Non-Gaussian Tensor Programs

NeurIPS'2022

Eugene Golikov* Greg Yang†

* École Polytechnique Fédérale de Lausanne, Switzerland
† Microsoft research, USA

Recall the limit theorems in probability theory:

1. **Law of Large Numbers**:
   An average of n iid random variables converges to their mean as $n \to \infty$.

2. **Central Limit Theorem**:
   An average of n iid zero-mean random variables scaled by $\sqrt{n}$ converges to a zero-mean Gaussian as $n \to \infty$.

Both state **universality**: the limits do not depend on the distribution of random variables albeit several first moments.

A universality principle for **neural networks**?

Conjecture

As width tends to infinity, two different iid random initializations induce identical training behavior as long as they sample weights with the same mean and variance.

When the above principle works:

$$f(\xi) = \frac{1}{n} V^\top \phi(g(\xi)), \quad g(\xi) = W\phi(U\xi), \qquad \xi \in \mathbb{R}, \quad U, V \in \mathbb{R}^n, \quad W \in \mathbb{R}^{n \times n}.$$

Consider two alternatives (G) and (R) for sampling W:

$$\text{(R)} \;\; W_{\alpha\beta} \sim \mathrm{Unif}([-\sqrt{3/n}, \sqrt{3/n}]) \qquad \text{or} \qquad \text{(G)} \;\; W_{\alpha\beta} \sim \mathcal{N}(0, 1/n).$$

The distribution of $g_\alpha(\xi)$ tends to the same Gaussian by CLT for both (R) and (G)!

When the above principle fails:

$$f(\xi) = \frac{1}{n} U^\top \phi(U\xi), \qquad \xi \in \mathbb{R}, \quad U \in \mathbb{R}^n.$$

Consider two alternatives (G) and (R) for sampling U:

$$\text{(R)} \quad U_\alpha = \pm 1 \text{ with prob. } 1/2 \qquad \text{or} \qquad \text{(G)} \quad U_\alpha \sim \mathcal{N}(0,1).$$

Suppose $\phi(x) = x 1_{[-\frac{1}{2},\frac{1}{2}]}(x)$. Then

$$f(1) = 0 \text{ with init (R)} \qquad \text{but} \qquad f(1) \to \mathbb{E}_z z\phi(z) > 0 \text{ with init (G)}$$

as $n \to \infty$, where $z \sim \mathcal{N}(0,1)$.

Take-away: Universality fails for **vector-shaped** weights.

Universality also fails for **scalar-shaped** weights:

$$f(\xi) = b + \frac{1}{n} U^\top \phi(U\xi), \qquad \xi \in \mathbb{R}, \quad b \in \mathbb{R}, \quad U \in \mathbb{R}^n.$$

The distribution of $f(\xi)$ depends on the distribution of b.

Universality also fails for **scalar-shaped** weights:

$$f(\xi) = b + \frac{1}{n} U^\top \phi(U\xi), \qquad \xi \in \mathbb{R}, \quad b \in \mathbb{R}, \quad U \in \mathbb{R}^n.$$

The distribution of $f(\xi)$ depends on the distribution of b.

Conjecture

As width tends to infinity, two different iid random initializations induce identical training behavior as long as

1. *They sample* **scalar***-shaped and* **vector***-shaped weights the same way, and*
2. *They sample* **matrix***-shaped weights with the same mean and variance.*

Definition ([Yang et al., 2022])
Let P be a parameter tensor in a neural network of any architecture. As width becomes large,

- if P's size remains constant, then we say P is **scalar-like**;

Definition ([Yang et al., 2022])
Let P be a parameter tensor in a neural network of any architecture. As width becomes large,

- if P's size remains constant, then we say P is **scalar-like**;
- if exactly one dimension of P becomes large, we say P is **vector-like**;

Definition ([Yang et al., 2022])
Let P be a parameter tensor in a neural network of any architecture. As width becomes large,

- if P's size remains constant, then we say P is **scalar-like**;
- if exactly one dimension of P becomes large, we say P is **vector-like**;
- if exactly two dimensions of P becomes large, we say P is **matrix-like**.

Principle (Universality in General Neural Network Initialization)

As width becomes large, two different iid random initializations of a neural network of any architecture induce identical training behavior as long as

1. *They sample* **scalar**- *and* **vector**-*like weights the same way, and*
2. *They sample* **matrix**-*like weights with the same mean and variance.*

It is a corollary of a universality principle for **tensor programs**.

We are given:

- L matrices $A^1, \ldots, A^L \in \mathbb{R}^{n \times n}$;

We are given:

- L matrices $A^1, \ldots, A^L \in \mathbb{R}^{n \times n}$;
- M_0 initial vectors $g^1, \ldots, g^{M_0} \in \mathbb{R}^n$;

We are given:

- L matrices $A^1, \ldots, A^L \in \mathbb{R}^{n \times n}$;
- M_0 initial vectors $g^1, \ldots, g^{M_0} \in \mathbb{R}^n$;
- M_0 initial scalars $c^1, \ldots, c^{M_0} \in \mathbb{R}$.

We are given:

- L matrices $A^1, \ldots, A^L \in \mathbb{R}^{n\times n}$;
- M_0 initial vectors $g^1, \ldots, g^{M_0} \in \mathbb{R}^n$;
- M_0 initial scalars $c^1, \ldots, c^{M_0} \in \mathbb{R}$.

A **tensor program** generates new vectors and scalars iteratively:

$$g^i_\alpha \leftarrow \sum_{\beta=1}^{n} W^i_{\alpha\beta} x^i_\beta, \quad c^i \leftarrow \frac{1}{n}\sum_{\beta=1}^{n} x^i_\beta, \quad \text{where } x^i_\alpha = \phi^i(g^1_\alpha, \ldots, g^{i-1}_\alpha; c^1, \ldots, c^{i-1}), \qquad (1)$$

and

- ϕ^i is a scalar function;
- $W^i = A^j$ or $W^i = A^{j\top}$ for some $j \in [L]$.

Tensor programs can express

1. A **forward pass** for a neural network of any architecture [Yang, 2019];

Tensor programs can express

1. A **forward pass** for a neural network of any architecture [Yang, 2019];
2. A **backward pass** for a neural network of any architecture [Yang, 2020a];

Tensor programs can express

1. A **forward pass** for a neural network of any architecture [Yang, 2019];
2. A **backward pass** for a neural network of any architecture [Yang, 2020a];
3. **Any number of gradient descent steps** [Yang and Littwin, 2021, Yang and Hu, 2021]:

Tensor programs can express

1. A **forward pass** for a neural network of any architecture [Yang, 2019];
2. A **backward pass** for a neural network of any architecture [Yang, 2020a];
3. **Any number of gradient descent steps** [Yang and Littwin, 2021, Yang and Hu, 2021]:

Proposition

Let the network have k outputs. For each t and input ξ, the network output after t GD steps $f_t(\xi)$ can be expressed as a set of k scalars c in some tensor program.

Tensor programs can express

1. A **forward pass** for a neural network of any architecture [Yang, 2019];
2. A **backward pass** for a neural network of any architecture [Yang, 2020a];
3. **Any number of gradient descent steps** [Yang and Littwin, 2021, Yang and Hu, 2021]:

Proposition

Let the network have k outputs. For each t and input ξ, the network output after t GD steps $f_t(\xi)$ can be expressed as a set of k scalars c in some tensor program.

Conjecture (Universality for tensor programs)

Scalars $c^1, \ldots, c^M$ in a tensor program converge and their limits depend only on mean and variance of the distribution of entries of $A^1, \ldots, A^L$.

Theorem (Gaussian Master Theorem, [Yang, 2020b])
Consider a Tensor Program with M vectors $g^1, \ldots, g^M \in \mathbb{R}^n$ and scalars $c^1, \ldots, c^M$. Suppose

1. *All initial vectors $g^1, \ldots, g^{M_0}$ have iid entries from $\mathcal{N}(0, 1)$;*

[1]A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is called pseudo-Lipschitz if there exist $C, d > 0$ such that for any $x, y \in \mathbb{R}^n$, $\|f(x) - f(y)\| \leq C\|x - y\|(1 + \|x\|^d + \|y\|^d)$.

Theorem (Gaussian Master Theorem, [Yang, 2020b])
Consider a Tensor Program with M vectors $g^1, \ldots, g^M \in \mathbb{R}^n$ and scalars $c^1, \ldots, c^M$. Suppose

1. *All initial vectors $g^1, \ldots, g^{M_0}$ have iid entries from $\mathcal{N}(0, 1)$;*
2. *All matrices A^i have iid entries from $\mathcal{N}(0, n^{-1})$;*

[1] A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is called pseudo-Lipschitz if there exist $C, d > 0$ such that for any $x, y \in \mathbb{R}^n$, $\|f(x) - f(y)\| \leq C\|x - y\|(1 + \|x\|^d + \|y\|^d)$.

Theorem (Gaussian Master Theorem, [Yang, 2020b])
Consider a Tensor Program with M vectors $g^1, \ldots, g^M \in \mathbb{R}^n$ and scalars $c^1, \ldots, c^M$. Suppose

1. *All initial vectors $g^1, \ldots, g^{M_0}$ have iid entries from $\mathcal{N}(0, 1)$;*
2. *All matrices A^i have iid entries from $\mathcal{N}(0, n^{-1})$;*
3. *All the nonlinearities ϕ^i are pseudo-Lipschitz[1];*

[1] A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is called pseudo-Lipschitz if there exist $C, d > 0$ such that for any $x, y \in \mathbb{R}^n$, $\|f(x) - f(y)\| \leq C\|x - y\|(1 + \|x\|^d + \|y\|^d)$.

Theorem (Gaussian Master Theorem, [Yang, 2020b])
Consider a Tensor Program with M vectors $g^1, \ldots, g^M \in \mathbb{R}^n$ and scalars $c^1, \ldots, c^M$. Suppose

1. *All initial vectors $g^1, \ldots, g^{M_0}$ have iid entries from $\mathcal{N}(0, 1)$;*
2. *All matrices A^i have iid entries from $\mathcal{N}(0, n^{-1})$;*
3. *All the nonlinearities ϕ^i are pseudo-Lipschitz[1];*
4. *All initial scalars $c^1, \ldots, c^{M_0}$ have almost sure limits as $n \to \infty$.*

[1] A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is called pseudo-Lipschitz if there exist $C, d > 0$ such that for any $x, y \in \mathbb{R}^n$, $\|f(x) - f(y)\| \leq C\|x - y\|(1 + \|x\|^d + \|y\|^d)$.

Theorem (Gaussian Master Theorem, [Yang, 2020b])
Consider a Tensor Program with M vectors $g^1, \dots, g^M \in \mathbb{R}^n$ and scalars $c^1, \dots, c^M$. Suppose

1. *All initial vectors $g^1, \dots, g^{M_0}$ have iid entries from $\mathcal{N}(0, 1)$;*
2. *All matrices A^i have iid entries from $\mathcal{N}(0, n^{-1})$;*
3. *All the nonlinearities ϕ^i are pseudo-Lipschitz*[1]*;*
4. *All initial scalars $c^1, \dots, c^{M_0}$ have almost sure limits as $n \to \infty$.*

Then, as $n \to \infty$, for any $i \in [M]$,

$$c^i \xrightarrow{a.s.} \mathring{c}^i, \tag{2}$$

where $\mathring{c}^i$ is a deterministic scalar given by a certain recurrent formula.

[1] A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is called pseudo-Lipschitz if there exist $C, d > 0$ such that for any $x, y \in \mathbb{R}^n$, $\|f(x) - f(y)\| \leq C\|x - y\|(1 + \|x\|^d + \|y\|^d)$.

Theorem (Non-Gaussian Master Theorem, ours)
Consider a Tensor Program with M vectors $g^1, \ldots, g^M \in \mathbb{R}^n$ and scalars $c^1, \ldots, c^M$. Suppose

1. *All initial vectors $g^1, \ldots, g^{M_0}$ have iid entries from $\mathcal{N}(0,1)$;*
2. *All matrices A^i have iid entries with zero mean, variance n^{-1}, and each k-th moment bounded by $\nu_k n^{-k/2}$;*
3. *All the nonlinearities ϕ^i are polynomially smooth*[2]*;*
4. *All initial scalars $c^1, \ldots, c^{M_0}$ have almost sure limits as $n \to \infty$ and all moments.*

Then, as $n \to \infty$, for any $i \in [M]$,

$$c^i \xrightarrow{\text{a.s. } \& \ L^p} \mathring{c}^i \quad \forall p \in [1, \infty) \tag{3}$$

for the same $\mathring{c}^i$ as in the Gaussian theorem.

[2]We call f polynomially smooth if it is smooth and each derivative of order $k \geq 0$ is polynomially bounded.

$$c^i \xrightarrow{\text{a.s. \& } L^p} \mathring{c}^i \quad \forall p \in [1, \infty) \tag{4}$$

for the same c^i no matter if matrix weights A are Gaussian or not:

Principle (Universality in Tensor Program Sampling)

As $n \to \infty$, two different iid random samplings of a TP's matrices and initial vectors result in identical limits of scalars as long as

1. *They sample all initial* **vectors** *and initial* **scalars** *the same way, and*
2. *They sample all* **matrix** *entries with the same variance and zero mean.*

Applications of Master theorem:

1. **NNGP correspondence**: Each pre-activation output of a neural network converges to a Gaussian process as width tends to infinity.[3]
2. **Convergence to a kernel method**: Under certain parameterization, SGD training dynamics converges to the training dynamics of a kernel method as width tends to infinity.[4]
3. **Random matrix theory**: Semi-circle and Marchenko-Pastur laws.
4. **Free Independence Principle**: at initialization, neural network's weights become freely independent from its hidden representations as width goes to infinity.[5]
5. **Hyperparameter transfer**: optimal training hyperparameters can be transfered from thin to wide nets under certain parameterization.[6]

---

[3] [Neal, 1995, Lee et al., 2017, Garriga-Alonso et al., 2018, Novak et al., 2018, Yang, 2019]
[4] [Jacot et al., 2018, Lee et al., 2019, Yang, 2020a, Yang and Littwin, 2021]
[5] [Yang, 2020b]
[6] [Yang and Hu, 2021, Yang et al., 2022]

Garriga-Alonso, A., Rasmussen, C. E., and Aitchison, L. (2018).
Deep convolutional networks as shallow gaussian processes.
arXiv preprint arXiv:1808.05587.

Jacot, A., Gabriel, F., and Hongler, C. (2018).
Neural tangent kernel: Convergence and generalization in neural networks.
In *Advances in neural information processing systems*, pages 8571–8580.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-Dickstein, J. (2017).
Deep neural networks as gaussian processes.
arXiv preprint arXiv:1711.00165.

Lee, J., Xiao, L., Schoenholz, S., Bahri, Y., Novak, R., Sohl-Dickstein, J., and Pennington, J. (2019).
Wide neural networks of any depth evolve as linear models under gradient descent.
In *Advances in neural information processing systems*, pages 8572–8583.

Neal, R. M. (1995).

BAYESIAN LEARNING FOR NEURAL NETWORKS.
PhD thesis, University of Toronto.

Novak, R., Xiao, L., Lee, J., Bahri, Y., Yang, G., Hron, J., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. (2018).
Bayesian deep convolutional networks with many channels are gaussian processes.
arXiv preprint arXiv:1810.05148.

Yang, G. (2019).
Tensor programs i: Wide feedforward or recurrent neural networks of any architecture are gaussian processes.
arXiv preprint arXiv:1910.12478.

Yang, G. (2020a).
Tensor programs ii: Neural tangent kernel for any architecture.
arXiv preprint arXiv:2006.14548.

Yang, G. (2020b).
Tensor programs iii: Neural matrix laws.

arXiv preprint arXiv:2009.10685.

Yang, G. and Hu, E. J. (2021).
Tensor programs iv: Feature learning in infinite-width neural networks.
In *International Conference on Machine Learning*, pages 11727–11737. PMLR.

Yang, G., Hu, E. J., Babuschkin, I., Sidor, S., Liu, X., Farhi, D., Ryder, N., Pachocki, J., Chen, W., and Gao, J. (2022).
Tensor programs v: Tuning large neural networks via zero-shot hyperparameter transfer.
arXiv preprint arXiv:2203.03466.

Yang, G. and Littwin, E. (2021).
Tensor programs iib: Architectural universality of neural tangent kernel training dynamics.
In *International Conference on Machine Learning*, pages 11762–11772. PMLR.