GeONet: a neural operator for learning the Wasserstein geodesic

DIGIMAT

Andrew Gracyk, Xiaohui Chen
University of Illinois at Urbana-Champaign, Department of Statistics

Introduction

We present GeONet, a neural operator deep learning framework for learning the Wasserstein geodesic.

GeONet is instantaneous, and allows for real-time predictions in the online setting, unlike traditional optimal transport numerical solvers.

Traditional methods require domain discretization and suffer from curse-of-dimensionality, unlike GeONet.

Finally, GeONet needs initial condition data only.

Background

The optimal transport problem was originally formulated by Monge, which seeks an optimal map such that

$$\min_{T:\mathbb{R}^d\to\mathbb{R}^d}\left\{\int_{\mathbb{R}^d} c(x,T(x))\,\mathrm{d}\mu_0(x) : T_\sharp\mu_0=\mu_1\right\}$$

for general cost function. We will consider quadratic cost $c(x,y)=\frac{1}{2}\|x-y\|_2^2$. The Monge problem induces a distance

$$W_2^2(\mu_0,\mu_1) := \min_\gamma\left\{\int_{\mathbb{R}^d\times\mathbb{R}^d}\|x-y\|_2^2\,\mathrm{d}\gamma(x,y)\right\}.$$

This can be re-expressed in a dual form

$$\frac{1}{2}W_2^2(\mu_0,\mu_1)=\sup\left\{\int_{\mathbb{R}^d}\varphi\,\mathrm{d}\mu_0+\int_{\mathbb{R}^d}\psi\,\mathrm{d}\mu_1 : \varphi(x)+\psi(y)\leqslant\frac{\|x-y\|_2^2}{2},\ \varphi\in L^1(\mu_0),\ \psi\in L^1(\mu_1)\right\}$$

The Benamou-Brenier dynamic formulation expresses the Wasserstein distance as a minimal flow kinetic energy problem

$$\frac{1}{2}W_2^2(\mu_0,\mu_1)=\min_{(\mu,\mathbf{v})}\int_0^1\int_{\mathbb{R}^d}\frac{1}{2}\|\mathbf{v}(x,t)\|_2^2\,\mu(x,t)\,\mathrm{d}x\,\mathrm{d}t$$
$$\text{subject to }\ \partial_t\mu+\mathrm{div}(\mu\mathbf{v})=0,\ \ \mu(\cdot,0)=\mu_0,\ \ \mu(\cdot,1)=\mu_1$$

where the probability density flow satisfies the continuity equation. To solve the above, we apply Lagrange multipliers, which yields a system of PDEs

$$\begin{cases}\partial_t\mu+\mathrm{div}(\mu\nabla u)=0,\ \ \partial_t u+\frac{1}{2}\|\nabla u\|_2^2=0,\\ \mu(\cdot,0)=\mu_0,\ \ \mu(\cdot,1)=\mu_1.\end{cases}$$

A neural operator is effectively a deep learning framework that learns a mapping $\Gamma^\dagger:\mathcal{A}\to\mathcal{U}$ between infinite-dimensional function spaces, typically defined through a PDE or differential operator. We construct a parametric map $\Gamma:\mathcal{A}\times\Theta\to\mathcal{U}$ to approximate the true operator, which can be done via a physics-informed risk

$$\theta^*=\mathrm{argmin}_{\theta\in\Theta}\ \ \mathbb{E}_{(a,u_0,u_T)\sim\mu}\Big[\big\|(\partial_t+\mathcal{D})\Gamma(a,\theta)\big\|_{L^2(\Omega\times(0,T))}^2$$
$$+\lambda_0\big\|\Gamma(a,\theta)(\cdot,0)-u_0\big\|_{L^2(\Omega)}^2+\lambda_T\big\|\Gamma(a,\theta)(\cdot,T)-u_T\big\|_{L^2(\Omega)}^2\Big]$$

Our method

DeepONets are a neural operator framework effective for physics-informed systems. This architecture is set up

$$\Gamma^\dagger(u_0,u_1)(x,t)\approx\sum_{\ell=1}^p\mathcal{B}_\ell(u_0(x_1),\dots,u_0(x_m),u_1(x_1),\dots,u(x_m))\mathcal{T}_\ell(x,t)$$

This architecture acts as a fine equivalence to the universal approximation theorem. It is our aim to learn a nonlinear operator mapping two input probability measures to the connecting geodesic, outlined by

$$\Upsilon^\dagger:\mathcal{P}_2(\Omega)\times\mathcal{P}_2(\Omega)\to\mathrm{AC}(\mathcal{P}_2(\Omega)),$$
$$(\mu_0,\mu_1)\mapsto\{\mu_t\}_{t\in[0,1]},$$

based on training data $\{(\mu_0^{(1)},\mu_1^{(1)}),\dots,(\mu_0^{(n)},\mu_1^{(n)})\}$. We implement 4 neural networks such that

$$\mathcal{C}(x,t,\mu_0,\mu_1,\theta^{\mathrm{cty}},\xi^{\mathrm{cty}})=\sum_\ell\mathcal{B}_\ell^{\mathrm{cty}}\mathcal{T}_\ell^{\mathrm{cty}},$$
$$\mathcal{H}(x,t,\mu_0,\mu_1,\theta^{\mathrm{HJ}},\xi^{\mathrm{HJ}})=\sum_\ell\mathcal{B}_\ell^{\mathrm{HJ}}\mathcal{T}_\ell^{\mathrm{HJ}}\ .$$

i.e., we have branch and trunk neural networks. Our optimization procedure is outlined as minimizing the empirical loss as a sum of continuity, HJ, and boundary losses

$$\phi^*,\psi^*\ \ =\ \ \mathrm{argmin}_{\phi,\psi\in\Theta\times\Xi}\ \ \mathcal{L}_{\mathrm{cty}}+\mathcal{L}_{\mathrm{HJ}}+\mathcal{L}_{\mathrm{BC}},$$

where

$$\mathcal{L}_{\mathrm{cty},i}\ =\ \frac{\alpha_1}{N}\left\|\frac{\partial}{\partial t}\mathcal{C}_{\theta,i}+\mathrm{div}(\mathcal{C}_{\theta,i}\nabla\mathcal{H}_{\xi,i})\right\|_{L^2(\Omega\times(0,1))}^2,$$
$$\mathcal{L}_{\mathrm{HJ},i}\ =\ \frac{\alpha_2}{N}\left\|\frac{\partial}{\partial t}\mathcal{H}_{\xi,i}+\frac{1}{2}\|\nabla\mathcal{H}_{\xi,i}\|_2^2\right\|_{L^2(\Omega\times(0,1))}^2,$$
$$\mathcal{L}_{\mathrm{BC},i}\ =\ \frac{\beta_0}{N}\left\|\mathcal{C}_{\theta,0,i}-\mu_0^{(i)}\right\|_{L^2(\Omega)}^2+\frac{\beta_1}{N}\left\|\mathcal{C}_{\theta,1,i}-\mu_1^{(i)}\right\|_{L^2(\Omega)}^2.$$

Training Algorithm

Algorithm 1 Train GeONet

Input: data pairs $(\mu_0^{(1)},\mu_1^{(1)}),\dots,(\mu_0^{(n)},\mu_1^{(n)})$; discretization size N; initialization of the neural network parameters $\phi\in\Theta\times\Xi,\psi\in\Theta\times\Xi$; weight parameters $\alpha_1,\alpha_2,\beta_0,\beta_1$.

1: **while** $\mathcal{L}_{\mathrm{total}}$ has not converged **do**
2: Independently draw N sample points $\{(x_\ell^i,t_\ell^i)\}_{\ell=1}^N$ from $U(\Omega)\times U(0,1)$
3: Compute $\Phi_i=\partial_t\mathcal{C}_{\phi,i}+\mathrm{div}(\mathcal{C}_{\phi,i}\nabla\mathcal{H}_\psi)$. ▷ `continuity residual`
4: Compute $\Psi_i=\partial_t\mathcal{H}_\psi+\frac{1}{2}\|\mathcal{H}_\psi\|_2^2$. ▷ `Hamilton-Jacobi residual`
5: Compute $B_{0,i}=\mathcal{C}_{\phi,0,i}-\mu_0^{(i)}(x_\ell^i),\ B_{1,i}=\mathcal{C}_{\phi,1,i}-\mu_1^{(i)}(x_\ell^i)$. ▷ `boundary residual`
6: Compute

$$\mathcal{L}_{\mathrm{cty}}=\frac{\alpha_1}{N}\sum_{i=1}^n\|\Phi_i\|_{L^2(\Omega\times(0,1))}^2,\quad\mathcal{L}_{\mathrm{HJ}}=\frac{\alpha_2}{N}\sum_{i=1}^n\|\Psi_i\|_{L^2(\Omega\times(0,1))}^2,$$
$$\mathcal{L}_{\mathrm{BC}}=\frac{1}{N}\sum_{i=1}^n\beta_0\|B_{0,i}\|_{L^2(\Omega)}^2+\beta_1\|B_{1,i}\|_{L^2(\Omega)}^2,$$

where $\|f\|_{L^2(\Omega\times(0,1))}^2:=\sum_{\ell=1}^N f^2(x_\ell,t_\ell)$ and $\|f\|_{L^2(\Omega)}^2:=\sum_{\ell=1}^N f^2(x_\ell)$.
7: Compute $\mathcal{L}_{\mathrm{total}}(\phi,\psi)=\mathcal{L}_{\mathrm{cty}}+\mathcal{L}_{\mathrm{HJ}}+\mathcal{L}_{\mathrm{BC}}$.
8: Minimize $\mathcal{L}_{\mathrm{total}}(\phi,\psi)$ to update ϕ and ψ. ▷ `minimize the loss function`
9: **end while**

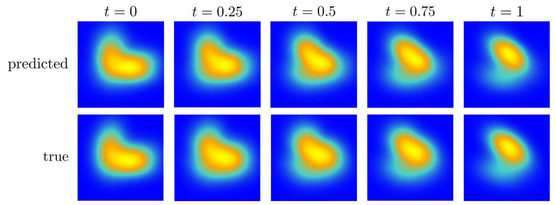
Experiments

We provide experimental results. We employ GeONet on Gaussian mixtures, which serve as continuous density estimators. Furthermore, we employ GeONet on real 32x32 CIFAR-10 images.

We begin by considering mixtures of the form

$$\mu_j(x)=\sum_{i=1}^{k_j}\pi_i\mathcal{N}(x|u_i,\Sigma_i)\qquad\text{subject to}\qquad\sum_{i=1}^{k_j}\pi_i=1$$

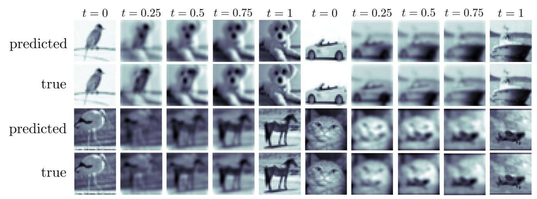
concentrated in a compact hypercube domain.



GeONet error for Gaussian mixtures

number of Gaussians	$t=0$	$t=0.25$	$t=0.5$	$t=0.75$	$t=1$
$k_0=3,k_1=2$	0.39 ± 0.33	2.6 ± 2.8	4.4 ± 5.1	2.9 ± 2.9	0.57 ± 0.61
$k_0=4,k_1=2$	0.73 ± 1.4	4.4 ± 8.1	7.1 ± 13	4.3 ± 6.3	0.58 ± 0.48
$k_0=5,k_1=2$	0.46 ± 0.39	3.3 ± 3.8	5.3 ± 5.9	3.6 ± 4.5	0.85 ± 2.2

Next, we employ GeONet on CIFAR-10 images.



MSE on CIFAR-10 data

Method	$t=0$	$t=0.25$	$t=0.5$	$t=0.75$	$t=1$
ER-GeONet	0.099 ± 0.14	12 ± 19	19 ± 27	11 ± 15	0.11 ± 0.32
GE-GeONet	0.76 ± 1.2	11 ± 19	18 ± 26	10 ± 15	0.47 ± 0.93

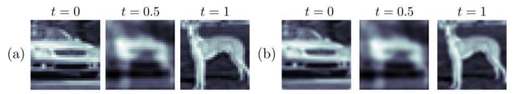
The above results were collected using a simple neural network architecture. We propose to implement a superior DeepONet architecture, which should be able to handle more sophisticated density geodesics.

We also intend to include baseline cases in our error analysis.

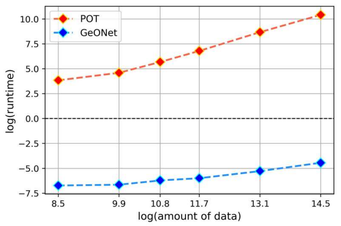
Method characteristics

GeONet is mesh-invariant, and is strong for super-resolution examples.

We may feed data along a low-dimensional discretization, and a geodesic along a high-dimensional domain can be produced.



GeONet requires a training procedure, but inference is instantaneous unlike traditional numerical solvers, which are particularly encumbered with a low regularization parameter and a high-dimensional domain discretization.



Acknowledgements

We would like to thank the DIGIMAT program for its support, as well as the UIUC Department of Statistics.

We would like to thank Duke University and the University of Chicago for the symposium as well.