SAND: Smooth Imputation Of Sparse And Noisy Functional Data With Transformer Networks

Ju-Sheng Hong*, Junwen Yao*, Jonas Mueller†, Jane-Ling Wang*
* University of California Davis † Cleanlab

See the Code!

Let's Connect!

Abstract

The transformer architecture has dominated in text and image processing, but its application to irregularly-spaced longitudinal data has been limited. We introduce SAND (Self-Attention on Derivatives), a simple module for transformers that encourages smoothness by modeling the sub-derivative of imputed curves. On the theoretical front, we prove the number of hidden nodes required by a network with SAND to achieve an ϵ prediction error bound for functional imputation. Extensive experiments demonstrate that SAND outperforms both standard transformers and transformers augmented with alternative approaches, as well as traditional statistical approaches like kernel smoothing and PACE.

Summary of Contributions

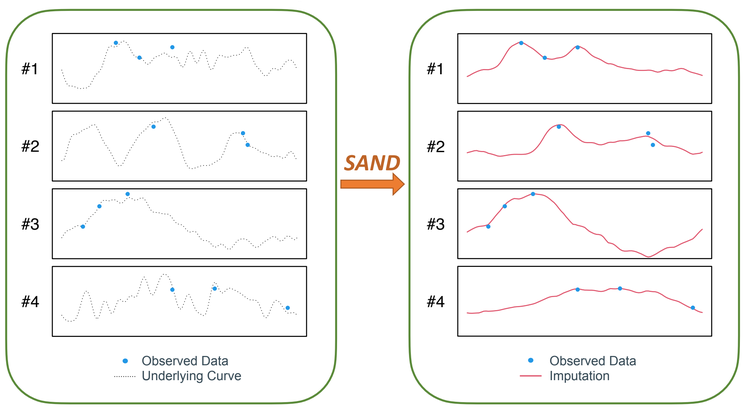
- **Empirical finding**: We demonstrate that the vanilla transformer outperforms traditional methods like PACE and neural processes for functional data imputation, though its imputations lack smoothness.
- **Novel contribution**: We introduce SAND (Self-Attention on Derivatives), a new layer that can be trained end-to-end with a transformer, ensuring smooth imputations without any assumptions on the data distribution.
- **Theoretical results**: We prove that imputations from SAND are first-order differentiable and provide an analytical expression relating the number of hidden nodes to the prediction error in SAND networks.

Sparse and Noisy Functional Data

- Functional data are random functions. e.g. energy consumption studies
- We focus on a univariate random curve $X(t)$ defined on $[0,1]$.
- Let n be the sample size. For $i \in \{1,\ldots,n\}$, the underlying curve $X_i(\cdot)$ is observed at time $t_{i1}, \ldots, t_{in_i}$ with observation: $Y_{ij} = X(t_{ij}) + \varepsilon_{ij}$.

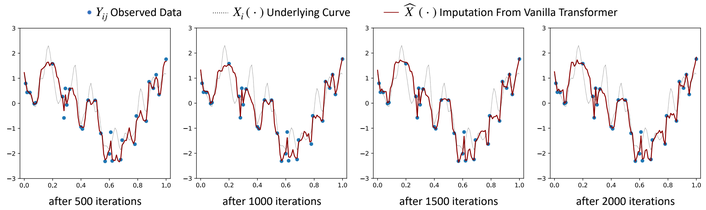
Scientific Motivation

Using observed data $\{(t_{ij}, Y_{ij})_{j=1}^{n_i}\}_{i=1}^{n}$ ***to recover the underlying curve*** $X_i(\cdot)$



Imputation Challenges with Vanilla Transformer

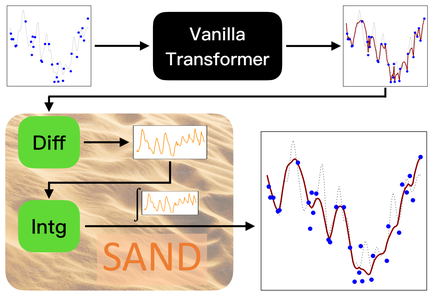
$$\text{Loss Function} = \sum_{i}\sum_{j=1}^{n_i}[\widehat{X}(t_{ij}) - Y_{ij}]^2 \text{ on the training set}$$



Why vanilla transformers may struggle with noisy data?

- When trained on noisy Y_{ij}, transformers produce zigzag-like imputations that reflect the noise (universal approximation property, [4]).
- During testing, if the input data also contains noise, well-trained models continue generating these zigzag patterns because the model has learned to replicate noise patterns present in the training data.

SAND — Self AtteNtion on Derivative



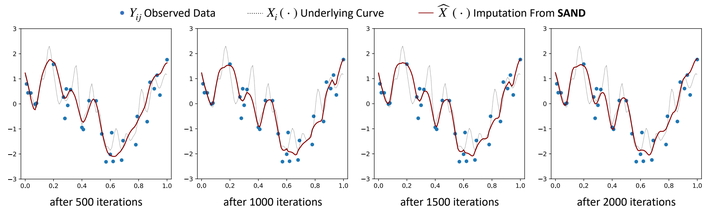
Technical Details of SAND

- Input: $\widetilde{T}$, an coarse imputation from a vanilla transformer
- Output: a smooth version of an input

$$\text{SAND}(\widetilde{T}) = (\widetilde{T})_1 + \text{Intg}[\text{Diff}(\widetilde{T})]$$

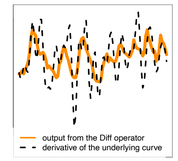
$$\text{where Diff}(\widetilde{T}) = \sum_{h=1}^{H} W_O^{(h)}\left(W_V^{(h)}\widetilde{T}\right)\left[\left(W_K^{(h)}\widetilde{T}\right)^{\top}\left(W_Q^{(h)}\widetilde{T}\right)\Big/\sqrt{h_d}\right]$$

and Intg is the cumulative summation operator.



SAND's Ability to Capture First Derivative

- Although Intg lacks direct access to the derivative and is not explicitly instructed to produce derivatives, it indirectly guides Diff to generate similar outcomes.
- This occurs even though Diff operates without direct information about the true derivative.



Simulation Study

- sample size $n = 10000$. Signal-to-noise ratio $= 4$

	$n_i = 30$		$n_i = 8$ to 12		$n_i = 3,4,5$	
	MSE(SD)	TV(SD)	MSE(SD)	TV(SD)	MSE(SD)	TV(SD)
PACE[1]	189.9(4.3)	187.1(2.0)	450.0(15)	201.9(2.1)	795.5(33)	209.5(2.2)
FACE[5]	284.6(8.8)	198.9(2.1)	488.2(16)	204.5(2.2)	807.1(32)	209.5(2.2)
mFPCA[6]	224.7(5.8)	192.0(2.1)	480.3(16)	204.0(2.2)	787.1(31)	**209.3**(2.2)
MICE[7]	176.7(3.7)	233.1(1.7)	721.6(27)	318.4(3.0)	1416(57)	332.7(2.8)
CNP[2]	290.4(11)	198.9(2.0)	551.3(21)	207.6(2.1)	920.3(52)	211.9(2.2)
GAIN[8]	261.9(6.8)	350.0(3.4)	1767(52)	743.3(5.1)	2065(51)	759.2(4.3)
1DS	262.9(6.0)	273.8(2.4)	735.3(22)	305.7(3.7)	1157(43)	263.3(3.1)
Transformers and our method						
VT[3]	169.8(3.2)	218.2(1.7)	436.7(15)	227.0(2.2)	798.6(35)	230.6(2.6)
VTP	169.0(3.5)	179.9(2.0)	425.3(14)	**199.4**(2.1)	**777.4**(36)	210.2(2.2)
SAND	**146.5**(2.7)	**164.6**(1.8)	**410.9**(13)	**196.8**(2.0)	**758.1**(43)	**206.8**(2.2)

Real Data

- Impute $n = 5500$ household's energy usage in London from Nov 13 — 14, 2013
- Impute $n = 890$ BMI curves from age 40 to 60 in Framingham Heart Study

	UK electricity						Framingham study	
	$n_i = 30$		$n_i = 8$ to 12		$n_i = 3,4,5$		$n_i = 3$ to 11	
	MSE(SD)	TV(SD)	MSE(SD)	TV(SD)	MSE(SD)	TV(SD)	MSE(SD)	TV(SD)
PACE	12.8(1.8)	**19.0**(1.1)	**30.1**(4.5)	**21.1**(1.2)	**39.6**(5.2)	**21.9**(1.2)	0.076(0.003)	**0.04**(0.002)
FACE	15.8(2.1)	21.3(1.2)	32.5(5.4)	22.6(1.2)	**39.6**(5.2)	23.0(1.2)	0.075(0.004)	**0.03**(0.002)
mFPCA	16.4(2.0)	22.2(1.2)	34.8(4.9)	23.2(1.2)	41.7(5.4)	23.3(1.2)	0.199(0.011)	0.325(0.013)
MICE	20.4(2.2)	67.8(3.3)	40.0(4.5)	65.4(2.8)	75.4(8.6)	71.4(1.5)	0.199(0.011)	0.32(0.013)
CNP	23.0(3.5)	21.4(1.2)	31.5(4.3)	22.1(1.2)	47.9(7.1)	**22.7**(1.2)	0.087(0.004)	**0.04**(0.002)
GAIN	31.9(3.7)	108(5.6)	75.4(8.2)	104(6.7)	99.6(15)	121(2.4)	0.212(0.019)	0.30(0.011)
1DS	17.3(2.2)	19.4(1.1)	50.0(7.0)	22.8(1.3)	105(18)	44.1(2.7)	0.135(0.009)	0.11(0.004)
VT	**10.7**(1.8)	20.6(1.1)	31.2(3.3)	23.2(1.3)	42.6(5.6)	38.5(2.5)	**0.004**(0.0002)	0.13(0.005)
SAND	**10.0**(1.9)	**15.7**(0.9)	**26.7**(3.0)	**20.1**(1.2)	**38.3**(5.1)	25.5(1.6)	**0.019**(0.001)	0.09(0.004)

Downstream Prediction Tasks

- [UK Electricity Dataset] Predict the average energy usage on Nov 15, 2013
- [Framingham Heart Study] Predict the average BMI from age 61 to 65

	UK electricity			Framingham study
	$n_i = 30$	$n_i = 8$ to 12	$n_i = 3,4,5$	$n_i = 3$ to 11
PACE	5.77(0.9)	11.8(2.2)	15.5(2.9)	2.23(0.7)
FACE	7.23(1.4)	10.8(2.1)	13.7(2.3)	2.10(0.6)
VT	6.22(1.1)	8.01(1.3)	**12.0**(1.9)	2.31(0.6)
SAND	**5.19**(0.6)	**7.39**(1.1)	**12.0**(2.0)	**1.76**(0.5)

References

1. Fang Yao, Hans-Georg Müller, and Jane-Ling Wang. Functional data analysis for sparse longitudinal data. Journal of the American Statistical Association, 2005.
2. Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. Conditional neural processes. In International Conference on Machine Learning, 2018.
3. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems, 2017.
4. Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations*, 2020.
5. Luo Xiao, Cai Li, William Checkley, and Ciprian Crainiceanu. Fast covariance estimation for sparse functional data. Statistics and computing, 2018
6. Jie Peng and Debashis Paul. A geometric approach to maximum likelihood estimation of the functional principal components from sparse longitudinal data. Journal of Computational and Graphical Statistics, 2009
7. Stef Van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in R. Journal of Statistical Software, 2011.
8. Jinsung Yoon, James Jordon, and Mihaela Schaar. Gain: Missing data imputation using generative adversarial nets. In International conference on machine learning. Proceedings of Machine Learning Research, 2018.