Supplementary Materials

for

Deep Learning for Functional Data Analysis with Adaptive Basis Layers

1 Proof of Theorem 1

Proof. Let $\tau_i : \mathbb{R}^q \to \mathbb{R}$ be a projection such that $\tau_i(v) = v_i$ for $v = (v_1, \ldots, v_q) \in \mathbb{R}^q$. It is straightforward to verify that τ_is are linear and continuous and $v = \big(\tau_1(v), \ldots, \tau_q(v)\big)$. Using this property, we can write $g = \big(\tau_1 \circ g, \ldots, \tau_q \circ g\big)$ with each $\tau_i \circ g : \mathcal{C}([0,1]) \to \mathbb{R}$ being a continuous linear functional. We further denote $c_i = \tau_i \circ g(f)$ and write $v_c = (c_1, \ldots, c_q)^\top$. By Riesz representation theorem, for each $i = 1, \ldots, q$, there exists $b_i \in \mathcal{L}_2([0,1])$ such that $\tau_i \circ g(f) = \langle f, b_i \rangle$ for every $f \in \mathcal{C}([0,1])$ and $\|\tau_i \circ g\|_{\text{op}} = \sup_{\|f\|=1} |\langle f, b_i \rangle| = \|b_i\|_2$. Since $\|f\|_2 \le 1$, we can check that $\|v_c\|_2 \le K$ where $K = \sqrt{q} \max_i \|b_i\|_2$.

Classical universal approximation results [Cybenko, 1989, Funahashi, 1989, Hornik, 1991, Stinchcombe, 1999] imply that for any $\epsilon > 0$, there exists a network with weights Θ^* such that $\sup_{\|v\|_2 \le 2K} |\text{nn}_{\Theta^*}(v) - h(v)| < \epsilon/2$. Since h is uniformly continuous on the compact set $D = \{v \in \mathbb{R}^q \,|\, \|v\|_2 \le 2K\}$, there exists $0 < \rho_\epsilon < K$ such that for any $v_1, v_2 \in D$, $\|v_1 - v_2\|_2 < \rho_\epsilon$ implies $|h(v_1) - h(v_2)| < \epsilon/2$.

On the other hand, it is well known that $\mathcal{C}([0,1])$ is dense in $\mathcal{L}_2([0,1])$. For each $i = 1, \ldots, q$, there exists $\tilde{b}_i \in \mathcal{C}([0,1])$ such that $\|\tilde{b}_i - b_i\|_2 < \rho_\epsilon/(2\sqrt{q})$. Classical universal approximation results imply that there exists a set of networks, each of which has weights Θ_i^*, such that $\sup_{t \in [0,1]} |\text{nn}_{\Theta_i^*}(t) - \tilde{b}_i(t)| < \rho_\epsilon/(2\sqrt{q})$. Denote $\tilde{c}_i = \langle \text{nn}_{\Theta_i^*}, f \rangle$ and write $\tilde{v}_c = (\tilde{c}_1, \ldots, \tilde{c}_q)^\top$. Then, we have

$$|\tilde{c}_i - c_i| \le |\langle \text{nn}_{\Theta_i^*} - \tilde{b}_i, f \rangle| + |\langle \tilde{b}_i - b_i, f \rangle| < \rho_\epsilon / \sqrt{q}$$

and thus $\|\tilde{v}_c - v_c\|_2 < \rho_\epsilon$.

Therefore, by linking these steps together, we have

$$\begin{aligned}
\sup_{\substack{f \in \mathcal{C}([0,1]) \\ \|f\|_2 \le 1}} |\widehat{\mathcal{T}}^*(f) - \mathcal{T}(f)| &\le \sup_{\substack{\|\tilde{v}_c - v_c\|_2 < \rho_\epsilon \\ v_c, \tilde{v}_c \in D}} |\text{nn}_{\Theta^*}(\tilde{v}_c) - h(v_c)| \\
&\le \sup_{v \in D} |\text{nn}_{\Theta^*}(v) - h(v)| + \sup_{\substack{\|v_1 - v_2\|_2 < \rho_\epsilon \\ v_1, v_2 \in D}} |h(v_1) - h(v_2)| \\
&< \epsilon/2 + \epsilon/2 = \epsilon.
\end{aligned}$$

For the second part of the claim, first note that X is a continuous random process defined on a compact interval. Its norm $\|X\|_2$ is a random variable on $\mathbb{R}$ and thus $\|X\|_2$ is stochastically bounded. In other words, for any $\delta > 0$, there exists a constant $M_\delta > 0$ such that $\mathbb{P}(\|X\|_2 \le M_\delta) > 1 - \delta$. Then, it is easy to verify that by using $\|X\|_2 \le M_\delta$ in replacement of $\|f\|_2 \le 1$ in the arguments above, we can obtain $\sup_{f \in \mathcal{C}([0,1]), \|f\|_2 \le M_\delta} |\widehat{\mathcal{T}}^\star(f) - \mathcal{T}(f)| < \delta$ for a suitable $\widehat{\mathcal{T}}^\star$. □

2 Proof of Theorem 2

Proof. Without loss of generality, we may assume that the model $\widehat{\mathcal{T}}_\Theta$ has one basis node in its BL and one hidden layer (with m nodes) in the subsequent network. Suppose that a numerical integration algorithm with weights $\{\omega_j\}_{j=1}^{J+1}$ and grid $\{t_j\}_{j=1}^{J+1}$ is used. Then, the model $\widehat{\mathcal{T}}_\Theta$ can be expressed as $\widehat{\mathcal{T}}_\Theta(X) = \sigma\big(\sum_{l=1}^m \theta_{l1} \sigma\big(\theta_{l2} \sum_{j=1}^{J+1} \omega_j X(t_j) \cdot \text{nn}_{\tilde{\Theta}}(t_j) + \theta_{l3}\big) + \theta_4\big)$, where σ is a nonlinear Lipschitz activation function (e.g., sigmoid or ReLU), and $\tilde{\Theta}$ denotes the parameters of the basis function network $\text{nn}_{\tilde{\Theta}}$. Note that here Θ denotes the collection of $\tilde{\Theta}$, θ_{l1}s, θ_{l2}s, θ_{l3}s, and θ_4.

It is straightforward to verify that both $\widehat{\mathcal{T}}_\Theta$ and $\nabla_\Theta \widehat{\mathcal{T}}_\Theta$ are bounded Lipschitz functions in Θ with bounded Lipschitz constants due to the condition $\sup_{t \in [0,1]} |X(t)| \le M_1$ and Assumption (i). We may also assume that

the the loss function $\ell(\cdot,\cdot)$ is non-negative, since we can always shift the bounded loss function by a positive constant so that its minimum value is non-negative. Using the condition $|Y| \leq M_2$, Assumption (ii), chain rule, and the fact that a composition of two Lipschitz functions is also a Lipschitz function, we conclude that $\ell\big(\widehat{\mathcal{T}}_\Theta(X), Y\big)$ and $\nabla_\Theta \ell\big(\widehat{\mathcal{T}}_\Theta(X), Y\big)$ are also Lipschitz in Θ with bounded Lipschitz constants. Therefore, the second part of the theorem follows directly from Theorem 3.12 in Hardt et al. [2016] by checking its conditions. □

3 Experiment Details

Table 1 summarizes the number of bases used in each of the four simulation settings and each of the nine tasks in the data experiments.

Table 1: Number of bases used for the four simulation settings and nine tasks on datasets. The error rate of each method (already in the main paper) is also reported right below the method. The smallest error is marked in bold.

Method	Case 1	Case 2	Case 3	Case 4	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8	Task 9
Raw data + NN	51	51	51	51	48	48	48	48	48	48	48	30	20
	0.015	0.038	0.275	0.334	0.099	0.284	0.124	0.296	0.380	0.488	0.472	0.406	0.373
B-spline (4) + NN	4	4	4	4	4	4	4	4	4	4	4	4	4
	0.050	0.984	0.971	0.369	-	-	-	-	-	-	-	-	-
B-spline (15) + NN	15	15	15	15	15	15	15	15	15	15	15	15	15
	0.013	0.019	0.206	0.251	0.094	0.306	0.137	0.326	0.335	**0.477**	0.429	0.413	0.387
$\text{FPCA}_{0.9}$ + NN	2	3	4	20	5	5	10	11	1	1	1	4	3
	0.917	0.023	0.134	0.855	-	-	-	-	-	-	-	-	-
$\text{FPCA}_{0.99}$ + NN	4	19	20	28	17	17	24	24	5	4	2	12	7
	0.003	0.036	0.239	0.667	0.119	0.339	0.143	0.306	0.363	0.493	0.431	0.429	0.378
AdaFNN	2	3	3	2	4	4	4	4	4	4	4	4	4
	0.001	**0.003**	**0.127**	**0.193**	**0.084**	**0.260**	**0.118**	**0.294**	**0.339**	**0.477**	**0.410**	**0.362**	**0.368**

3.1 Model description

Preprocessing before training. For all tasks, the functional input is standardized entry-wise using function `StandardScaler` in `Python` package `sklearn.preprocessing`. The response is also standardized using the same function in all regression tasks.

Basis Layer in AdaFNN. As each basis node in the Basis Layer is implemented as a micro network, the scale of its output can vary with the initialization of the micro network weights. To stabilize the numerical integration at a basis node, we can normalize the output of the micro networks, i.e., scale the output of each basis node such that its numerical integral is equal to 1. Another benefit of this normalization is that we do not have to normalize the learned basis functions again when applying the orthogonality/sparsity regularization.

B-spline scores. The B-spline scores used to represent functional inputs as a baseline method are computed using the spline functions (e.g., `smooth.spline`) in `R`, and the coefficients are computed individually for each curve.

FPCA scores. We use the function `FPCA` in the `R` package `fdapace` [Chen et al., 2019] to compute functional principal component scores. Principal components are estimated and selected based on the training dataset first and then used to compute the principal component scores of curves in the test dataset.

3.2 Data description

Simulated Datasets: Simulation datasets are generated based on the following model

$$X(t) = \sum_{k=1}^{50} c_k \phi_k(t), \quad t \in [0,1],$$

where terms on the right hand are defined as:

1. $\phi_1(t) = 1$ and $\phi_k(t) = \sqrt{2}\cos((k-1)\pi t)$, $k = 2, \ldots, 50$;
2. $c_k = z_k r_k$, and r_k are i.i.d. uniform random variables on $[-\sqrt{3}, \sqrt{3}]$.

Four simulation cases correspond to different configurations of z_ks:

1. In Case 1, $z_1 = 20, z_2 = z_3 = 5$, and $z_k = 1$ for $k \geq 4$. The response is $y = (\langle \phi_3, X \rangle)^2$;
2. In Case 2, $z_1 = z_3 = 5, z_5 = z_{10} = 3$, and $z_k = 1$ for other k. The response is $y = (\langle \phi_5, X \rangle)^2$.
3. Case 3 has the same configurations as Case 2. The observed response is
$$\tilde{y} = y + \epsilon = (\langle \phi_5, X \rangle)^2 + \epsilon, \quad \epsilon \sim N(0, 3/10).$$
For each time point t_j, the observed $X(t_j)$ is
$$\tilde{X}(t_j) = X(t_j) + \eta_j, \quad \eta_j \overset{\text{i.i.d.}}{\sim} N(0, 114/10).$$
4. In Case 4, $z_k = 1$ for all k. The response is $y = \langle \beta_2, X \rangle + (\langle \beta_1, X \rangle)^2$, where
$$\beta_1(t) = (4 - 16t) \cdot 1\{0 \leq t \leq 1/4\}$$
and
$$\beta_2(t) = (4 - 16|1/2 - t|) \cdot 1\{1/4 \leq t \leq 3/4\}.$$
The observed response is
$$\tilde{y} = y + \epsilon, \quad \epsilon \sim N(0, 1/10).$$
For each time point t_j, the observed $X(t_j)$ is
$$\tilde{X}(t_j) = X(t_j) + \eta_j, \quad \eta_j \overset{\text{i.i.d.}}{\sim} N(0, 5).$$
5. Case 5 has the same setup as Case 4, but with double the noise variance in Y.

In each case, 4000 curves are generated, among which 3200 are used for training while the rest 800 are left for testing. During training, 20% of the training data, i.e., 640 curves, are held out as a validation set.

Case 1 is designed to illustrate the weakness of the FPCA+NN approach, where the first two principle components explain at least 90% of the variability of the functional input X and are selected, but the true signal ϕ_3, which explains a very small fraction of the variability of X, is in the later principal components.

Case 2 is designed to illustrate the weakness of the B-spline+NN approach; here the true signal ϕ_5 in the functional covariate cannot be well represented by a small set of, e.g., four, B-splines.

Case 3 is designed to illustrate that AdaFNN is able to learn meaning basis functions and does better than other baseline methods in the appearance of both measurement error and noise.

Case 4 is designed to illustrate that AdaFNN can be used to select relevant domains and achieve smaller prediction error.

Case 5 is designed to illustrate the effectiveness of the regularizers.

Real Datasets:

Electricity Data [UK Power Networks, 2015]. The study contains electricity consumption readings for 5567 London households which participated in the Low Carbon London project from November 2011 to February 2014. Every half hour, there is a recording of the total electricity usage during the past half hour, so the total number of daily observations is 48. The observation periods vary among households. So, we select a period of 5 weeks in which the number of households in the project is the highest. This results in 5503 households. Since daily consumption is noisy, we take the average daily consumption during the first week period (to produce a smoother curve) as our functional covariate $X(t)$ for all prediction tasks. The training and test split is $4:1$, and 20% of the training data are held out for validation during the training process.

NHANES Data [NCHS, CDC 2020]. The study contains wearable device readings of physical intensity of 7742 subjects during a one week period. According to the data documentation, the device was the ActiGraph AM-7164 (formerly the CSA/MTI AM-7164), manufactured by ActiGraph located in Ft. Walton Beach, FL. It is programmed to detect and record the magnitude of acceleration or "intensity" of movement. The original intensity readings were summed over 1-minute epoch. To construct the functional input, we further aggregate the readings over 30 minute intervals, and this results in 48 observations per day. We take the average daily intensity curve for the whole week as the functional input $X(t)$. The prediction tasks here is to classify the health conditions of individual subjects using their physical intensity curve $X(t)$. Since not everyone reported their health conditions and there are invalid and incomplete information in some subjects, we ended up with 6555 subjects for Task 5, 2416 for Task 6, and 2412 for Task 7. The training and test split is taken to be $4:1$, and 20% of training data are held out for validation during the training process.

Mexfly [Carey et al., 2005] and Medfly[Chiou et al., 2003]. These two datasets are very similar, so we describe them together. The Mexfly data contain 1072 subjects and the Medfly data consist of 1000 subjects. For both data, the number of eggs laid daily was recorded for individual flies in the study until death. It is of biological interest to explore how early reproduction shapes lifetime reproduction, defined as the total number of eggs laid in a lifetime, which is perhaps the single most important biological question from the evolution point of view. We thus aim to predict the lifetime reproduction of a fly using its early reproduction trajectory $X(t)$, which is the number of eggs laid daily from birth till a specified day M shortly before peak reproduction period. This day M is 20 for the Medflies and 30 for the Mexflies. Next, we remove flies that died before day M and the resulted sample size is 872 for Mexflies (Task 8) and 870 for Medflies (Task 9). The training and testing split $4:1$ is the same as before.

Additional References for the Supplement

J. R. Carey, P. Liedo, H.-G. Müller, J.-L. Wang, D. Senturk, and L. Harshman. Biodemography of a long-lived tephritid: reproduction and longevity in a large cohort of female mexican fruit flies, anastrepha ludens. *Experimental Gerontology*, 40(10):793–800, 2005.

Y. Chen, C. Carroll, X. Dai, J. Fan, P. Z. Hadjipantelis, K. Han, H. Ji, H.-G. Mueller, and J.-L. Wang. *fdapace: Functional Data Analysis and Empirical Dynamics*, 2019. R package version 0.5.0.

J.-M. Chiou, H.-G. Müller, J.-L. Wang, and J. R. Carey. A functional multiplicative effects model for longitudinal data, with application to reproductive histories of female medflies. *Statistica Sinica*, 13(4): 1119, 2003.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989.

K.-I. Funahashi. On the approximate realization of continuous mappings by neural networks. *Neural Networks*, 2(3):183–192, 1989.

M. Hardt, B. Recht, and Y. Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, volume 48, pages 1225–1234. JMLR, 2016.

K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

National Center for Health Statistics (NCHS), Centers for Disease Control and Prevention (CDC). National health and nutrition examination survey data., 2020.

M. B. Stinchcombe. Neural network approximation of continuous functionals and continuous functions on compactifications. *Neural Networks*, 12(3):467–477, 1999.

UK Power Networks. Smartmeter energy consumption data in london households, 2015.