Lecture 8 : Infinite-width limits

Previous lectures : as a first approximation of highly overparametrized neural networks, we can consider "infinite width limits"

→ The hope is that:

1. There will be some simplifications that result in a compact description of the dynamics wich is amenable to analysis

2. It's a good approximation of neural networks used in practice.

3. Conversely, it can inform us how to scale NNs (Common practice : grid-search over hyperparameters → if we had a more principled approach, we could save time and money + now models are too big to do grid-search directly)

Right now, we saw two different limits:

* **NTK scaling:** As $M \to \infty$ with multi-layer NNs
SGD concentrates on kernel method

$$\mathrm{NTK}(x_1, x_2) = h_L(\langle x_1, x_2 \rangle) \to \text{closed form}$$

easy to compute

* **MF scaling:** As $M \to \infty$, 2-layer NNs
SGD concentrates on Wasserstein gradient flow

$$f = \int \sigma(x;\theta)\, \rho(\theta) \qquad \partial_t \rho_t = \nabla_\theta \cdot [\rho_t \nabla_\theta \Psi(\theta;\rho_t)]$$

→ Very different behavior between the 2 limits

$\boxed{\text{What about limits with other scalings?}}$

Many papers have explored how to scale different hyperparameters with width, and how to "transfer" these hyperparameters between models

Today's goal: briefly present a very simple framework
"ABC"-parametrization [Yang, Hu, 2020]

Fully-connected multi-layer NNs + SGD
and track the scaling of 3 sets of hyperparameters as
width $\to \infty$.

Model: L-hidden layer NNs

all layers are width M

* input $x \in \mathbb{R}^d$
* weight matrices: $W^1 \in \mathbb{R}^{M \times d}$

$W^2, W^3, \dots, W^L \in \mathbb{R}^{M \times M}$

$W^{L+1} \in \mathbb{R}^{1 \times M}$

* non-linearity $\sigma : \mathbb{R} \to \mathbb{R}$

Compute the output iteratively: $h^1(x) = W^1 x$

"preactivation"

$\ell = 1, \dots, L$

$$z^\ell(x) = \sigma(h^\ell(x)) \in \mathbb{R}^M \qquad h^{\ell+1}(x) = W^{\ell+1} z^\ell(x) \in \mathbb{R}^M$$

$$\begin{aligned} f(x) = W^{L+1} z^L(x) &= W^{L+1} \sigma(h^L(x)) \\ &= W^{L+1} \sigma(W^L z^{L-1}(x)) \\ &= W^{L+1} \sigma(W^L \phi(h^{L-1}(x))) \\ &= W^{L+1} \circ \sigma \circ W^L \circ \dots \circ \sigma \circ W^1 x \end{aligned}$$

We train this NN with batch-1 SGD

Some loss: $\ell(y, f(x; \Theta))$

$$\Theta_{k+1} = \Theta_k - \eta \nabla_{\Theta} \ell(y_k, f(x_k; \Theta_k))$$

$$= \Theta_k - \eta \, \ell'(y_k, f(x_k; \Theta_k)) \nabla_{\Theta} f(x_k; \Theta_k)$$

$$f_k(x) := f(x; \Theta_k)$$

$\longrightarrow$ same with $z_k^\ell(x)$ $\quad h_k^\ell(x)$

$$\chi_k = \ell'(y_k, f(x_k; \Theta_k))$$

We want to consider $M \to \infty$ limit in this dynamic and how we should scale 3 sets of hyperparameters

(a) <u>Layer normalization</u>: $W^\ell = \frac{1}{M^{a_\ell}} \overline{W}^\ell$

$\hookrightarrow$ $\overline{W}^\ell$ are the trainable parameters

(b) Initialization : $\overline{W}_0^\ell$ has iid $N\left(0, \frac{1}{M^{2b_\ell}}\right)$

(c) Learning rate : $\eta = \frac{\overline{\eta}}{M^c}$

$$\overline{W}_{k+1}^\ell = \overline{W}_k^\ell - \frac{\overline{\eta}}{M^c} \chi_k \nabla_{\overline{W}^\ell} f_k(x_k)$$

Set of hyperparameters $\{a_\ell, b_\ell\}_{\ell=1}^{L+1} \cup \{c\}$

$$\boxed{\begin{array}{l}\text{Wich choice of abc leads to meaningful limits?} \\ \text{What is the behavior of these different limits ?}\end{array}}$$

Rmk: abc parametrization is degenerate

- At initialization: $W_0^\ell = \frac{\overline{W}^\ell}{M^{a_\ell}} \sim N\left(0, \frac{1}{M^{2(a_\ell+b_\ell)}}\right)$

  ↳ For $a_\ell + b_\ell = \text{cste}$ NN at initializat° is the same

- When updating:

$$h^{\ell}_{k+1}(x) = W^{\ell}_{k+1}\, z^{\ell}_{k+1}(x)$$

$$= \frac{1}{M^{a_\ell}} \left(W^{\ell}_{k} - \frac{\eta}{M^{c}} \chi_k \nabla_{W^\ell} f_k(x_k) \right)$$

$$= h^{\ell}_{k}(x) - \frac{\eta}{M^{c+2a_\ell}} \chi_k \nabla_{W^\ell} f_k(x_k)$$

So NN during SGD is invariant under the following reparametrization

$\forall \theta \in \mathbb{R}$,

$$\left.\begin{aligned} a_\ell &\leftarrow a_\ell + \theta \\ b_\ell &\leftarrow b_\ell - \theta \end{aligned}\right\} \text{invariance at initialization}$$

$$\left. c \leftarrow c - 2\theta \right\} \text{invariance updates}$$

What can go wrong when taking $M \to \infty$?

- $h^{l}_{k}(x) \to \infty$ (e.g. c too small)

learning rate too large

↳ blow-up of preactivation

"parametrization is unstable"

- $f_k(x) = f_0(x)$ (e.g. c too large)

learning rate too small

↳ network does not change in finite time

"parametrization is trivial"

Example: 1-hidden layer $(L=1)$

For simplicity $x \in \mathbb{R}$ $\quad W \in \mathbb{R}^{M \times 1}$ $\quad V \in \mathbb{R}^{1 \times M}$

$$f(x) = V\, z(x) \qquad z(x) = \sigma(h(x))$$
$$h(x) = W x$$

$$V = \frac{\bar{V}}{M^{a_2}} \qquad W = \frac{\bar{W}}{M^{a_1}}$$

* <u>NT parametrizat°</u>:

$$a_1 = 0 \qquad b_1 = 0$$
$$a_2 = \frac{1}{2} \qquad b_2 = 0 \qquad \Big\} \ \frac{1}{\sqrt{M}} \sum v_j \sigma(w_j^T x)$$
$$c = 0$$

* <u>MF parametrization</u>:

$$a_1 = b_1 = b_2 = 0$$
$$a_2 = 1 \qquad \Big\} \ \frac{1}{M} \sum v_j \sigma(w_j^T x)$$
$$c = -1 \longrightarrow \eta M$$

<u>Derivatives</u>:

$$\nabla_z f(x) = V \qquad \nabla_h f(x) = V \odot \sigma'(h(x))$$

$$\nabla_{\bar{V}} f(x) = \frac{1}{M^{a_2}} z(x) \qquad \nabla_{\bar{W}} f(x) = \frac{1}{M^{a_1}} V \odot \sigma'(h(x))\, x$$

Update after one step:

$$\Delta h_1(x) = h_1(x) - h_0(x) = -\frac{1}{M^{2a_1+c}} x_0\, x_0 \left[V_0 \odot \sigma'(h_0(x))\right]$$

$$\Delta f_1(x) = V_0\, \Delta z_1(x) + \Delta V_1\, z_1(x)$$

$$= V_0\, \Delta z_1(x) - \frac{1}{M^{2a_2+c}}\, z_1(x_0)^T z_1(x)$$

NTP & MFP: $z_0(x)$ has coordinates $\Theta(1)$

Feature evolution?

NTP: $\Delta h_1(x) = \Theta(1) \cdot \underset{\Theta\left(\frac{1}{\sqrt{M}}\right)}{\underline{V_0}} \odot \Theta(1)$

each coordinate of preactivation move $\Theta\left(\frac{1}{\sqrt{M}}\right)$

MFP: $\Delta h_1(x) = \Theta(M) \cdot \Theta\left(\frac{1}{M}\right) \odot \Theta(1)$

$= \Theta(1)$

each coordinate of preactivat° move $\Theta(1)$

Feature "kernel" evolution:

$$F_k(x, x') = \frac{1}{M} z_k(x)^T z_k(x')$$

$$\lim_{M \to \infty} F_k(x, x') = \lim_{M \to \infty} F_0(x, x') \qquad \text{NTP}$$

the feature kernel does not evolve

$$\lim_{M \to \infty} F_k(x, x') \neq \lim_{M \to \infty} F_0(x, x') \qquad \text{MFP}$$

the kernel changes!

General picture

Symmetries of abc parametrizat°:

$\forall \theta \in \mathbb{R}$, $\quad a_\ell \leftarrow a_\ell + \theta \quad b_\ell \leftarrow b_\ell - \theta \quad c \leftarrow c - 2\theta$

result in the same limiting model.

Feature movements scaling: $\Delta z_k^L(x) = \Theta(M^{-r})$

$$r := \min(a_{L+1} + b_{L+1},\ 2a_{L+1} + c) + c - 1$$

$$+ \min_{\ell=1}^{L} \left[2a_\ell + \mathbb{1}(\ell = 1) \right]$$

Ex. NTP $r = \frac{1}{2}$ $\qquad$ MFP $r = 0$

$\hookrightarrow$ to avoid blow-up $r \geq 0$ $\qquad$ feature learning $r = 0$

(**Proof:** look at one step, easy)

Stable abc parametrization:

It is stable IFF

(1) preactivat° at initialization are $\Theta(1)$
and $f_0 = O(1)$

$$\boxed{\begin{array}{ccc} \ell=1 & 2\le \ell\le L & \ell=L+1 \\ a_1+b_1=0 & a_\ell+b_\ell=\frac{1}{2} & a_{L+1}+b_{L+1}\geqslant\frac{1}{2} \end{array}}$$

(2) features don't blowup $\Delta x_t^\ell = O(1)$

$$\boxed{r\geqslant 0}$$

(3) f_k doesn't blowup

$$\boxed{2a_{L+1}+c\geqslant 1 \qquad a_{L+1}+b_{L+1}+r\geqslant 1}$$

Proof: easy (do one step)

Nontrivial abc parametrizations

A stable parametrization is non-trivial

IFF $\quad a_{L+1} + b_{L+1} + r = 1 \quad$ or $\quad 2a_{L+1} + c = 1$

Proof: $\Delta f_1(x) = \Delta W_1^{L+1} z_1^L(x) + W_0^{L+1} \Delta z_1^L(x)$

$\Theta\left(\frac{1}{M^{2a_{L+1}+c}}\right) \cdot \Theta(1) \qquad \Theta\left(\frac{1}{M^{a_{L+1}+b_{L+1}}}\right) \Theta\left(n^{-r}\right)$

$\lim_{M \to \infty} \Delta f_1(x) \neq 0$ IFF one of the 2 terms $\neq 0$

Dynamical dichotomy

Def: [Feature learning] abc parametrization admits feature learning if as $M \to \infty$,

$$\boxed{\frac{1}{M}\left(z_k^L(x)^T z_k^L(x') - z_0^L(x)^T z_0^L(x') \right) = \Omega(1)}$$

for some time $k \geq 1$ and input x, x'

(That is $\Delta z_k^L(x)$ has $\Omega(1)$ coordinates)

Def: [Kernel regime] abc parametrization is in the kernel regime iff for any $k \geq 0$,

$$\boxed{\lim_{M\to\infty} f_{k+1}(x) = \lim_{M\to\infty} f_k(x) - \eta K(x, x_k)\, \ell'(y_k, f_k(x_k))}$$

where $K(x, x') = \lim_{M\to\infty} \langle \nabla_\Theta f(x), \nabla_\Theta f(x') \rangle$

Thm: A non-trivial stable abc parametrization

* admits feature learning iff $r = 0$
* is in kernel regime iff $r > 0$

Rmk:

- For $r = 0$ and $r > 0$ plenty of dynamics possible (e.g. some layers don't move)
- Higher-order Taylor correct° to NTK are not valid ∞^k width limits

Examples

NTP: $a_\ell = \begin{cases} 0 & \ell = 1 \\ \frac{1}{2} & \ell \geq 2 \end{cases}$ $\quad b_\ell = 0 \quad c = 0$

Then $r = \frac{1}{2}$ kernel regime

RFP: $c = 1 \quad a_{L+1} = 0 \quad b_{L+1} = \frac{1}{2}$

$a_\ell = 0 \quad b_1 = 0$

$b_\ell = \frac{1}{2} \quad 2 \leq \ell \leq L$

Then $r = \frac{1}{2}$

↳ only train last layer ("random feature model")

MFP: $L = 1 \quad a_1 = 0 \quad b_\ell = 0$

$a_2 = 1 \quad c = -1$

Then $r = 0$ feature learning regime

SP: "standard parametrization" available as default setting in PyTorch

It has $a_\ell = 0$ for all ℓ

$b_1 = 0$ and $b_\ell = \frac{1}{2}$ for $\ell \geq 2$

$c = 0$

↳ learning rate is too large, not a stable abc parametrization. To make it stable, take $c = 1$

Then $r = \frac{1}{2}$ kernel regime

"μP": "maximal update parametrization"

↳ parametrization that allows each parameters to be updated maximally without blowing up

i.e. $\Delta h_k^\ell(x) = \Theta(1)$ entries for all $\ell = 1, \dots, L$

$\Theta(n^{-r_\ell})$ $\quad r_\ell = \min(a_{L+1} + b_{L+1}, 2a_{L+1} + c) + c - 1 + 2a_\ell + \mathbb{1}(\ell = 1)$

$r = \min_{1 \leq \ell \leq L} r_\ell \implies \mu P \quad r_\ell = 0$ for all ℓ

This gives:

$$a_\ell = \begin{cases} -1/2 & \ell = 1 \\ 0 & 2 \leq \ell \leq L \\ 1/2 & \ell = L+1 \end{cases}$$

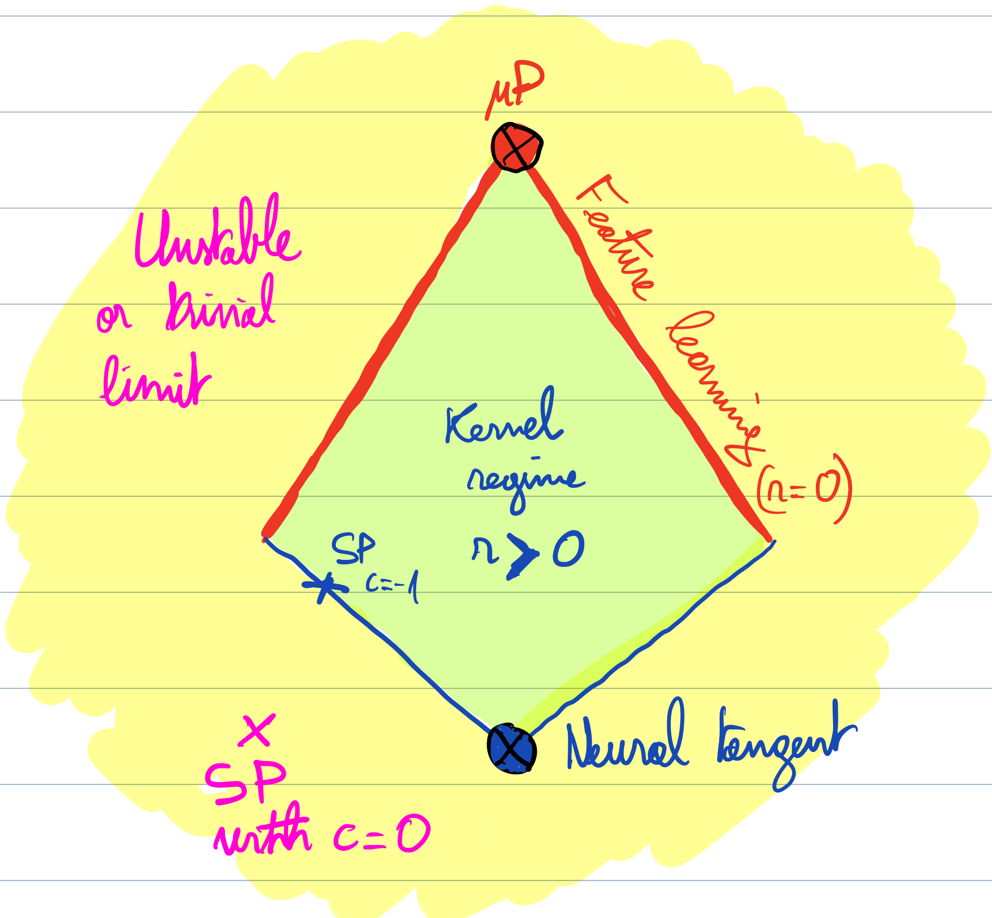
$$b_\ell = \frac{1}{2}$$

$$c = 0$$

This coincides with MFP for $L=1$

Summary

In space $\{a_\ell, b_\ell\} \cup \{c\}$ non trivial stable parametrization form a polytop



Application: • μP seem to work well

• "one-shot transfer learning": Let's we want to train a model with 10 B parameters ($M = 10^6$)
how to choose learning rate / layer normalizat° / variance at initialization

Take a smaller model with width M_0
(e.g. $M_0 = 10^5$, i.e. $\simeq$ 100M parameters)

do grid search and find optimal η_0, $N_{\ell,0}$, $v_{\ell,0}$

then for model with width M_1, use directly

$$\eta_1 = \eta_0 \left(\frac{M_0}{M_1}\right)^c \qquad N_{\ell,1} = N_{\ell,0}\left(\frac{M_1}{M_0}\right)^{a_\ell} \qquad v_{\ell,1} = v_{\ell,0}\left(\frac{M_0}{M_1}\right)^{2b_\ell}$$

<u>e.g.</u> take μP

[Greg Yang et al., 22] Using this idea with μP, took GPT-3

40M parameters $\longrightarrow$ 6.7B parameters

↳ the model they get 1) show better performance
2) only 7% of total pretraining cost

[OpenAI, Anthropic, xAI: have empirical heuristics to scale hyperparameters ... but it's not μP...]