On Exact Computation with an Infinitely Wide Neural Net
Arora et al. (2019)

Review by Adonis Jamal

CentraleSupélec

Context and Motivation

The Goal: Compute the exact performance of infinitely wide Convolutional Neural Networks (CNNs) without Monte Carlo approximations.
The Gap: Prior work could not combine exactness with CNN architectures (pooling was the bottleneck).

Method	Exact?	CNN Support?	Pooling?
Standard NTK [2]	Yes	No (FC only)	No
Monte Carlo Approx.	No	Yes	Yes
CNTK (This Paper)	**Yes**	**Yes**	**Yes**

Key Contribution: Derivation of the **Convolutional Neural Tangent Kernel (CNTK)** using a dynamic programming approach that runs efficiently on GPUs [1].

Theoretical Guarantee: Lazy Training

Theorem: As width $m \to \infty$, a fully-trained net is equivalent to Kernel Regression using the CNTK.

Gradient Flow

Training evolves as $\frac{du(t)}{dt} = -H(t)(u(t)-y)$, where $H(t)$ is the tangent kernel.

Frozen Kernel

Weights stay close to initialization ($\|W(t)-W(0)\|_F \to 0$), the kernel remains constant: $H(t) \approx H(0) = \Theta_{CNTK}$.

The Solution

$$f^*(x) = K(x, X_{\text{train}})K(X_{\text{train}}, X_{\text{train}})^{-1}Y_{\text{train}}$$

The Exact CNTK Algorithm

Input Images x, x' → Base Covariance $\Sigma^{(0)} = x^\top x'$ → **Layerwise Recursion** $\Sigma^{(h)}$ (Activation) $\dot{\Sigma}^{(h)}$ (Gradient) → Final Kernel $\Theta^{(L)}(x, x')$ → Regression $y^* = K_{te}K_{tr}^{-1}y$

Closed-Form ReLU Kernels

For input correlation $\rho = \frac{\Sigma_{12}}{\sqrt{\Sigma_{11}\Sigma_{22}}}$, the ReLU integrals have closed forms:

- Activation: $\Sigma = \frac{\sqrt{\Sigma_{11}\Sigma_{22}}}{2\pi}\left(\rho(\pi - \arccos\rho) + \sqrt{1-\rho^2}\right)$
- Gradient: $\dot{\Sigma} = \frac{1}{2\pi}(\pi - \arccos\rho)$

CNTK Recurrence (Dynamic Programming)

The final kernel $\Theta^{(L)}$ is computed recursively, combining the activation ($\Sigma \propto K$) and gradient ($\dot{\Sigma} \propto \dot{K}$) covariances:

$$\Theta^{(h)} = \underbrace{\dot{K}^{(h)} \odot \Theta^{(h-1)}}_{\text{Backward Pass}} + \underbrace{K^{(h)}}_{\text{Forward Pass}}$$

Global Average Pooling (GAP)

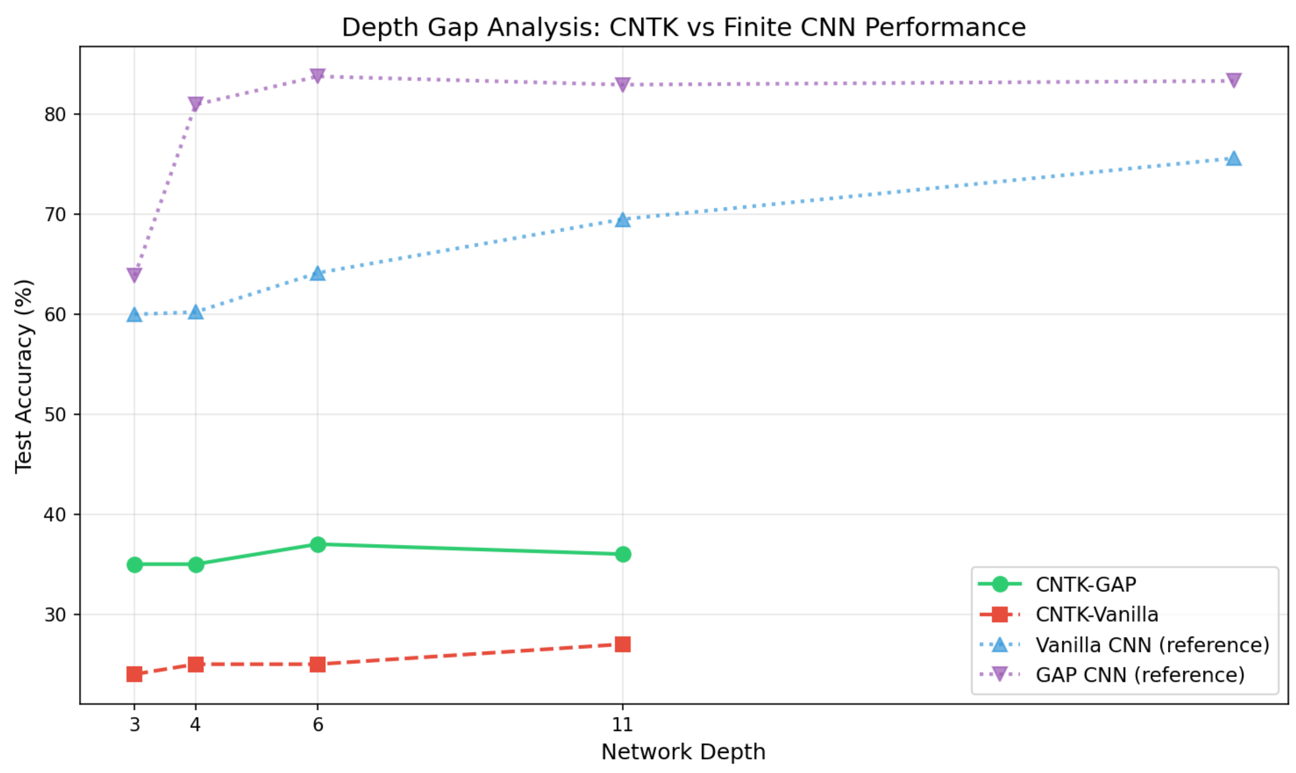
Standard CNNs flatten the output, hurting shift invariance. GAP averages the kernel over spatial dimensions ($P \times P$) at the final layer, making the kernel translation invariant and significantly boosting accuracy.

Reproduction and Results

Setup: PyTorch implementation on CIFAR-10 subset ($N_{train} = 200$, $N_{test} = 100$).
Computational Complexity: $O(N^2P^2Q^2L)$ time, quadratic in dataset size, limiting scalability.

Architecture	Paper Acc.	Reproduced Acc.
CNTK-Vanilla	66.03%	25.00%
CNTK-GAP	**76.73%**	**37.00%**

Depth Gap: While Finite CNNs benefit from depth, the infinite CNTK performance saturates or even degrades after ≈ 11 layers.



**Lower absolute accuracy is expected due to small dataset size and lack of kernel regularization; qualitative trends match the paper.*

Critical Discussion

Performance Gap and Scalability

Despite exact computation, CNTK (77%) still trails Finite CNNs (83%). Moreover, exact CNTK scales as $O(N^2)$, making it computationally intractable for datasets like ImageNet without approximation.

Feature Learning

The "Lazy Training" regime fixes features at initialization. The performance gap suggests that *feature adaptation* (weights moving far from init) is crucial for SOTA performance.

Impact

Despite the performance gap, CNTK provides a deterministic way to study deep learning optimization without the noise of SGD sampling or initialization.

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, 2019.

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