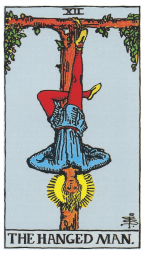

Learnable Spaces Conceptual Paper

Alger Wilson
algerwilson@proton.me
https://github.com/Alger-D-Wilson/LearnableCoordinateSpaces

October 1, 2025

1 Introduction

The Artificial Neural Network(ANN's) have been at the forefront of learning. The main issues with ANNS are contuinal learning/catastrophic forgeting. In this paper I define Learnable Space which in theory will lead to intelligence, there is one known Learbable Space that has already led to intelligence. But thats just a theory a?, the working project right now is a Light Learnable Coordinate Space. that also in theory should be a universal approximator, it might also have some cool traits? Intill a working expample and multiple iterations of the idea is worked on we wont know, but the idea that if you operate in a octree space you can get some very sparse compuation, this depending on if tweaks are made to intial idea is that the model is event driven, and you'll need to read the Light LC's section first, but theres also the idea of leaving many untrained drops on the edge of the universe and train contiully on them?

2 Defining Learnable Spaces

In mathmatics a "space" can be defined as a set with structure mapping the relationship of the elements in the set. There is a unviverse of possible Learnable Spaces of wich a space is populated with some function or functions and equipped with some optimization algorithm or algorithms.

3 Learnable Coordinate Spaces Base

This leaves a lot of choiches for defining the intial parameters of a learnable space. Intuition leads to the idea of just copying the type of space that is the unviverse, with the universe being a 3d space combined with a 4th dimension of time, otherwise known as the spacetime continuum credit to Minkowski.Our 3d space simplification of the universe can be defined as a Euclidean space since the cosmological curvature parameter estimated by the Planck mission is said to be $\Omega_K = 0.0007 \pm 0.0019$. Euclidean Space defintion below.

$$\mathbb{R}^3 = \mathbb{R} \times \mathbb{R} \times \mathbb{R} = \{(x, y, z) \mid x, y, z \in \mathbb{R}\}$$

For the 4th dimension of time since this is a very rough draft of a learnable space we will simply define time to be the progress of the learnable space, with our Planck time t_P simply being a discrete optimization step in some evolution function $\mathcal{F}$ for our universe $\Psi(t)$.

$$\Psi(t + t_P) = \mathcal{F}(\Psi(t))$$

This give's us a Coordinate space with a 4d time dimension. To make it learnable add any type of computation/functions alongside a optimization algorithm that can be used to minimize a loss function.This in theory is a universal approximator.

4 Light LC's

An early reasonable implementiation of LC'S would be a phyiscs-lite based Light Learnable Coordinate Space. We will build the space using the Learnable Coordinate base and we will populate the space with "learnable drops" these drops will be perfectly spherical and will all be 'water' for easy of computation sake. These drops will be randomly scattered in our space, defined as a set $\mathcal{S}$ of N drops.

$$\mathcal{S} = \{D_1, D_2, \ldots, D_N\}$$

Each drop D_i can be defined as a sphere with its tuple of properties position $\mathbf{p}_i$, a constant radius r, a weight W, and an energy value E and is IOR value of 1.333 for water.

$$D_i = (\mathbf{p}_i, r, W, E, radiance, 1.333)$$

Into this populated space goes some batch $\mathcal{B}$ of data from some dataset $\mathcal{D}$. We will spawn those data points in a form of a ray defined as tuple containing its associated data δ; its origin point, $\mathbf{p}$; and its normalized direction vector, $\hat{\mathbf{d}}$.

$$R = (\delta, \mathbf{p}, \hat{\mathbf{d}})$$

Defining the origin point and its normalized direction will vary case by case and need expermimenting with. My current ideas on defining origin point of the ray is as followed

- Use a focal point to which rays enter the space from outside the universe.
- Spawn from center coordinate 0,0,0 possible define normalized vector from postion in sequence as an idea.
- For areas where embeddings can be used such as NLP, use frozen embedding that spawn ray with origin and normalized vector.

This ray will be used to propagate our learnable space and we define the movement of the ray as just a straight line/Rectilinear Propagation. These rays will hit the nearest 'drop' in its path and then refract and relfect, instead of using the Fresnel equations which are computationaly heavy for this behavior we will use the standard Schlick Approximation commonly used in computer graphics.The reflectance F_r as a function of the angle of incidence θ_i is

$$F_r(\theta_i) \approx R_0 + (1 - R_0)(1 - \cos(\theta_i))^5$$

where R_0 is the reflection coefficient for light incoming parallel to the normal defined as

$$R_0 = \left(\frac{n_1 - n_2}{n_1 + n_2}\right)^2$$

To handle computational cost of doubling rays we need a method to terminate them. We will use an energy partitioning function which takes the incident radiance L_{in} and $theta_i$ as inputs. If L_{in} is below our threshold *epsilon* then we terminate the ray, otherwise the function partitions the radiance of the reflected and refracted rays as $(L_{\text{reflect}}, L_{\text{refract}})$ based on Schlick Approximation for reflectance $F_r(\theta_i)$

$$(L_{\text{reflect}}, L_{\text{refract}}) = \begin{cases} (0, 0) & \text{if } L_{\text{in}} < \epsilon \\ (L_{\text{in}} \cdot F_r(\theta_i), L_{\text{in}} \cdot (1 - F_r(\theta_i))) & \text{otherwise} \end{cases}$$

This is not differntiable howeve at one point so what do we do? We could lowkey just ignore it. Or we could use a smooth cutoff by using a sigmoid gate. Let $\sigma(x)$ be its standar function use this to define a gating function, $g(L_{\text{in}})$, steepness of this transition is controlled by a constant k The final energy partitioning function is then given by:

$$(L_{\text{reflect}}, L_{\text{refract}}) = g(L_{\text{in}}) \cdot (L_{\text{in}} \cdot F_r(\theta_i), L_{\text{in}} \cdot (1 - F_r(\theta_i)))$$

This formulation ensures the entire process is fully differentiable while closely approximating a hard cutoff for a large k. Determining when rays should be used to predict $\hat{y}$ and how to go about it has options aswell that needs to be considered and tested.

- Have the prediction be a weighted sum of rays below a certain energy threshold and depth limit.
- Do the same above but if a ray exits the universe aswell add that to final prediction.
- others?

minimizing the loss of the universe for the global objectavie function we run back prop on the ray march and all its computations. $L(haty, y)$, we can compute the gradient $(\nabla_\Theta L)$ in regards to learnable params Weight and Energy