The orbits of neural networks

William Chuang

October 1, 2023

1 Introduction

Inspired by [10, 3, 4], consider the following question: let a training set and a trained neural network be given where the trained neural network already converged to a local minimum, meaning given a positive real number $\epsilon > 0$, then each sample X_i in the given training set, there is $N \in \mathbb{N}$ such that for the number of epoch greater than N, the model is in the epsilon ball around the local minimum of the loss surface (i.e. its parameter space or moduli space). Then, is there a way to find equivalent models of this trained model?

Definition 1. *Two models are equivalent if they both converge on a fixed training set.*

It is necessary to ask this question so that the trained model can be understood better to a point that the distribution of its weights can be studied using random matrix theory, and if all the other equivalent models can be derived instantaneously, then this is also equivalent to apply quantum computing to find all those equivalent solutions in parallel.

If a model can be replaced by another equivalent model, then the other model might be at the different point on the loss surface, then the barrier between the local minimum to the nearest lower local minimum might have a lower height or in some better case with a negative height, meaning the model can be trained to a better model with a smaller error by using a new set of training data to descent to that lower local minimum. Hence, it is also sufficient to ask the question: how to find equivalent models of a given trained model? If the answer to this question is unknown, suppose a trained model cannot distinguish the fake data generated by an adversary model from the true data, or if a model needs to go through a debugging process for some false classified input, then the model not only has to be retrained, but it cannot be justified it is the best solution due to the process to reach the solution is random. Thus, on the contrary, if all of its equivalent models can be found immediately, then chances are, within the set of equivalent models, there might exist a model that will not take the fake input or falsely classified a corner case, meaning a model can be replaced instantly without going through a new training process.

2 Hyperbolic-orbit Algorithm

Definition 2. *Hyperbolic-orbit Algorithm is an algorithm that implements hyperbolic distance to replace matrix product.*

To introduce the algorithm, it is natural to start to build up the concept and tools from the Euclidean space. In Euclidean space, equivalent models of a trained model can be obtained by redefining matrix multiplication using Euclidean distance $d_E(x,y) = |x-y|$ for all $x, y \in \mathbb{R}$. For instance, let A be in $\mathbb{R}^{n\times m}$, B be in $\mathbb{R}^{d\times m}$, , and their entries are denoted by a_{ki} and b_{li} where $k \in \{1,...,n\}$, $l \in \{1,...,d\}$, and $i \in \{1,...,m\}$. Then, $(AB^t)_{kl} = \sum\limits_{i=1}^{m} a_{ki}b_{il}$.

Instead of using matrix multiplication, a new operation using Euclidean distance is defined in the following:

$$\left(A \odot_E B^t\right)_{kl} := \sum_{i=1}^{m} d_E\left(a_{ki}, b_{il}\right).$$

Now, let the above A and B be any two matrices that need to be multiplied in a given model that already converges under the above new operation $\odot_E$ between matrices A and B. Then by adding a real number α to A and B, infinite many equivalent models can be derived, since

$$\left((\alpha + A) \odot_E \left(\alpha + B^t\right)\right)_{kl} := \sum_{i=1}^{m} d_E\left(\alpha + a_{ki}, \alpha + b_{il}\right) = \sum_{i=1}^{m} d_E\left(a_{ki}, b_{il}\right) = \left(A \odot_E B^t\right)_{kl}.$$

In other words, the original given model is unchanged. This operation results in a translation of the loss surface.

Alternatively, every entry of the matrices $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{d\times m}$ can be embedded into the upper half-plane $\mathbb{H}^2$. That is, the entries of A and B become complex numbers with positive imaginary part: $a_{ki} \in \mathbb{H}^2$ and $b_{il} \in \mathbb{H}^2$.

Define a new binary operation between A and B using hyperbolic length on the upper-half plane $\mathbb{H}^2$[2]:

$$\left(A \odot_{\mathbb{H}^2} B^t\right)_{kl} := \sum_{i=1}^{m} d_{\mathbb{H}^2}\left(a_{ki}, b_{il}\right).$$

Then on the upper-half plane, the projective special linear group $\mathrm{PSL}(2,\mathbb{R})$ is its automorphism group. Furthermore, elements of the group are conformal mapping that preserve not only angles but hyperbolic distance. Hence, $\mathrm{PSL}(2,\mathbb{R})$ is also the isometry group of the upper-half plane.

To derive an equivalent model, take any $M \in \mathrm{PSL}(2,\mathbb{R}) \setminus \{\mathrm{id}\}$, then there exists an isomorphism to map M to a linear fractional (Möbius) map $T_M(z) = \frac{az+b}{cz+d}, a, b, c, d \in \mathbb{R}, z \in \mathbb{C}$.

Definition 3 (Hyperbolic-length product)**.** *Define the hyperbolic-length product is the following new operation:*

$$\left(T_M(A) \odot_{\mathbb{H}^2} T_M(B^t)\right)_{kl} := \sum_{i=1}^{m} d_{\mathbb{H}^2}\left(T_M\left(a_{ki}\right), T_M\left(b_{il}\right)\right).$$

Definition 4. *Let W be a neural network model and assume W converged already with the hyperbolic-length product implemented. Then with the above notations, the orbit of W of the given weights $\{a_{ki}, b_{il}\}$ (points in the hyperbolic space) is the set:*

$$\{T_M(a_{ki})\} \cup \{T_M(b_{il})\}$$

for all $M \in PSL(2,\mathbb{R})$, and for all $k \in \{1, ..., n\}$, $l \in \{1, ..., d\}$, and $i \in \{1, ..., m\}$.

Proposition 1. *With the above notations, then the newly defined product is an invariant:*

$$\left(T_M(A) \odot_{\mathbb{H}^2} T_M(B^t)\right)_{kl} = \sum_{i=1}^{m} d_{\mathbb{H}^2}\left(T_M\left(a_{ki}\right), T_M\left(b_{il}\right)\right) = \sum_{i=1}^{m} d_{\mathbb{H}^2}\left(a_{ki}, b_{il}\right) = \left(A \odot_{\mathbb{H}^2} B^t\right)_{kl}.$$

Proof. This result follows immediately from the property of the Möbius mapping T_M when it is corresponding to a matrix M in the projective special linear group $\mathrm{PSL}(2,\mathbb{R})$, then T_M is an isometry under the hyperbolic metric. $\square$

Let $\mathbb{B}^2 := \{z \in \mathbb{C} : |z| < 1\}$ be the Poincare's disc and Ψ be the Cayley's transformation $\Psi : \mathbb{H}^2 \to \mathbb{B}^2$. Then, $\Psi\left(T_M(z)\right) = \frac{a'z+b'}{c'z+d'}$ where $a', b', c', d' \in \mathbb{C}, z \in \mathbb{C}$.

Proposition 2. *With the above definition and proposition, then the value of the product has the same value when the points a_{ki} and b_{il} are sent to Poincare's disc:*

$$\left(T_M(A) \odot_{\mathbb{H}^2} T_M(B^t)\right)_{kl} = \sum_{i=1}^{m} d_{\mathbb{H}^2}\left(T_M\left(a_{ki}\right), T_M\left(b_{il}\right)\right) = \sum_{i=1}^{m} \rho_{\mathbb{B}^2}\left(\Psi\left(T_M\left(a_{ki}\right)\right), \Psi\left(T_M\left(b_{il}\right)\right)\right).$$

Proof. Since for each $z_1, z_2 \in \mathbb{H}^2$, $d_{\mathbb{H}^2}(z_1, z_2) = \rho_{\mathbb{B}^2}(\Psi(z_1), \Psi(z_2))$. □

The 2-dimensional hyperbolic-length product can be generalized to n-dimension[5, 1, 2, 6, 7, 8], for $n \geq 3$. Let A, B be embedded into $\mathbb{H}^n, n \geq 3$.

Definition 5 (Hyperbolic-length product)**.** *Define n-dime the hyperbolic-length product is the following new operation:*

$$\left(T_M(A) \odot_{\mathbb{H}^n} T_M(B^t)\right)_{kl} := \sum_{i=1}^{m} d_{\mathbb{H}^n}\left(T_M\left(a_{ki}\right), T_M\left(b_{il}\right)\right).$$

This n-dimension result could be useful for applying the hyperbolic-orbit algorithm to any trained models that were trained without implementing the hyperbolic-length product during the training. Likewise, for higher dimensions, the hyperbolic-length product is also an invariant:

Proposition 3. *With the above notations, then the newly defined product is an invariant:*

$$\left(T_M(A) \odot_{\mathbb{H}^n} T_M(B^t)\right)_{kl} = \sum_{i=1}^{m} d_{\mathbb{H}^n}\left(T_M\left(a_{ki}\right), T_M\left(b_{il}\right)\right) = \sum_{i=1}^{m} d_{\mathbb{H}^n}\left(a_{ki}, b_{il}\right) = \left(A \odot_{\mathbb{H}^n} B^t\right)_{kl}.$$

3 Examples

3.1 Self-attention mechanism in any transformer-based models with the hyperbolic-length product implemented before training

All layers of a given neural network model can be implemented with the above hyperbolic-orbit algorithm. The following is an example when the hyperbolic-orbit algorithm is applied to the self-attention mechanism, i.e. transformer-based models. Let $A = Q \in \mathbb{C}^{n \times d_q}$ and $B = K^t \in \mathbb{C}^{d_k \times n}$ in self-attention mechanism[9] using dynamic scaling factor β (defined in our another work in progress), and by convention let $d_q = d_k$.

3.2 Apply to models that were trained without using the hyperbolic-length product

Hyperbolic-orbit Algorithm can be implemented into any models that is trained in a traditional method immediately. Let Ω be a model that is trained without using hyperbolic-length product. Once the matrix multiplication between two matrices called A and B within Ω is chosen to be replaced with hyperbolic-length product, the matrices $A \in \mathbb{R}^{n\times m}$ and $B \in \mathbb{R}^{m\times d}$ can be easily embedded into $\mathbb{H}^2$.

In matrix A, the i-th column which has n entries, and in matrix B, the i-th row which has d entries will have $n \times d$ products and this gives $n \times d$ equations to determine where to embedded a_{ki} and b_{il}.

Denote the images of a_{ki} and b_{il} in $\mathbb{H}^2$ using a'_{ki} and b'_{il}. The following embedding algorithm can test if the hyperbolic-orbit algorithm is applicable. For each i in 1,...,m,

For each l in 1,...,d

Consider hyperbolic circles center at b'_{il} with radii $d_{\mathbb{H}^2}(a'_{ki}, b'_{il}) = a_{ki}b_{il}$, and k from 1 to n. Consider hyperbolic circles center at b'_{ij} for each $j \neq l$ with radii $d_{\mathbb{H}^2}(a'_{ki}, b'_{ij}) = a_{ki}b_{ij}$, and k from 1 to n. If for each k, each pair of hyperbolic circles that are corresponding to the same k, one has radius $d_{\mathbb{H}^2}(a'_{ki}, b'_{il})$ and the other has radius $d_{\mathbb{H}^2}(a'_{ki}, b'_{ij})$ has an interception, then do the next j of b'_{ij}. Finally, if there exist interceptions in the above process for all i, then A and B can be embedded into $\mathbb{H}^2$ and use the hyperbolic-orbit algorithm to replace matrix multiplication.

After implemented the hyperbolic-length product and hyperbolic-orbit algorithm to the trained model, with the same training set the trained the model, the new modified model will give exactly the same output. The only distinction will happen only when the old and new model were fed in samples that were out of the training set.

The above two examples show that implement the hyperbolic-orbit algorithm before training is way easier than the other way around. To solve this problem, instead of embedding A and B into 2-dimensional hyperbolic space, hyperbolic spaces in higher dimensions could be the solution, since for each $n \geq 3$, $\mathbb{H}^n$ has an isometry group. This is the sufficient condition for generalizing

2-dimensional hyperbolic-length product to n-dimension.

The hyperbolic-orbit algorithm can be implemented to all trained models that were trained while without implementing the hyperbolic-length product. The reason is in the worst case, matrices A and B can be implemented into $(n \times d)$-dimension, since there is always a solution in $\mathbb{H}^{n\times d}$.

4 Discussion

Furthermore, with the hyperbolic-orbit algorithm, it is possible to find an appropriate Möbius mapping that is corresponding to $M \in \mathrm{PSL}(2,\mathbb{R})$ such that training a 3-node neural network become not necessarily NP-hard in [3]. Further, if every layer of a given neural network is implemented with the hyperbolic-orbit algorithm introduced in this paper, by combining with perturbations and parallel computing, the steepest descent path toward the global minimum could be designed. The algorithm introduced by the paper can help to save training time for parallel computing in the ensemble method, that is, instead of starting over from the beginning to train N' models in parallel, $N' \in \mathbb{N}$, the training can start with N' models that are equivalent to a trained model that already has an error that cannot be reduced by using the old training set–but on the orbits of the trained model, each equivalent model on the orbit can be kept training with a new training set by updating the weight matrices using back-propagation with the gradient descent algorithm. Then, in each epoch, the best model could be selected and instantaneously its N' equivalent models could be obtained using the hyperbolic-orbit algorithm.

References

[1] Lars V. Ahlfors, *Möbius transformations in several dimensions*, Ordway Professorship Lectures in Mathematics, University of Minnesota, School of Mathematics, Minneapolis, Minn., 1981. MR 725161

[2] Alan F. Beardon, *The geometry of discrete groups*, Graduate Texts in Mathematics, vol. 91, Springer-Verlag, New York, 1995, Corrected reprint of the 1983 original. MR 1393195

[3] Avrim Blum and Ronald Rivest, *Training a 3-node neural network is np-complete*, Advances in neural information processing systems **1** (1988).

[4] J Stephen Judd, *Neural network design and the complexity of learning*, MIT press, 1990.

[5] M. È. Kapovich and L. D. Potyagaĭlo, *On the absence of finiteness theorems of Ahlfors and Sullivan for Kleinian groups in higher dimensions*, Sibirsk. Mat. Zh. **32** (1991), no. 2, 61–73, 212. MR 1138441

[6] Michael Kapovich, *Kleinian groups in higher dimensions*, Geometry and dynamics of groups and spaces, Progr. Math., vol. 265, Birkhäuser, Basel, 2008, pp. 487–564. MR 2402415

[7] John G. Ratcliffe, *Foundations of hyperbolic manifolds*, Graduate Texts in Mathematics, vol. 149, Springer, Cham, [2019] ©2019, Third edition [of 1299730]. MR 4221225

[8] José Seade and Alberto Verjovsky, *Higher dimensional complex Kleinian groups*, Math. Ann. **322** (2002), no. 2, 279–300. MR 1893917

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, *Attention is all you need*, Advances in neural information processing systems **30** (2017).

[10] Chulhee Yun, Suvrit Sra, and Ali Jadbabaie, *Global optimality conditions for deep neural networks*, arXiv preprint arXiv:1707.02444 (2017).