Implicit Interpolation with Lipschitz Regularization

October 31, 2023

Lorenzo Del Signore

Abstract

Shape interpolation is a critical task in computer graphics and 3D modeling, and this project explores the use of a neural implicit field, a deep learning model, for this purpose. This report present a method to improve implicit shape interpolation by applying Lipschitz regularization to the neural implicit field. The use of Lipschitz regularization results in the smoothing of the neural implicit field of the neural network, consequently enabling more meaningful and visually coherent interpolations between latent codes. The code is available at https://github.com/lorenzo-delsignore/lipschitz-regularization

1. Introduction

Neural implicit fields have become a valuable representation for 3D shapes. These fields are typically implemented as neural networks that map latent descriptors and 3D coordinates to implicit function values, in this work a signed distance function (SDF). A SDF is a continuous function that, for a given spatial point, outputs the point's distance to the closest surface, whose sign encodes whether the point is inside (negative) or outside (positive) of the watertight surface. The latent descriptor essentially acts as a control point for deforming the 3D shape it represents. Therefore, ensuring smoothness in the latent descriptor space is crucial for performing shape-editing tasks. To encourage smoothness in the latent space even far away from the training set a Lipschitz bound can be used. Previous Lipschitz networks weren't practical for geometry due to the need for pre-determined bounds and extensive tuning. This work implement a smoothness regularizer that learns the Lipschitz bound on the neural field's latent vector, making it more adaptable to geometry tasks. (Liu et al., 2022)

Email: Lorenzo Del Signore <delsignore.1605952@di.uniroma1.it>.

Deep Learning and Applied AI 2023, Sapienza University of Rome, 2nd semester a.y. 2022/2023.

2. Related work

In the literature various methods have been proposed to constrain the Lipschitz constant of neural networks, which influences the stability and behavior of the network:

- **Spectral Normalization:** This technique, introduced by (Miyato et al., 2018), normalizes weight matrices by dividing them by their largest eigenvalue to ensure that the network is 1-Lipschitz under the L2 norm.
- **L1 and L-infinity Norms:** (Gouk et al., 2020) rely on different weight normalization methods to constrain the Lipschitz bound under L1 and L-infinity norms.
- **Orthonormalization:** (Anil et al., 2019) obtain 1-Lipschitz networks by orthonormalizing each weight matrix.
- **Soft Constraints:** (Terjék, 2020) proposes regularization to softly encourage a network to be Lipschitz-constant but not strictly 1-Lipschitz, as strict constraints may lead to optimization issues.

3. Method

3.1. Data preparation

To prepare the data a normalization is applied to each mesh scaling them between [0,1] and a sample of 100,000 spatial points from its surface are taken. These points are then perturbed by random noise within a given range. The perturbed points are divided into two sets, one inside the mesh and the other farther away. Then, for each perturbed point the signed distance is computed, that is the ground truth.

3.2. Neural netwrok Architecutre

The neural implicit field is represented as a MLP that is composed by several fully connected layers (in the experiment 5 with 256 neurons) in which each fully connected layer is followed by a Tanh activation function. The input point x is multiplied with one hundred (100x) to avoid the possibility that the network smoothness in the latent code is bounded by spatial smoothness. The last layer is a fully connected layer that gets the predicted signed distance.

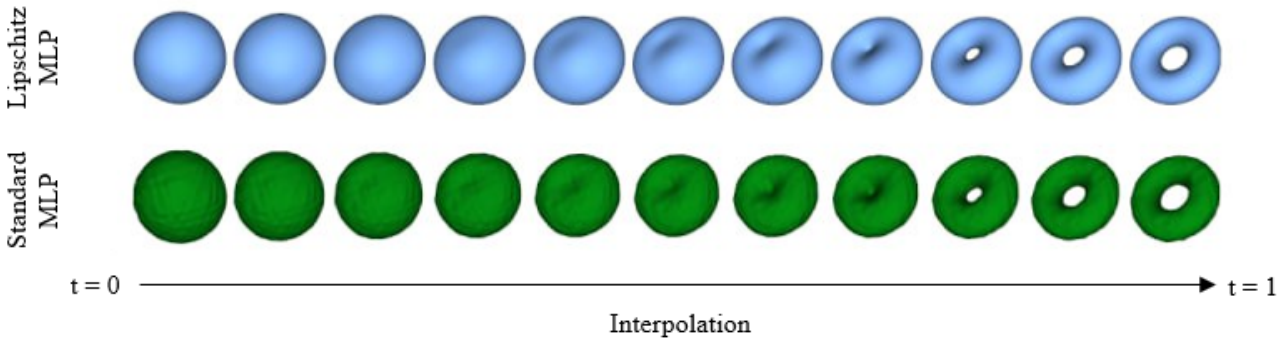

Figure 1: The interpolation between a Sphere and a Torus going from the latent code at t=0 (Sphere) to the latent code at t=1 (Torus).

3.3. Lipschitz Regularization

To satisfy the Lipschitz continuity in respect to the latent code $t \in \mathbb{R}^{|t|}$ the neural network f_θ, which takes as input a tuple of a location $x \in \mathbb{R}^{|d|}$ and a latent code t, should satisfy this constraint:

$$||f_\theta(x, t_0) - f_\theta(x, t_1)||_p \leq c\,||t_0 - t_1||_p$$

for all possibile combinations of x,t_0,t_1. The Lipschitz constant c bounds how fast this function f_θ can change and in a MLP is possible to estimate c by adding a weight normalization to each fully connected layer, parameterized by a learnable Lipschitz variable c. The normalization scheme depends on the choice of different matrix p-norms. The second step is to augment the loss with a Lipschitz term:

$$L(\theta) = L(\theta) + \alpha \prod_{i=1}^{L} \text{softplus}(c_i)$$

with c_i denoting the collection of the per-layer Lipschitz constant used in the architecture and $\text{softplus}(c_i) = \ln(1 + e^{c_i})$. The product of per-layer Lipschitz constants is the Lipschitz bound of the network.

3.4. Training

The losses used were two: the MSE loss which minimize the distance between the ground truth signed distance and the predicted one by the model and the Lipschitz loss.

4. Results

In the experiment the $p = \infty$ norm is used in which weights normalization is simply scaling individual rows to have a maximum absolute row sum smaller than a prescribed bound. The prescribed bound is calculated as $\text{softplus}(c_i)$, where c_i is the trainable parameter for that weight matrix. The shapes used as reference were three: Torus, Sphere and a Human. Once the model is trained the underlying surface is implicitly represented by the isosurface of $\{x \in \mathbb{R}^d | \text{SDF}(x) = 0\}$ with d = 3. A view of this implicit surface can be rendered through rasterization of a mesh obtained with the Marching Cubes algorithm. As shown in Figure 1 the Lipschitz MLP achieves smooth interpolation, while the Standard MLP obtain rough results. Ideally a linear interpolation between two meshes should linearly interpolate the volume and area. As shown in Figure 2 the Lipschitz MLP obtained better results.

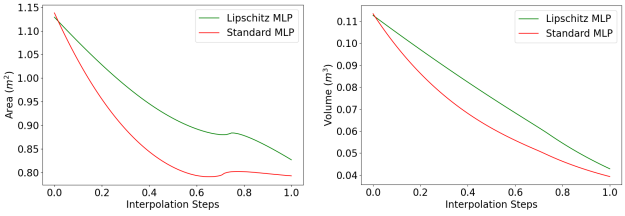

Figure 2: Comparision between the Lipschitz MLP and Standard MLP in terms of volume and area of the meshes obtained during the interpolation between a Sphere and a Torus

Additional studies on the performance of this technique using more complex shapes (e.g. humans) reveal the possibility of oversmoothing, where the model fails to learn the correct reconstruction of the shape.

5. Discussion and conclusions

The interpolation in the experiment obtained it's similar to an interpolation in which the metrics interpolate linearly and the regularization implemented is defined on a loose upper bound of the true Lipschitz constant. More tests can be done using different type of p-norms. Lipschitz regularization is a generic technique to encourage smooth neural network solutions and it will be interesting to explore this technique in other geometric tasks.

References

Anil, C., Lucas, J., and Grosse, R. Sorting out lipschitz function approximation, 2019.

Gouk, H., Frank, E., Pfahringer, B., and Cree, M. J. Regularisation of neural networks by enforcing lipschitz continuity, 2020.

Liu, H.-T. D., Williams, F., Jacobson, A., Fidler, S., and Litany, O. Learning smooth neural functions via lipschitz regularization, 2022.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks, 2018.

Terjék, D. Adversarial lipschitz regularization, 2020.