK overfits or test predictions blow up), bottom directions are “good” and allow for large movement.
- Regularization encourages movement in good directions.
- QuadNTK and NTK can jointly fit low-degree plus sparse signal.
- End-to-end convergence and generalization guarantee with provable sample complexity improvement.
- **Future Directions:**
    - How to leverage higher-order terms in Taylor expansion?
    - How to increase depth to jointly learn a hierarchical representation?

Conclusion

- Feature covariance matrix has 3 sets of directions: Top directions fit degree $\leq k$ signal, middle directions are “bad” (NTK overfits or test predictions blow up), bottom directions are “good” and allow for large movement.
- Regularization encourages movement in good directions.
- QuadNTK and NTK can jointly fit low-degree plus sparse signal.
- End-to-end convergence and generalization guarantee with provable sample complexity improvement.
- **Future Directions:**
  - How to leverage higher-order terms in Taylor expansion?
  - How to increase depth to jointly learn a hierarchical representation?
  - How does the QuadNTK relate to feature learning?

Thanks for Listening!

References:

[1] Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In Advances in Neural Information Processing Systems (NeurIPS), 2019.
[2] Behrooz Ghorbani, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. Linearized two-layers neural networks in high dimension. The Annals of Statistics, 49:1029–1054, 2021
[3] Andrea Montanari and Yiqiao Zhong. The interpolation phase transition in neural networks: Memorization and generalization under lazy training, 2020. URL https://arxiv.org/ abs/2007.12826.
[4] Yu Bai and Jason D. Lee. Beyond linearization: On quadratic and higher-order approximation of wide neural networks. In International Conference on Learning Representations (ICLR), 2020.