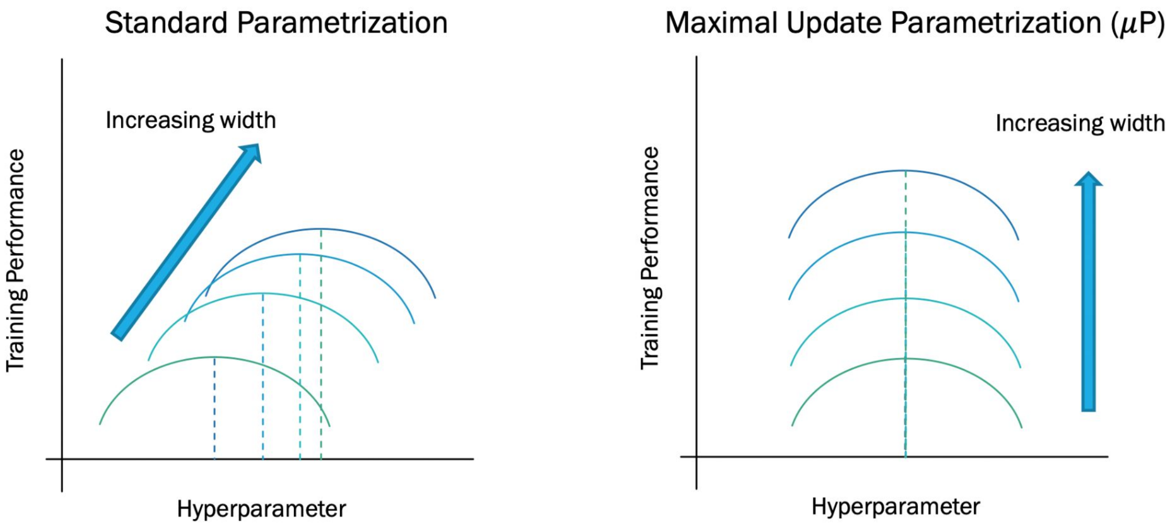
Maximal Update Parametrization

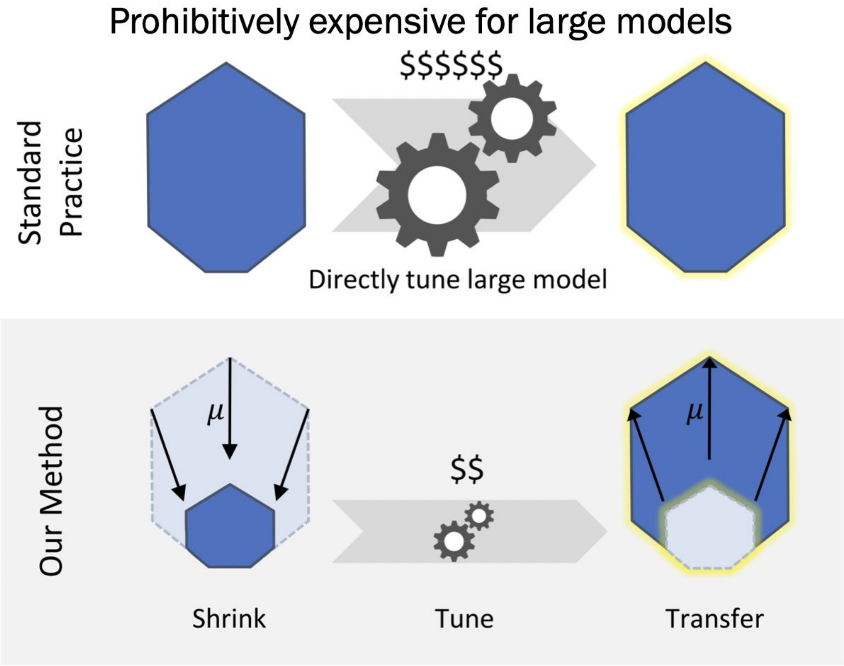
Why?



“Transfer” = optimal hyperparameter remains stable with model size

- hyperparameter tuning for a large model is computationally expensive
- it is possible to tune hyperparameters on a smaller model and then transfer them to a wider model, using Maximal Update Parametrization

μTransfer: Zero-Shot Hyperparameter Transfer

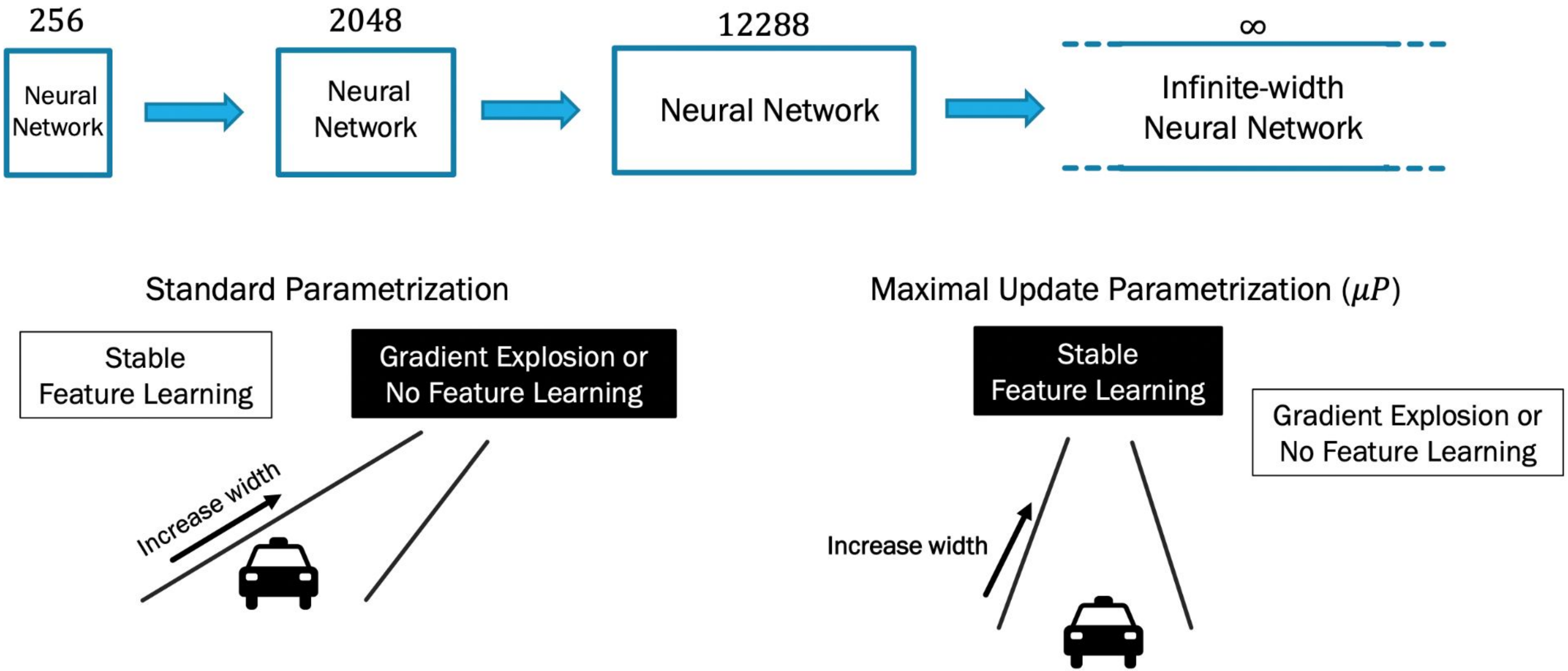


- Preserves hyperparameter optimum across width
- Allows zero-shot hyperparameter transfer
- Efficient tuning
- Can tune enormous models only on a single GPU
- Very fast

μTransfer: Zero-Shot Hyperparameter Transfer

Algorithm 1 Tuning a Large Target Model via μTransfer

1: Parametrize target model in Maximal Update Parametrization (μP)
2: Tune a smaller version (in width and/or depth) of target model
3: Copy tuned hyperparameters to target model


256
Neural Network
2048
Neural Network
12288
Neural Network
∞
Infinite-width Neural Network
Standard Parametrization
Stable Feature Learning
Gradient Explosion or No Feature Learning
Increase width
Maximal Update Parametrization (μP)
Stable Feature Learning
Gradient Explosion or No Feature Learning
Increase width

Neural Tangent Kernel

The naive first-order Taylor expansion is given by:

$$f(x;\theta) - f(x;\theta_0) \approx \langle \nabla_\theta f(x;\theta_0), \theta - \theta_0 \rangle$$

$$K(x, z) = \langle \nabla_\theta f(x;\theta_0), \nabla_\theta f(z;\theta_0) \rangle$$

$$f_t - f_{t-1} \approx -\eta K \mathcal{L}'(f_t, y)$$

Why NTK doesn't learn features?

Naïve first order Taylor expansion

$$f(x;\theta) - f(x;\theta_0) \approx \langle \nabla_\theta f(x;\theta_0), \theta - \theta_0 \rangle$$

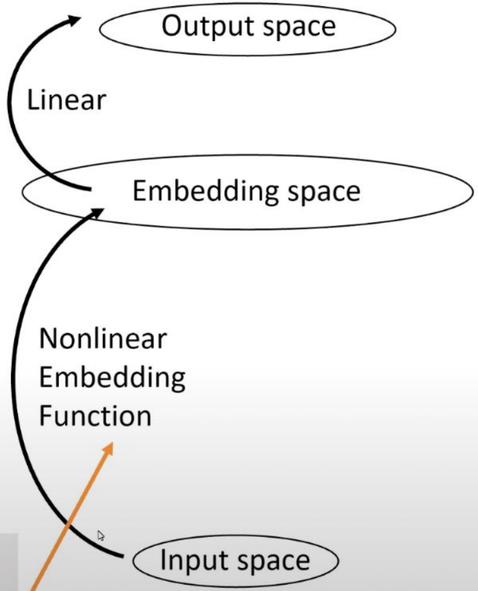
so

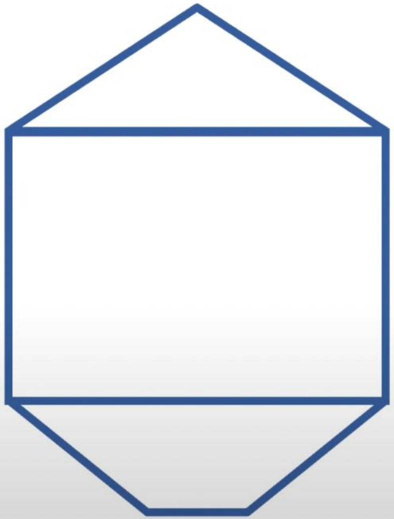
$$f_t - f_{t-1} \approx -\eta K \mathcal{L}'(f_t, y)$$

Where $\mathcal{L}$ is loss, y is label, and K is the kernel

$$K(x,z) = \langle \nabla_\theta f(x;\theta_0), \nabla_\theta f(z;\theta_0) \rangle$$

Implies $\theta - \theta_0$ is small
$\Rightarrow$
Embedding function cannot change too much

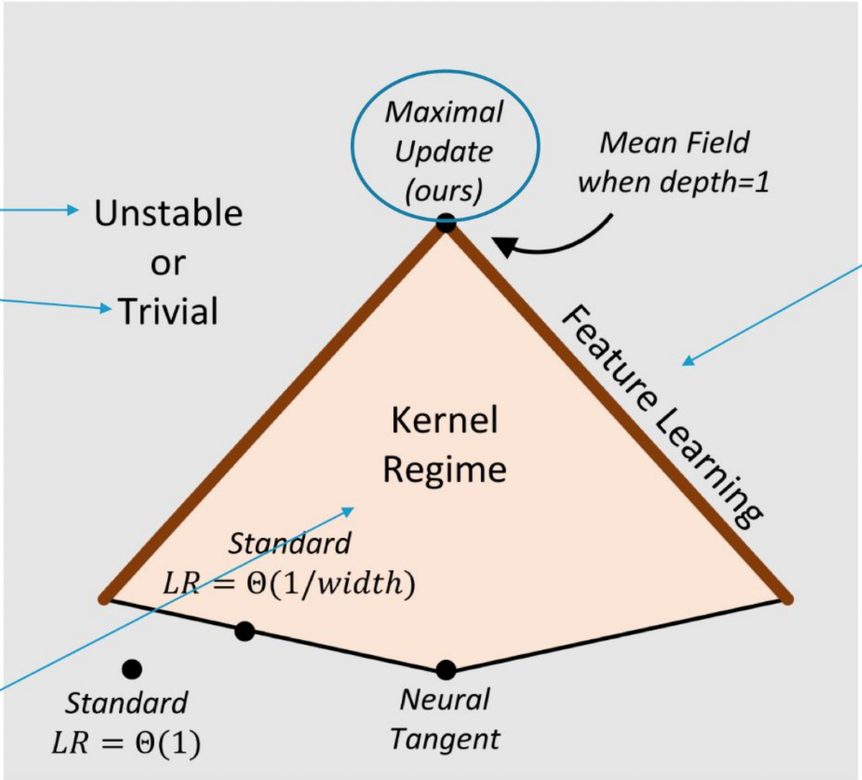




A Caricature of Space of Parametrizations

Training blows up

Training stuck at init



- Feature learning
- Function evolution cannot be described purely in the function space

- Function evolution described by functional equation for some kernel K
  $f_{t+1} - f_t = -\eta K \mathcal{L}'(f_t)$
- No feature learning

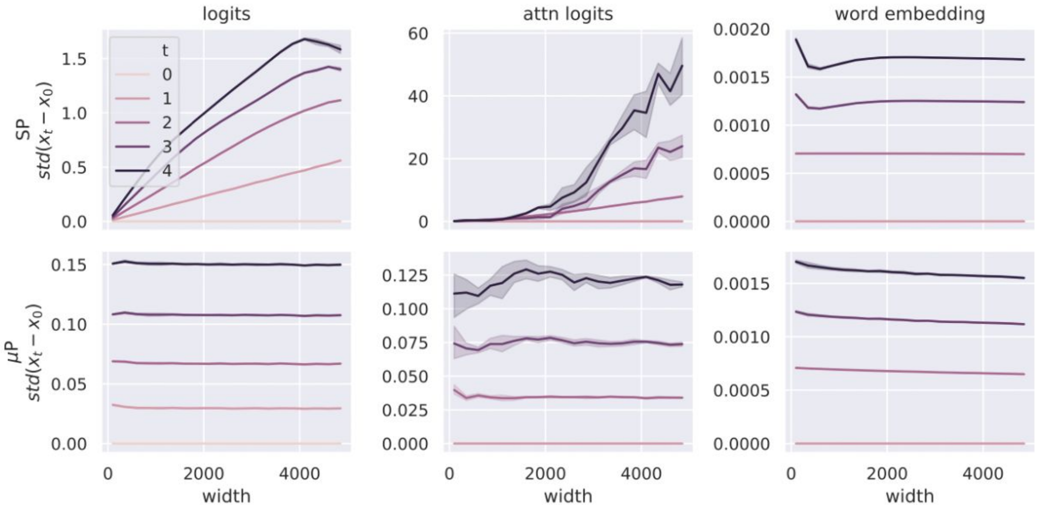
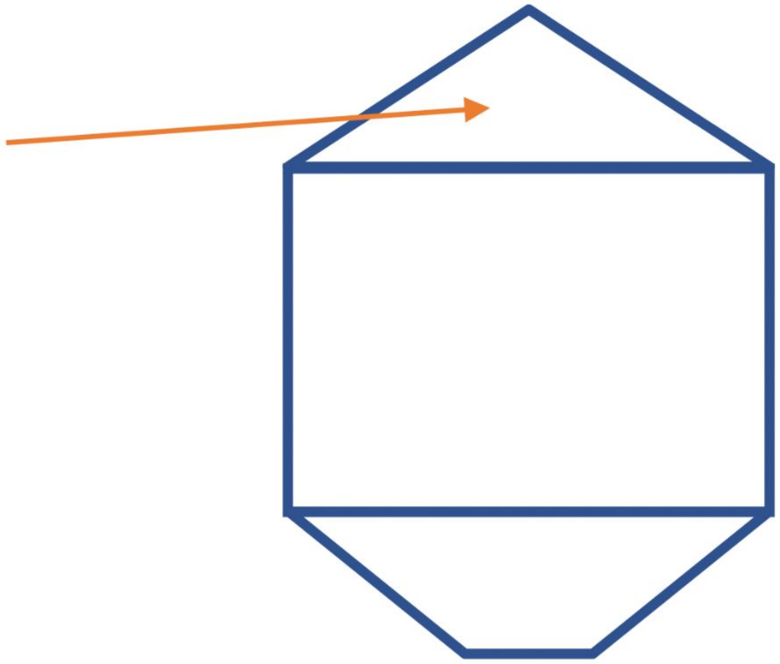


Figure 5: **Logits and attention logits, but not word embeddings, of a Transformer blow up with width in SP after 1 step of training.** In contrast, all three are well-behaved with width in μP. Here we measure how much different values change coordinatewise from initialization over 4 steps of Adam updates, as a function of width. Specifically, we plot the standard deviation of the coordinates of $x_t - x_0$, for $t = 0, \ldots, 4$, and $x \in$ {logits, attention logits, word embeddings}, where $t = 0$ indicates initialization.

Standart Parametrization doesn't learn features

- Intuition why
  - The last layer weights get too much gradient, relative to weights in the body
  - We want to use larger learning rate to enable feature learning, but then the logits would blow up.



Maximal Update Parametrization

- Modify Standard Param to get Maximal Update Param
  - Last layer: divide logits by $\sqrt{n}$ and use $\Theta(1)$ learning rate
    - i.e. $a_{L+1} = \frac{1}{2}, c = 0$
    - i.e. $f(\xi) = \frac{1}{\sqrt{n}} w^{L+1} x^L(\xi)$ where $w_{\alpha\beta}^{L+1} \sim \mathcal{N}\left(0, \frac{1}{n}\right)$
    - This alone suffices to enable feature learning
  - First layer: increase the gradient by n by setting $a_1 = -\frac{1}{2}, b_1 = 1/2$
    - i.e. $h^1(\xi) = \sqrt{n} w^1 \xi$ where $w_{\alpha\beta}^1 \sim \mathcal{N}\left(0, \frac{1}{n}\right)$
    - Needed to enable feature learning in *every* layer

An *abc-parametrization* is given by a set of numbers $\{a_l, b_l\}_l \cup \{c\}$ s.t.

a) Parametrize each $W^l = n^{-a_l} w^l$ where w^l is trained instead of W^l
b) Initialize each $w^l_{\alpha\beta} \sim \mathcal{N}(0, n^{-2b_l})$
c) SGD learning rate is ηn^{-c} for some width-independent η.

	Definition	NTK	Standard	Standard ($1/n$ LR)	Mean Field ($L = 1$)	Maximal Update
a_l	$W^l = n^{-a_l} w^l$	$\begin{cases} 0 & \text{if } l = 1 \\ \frac{1}{2} & \text{if } l > 1 \end{cases}$	0	0	$\begin{cases} 0 & \text{if } l = 1 \\ 1 & \text{if } l = 2 \end{cases}$	$\begin{cases} -\frac{1}{2} & \text{if } l = 1 \\ 0 & \text{if } 2 \le l \le L \\ \frac{1}{2} & \text{if } l = L+1 \end{cases}$
b_l	$w^l_{\alpha\beta} \sim \mathcal{N}(0, n^{-2b_l})$	0	$\begin{cases} 0 & \text{if } l = 1 \\ \frac{1}{2} & \text{if } l > 1 \end{cases}$	$\begin{cases} 0 & \text{if } l = 1 \\ \frac{1}{2} & \text{if } l > 1 \end{cases}$	0	1/2
c	$LR = \eta n^{-c}$	0	0	1	-1	0