

Neural Tangent Kernel

W

Neural Tangent Kernel Formula

L-layer NN. For $h = 1, \dots, L$:

$$\Sigma^{(0)}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{x}',$$

$$\mathbf{\Lambda}^{(h)}(\mathbf{x}, \mathbf{x}') = \begin{pmatrix} \Sigma^{(h-1)}(\mathbf{x}, \mathbf{x}) & \Sigma^{(h-1)}(\mathbf{x}, \mathbf{x}') \\ \Sigma^{(h-1)}(\mathbf{x}', \mathbf{x}) & \Sigma^{(h-1)}(\mathbf{x}', \mathbf{x}') \end{pmatrix} \in \mathbb{R}^{2 \times 2},$$

$$\Sigma^{(h)}(\mathbf{x}, \mathbf{x}') = c_\sigma \mathbb{E}_{(u,v) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Lambda}^{(h)})} [\sigma(u) \sigma(v)],$$

$$\dot{\Sigma}^{(h)}(\mathbf{x}, \mathbf{x}') = c_\sigma \mathbb{E}_{(u,v) \sim \mathcal{N}(\mathbf{0}, \mathbf{\Lambda}^{(h)})} [\dot{\sigma}(u) \dot{\sigma}(v)].$$

Final output:

$$\Theta^{(L)}(\mathbf{x}, \mathbf{x}') = \sum_{h=1}^{L+1} \left(\Sigma^{(h-1)}(\mathbf{x}, \mathbf{x}') \cdot \prod_{h'=h}^{L+1} \dot{\Sigma}^{(h')}(\mathbf{x}, \mathbf{x}') \right)$$

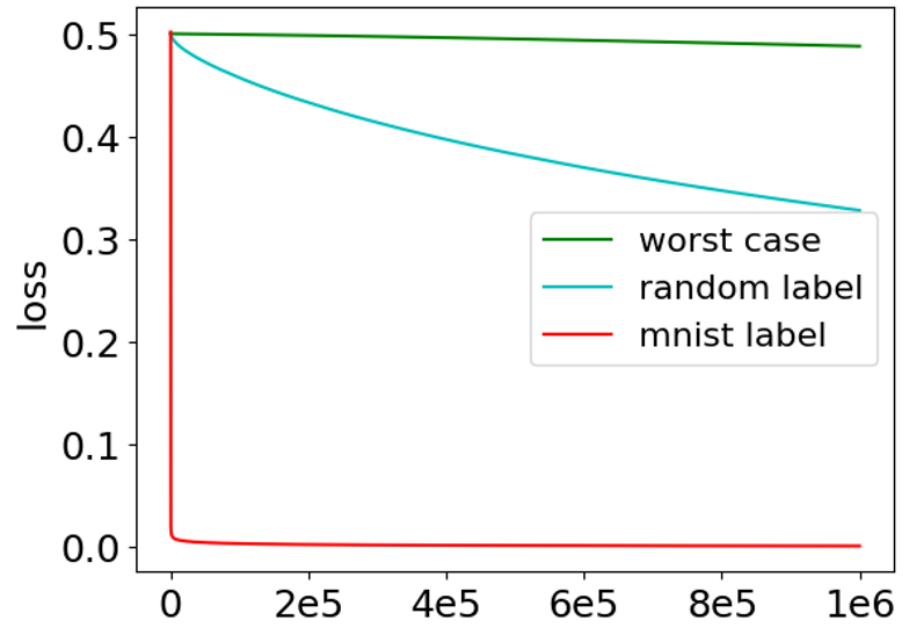


L-layer recursion.
Encodes NN's architecture.

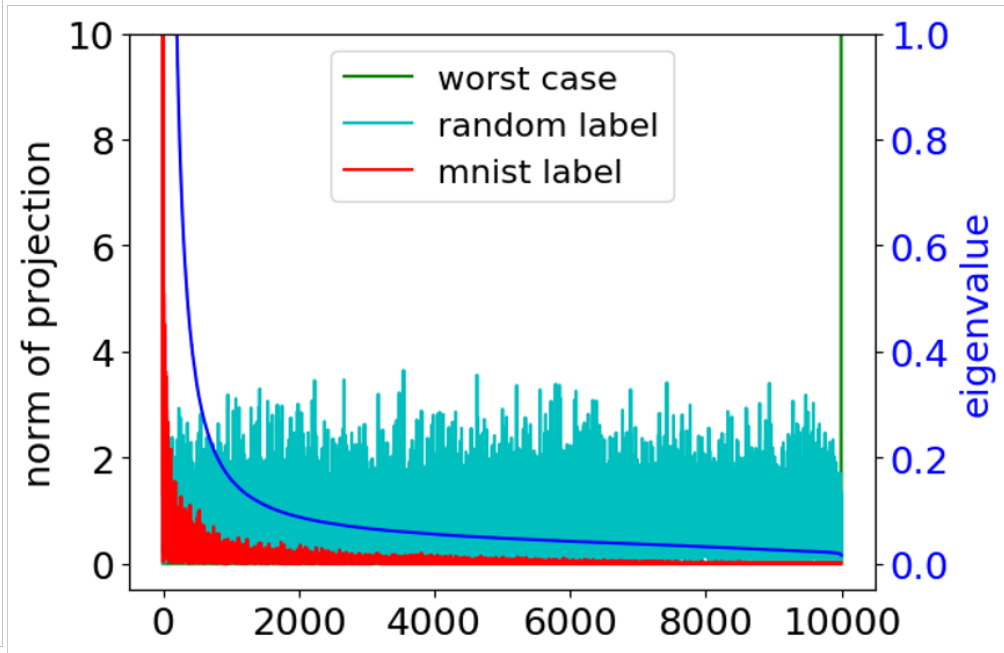


Dependency on the derivative:
Gradient descent algorithm.

What determines the convergence rate?



Convergence Rate



Projections

Neural Tangent Kernel

Recipe for designing new kernels

$$f_{\text{NN}}(\theta_{\text{NN}}, x) \rightarrow k(x, x') = \mathbb{E}_{\theta_{\text{NN}} \sim \mathcal{W}} \left[\left\langle \frac{\partial f_{\text{NN}}(\theta_{\text{NN}}, x)}{\partial \theta_{\text{NN}}}, \frac{\partial f_{\text{NN}}(\theta_{\text{NN}}, x')}{\partial \theta_{\text{NN}}} \right\rangle \right]$$

Transform a neural network of **any architecture to a kernel!**

Fully-connected NN → Fully-connected NTK

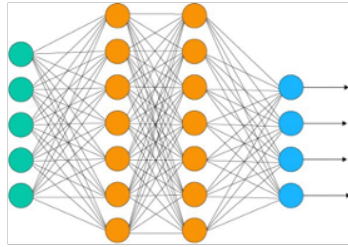
Convolutional NN → Convolutional NTK

Graph NN → Graph NTK

.....

Fully-Connect NTK

$$\begin{pmatrix} -0.1 \\ 0.2 \\ \dots \\ 0.9 \end{pmatrix}$$



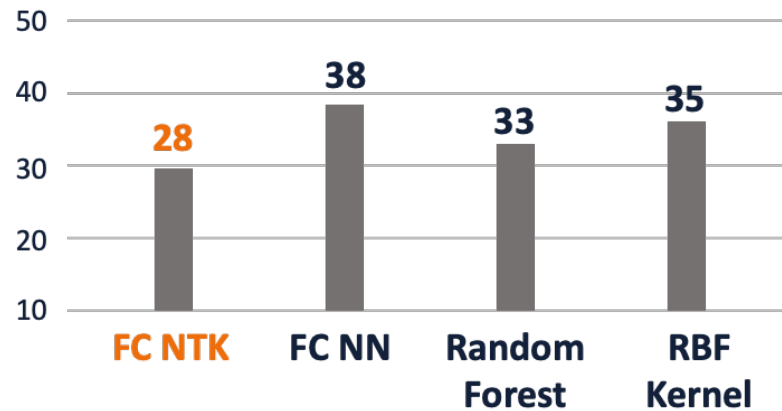
$$\mathcal{K} \left(\begin{pmatrix} -0.1 \\ 0.2 \\ \dots \\ 0.9 \end{pmatrix}, \begin{pmatrix} -0.3 \\ 0.5 \\ \dots \\ -0.8 \end{pmatrix} \right)$$

Features

FC NN

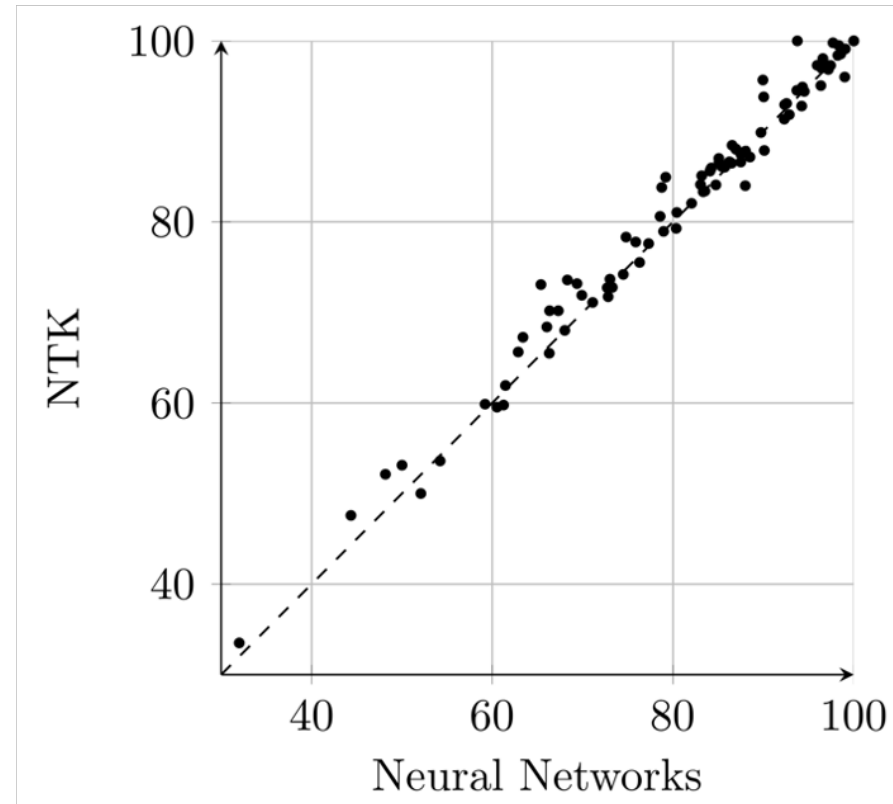
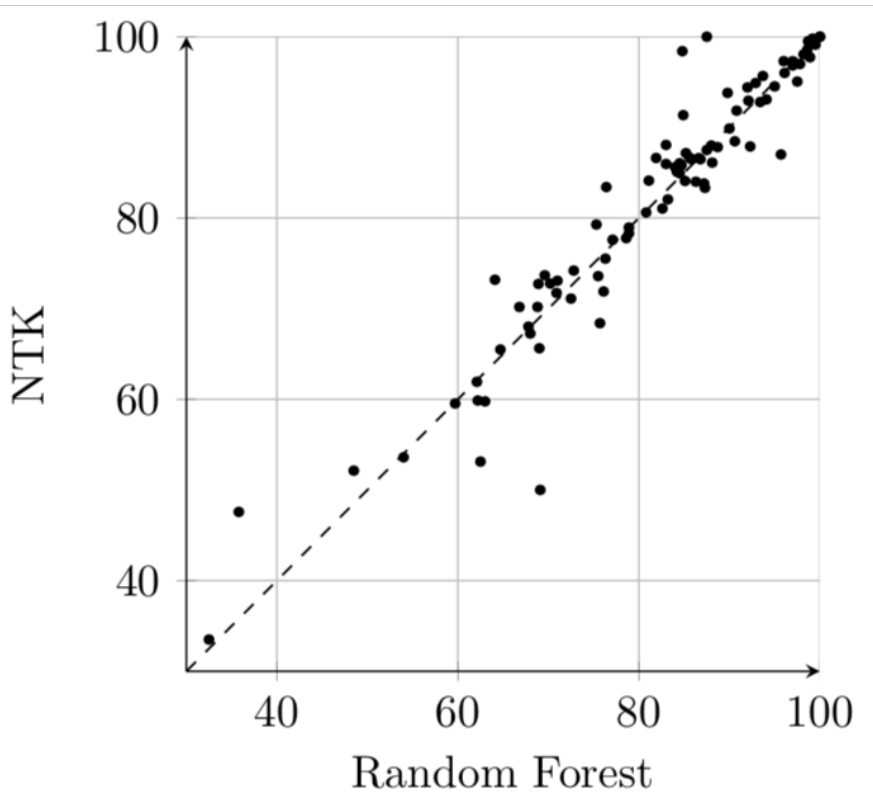
FC NTK

Avg Rank



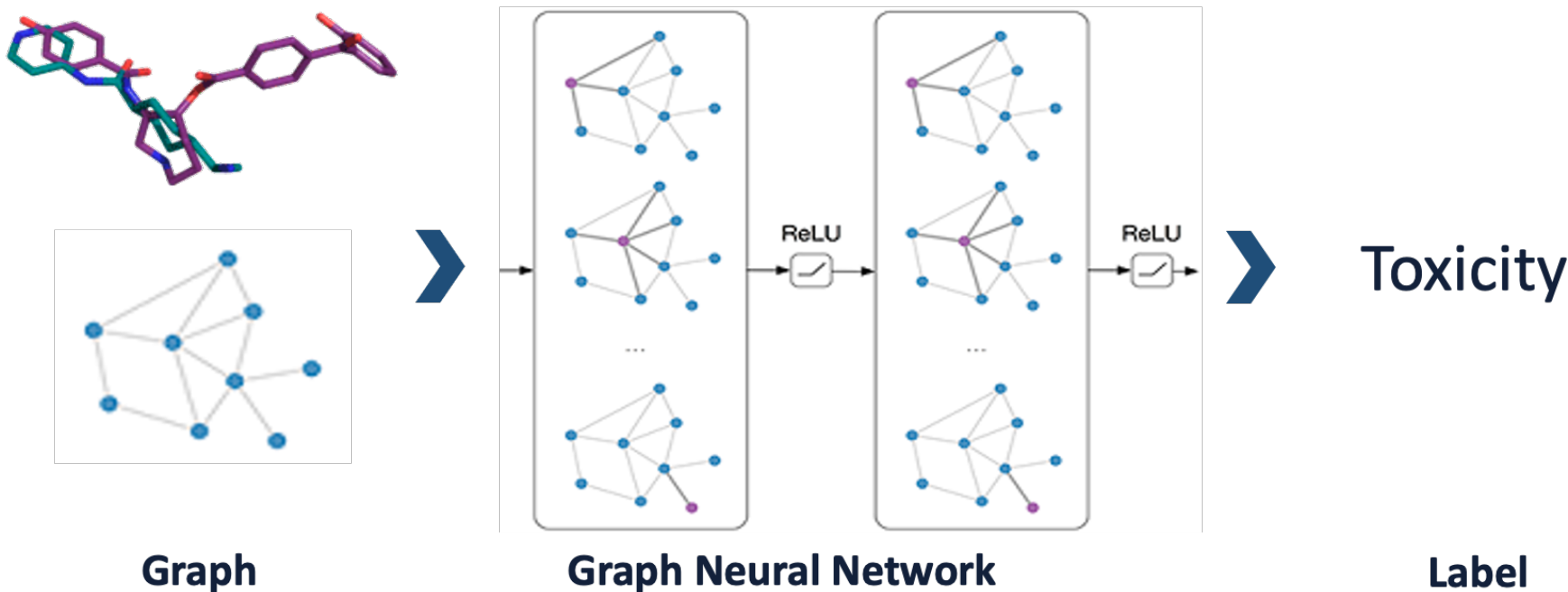
Classifier	Avg Acc	P95	PMA
FC NTK	82%	72%	96%
FC NN	81%	60%	95%
Random Forest	82%	68%	95%
RBF Kernel	81%	72%	94%

Pairwise Comparisons



Classification
Accuracy

Graph Neural Network



Graph Neural Tangent Kernel



Graph

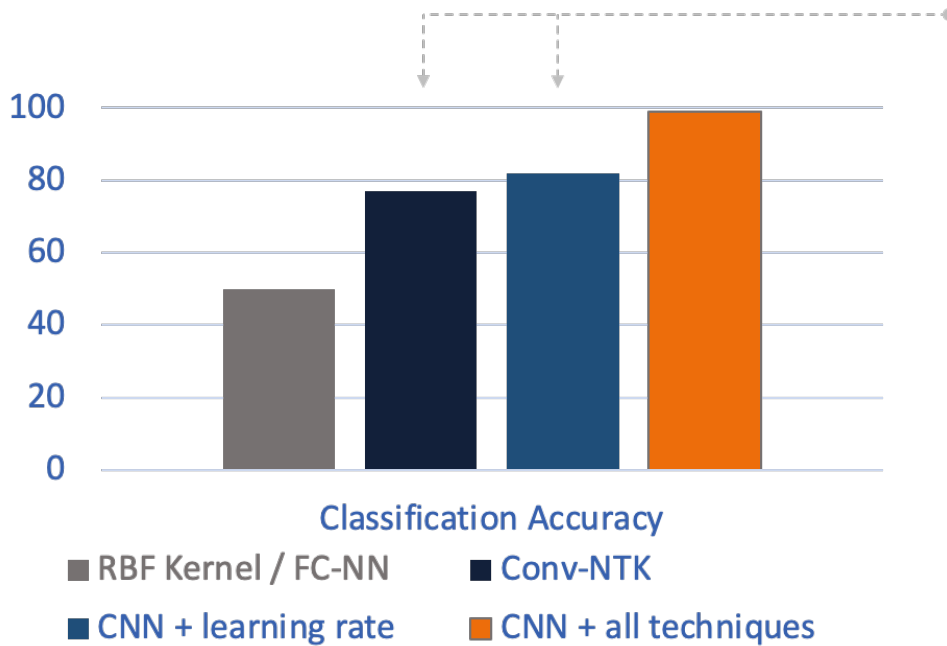
Graph NN

Graph NTK

	Method	COLLAB	IMDB-B	IMDB-M	PTC
GNN	GCN	79%	74%	51%	64%
	GIN	80%	75%	52%	65%
GK	WL	79%	74%	51%	60%
	GNTK	84%	77%	53%	68%

Gap between NN and NTK

CIFAR-10 Image Classification



Open Problems:

Why there is a gap:
finite-width?
learning rate?

Understanding techniques:

batch-norm
dropout
data-augmentation

...