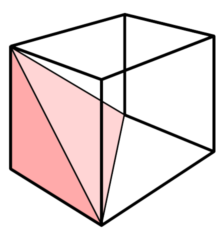
Measuring the "size" of hypothesis spaces over $\mathbb{R}$ from the perspective of No Free Lunch

Yan King Yin

May 3, 2024

Motivating example

We want to learn, ie, use gradient descent of a neural network to approximate, a symmetric function $y = \hat{f}(\vec{x})$ satisfying the condition $\hat{f}(x_1, ... x_n) = \hat{f}(\sigma \cdot \{x_1, ... x_n\})$ where $\sigma \in \mathfrak{S}_n$ is a permutation. The **input space** is $X = \mathbb{R}^n = n$-dimensional hypercube:



(1)

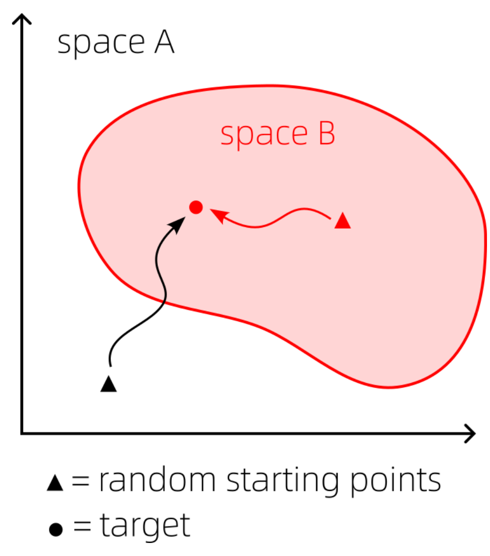
Permutation symmetry implies that only one corner of the hypercube need be considered, this is the **fundamental domain**. The **volume** of this domain over the entire hypercube shrinks exponentially as n grows, so it appears that this symmetry is very significant for efficiency consideration. We want to quantify the notion of this efficiency.

Finding the right space setting

We want to compare the **hypothesis spaces** A and B:

- A = unconstrained, fully-connected neural network with L layers and N total weights. The parameter space $A = \Theta_A = \mathbb{R}^N$.
- B = symmetric neural network with a special structure but nonetheless its parameter space is also of the form $B = \Theta_B = \mathbb{R}^M$.

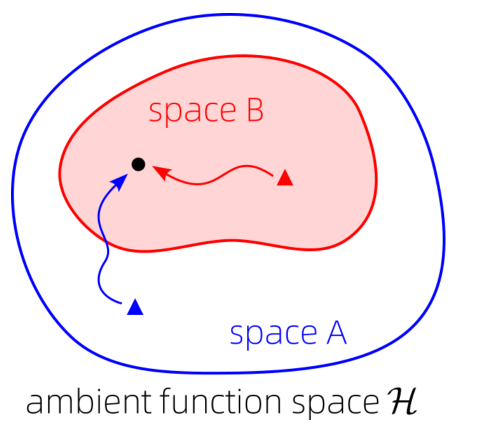
My first idea is to embed the space B into space A. This will lead to "unwieldy" elements of B trying to approximate A, for example, consider if A's activation function is sigmoid but B's activation function is RELU.



(2)

But the general idea is: If solutions are only found in space B, then limiting our search within B will lead to faster convergence as there is no need to waste time "wandering" in space A.

A better idea which is much more convenient is to consider A and B as both embedded in an ambient **function space** $\mathcal{H}$ that is "fine" enough to contain all points of A and B:



Embeddings:
$\Theta_A \hookrightarrow \mathcal{H}$
$\Theta_B \hookrightarrow \mathcal{H}$

(3)

Measure roughness of landscape?

My second idea is to measure the "roughness" of the landscapes A and B in the function space $\mathcal{H}$. The objective function is usually of the form

$$\text{loss} = \sum_i d(f(x_i; \Theta), y_i) + \text{Reg}(f) \tag{4}$$

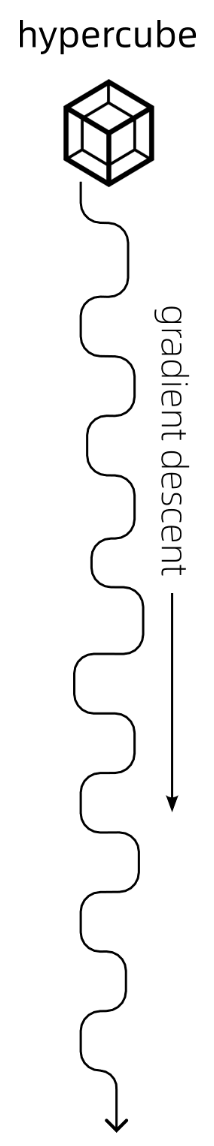
where d is some distance metric and Reg is a regularization function.

The models in A or B live in the parameter spaces Θ_A or Θ_B and both are embedded into the space $\mathcal{H}$. However the **paths** of gradient descent in A or B could still be different, so we have to be cautious about that.

In low-dimension space, we imagine being "trapped" in some kind of pit, so we may want to measure the roughness of the landscape by "undulations" of gradients. But our next consideration suggests that this may not be a very effective measurement...

Gradient descent in high-dimension space

It may be helpful to visualize gradient descent in high-dimension space with Fig.(5). Imagine the hypercube as having millions of vertices (as the number of weights). The "landscape" is a surface sitting "over" the hypercube but we visualize it downwards.



(5)

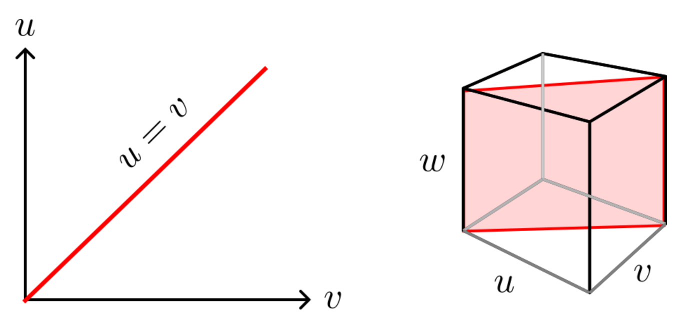
It is very "easy" for gradient descent to find a way "down" because there are so many dimensions to choose from.

Also, a "local minimum" is very rare as it requires millions of gradients to be pointing "up" at the same spot. In a model with a massive number of weights (as in the case of current LLMs), for all practical purposes, a local minimum is just as good as an acceptable solution. In this sense, *the phenomenon of "getting stuck in local minima" disappears.*

The "symmetric" parameter space

The following figure may help visualize the parameter space of our **symmetric** motivating example (but not for the AGI problem).

In our example, $\Theta_A = (u, v)$ but Θ_B is just the diagonal space $\langle u = v \rangle$. The case with 3 parameters, of which 2 are equal, is shown on the right:



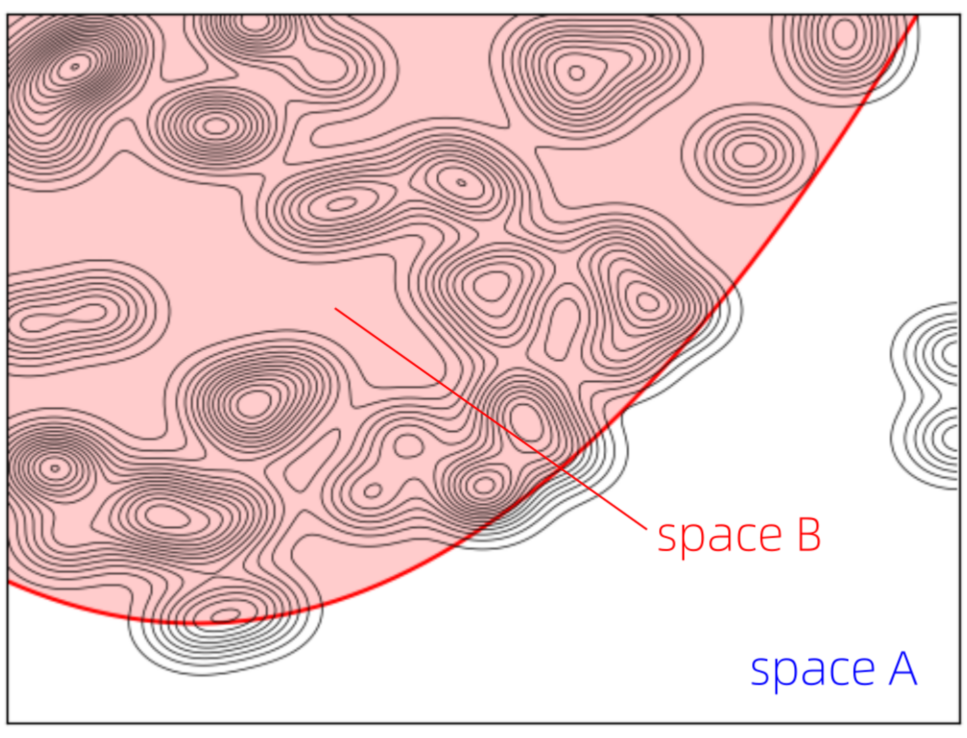
(6)

In general, multiple dimensions can collapse to one, when the weights are shared (equal).

Distribution of solution "attractors"

I find the most helpful way to visualize the AGI landscape is to imagine solution **attractors** to be more *densely* populated in the more-structured space B.

The following is a diagram I made up using a Python script and Matplotlib with randomly generated plots. One can imagine "attractors" for **gradient descent** that are distributed over the hypothesis space that lead to solutions (AGIs). I made the distribution of attractors in space B **denser** than in space A. So if an algorithm randomly starts to search, it will have a better chance of success to start within space B. The "distances" traveled by gradient descent are also different for A and B because their parameter spaces are different.



(7)

Perhaps the most crucial thing we want to prove, or at least find indirect evidence for, is whether solution attractors are **sparser** in space A than B. This is obviously a **problem-specific** question. In some cases, such as the learning of symmetric functions, we know *a priori* that solutions *exclusively* lie in space B. But the issue is very hard to analyze for AGI.

1

This

- Here

②

This

- Here

③

This

- Here