Data Continuity Matters

Improving Sequence Modeling with Lipschitz Regularizer

Eric Qu[1,2] Xufang Luo[1] Dongsheng Li[1]

[1]Microsoft Research
[2]Duke Kunshan University

May 3, 2023

Motivation

Sequence Models Works Well On Specific Tasks

Transformers		State Space Models	
Text	Gene	Audio	Time-series

But Why?

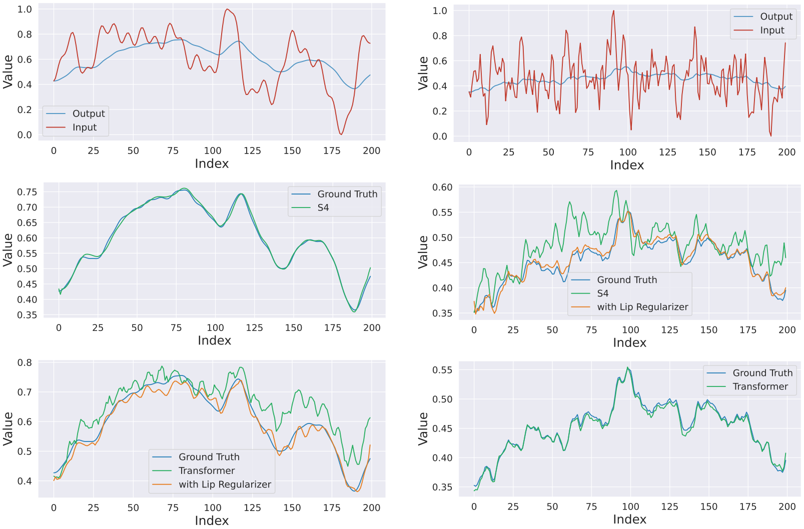
Motivation

Sequence models have preferences in **Data Continuity**

- Transformers $\rightarrow$ Discrete Data
- State Space models $\rightarrow$ Continuous Data

- Proof of Concept Experiment
  - Generate discrete and continuous input sequence
  - Map them to the output sequence with the same function
  - Use Transformer and S4 Model to learn this mapping

Motivation



(a) High Continuity (b) Low Continuity

Motivation

Sequence Models + Unpreferred Data Continuity

$\Downarrow$

Deteriorated Performance

- Solution: a regularizer that alters input data continuity!
- Apply the regularizer to the input embedding

Lipschitz Regularizer

- Continuity Measure: Lipschitz Constant
- For a sequence $x_0, x_1, \ldots, x_n$, view it as a sample of $f(t)$:

$$f(t_0) = x_0,\ f(t_1) = x_1,\ \ldots\ f(t_n) = x_n,$$

  where $t_0, t_1, \ldots, t_n$ are time steps.
- The Lipschitz constant L_f of $f(t)$ is

$$L_f = \max_{t_i, t_j \in \{0,1,\ldots,n\}} \frac{|f(t_i) - f(t_j)|}{|t_i - t_j|} = \max_{i,j \in \{0,1,\ldots,n\}} \frac{|x_i - x_j|}{|i - j|}.$$

- By Mean Value Theorem

$$L_f = \max_{i,j \in \{0,1,\ldots,n\}} \frac{|x_i - x_j|}{|i - j|} = \max_{k \in \{0,1,\ldots,n-1\}} |x_{k+1} - x_k|.$$

Lipschitz Regularizer

- To help with optimization, we introduce a surrogate:
  - Max $\rightarrow$ Mean
  - L1 norm $\rightarrow$ L2 norm

Definition 1: Lipschitz Regularizer

Suppose the sequence is $x_0, x_1, \ldots, x_n$. We define the Lipschitz Regularizer as follows:

$$\mathcal{L}_{\text{Lip}} = \frac{1}{n}\sum_{i=0}^{n-1}(x_{i+1} - x_i)^2 \tag{1}$$

State Space Model

- State Space models prefer continuous input
  - Assumption: input is a discrete sample of a continuous func
  - Higher input continuity $\rightarrow$ Lower HiPPO Leg-S error rate
- Use LipReg to make input continuous
  - Introduce a 1D Convolution embedding layer
  - Apply LipReg on the embedding

$$\mathcal{L}(y, \hat{y}, \hat{l}) = \mathcal{L}_{\mathrm{S4}}(y, \hat{y}) + \lambda \mathcal{L}_{\mathrm{Lip}}(\hat{l})$$

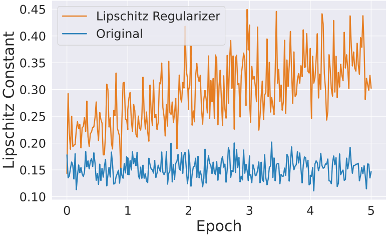
	ListOps	Text	Retrieval	Image	Image-c	Path	Path-c	PathX	PathX-c
S4	59.53	86.51	91.07	88.54	84.27	**94.02**	89.11	**96.03**	92.41
S4 + Emb	58.94	87.12	90.28	87.25	85.13	92.37	90.32	93.87	92.81
S4 + Emb + Lip	**61.37**	**89.74**	**93.83**	**89.19**	**88.43**	93.52	**91.39**	95.72	**94.36**

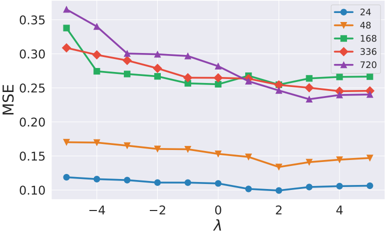
- Transformers prefer discrete input

$$\mathcal{L}(y, \hat{y}, \hat{l}) = \mathcal{L}_{\text{Transformer}}(y, \hat{y}) - \lambda \mathcal{L}_{\text{Lip}}(\hat{l})$$

Methods		Transformer		Transformer + Lip		Informer		Informer + Lip		Autoformer		Autoformer + Lip	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh_1	24	0.07047	0.20586	**0.07019**	**0.20570**	0.09842	0.24747	**0.08882**	**0.23674**	0.05567	0.18596	**0.05504**	**0.18495**
	48	0.18902	0.37046	**0.16716**	**0.34974**	0.15845	0.31907	**0.12615**	**0.28333**	0.07860	0.22324	**0.07422**	**0.21398**
	168	0.39773	0.55569	**0.30811**	**0.48183**	0.18314	0.34619	**0.10579**	**0.25552**	0.09232	0.24037	**0.08983**	**0.23544**
	336	0.41523	0.56902	**0.41324**	**0.56402**	0.22164	0.38720	**0.11810**	**0.26959**	0.10462	0.25484	**0.10461**	**0.25483**
	720	0.65586	0.75324	**0.62233**	**0.73160**	0.26883	0.43506	**0.13131**	**0.28731**	**0.12069**	**0.27791**	0.12394	0.27833
ETTh_2	24	0.09449	0.24259	**0.07560**	**0.20989**	0.09309	0.24015	**0.08626**	**0.22559**	0.11136	0.26315	**0.09345**	**0.25515**
	48	0.15016	0.30996	**0.13229**	**0.29278**	0.15483	0.31445	**0.13684**	**0.28936**	0.15137	0.30316	**0.14945**	**0.30129**
	168	0.25197	0.41087	**0.21046**	**0.37453**	**0.23193**	**0.38947**	0.30071	0.43671	0.20403	0.35646	**0.18370**	**0.33714**
	336	0.22258	0.38170	**0.20867**	**0.37298**	0.26321	0.41659	**0.24875**	**0.40827**	0.22188	0.37417	**0.21195**	**0.36425**
	720	0.21932	0.38844	**0.18445**	**0.35793**	0.27722	0.43063	**0.23646**	**0.39648**	0.25612	0.40089	**0.25604**	**0.40085**
ETTm_1	24	0.01279	0.08410	**0.01210**	**0.08312**	0.03016	0.13717	**0.01815**	**0.09147**	0.02317	0.11778	**0.02300**	**0.10107**
	48	0.08974	0.25869	**0.02872**	**0.12820**	0.06944	0.20255	**0.05848**	**0.19686**	0.04130	0.15783	**0.03931**	**0.15601**
	96	0.05341	0.17696	**0.05182**	**0.15017**	0.19414	0.37236	**0.13336**	**0.30091**	0.05432	0.18033	**0.05258**	**0.17605**
	288	0.22354	0.40455	**0.13780**	**0.29825**	0.40140	0.55355	**0.30266**	**0.46864**	0.11893	0.27181	**0.07521**	**0.21728**
	672	**0.40726**	**0.55824**	**0.40726**	0.55826	0.51164	0.64390	**0.27543**	**0.45377**	**0.09156**	0.23690	0.09280	**0.23621**
Weather	24	0.00223	0.03468	**0.00154**	**0.02497**	0.11676	0.25142	**0.11256**	**0.23844**	0.00740	0.06422	**0.00736**	**0.06329**
	48	0.00422	0.04106	**0.00292**	**0.03026**	**0.17822**	**0.31846**	0.19134	0.32408	0.01002	**0.07648**	**0.00978**	0.07727
	168	0.00537	0.05975	**0.00319**	**0.04464**	0.26585	0.39764	**0.25138**	**0.37400**	0.01038	0.07082	**0.00528**	**0.05638**
	336	0.00524	0.05772	**0.00417**	**0.03673**	0.29713	0.41571	**0.24748**	**0.37725**	0.00729	0.06492	**0.00566**	**0.05888**
	720	0.00933	0.07630	**0.00272**	**0.03823**	0.35875	0.46647	**0.26479**	**0.39214**	0.00960	0.08758	**0.00925**	**0.07136**
Count		2		39		4		36		4		36	

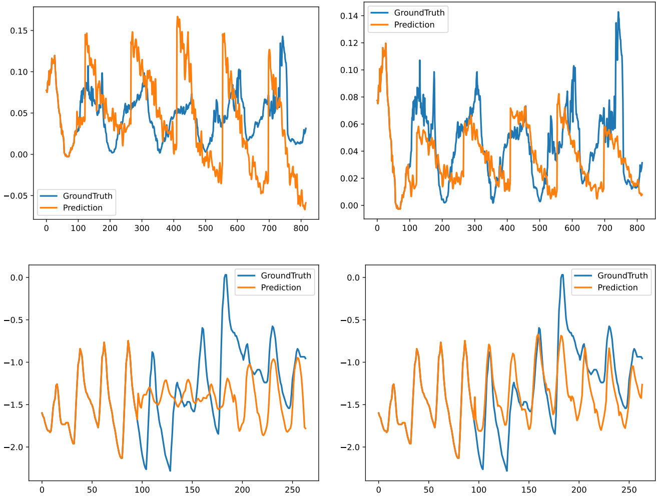
- Left: Change of L_f during training (ETTh$_2$, 24h)
  - LipReg effectively altered input continuity
- Right: MSE with different λ (ETTh$_2$)
  - Transformers prefer low continuity





Transformers
Time Series Prediction

Left: original; Right: with LipReg



More Experiments

- More Experiments:
  - Fine-tune Vision Transformers
  - Speech Transformers
  - Neural ODE

- In the Frequency Domain, the Lipschitz Regularizer is:

$$\begin{aligned}\sum_{i=0}^{n-1}(x_{i+1}-x_i)^2 &\approx \int_{\mathbb{R}} \left(\frac{\mathrm{d}f(t)}{\mathrm{d}t}\right)^2 \mathrm{d}t \\ &= \int_{\mathbb{R}} (2\pi i\xi)^2\, \hat{f}^2(\xi)(-\mathrm{d}\xi) \\ &= 4\pi^2 C\mathbb{E}_{p(\xi)}[\xi^2]\end{aligned}$$

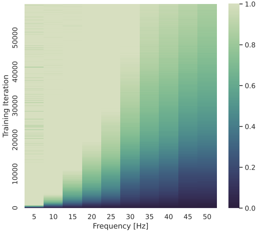
- An expectation over the frequency of the function
- Use it to penalize the frequency of the neural network

Spectral Bias

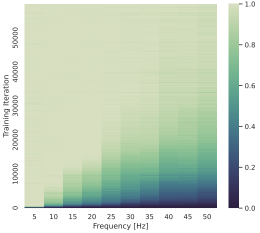
- Spectral Bias: low-frequency part is learned first
- Use LipReg to penalize the low-frequency part of NN

$$\mathcal{L}(y, \hat{y}) = \mathcal{L}_{\text{MSE}}(y, \hat{y}) - \lambda e^{-\epsilon t} \mathcal{L}_{\text{Lip}}(\hat{y})$$

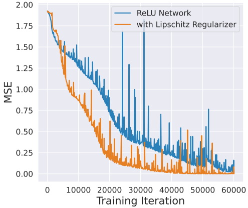
- Experiment: Curve fitting with ReLU Network
  - Training Iteration & Error in Frequency
  - Significantly reduce Spectral Bias



(a) Without LipReg



(b) With LipReg



(c) Training Loss

Thank you for your attention!

Link to Paper

Link to Code

Poster No. 66