

DeepFunction: Deep Metric Learning-based Imbalanced Classification for Diagnosing Threaded Pipe Connection Defects using Functional Data

Yukun Xie^{a,b}, Juan Du^{a,c*}, Chen Zhang^d

^aSmart Manufacturing Thrust, Systems Hub, The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China; ^bAcademy of Interdisciplinary Studies, The Hong Kong University of Science and Technology, Hong Kong SAR, China; ^cDepartment of Mechanical and Aerospace Engineering, The Hong Kong University of Science and Technology, Hong Kong SAR, China; ^dDepartment of Industrial Engineering, Tsinghua University, Beijing, China

Contact by Email: juandu@ust.hk

DeepFunction: Deep Metric Learning-based Imbalanced Classification for Diagnosing Threaded Pipe Connection Defects using Functional Data

Abstract

In modern manufacturing, most of the product lines are conforming. Few products are nonconforming but with different defect types. The identification of defect types can help further root cause diagnosis of production lines. With the sensing development, signals of process variables can be collected in high resolution, which can be regarded as multichannel functional data. They have abundant information to characterize the process and help identify the defect types. Motivated by a real example from the pipe tightening process, we focus on defect classification where each sample is a multichannel functional data. However, the available samples for each defect type are limited and imbalanced. Moreover, the functions are incomplete since the pre-tightening process before the pipe tightening process is unobserved. To classify the defect samples based on imbalanced, multichannel, and incomplete functional data is very important but challenging. Thus, we propose an innovative classification framework based on deep metric learning using functional data (DeepFunction). The framework leverages the power of deep metric learning to train on imbalanced datasets. A neural network specially crafted for processing functional data is also proposed to handle multichannel and incomplete functional data. The results from a real-world case study demonstrate the superior accuracy of our framework when compared to existing benchmarks.

Keywords: imbalanced classification, incomplete functional data, functional neural network, pipe tightening process.

1. Introduction

Threaded pipe connections have extensive applications in various industries, particularly in petroleum drilling and transportation. The paramount importance of high-quality threaded pipe connections (*VAM book*, 2023) in ensuring safety during petroleum transportation is underscored by the fact that defective threaded pipe connections incur an annual cost of approximately half a billion USD (*Guangjie et al.*, 2006). Notably, a significant portion of the quality issues in threaded pipes arise from nonconforming connections. Thus, meticulous examination of the connection quality is a critical factor in determining the overall quality of threaded pipes.

The manufacturing process for threaded pipe connections depicted in Figure 1 (a) encompasses a pre-tightening process followed by a pipe tightening process (*Honglin et al.*, 2014). During the pre-

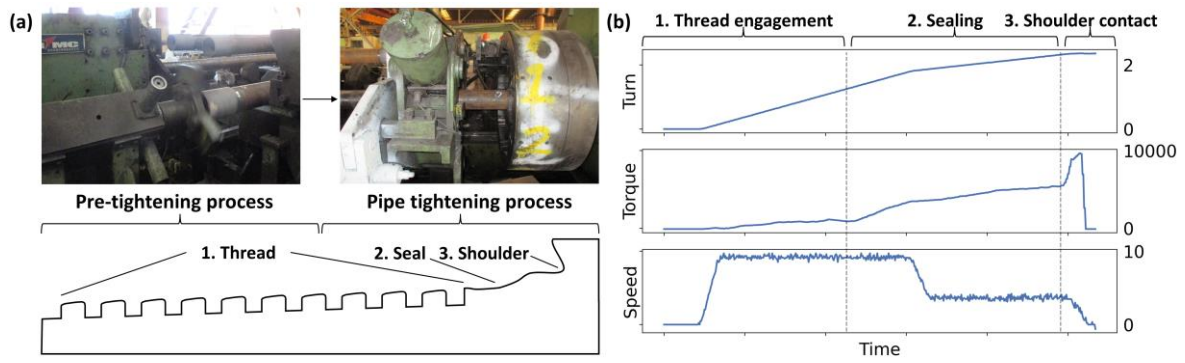


Figure 1. (a) Manufacturing process and structure of threaded pipe connections (Du *et al.*, 2017); (b) Sensor signals acquired in the pipe tightening process.

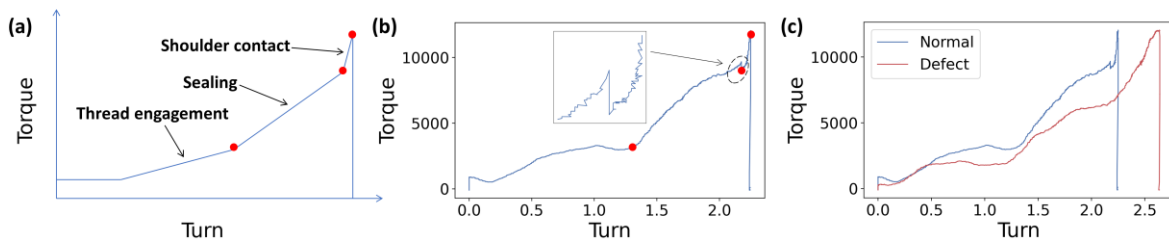


Figure 2. (a) Physical pattern of torque-turn function; (b) Torque-turn function from real sensor signals; (c) Torque-turn functions from one normal connection and one nonconforming connection with defect.

tightening process, most threads are initially screwed, with the final connection achieved in the subsequent pipe tightening process. As depicted in Figure 1 (b), sensors are positioned on pipe tightening machines to record process variables, including turns, torque, and tightening speed. These sensor signals provide a wealth of process-related information and reveal distinct phases, such as thread engagement, sealing, and shoulder contact, within the pipe tightening process. Consequently, the connection quality examination can be based on these functional data from sensors.

Currently, practitioners within the petroleum industry rely on torque-turn functions derived from sensor signals to manually detect nonconforming pipe connections (VAM book, 2023). Assuming that the torque-turn functions are piecewise linear, based on physical analysis (Mayne and Margolis, 1982), practitioners manually identify the change points between different phases of the pipe tightening process, as illustrated in Figure 2 (a). The torque values at these identified change points are then checked against the specifications of the VAM connections (VAM book, 2023). However, it is essential to note that oscillations in real sensor signals, as depicted in Figure 2 (b), will introduce bias to the

manually identified change points. In addition, manual labeling can only determine whether a connection is nonconforming. It is necessary for expert intervention to further diagnose the defects and implement appropriate remedial measures, such as the diagnosis of the nonconforming connection in Figure 2 (c). Therefore, establishing a classification method based on the functional data from sensors is warranted to automatically detect nonconforming connections and diagnose the connection defects, thereby enhancing manufacturing process efficiency and quality.

To construct the classification method, the characteristics of the manufacturing process pose the following challenges:

1. **Imbalanced Classification:** The ratio between conforming and nonconforming connections is highly skewed. Owing to the stability of the manufacturing process, there are scarce samples for each defect category. Both label imbalance and data scarcity pose significant challenges, such as overfitting, for the classification problem. As illustrated in Figure 3, the random samples from a threaded pipe connection assembly line exemplify the class imbalances in our case.

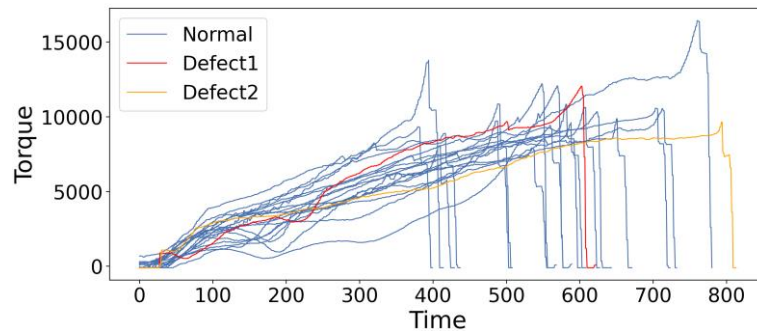


Figure 3. 20 samples of torque-time function in our case.

2. **Multichannel Functional Data:** Each sample is a multichannel function due to multiple process variables collected by sensors. As illustrated in Figure 1 (b), each channel exhibits a distinct range of values, with some ranging in tens and others in thousands.
3. **Incomplete Observation of Processes:** The incomplete observation of each process, particularly the unobserved pre-tightening process depicted in Figure 1 (a), results in incomplete functional data for each process. Traditional imputation or interpolation methods cannot effectively handle substantial and variable missing data segments within each function. Additionally, the inconsistency of the pre-tightening process results in varying-length observations for each pipe

tightening process, as shown in Figure 3. Thus, our method should accurately classify each sample despite the incomplete functional data.

The practical challenges described above are common in production lines, and the classification problem becomes even more challenging when the challenges are present simultaneously. Learning from multichannel functional data is a novel topic in the context of imbalanced classification. Although there have been numerous studies on imbalanced classification in recent years, the applicability of these methods to functional data remains uncertain. Moreover, the imbalanced classification problem needs to take into account the influence of learning from incomplete samples due to partial observability of the process. In the area of functional data analysis, there has been a functional neural network (Yao *et al.*, 2021) to learn from multichannel functions with varying-length observations. Prior research has also addressed challenges associated with incomplete functional data (Delaigle and Hall, 2013; James and Hastie, 2001; Kneip and Liebl, 2020; Kraus, 2015). However, the methods assume that the domain range for each sample is known and the pre-specification of domain ranges is required, which may not be practical for data from real production lines, such as the pipe tightening process in our case. Consequently, our method endeavors to tackle the practical challenges while leveraging existing research, including imbalanced classification and functional data analysis.

To overcome the challenges of classifying the imbalanced, multichannel, and incomplete functional data, we propose a novel framework combining a functional neural network with deep metric learning (DeepFunction). Specifically, a functional neural network is designed to directly learn fixed-length representations from multichannel functional data of varying lengths. To account for unobserved manufacturing processes, we employ a functional basis to pad the functional data before network encoding. In addition, we introduce a contrastive loss function tailored for deep metric learning on highly imbalanced functional datasets, thus facilitating efficient network training.

The contributions of our framework can be summarized as follows:

1. We propose a classification method for highly imbalanced functional data from manufacturing. By employing contrastive learning to address the imbalanced multi-class classification challenge, our framework realizes the identification of nonconforming pipe connections and the diagnosis of defect types.

2. With the application of functional neural networks, our framework can directly learn from the multichannel functional data. Our functional neural network surpasses traditional learning methods for functional data by enabling learnable and intricate transformations of functional data.
3. To handle the incomplete functional data, domain knowledge is incorporated into the padding mechanism of functional neural networks, thereby alleviating the influence of the unobserved parts of pipe tightening process. Moreover, the convolution-based functional basis can directly encode functional inputs without requiring pre-specification of domain ranges.

The remainder of the paper is organized as follows. In Section 2, we discuss related works in imbalanced classification and classification of multichannel and incomplete functional data. Section 3 provides a preliminary discussion of deep metric learning and functional neural networks. Section 4 presents a detailed illustration of our proposed framework, and Section 5 offers an analysis of the evaluation results obtained from a real dataset of threaded pipe connections. Finally, Section 6 concludes the paper.

2. Related Works

2.1 *Imbalanced Classification*

Considerable research has been dedicated to addressing the challenge of imbalanced classification. Traditional strategies for dealing with class imbalance can be broadly categorized into two groups: class rebalancing and data augmentation. Class rebalancing methods aim to balance the influence of different labels on the decision boundary. One representative is cost-sensitive learning (Elkan, 2001), which assigns larger weights to the minority labels in the loss functions. Data augmentation methods generate synthetic samples for minority labels to alleviate the imbalance problem. Examples of data generation methods include random oversampling, synthetic minority oversampling technique (SMOTE) (Chawla *et al.*, 2002), and deep learning methods such as generative adversarial networks (GAN) (Radford *et al.*, 2015). However, both class rebalancing and data augmentation methods lead to overfitting when defect samples are scarce. Although there have been cases where pre-trained models were used for the data scarcity issue, a massive amount of industrial data for model pre-training is not available in our case.

In recent years, the representation learning methods have shown great success in imbalanced classification given limited training samples. One branch of representation learning methods first learns the unsupervised representations of normal samples, and the model is then transferred to defect samples (Mou *et al.*, 2023). However, unsupervised learning can be challenging in cases where samples have varying lengths and are incomplete, as in our manufacturing dataset.

Another promising branch is to learn the supervised representations of each label based on deep metric learning. A neural network encoder first maps input data into the feature space, and the contrastive loss is then used to train the encoder and guarantee that the features are discriminative. Although existing studies have proposed losses based on contrastive learning to address imbalanced datasets (Khosla *et al.*, 2020; Wang *et al.*, 2021; Zhu *et al.*, 2022), these losses are primarily designed for image or speech data, and limited research focuses on classification using functional data in the literature. Therefore, both the contrastive loss and the neural network encoder need to adapt to the functional data in our case.

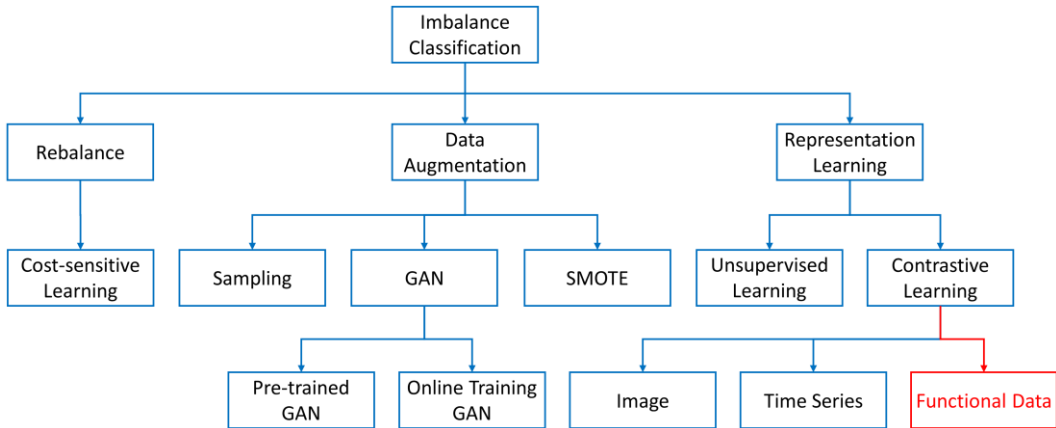


Figure 4. Methodology tree of imbalanced classification.

The methodology tree of imbalanced classification is in Figure 4, and our focus of research is the branch in red color. The theoretical knowledge of deep metric learning is given in Section 3.1.

2.2 Classification of Multichannel and Incomplete Functional Data

In modern manufacturing, sensors are placed throughout the manufacturing process and the data collected with sensors can be viewed as functional data. The functional data is one type of high-dimensional data, where a process variable is characterized by a functional relationship with other process variables. We use $\mathbf{x}(t), t \in [a, b]$ to denote one sample of functional data defined on a

compact interval. Since only a limited number of observations can be observed from $\boldsymbol{x}(t)$, we use $\boldsymbol{X} \in \mathbb{R}^{T \times C}$ to denote the T observations from multichannel function $\boldsymbol{x}(t)$ with C channels.

Due to the production environment and sensor locations, the manufacturing process cannot be observed completely, such as our threaded pipe connection case and clinical data (Elías *et al.*, 2022; Kraus, 2015). Therefore, the observations from the function $\boldsymbol{x}(t)$ are available only within a subset of $[a, b]$. Such datasets, known as incomplete functional data or partially observed functional data, are complete only within a specific interval $[a^*, b^*]$, where $a \leq a^* < b^* \leq b$.

To model and further classify the multichannel and incomplete functional data, current methods can be classified into three major categories: distance measure-based, reconstruction-based, and low-dimensional representation-based methods.

Distance measure-based methods quantify the dissimilarity between samples by defining distance measures. The representative distance measures that can be applied to functional data include dynamic time warping (DTW) and integrated depth for partially observed functional data (POIFD) (Elías *et al.*, 2022). However, it is important to note that DTW was originally designed for sequence alignment, and its ability to represent differences between different labels in a multi-class classification problem may be affected. POIFD (Elías *et al.*, 2022) assesses the centrality of functional data within a set of labeled functions. Nevertheless, POIFD (Elías *et al.*, 2022) is unsuitable for our case because it requires a pre-specified domain range for each sample. Furthermore, POIFD only treats multichannel functional data as a weighted sum of univariate functions, with the weights also requiring pre-specification.

Reconstruction-based methods aim to reconstruct the missing portions of a function and subsequently apply conventional classification techniques. Reconstruction is typically data-driven and can be based on fragments from other samples (Delaigle and Hall, 2013) or methods such as functional principal component analysis (FPCA) (Kneip and Liebl, 2020; Kraus, 2015). However, reconstruction is impractical when the domain range is unknown for each function. Moreover, data-driven reconstruction cannot leverage the physical mechanisms of the manufacturing process.

Low-dimensional representation-based methods directly transform functional data into low-dimensional representations with fixed lengths. Beginning with pioneering work (James and Hastie, 2001), various linear classifiers (Kraus and Stefanucci, 2019; Park *et al.*, 2022) have been proposed.

However, linear classifiers are not applicable because the assumptions of equal covariance for each label may not align with our dataset from the manufacturing process.

Apart from the limitations of each category of methods, most existing classification methods mentioned above focus primarily on binary classification, with limited exploration of imbalanced multi-class classification problems.

More recently, functional neural networks have shown great success in learning low-dimensional representations from functional data (Perdices *et al.*, 2021; Wang *et al.*, 2019; Yao *et al.*, 2021). With the power of neural networks, these approaches can directly encode multichannel functional data with varying-length observations and perform more intricate transformations of functional data, leading to improved discrimination results. Another benefit of a functional neural network is that we can account for the label imbalance during training, which has not been explored by current classification methods in functional data analysis. Details of the functional neural networks are introduced in Section 3.2.

3. Preliminary

3.1 Deep Metric Learning

The process of imbalanced classification based on deep metric learning can be decoupled into two stages: supervised representation learning and classifier training. During supervised representation learning, an encoder maps input data into a discriminative feature space. In the subsequent classifier training stage, the classifier is trained based on the learned representations.

When learning the representations, contrastive loss is utilized to minimize the distance between data representations from the same labels while maximizing the distance between data from different labels in the feature space. Suppose we have one sample i , and its representation is denoted as \mathbf{z}_i . The representation from samples having the same label as sample i is denoted as \mathbf{z}_p , and the representation from the other samples is denoted as \mathbf{z}_k . The basic contrastive loss for sample i is formulated as the following triplet contrast:

$$\mathcal{L}(i) = -\log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p)}{\exp(\mathbf{z}_i \cdot \mathbf{z}_k)}. \quad (1)$$

In the settings of supervised learning, multiple samples are known to have the same label. (Khosla et al., 2020) proposed the supervised contrastive loss when multiple positive samples and negative samples presents:

$$\mathcal{L}(i) = \frac{1}{|B_i| - 1} \sum_{p \in B_i \setminus \{i\}} -\log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p)}{\sum_{k \in B \setminus \{i\}} \exp(\mathbf{z}_i \cdot \mathbf{z}_k)}, \quad (2)$$

where B denotes the set of all training samples and B_i denotes the set of training samples having the same label as sample i .

To further alleviate the influence of imbalance samples, modified versions of Equation (2) have been proposed. (Wang et al., 2021) first learned the prototype of each label and then forced the representations of each label to be close to the prototype. (Zhu et al., 2022) balanced the contribution of each label to the loss function by adding the weights for the contrast between sample i and negative samples:

$$\mathcal{L}(i) = \frac{1}{|B_i| - 1} \sum_{p \in B_i \setminus \{i\}} -\log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p)}{\sum_{j=1}^J \frac{1}{|B_j|} \sum_{k \in B_j} \exp(\mathbf{z}_i \cdot \mathbf{z}_k)}, \quad (3)$$

where we suppose the training set B has J labels and the set of training samples under label j is denoted as B_j .

3.2 Functional Neural Network

The purpose of the functional neural network is to use a set of functional bases to project the functional input $\mathbf{x}(t)$ and extract representations. Denote the c th basis function as $\boldsymbol{\beta}_{(c)}(t), c = 1, \dots, C_1$, the score of $\mathbf{x}(t)$ regarding to the basis $\boldsymbol{\beta}_{(c)}(t)$ is as follows:

$$\langle \boldsymbol{\beta}_{(c)}(t), \mathbf{x}(t) \rangle = \int \boldsymbol{\beta}_{(c)}(t) \mathbf{x}(t) dt, \quad (4)$$

where both $\boldsymbol{\beta}_{(c)}(t)$ and $\mathbf{x}(t)$ have infinite dimensions. Notably, we use the case when dealing with single-channel functions as an illustration, and we only need to perform the same operation for each channel when dealing with multichannel functions.

There are two kinds of functional neural networks that incorporate Equation (4) into the neural networks. The first way is to use FPCA or pre-specified basis functions to model $\mathbf{x}(t)$ and use the result $\langle \boldsymbol{\beta}_{(c)}(t), \mathbf{x}(t) \rangle$ as the input of neural networks (Perdices et al., 2021). The second way is to directly use neural networks to approximate the functional basis (Wang et al., 2019; Yao et al., 2021).

The data encoded by the neural network is actually the observations of $\mathbf{x}(t)$, denoted as $\mathbf{X} = [\mathbf{x}(t_1), \dots, \mathbf{x}(t_T)]^T$. The discrete version of Equation (4) is then as follows:

$$\langle \boldsymbol{\beta}_{(c)}(t), \mathbf{x}(t) \rangle = \mathbf{X}^T \boldsymbol{\beta}_{(c)}, \quad (5)$$

where the basis vector $\boldsymbol{\beta}_{(c)} \in \mathbb{R}^{T \times 1}$ is expected to have equal length as \mathbf{X} . The basis function is learned through a multilayer perceptron (MLP), denoted as $\text{nn}(t), t \in [a, b]$. Given input \mathbf{X} with any length T , the basis vector is adaptively derived from $\text{nn}(t)$ as $\boldsymbol{\beta}_{(c)} = [\text{nn}(t_1), \dots, \text{nn}(t_T)]^T$. Compared to the first kind of functional neural network, the selection of functional bases or a prior is not required, and the information contained in response data can be used during learning.

One drawback of current functional neural networks is that the MLP-based functional basis lacks shift-invariance to the input and necessitates pre-specification of the domain range $[a, b]$ for each function. Although there has been practice developing functional neural networks shift-invariant to the input (Heinrichs *et al.*, 2023), the domain range for each function and the pre-specified basis functions are still required in (Heinrichs *et al.*, 2023). It is common for the functional data acquired in manufacturing cases to have varying-length observations and unknown domain ranges. Therefore, current functional neural networks can be further improved to apply in manufacturing cases, such as our pipe tightening process.

4. Methodology

In our pipe tightening process, certain characteristics necessitate the direct learning of multichannel and incomplete functional data without relying on model assumptions or requiring pre-specification of domain ranges. Additionally, the challenge posed by the imbalanced nature of the dataset should be addressed effectively. Therefore, there is a compelling need for a learning method tailored to incomplete and imbalanced functional data derived from manufacturing processes.

Our framework considers the underlying physical mechanisms of the manufacturing process and presents a neural network designed to encode multichannel and incomplete functional data directly. The core of the framework is a functional neural network-based encoder capable of transforming sensor-derived functional data into low-dimensional features. Notably, the encoder can be trained to ensure that the resulting features are discriminative across different labels. To address the challenges associated with the imbalanced nature of the dataset, we introduced a contrastive learning-based deep metric

learning framework. The framework facilitates the training of the encoder using a highly imbalanced dataset, ultimately leading to a feature space in which defect classification can be performed effectively.

Section 4.1 introduces the general framework. The functional neural network used to encode multichannel and incomplete functional data is elaborated in Section 4.2, and the contrastive loss designed for the imbalanced functional dataset is presented in Section 4.3. In Section 4.4, we provide guidelines for tuning the hyperparameters.

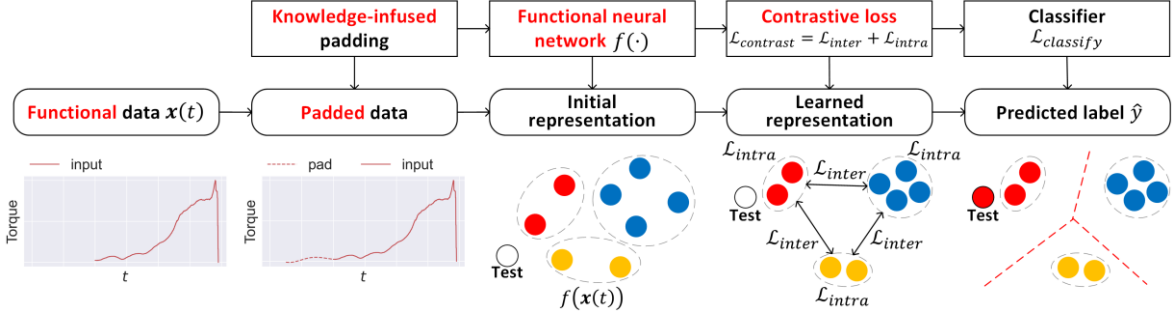


Figure 5. Flowchart of the framework.

4.1 General Framework

This section introduces the proposed framework for the classification of imbalanced and incomplete functional data. The general framework follows the paradigm of deep metric learning: each functional datum $\mathbf{x}(t)$ is first encoded into a low-dimensional representation and then classified based on discriminative representation. As shown in Figure 5, $\mathbf{x}(t)$ first undergoes knowledge-infused padding, which reconstructs the padded area required for the neural network. The proposed functional neural network encoder $f(\cdot)$ then derives a fixed-dimensional representation $f(\mathbf{x}(t))$. The initial representations of the samples from the different labels are mixed in the feature space. Therefore, a contrastive loss $\mathcal{L}_{contrast}$ is proposed, and the encoder $f(\cdot)$ is trained through backpropagation. After the representation learning stage, the learned representations under the same label are pulled together and the learned representations between different labels are pushed away. Finally, a classifier $\mathcal{L}_{classify}$, such as a support vector machine (SVM), is trained on the learned representations. The functional neural network encoder $f(\cdot)$ and the classifier $\mathcal{L}_{classify}$ after training are then used to predict the unlabeled samples.

It should be noted that the proposed functional neural network and contrastive loss can work collaboratively and enhance the performance of each other. The padding mechanism of a functional

neural network considers the physical mechanism of the manufacturing process, which guarantees that the padded values do not introduce bias to the functional input or subsequent contrastive loss. In return, contrastive loss enables the update of the functional neural network and improves the ability of the network to transform functional data after each iteration.

4.2 Functional Neural Network Encoder

As mentioned in Section 3.2, existing MLP-based functional neural networks (Wang et al., 2019; Yao et al., 2021) are not applicable in our case because the domain range for each function is unknown owing to partial observability. The neural network encoder should be shift-invariant and able to accept observations of functions with different ranges and lengths as inputs. Therefore, we developed a functional neural network based on dilated convolution (Oord *et al.*, 2016) and knowledge-infused padding.

The benefits of dilated convolution include shift-invariant features guaranteed by the convolution mechanism and the ability to accept varying-length inputs through dilated convolution. Assuming a dilation convolution layer with kernel size k_w and dilation size d and that the dilation is doubled for each layer, a convolution filter is applied over the input and skip input values with step 2^d . The receptive field grows exponentially with d , and the output length of the dilated convolution is still the input length T . Thus, the dilated convolution can use functional data $\mathbf{x}(t)$ with varying-length observations as the input.

Although dilated convolution has shown promise for handling functional data of varying lengths, the padding mechanism remains a challenge. Traditional padding methods can introduce bias into the distribution of functions, which can negatively affect the performance of the neural networks. Another challenge is designing a functional neural network with dilated convolution as the basic module. We illustrate the proposed solution for the two challenges in Section 4.2.1 and 4.2.2.

4.2.1 Knowledge-infused Padding

In our case, the potential bias introduced by padding is a significant concern because the padding length increases exponentially with the dilation size in the dilated convolution. For example, the padding length is $(k_w - 1)2^d$ for dilation size d when dilation is doubled for each layer. Therefore,

the padded values will significantly influence the original distribution of the functional data. To address the problem, we propose a novel approach called knowledge-infused padding. Similar to the reconstruction-based methods introduced in Section 2.2, our knowledge-infused padding attempts to reconstruct functions with a functional basis to generate the padded data. The selection of bases is a critical factor that directly affects the padding quality. Therefore, in this study, the selection of the functional basis is based on the physical mechanisms of the manufacturing process. Specifically, if the process exhibits a periodic pattern, the Fourier basis will be chosen to reconstruct the functional data collected from the process.

Algorithm 1. Knowledge-infused Padding

Input: Functional data $\mathbf{x}(t), t = 1, \dots, T$; Convolution layer with kernel size k_w and dilation size d ; Average pooling layer with kernel size k_w ; Basis functions $\{\phi_{(g)}(t)\}, g = 1, \dots, G$;

Output: Padded data $\mathbf{x}(t), t = 1, \dots, T'$

- 1: Obtain smoothed function $\bar{\mathbf{x}}(t) = \frac{1}{k_w} \sum_{s=t}^{t+k_w} \mathbf{x}(s), t = 1, \dots, T - k_w$
 - 2: Fit $\bar{\mathbf{x}}(t)$ with functional basis $\bar{\mathbf{x}}(t) = \sum_{g=1}^G c_g \phi_{(g)}(t)$
 - 3: Interpolate $\bar{\mathbf{x}}(t)$ to length $T' = T + 2(k_w - 1)2^d$
 - 4: Obtain the padded data $\mathbf{x}(t)$:
 - 5: Set $\mathbf{x}(t) = \bar{\mathbf{x}}(t), t = 1 - (k_w - 1)2^d, \dots, 0$
 - 6: Set $\mathbf{x}(t) = \bar{\mathbf{x}}(t), t = 1, \dots, T$
 - 7: Set $\mathbf{x}(t) = \bar{\mathbf{x}}(t), t = T + 1, \dots, T + (k_w - 1)2^d$
-

Compared with the reconstruction-based method, the reconstruction area in our framework is only the padded part in the convolution, rather than the entire missing part. The reason is that we assume the most discriminative patterns exist in the common domain, which is complete for every sample. After specifying the functional basis $\{\phi_{(g)}(t)\}, g = 1, \dots, G$, the knowledge-infused padding module is given in Algorithm 1 for any convolution layer with kernel size k_w and dilation size d .

4.2.2 Network Structure

To encode the functional data $\mathbf{x}(t)$ for downstream classification tasks, the encoder should consider the functional data structures while ensuring that the entire encoder is trained to extract the

discriminative features. Our proposed functional neural network $f(\cdot)$ achieves this by utilizing dilated convolution as the foundational structure. The details of the network structure are shown in Figure 6.

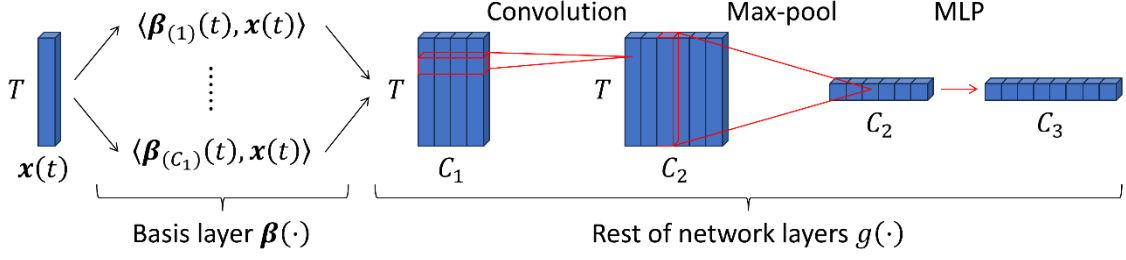


Figure 6. Structure of the functional neural network $f(\cdot)$.

Given the final representation size C_3 , the network shown in Figure 6 directly encodes the function $\mathbf{x}(t)$ to obtain the representation $\mathbf{z} \in \mathbb{R}^{1 \times C_3}$ after knowledge-infused padding. Specifically, the network is composed of two parts: a linear transform with a basis layer $\beta(\cdot)$ and a nonlinear transform with a traditional neural network $g(\cdot)$. The output for $\mathbf{x}(t)$ encoded by $f(\cdot)$ can be formulated as $f(\mathbf{x}(t)) = g(\beta(\mathbf{x}(t)))$.

Algorithm 2. Basis layer $\beta(\cdot)$ based on dilated convolution

Input: Functional data $\mathbf{x}(t)$ and its observations $\mathbf{X} \in \mathbb{R}^{T \times 1}$; D layers of dilated convolution $\{\mathbf{b}_{(c)}^{(l)}(t), \mathbf{h}_{(c)}^{(l)}(t)\}, c = 1, \dots, C_1, l = 0, \dots, D - 1$;

Output: $\beta(\mathbf{x}(t)) \in \mathbb{R}^{T \times C_1}$

1. **for** $c = 1, \dots, C_1$:
 2. Obtain $\mathbf{B}_{(c)}^{(0)} \in \mathbb{R}^{T \times 1}$ and $\mathbf{H}_{(c)}^{(0)} \in \mathbb{R}^{T \times T}$ from convolution $\{\mathbf{b}_{(c)}^{(0)}(t), \mathbf{h}_{(c)}^{(0)}(t)\}$
 3. Calculate $\mathbf{u}_{(c)}^{(0)} = \mathbf{B}_{(c)}^{(0)} + \mathbf{H}_{(c)}^{(0)}\mathbf{X}$
 4. **for** $l = 1, \dots, D - 1$:
 5. Obtain $\mathbf{B}_{(c)}^{(l)} \in \mathbb{R}^{T \times 1}$ and $\mathbf{H}_{(c)}^{(l)} \in \mathbb{R}^{T \times T}$ from convolution $\{\mathbf{b}_{(c)}^{(l)}(t), \mathbf{h}_{(c)}^{(l)}(t)\}$
 6. Calculate $\mathbf{u}_{(c)}^{(l)} = \mathbf{B}_{(c)}^{(l)}(t) + \mathbf{H}_{(c)}^{(l)}\mathbf{u}_{(c)}^{(l-1)}$
 7. **end for**
 8. Set $\langle \beta_{(c)}(t), \mathbf{x}(t) \rangle = \mathbf{u}_{(c)}^{(D-1)}$
 9. **end for**
 10. Set $\beta(\mathbf{x}(t)) = [\langle \beta_{(1)}(t), \mathbf{x}(t) \rangle, \dots, \langle \beta_{(C_1)}(t), \mathbf{x}(t) \rangle]$
-

The basis layer $\beta(\cdot)$ is implemented by stacking D dilated convolution layers without activation and performs similarly to the basis layer in (Yao et al., 2021). We denote the dilated convolution layers as $\{\mathbf{b}_{(c)}^{(l)}(t), \mathbf{h}_{(c)}^{(l)}(t)\}, c = 1, \dots, C_1, l = 0, \dots, D - 1$ with bias $\mathbf{b}_{(c)}^{(l)}(t)$ and weight $\mathbf{h}_{(c)}^{(l)}(t)$. The procedures in $\beta(\cdot)$ for the input of single-channel functions are shown in Algorithm 2. When dealing with multichannel functions with C channels, we need to perform Algorithm 2 on each channel of the multichannel function.

The essence of the functional structure lies in the functional basis implemented by convolution operation. As shown in Algorithm 2, we do not add activation functions for the D layers of the dilated convolution $\{\mathbf{b}_{(c)}^{(l)}(t), \mathbf{h}_{(c)}^{(l)}(t)\}$, which means that the entire basis layer $\beta(\cdot)$ does not contain nonlinearity. By transforming the convolution weights $\mathbf{h}_{(c)}^{(l)}(t)$ into a Toeplitz matrix $\mathbf{H}_{(c)}^{(l)} \in \mathbb{R}^{T \times T}$ and bias $\mathbf{b}_{(c)}^{(l)}(t)$ as a vector $\mathbf{B}_{(c)}^{(l)} \in \mathbb{R}^{T \times 1}$, we can derive $\langle \beta_{(c)}(t), \mathbf{x}(t) \rangle = \mathbf{B}_{(c)} + \mathbf{H}_{(c)}\mathbf{X}$. $\mathbf{B}_{(c)}$ and $\mathbf{H}_{(c)}$ are derived as follows:

$$\mathbf{B}_{(c)} = \mathbf{B}_{(c)}^{(D-1)} + \mathbf{H}_{(c)}^{(D-1)}\mathbf{B}_{(c)}^{(D-2)} + \dots + \left(\prod_{l=1}^{D-1} \mathbf{H}_{(c)}^{(l)}\right)\mathbf{B}_{(c)}^{(0)}, \quad (6)$$

$$\mathbf{H}_{(c)} = \prod_{l=0}^{D-1} \mathbf{H}_{(c)}^{(l)}, \quad (7)$$

where matrix $\mathbf{H}_{(c)}$ is the Toeplitz matrix for all D convolution layers and can be viewed as a matrix basis. Therefore, the output of the basis layer $\beta(\cdot)$ is still functional data. Compared with the MLP-based functional neural network (Wang et al., 2019; Yao et al., 2021), $\beta(\cdot)$ based on convolution makes the network shift-invariant to the input functions; Compared with current convolution-based functional neural network (Heinrichs et al., 2023), $\beta(\cdot)$ is adaptive and can accept the functional input without specifying the domain range and basis functions.

Moreover, we can interpret the C_1 channels of $\beta_{(c)}(t)$ in $\beta(\cdot)$ as C_1 functional bases. Similar to traditional functional modeling with a set of bases (Ramsay *et al.*, 2005), C_1 functional bases $\beta_{(c)}(t)$ can be interpreted well through visualization and identification of the most important bases.

After transforming $\mathbf{x}(t)$ into $\beta(\mathbf{x}(t))$ with basis layers, the neural network $g(\cdot)$ further transforms $\beta(\mathbf{x}(t))$ into representation \mathbf{z} through a convolution layer with C_2 channels and a $C_2 \times C_3$ MLP structure. The bias and weight for the convolution layer are denoted as \mathbf{B}' and \mathbf{H}' , and the bias and weight for the MLP layer are denoted as \mathbf{E} and \mathbf{R} . The procedure for nonlinear

transformation $g(\cdot)$ is shown in Algorithm 3. The representation \mathbf{z} is then used to calculate the loss functions in Section 4.3.

Algorithm 3. Nonlinear transformation $g(\cdot)$

Input: Output of basis layer $\boldsymbol{\beta}(\mathbf{x}(t)) \in \mathbb{R}^{T \times C_1}$; Dilated convolution with bias $\mathbf{B}' \in \mathbb{R}^{T \times C_2}$, weights $\mathbf{H}' \in \mathbb{R}^{C_1 \times C_2}$, and LeakyReLU activation; AdaptiveMaxPooling layer; MLP layer with bias $\mathbf{E} \in \mathbb{R}^{1 \times C_3}$ and weights $\mathbf{R} \in \mathbb{R}^{C_2 \times C_3}$;

Output: $\mathbf{z} \in \mathbb{R}^{1 \times C_3}$

- 1: Calculate $\mathbf{v}_1 \in \mathbb{R}^{T \times C_2} = \mathbf{B}' + \boldsymbol{\beta}(\mathbf{x}(t))\mathbf{H}'$
 - 2: Calculate $\mathbf{v}_2 \in \mathbb{R}^{T \times C_2} = \text{LeakyReLU}(\mathbf{v}_1)$
 - 3: Calculate $\mathbf{v}_3 \in \mathbb{R}^{1 \times C_2} = \text{AdaptiveMaxPool}(\mathbf{v}_2)$
 - 4: Calculate $\mathbf{z} \in \mathbb{R}^{1 \times C_3} = \mathbf{E} + \mathbf{v}_3\mathbf{R}$
-

The details and proofs for constructing the Toeplitz matrix form of $\mathbf{B}_{(c)}$ and $\mathbf{H}_{(c)}$ in functional neural network $\boldsymbol{\beta}(\cdot)$ are provided in Appendix A.1. We also illustrate how to visualize the functional bases learned from $\boldsymbol{\beta}(\cdot)$ in Appendix A.2. Guidelines for setting hyperparameters D, C_1, C_2, C_3 are also provided in Section 4.4.

4.3 Contrastive Loss

To train the functional neural network $f(\cdot)$ in Section 4.2, a loss function should be proposed for tackling label imbalance and data scarcity issues when learning from the highly imbalanced functional dataset. We employ deep metric learning to train $f(\cdot)$, and the contrastive loss $\mathcal{L}_{contrast}$ is used.

Suppose that a dataset has J labels and N_j samples for label j . We first generate a mini-batch B in each epoch through stratified sampling, and sample size M_j for each label satisfies $M_1/N_1 = \dots = M_j/N_j = \dots = M_J/N_J$. Similar to Section 3.1, one sample i is selected as the anchor sample and its representation is $\mathbf{z}_i = f(\mathbf{x}^{(i)}(t))$. The representations of positive samples and negative samples are denoted as $\mathbf{z}_p = f(\mathbf{x}^{(p)}(t))$ and $\mathbf{z}_k = f(\mathbf{x}^{(k)}(t))$, respectively. B_i denotes the set of samples having the same label as sample i and B_j denotes the set of samples under label j . $\mathcal{L}_{contrast}$ is then calculated after all samples in the mini-batch B are encoded by $f(\cdot)$. The formula for $\mathcal{L}_{contrast}$ is given in Equation (8-10):

$$\mathcal{L}_{contrast}(i, Q) = \mathcal{L}_{inter}(i) + \mathcal{L}_{intra}(i, Q), \quad (8)$$

$$\mathcal{L}_{inter}(i) = \frac{1}{|B_i| - 1} \sum_{p \in B_i \setminus \{i\}} -\log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_p)}{\sum_{j=1}^J \frac{1}{|B_j|} \sum_{k \in B_j} \exp(\mathbf{z}_i \cdot \mathbf{z}_k)}, \quad (9)$$

$$\mathcal{L}_{intra}(i, Q) = \frac{1}{|B_i| - 1} \sum_{p \in B_i \setminus \{i\}} -\log \frac{\frac{1}{|Q|} \sum_{q \in Q} \exp(f(q(\mathbf{x}^{(i)}(t))) \cdot f(q(\mathbf{x}^{(p)}(t))))}{\exp(\mathbf{z}_i \cdot \mathbf{z}_k)}. \quad (10)$$

As shown in Equation (8), $\mathcal{L}_{contrast}$ is composed of an inter-class loss \mathcal{L}_{inter} and an intra-class \mathcal{L}_{intra} . Considering the imbalanced distribution of labels, \mathcal{L}_{inter} in Equation (9) is the same as (Zhu et al., 2022) which distinguishes the representations between imbalanced labels. For the contrast between the representations of anchor samples and negative samples $\exp(\mathbf{z}_i \cdot \mathbf{z}_k)$, the weights $1/|B_j|$ balance the influence of different sample size for each label.

Regarding the data scarcity issue, we propose an intra-class loss \mathcal{L}_{intra} in Equation (10) to avoid overfitting. To obtain more samples under one label, we augment the anchor sample $\mathbf{x}^{(i)}(t)$ and positive sample $\mathbf{x}^{(p)}(t)$. Notably, traditional augmentation methods used for images or time series may introduce bias into the distribution of functional data. We modify the augmentation method during the generation of the training samples to be suitable for functional data, aligning it with the characteristics of the manufacturing dataset. The augmentation is achieved by applying a set of smoothing kernels, denoted as set Q . Each smoothing kernel $q \in Q$ is implemented through the average pooling layers with kernel size k_q , and we can set kernel set Q in advance. In contrast to \mathcal{L}_{inter} , the contrast pairs in \mathcal{L}_{intra} are limited to samples within the same label and are applied using the same smoothing kernel.

$$q(\mathbf{x}^{(i)}(t)) = \frac{1}{k_q} \sum_{s=t}^{t+k_q} \mathbf{x}^{(i)}(s), t = 1, \dots, T - k_q, \quad (11)$$

$$q(\mathbf{x}^{(p)}(t)) = \frac{1}{k_q} \sum_{s=t}^{t+k_q} \mathbf{x}^{(p)}(s), t = 1, \dots, T - k_q. \quad (12)$$

4.4 Hyperparameter Tuning

In this section, we present the guidelines for tuning the hyperparameters in our framework. The hyperparameters include D, C_1, C_2, C_3 in the functional neural network $f(\cdot)$ and the kernel set Q in the contrastive loss $\mathcal{L}_{contrast}$.

The number of convolution layers D and the number of bases C_1 determine the performance of basis layer $\beta(\cdot)$. The ability to fit complex functions increases with increasing D , whereas the padded length $2 * (k_w - 1) * 2^D$ grows exponentially with D . Therefore, D is set to satisfy the requirement

that the padded length should not exceed the raw data length $2 * (k_w - 1) * 2^D < T$ for all the functions. The number of bases C_1 can first be set as an initial value $J * C * S$, where S is the number of estimated piecewise linear segments for one function. During training, additional bases among the C_1 bases can be pruned by adding a channel-wise attention module to the neural network $f(\cdot)$. Channels C_2 and C_3 are the parameters for the traditional neural network $g(\cdot)$, and the network structure is the same as in previous practices (Franceschi *et al.*, 2019; Yue *et al.*, 2022). Therefore, C_2 and C_3 can be set using the same settings as (Franceschi *et al.*, 2019).

The setting of smoothing kernels $q \in Q$ can follow the procedure of smoothing functional data with a roughness penalty introduced in (Ramsay *et al.*, 2005).

5. Evaluation

In this section, we apply the proposed framework to a real dataset obtained from the manufacturing of threaded pipe connections to demonstrate its effectiveness. A total of 658 samples are manually labeled by the manufacturer, including normal connections and two types of nonconforming threaded pipe connections. The dataset has a substantial class imbalance, with 599 samples for normal connections, 28 samples for one defect type, and 31 samples for another defect type. As shown in Figure 7, each sample is characterized by multichannel functional data, with varying-length observations during the thread engagement phase of the manufacturing process.

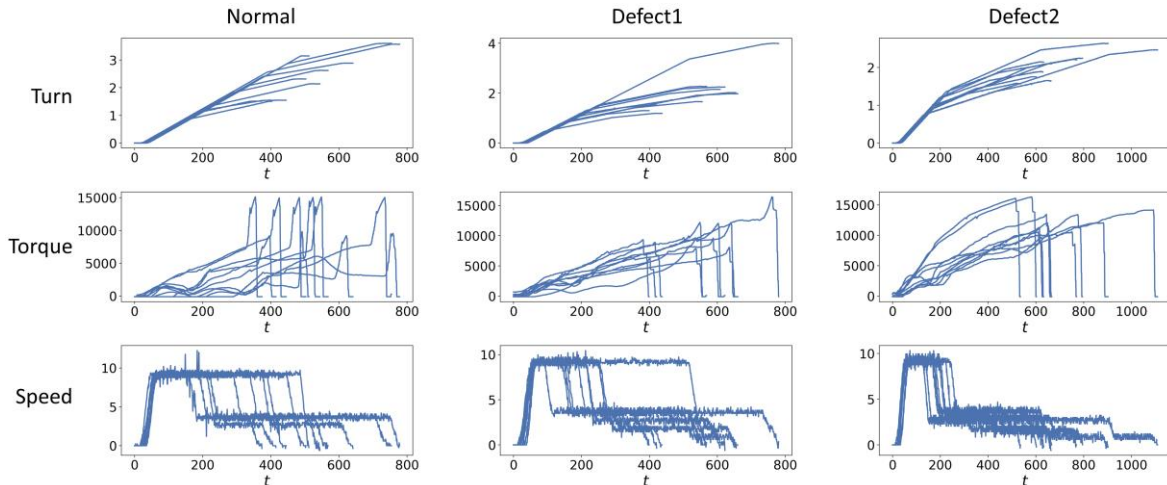


Figure 7. Multichannel functions for each label in the steel pipe dataset.

To configure our framework, we first select the basis functions for the knowledge-infused padding introduced in Section 4.2.1. Taking the torque function in the manufacturing process as an example, we

choose the Fourier basis because the missing thread engagement process exhibits a periodic pattern. The results obtained using the Fourier basis and other traditional padding methods are shown in Figure 8. Compared with traditional padding methods, our knowledge-infused padding does not affect the distribution of the original functions. Similarly, we select the monomial basis for the turns and tightening speed functions in the manufacturing process since both the turns and tightening speed during the thread engagement process can be approximated with polynomials.

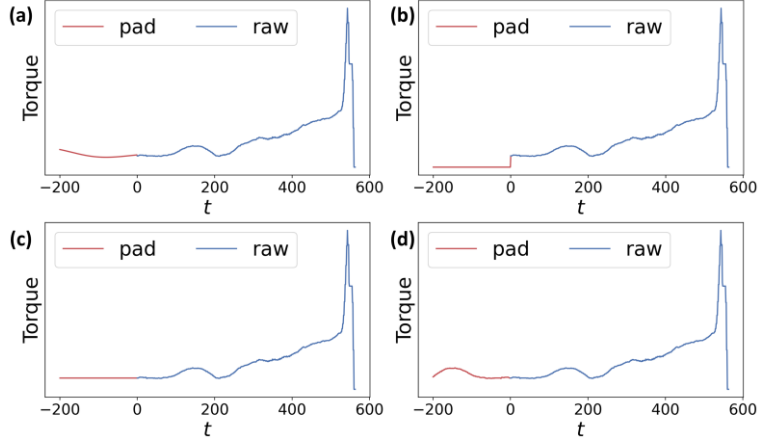


Figure 8. Padding results obtained using different padding methods. (a) Knowledge-infused padding; (b) Zero padding; (c) Replication padding; (d) Reflective padding.

The other hyperparameters are set as follows: we set the number of convolution layers $D = 5$ and number of bases $C_1 = 40$, following the guidelines outlined in Section 4.4. The number of channels for other parts of the neural network, denoted as $C_2 = 160, C_3 = 320$, is consistent with the values used in (Yue et al., 2022). Average pooling kernels of size $\{1,3,5\}$ are used in Q .

To demonstrate the efficacy of our framework when dealing with incomplete and imbalanced functional data, we visualize the representations of functional data, as shown in Figure 9. The fixed-dimensional representations are initially obtained from the multichannel and incomplete functions using functional neural networks, and are then visualized using the t-SNE technique (Van der Maaten and Hinton, 2008). We present the results from both the untrained network and the network after training to demonstrate the impact of representation learning. Figure 9 (a) illustrates that representations from different functional data samples are mixed in the feature space when using an untrained network. However, after representation learning through training with the contrastive loss in Section 4.3, the representations, as depicted in Figure 9 (b), become distinctly separated in the feature space. The

separation facilitates subsequent classification, demonstrating that our framework enables the accurate classification of incomplete and imbalanced functional data.

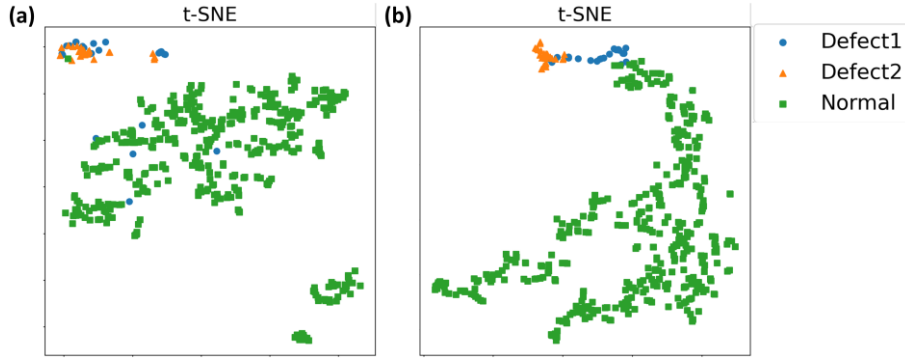


Figure 9. Visualization of the representations with t-SNE. (a) Representations from the untrained functional neural network. (b) Representations from the functional neural network after training.

For comparison, we select three benchmark methods, each representing one of the categories introduced in Section 2.2. The baseline method for classifying the functional data is DTW combined with a k-nearest neighbor (KNN) classifier. Additionally, we utilize POIFD (Elías et al., 2022), a benchmark method for evaluating functional depth, which considers partial observability. The functional depth needs to be combined with depth-based classifiers (Cuesta-Albertos *et al.*, 2017; Li *et al.*, 2012) to classify functional data. To apply POIFD to our dataset, we use the R package proposed in (Elías et al., 2022) to calculate depth measures and implement a depth-based classifier following (Cuesta-Albertos et al., 2017). Sparse functional linear discriminant analysis (SFLDA) (Park et al., 2022) is chosen as a representative low-dimensional representation-based method, and its implementation is based on the open code available in (Park et al., 2022). To ensure fair comparisons, we adopt the same SVM classifier as the backbone for classifier training in both our framework and the depth-based classifier using POIFD.

To evaluate the performance, we employ balanced accuracy and macro-F1 as evaluation metrics, which are commonly used for imbalanced classification. We employ a 5-fold cross-validation on the threaded pipe connection dataset. In each fold, we randomly select 25% samples from the training set as the validation set. The average results of 5-fold cross-validation using our framework and competing methods are presented in Table 1. As shown in Table 1, our proposed framework outperforms the competing methods in terms of both balanced accuracy and macro-F1 metrics. To provide a more detailed analysis of the classification accuracy, we present the recall rates for each label obtained from

5-fold cross-validation in Table 2. Notably, all methods perform well in classifying normal samples. However, the distinguishing factor lies in the recall rates of defect labels. Our framework exhibits higher recall rates for minority labels than other methods, emphasizing its superiority in handling imbalanced datasets and effectively identifying nonconforming threaded pipe connections in the manufacturing process. The detailed classification results of each fold are shown in Appendix A.3.

Table 1. Comparison of performance on the threaded pipe connection dataset

Framework	Balanced Accuracy	Macro-F1
DTW	0.750	0.744
POIFD	0.732	0.745
SFLDA	0.816	0.814
DeepFunction	0.894	0.907

Table 2. Comparison of recall rates for each label

Framework	Label	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5
DTW	Defect 1	0.50	0.50	0.83	0.80	0.60
	Defect 2	0.50	0.83	0.67	0.71	0.33
	Normal	1.00	0.99	1.00	1.00	0.99
POIFD	Defect 1	0.50	0.50	0.17	0.40	0.40
	Defect 2	0.83	1.00	0.50	0.86	0.83
	Normal	0.99	1.00	1.00	1.00	1.00
SFLDA	Defect 1	0.50	0.17	0.50	1.00	0.60
	Defect 2	1.00	0.83	1.00	1.00	0.67
	Normal	1.00	0.98	1.00	1.00	1.00
DeepFunction	Defect 1	0.83	0.67	0.67	0.80	0.60
	Defect 2	1.00	1.00	1.00	0.86	1.00
	Normal	1.00	0.99	1.00	1.00	1.00

6. Conclusion

This study introduces a novel deep metric learning framework tailored for the classification of imbalanced, multichannel, and incomplete functional data within the context of manufacturing processes. While prior research on classifying multichannel and incomplete functional data exists, addressing the challenge of imbalanced classification has not been adequately explored within the realm of functional data analysis. To facilitate an imbalanced classification, we propose a novel functional

neural network that encodes multichannel and incomplete functional data, allowing it to be trained effectively on highly imbalanced datasets. Furthermore, the padding mechanism and contrastive loss associated with the functional neural network are specifically tailored to address the characteristics of functional data derived from manufacturing processes. We also provide comprehensive guidelines for tuning the hyperparameters of the framework to ensure optimal performance.

To validate the effectiveness of our framework, we apply it to a real dataset derived from the manufacture of threaded pipe connections. The results demonstrate that our framework outperforms existing benchmark methods designed for incomplete functional data. This validation underscores the utility of our framework in achieving the accurate identification and classification of nonconforming threaded pipe connections in industrial manufacturing processes.

Acknowledgements

The authors acknowledge the generous support from the National Natural Science Foundation of China (No. 72001139 and No. 72371219), Guangdong Basic and Applied Basic Research Foundation (No. 2023A1515011656), and Guangzhou-HKUST(GZ) Joint Funding Program under Grant No. 2023A03J0651.

A. Appendix

A.1 Proofs for the Functional Neural Network $\beta(\cdot)$

In this section, we introduce the details and proofs for constructing the Toeplitz matrix form of $\mathbf{B}_{(c)}$ and $\mathbf{H}_{(c)}$ in the basis layer $\beta(\cdot)$.

The Toeplitz matrix is a square matrix $\mathbf{P}^{N \times N}$ where the values along the negative sloping diagonals are equal. Given the first column $[p_0, \dots, p_{N-1}]^T$ and first row $[p_0, \dots, p_{-N+1}]$, the matrix is in the following form:

$$\mathbf{P}^{N \times N} = \begin{bmatrix} p_0 & p_{-1} & p_{-2} & \cdots & p_{-N+2} & p_{-N+1} \\ p_1 & p_0 & p_{-1} & \cdots & p_{-N+3} & p_{-N+2} \\ p_2 & p_1 & p_0 & \cdots & p_{-N+4} & p_{-N+3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ p_{N-2} & p_{N-3} & p_{N-4} & \cdots & p_0 & p_{-1} \\ p_{N-1} & p_{N-2} & p_{N-3} & \cdots & p_1 & p_0 \end{bmatrix}.$$

For the functional data $\mathbf{x}(t)$, a Toeplitz matrix always exists for convolution weights $\mathbf{h}_{(c)}^{(l)}(t)$ given any length T of the observations $\mathbf{x}(t)$. Suppose the weights $\mathbf{h}_{(c)}^{(l)}(t)$ have a convolution kernel $\mathbf{w}_{(c)}^{(l)} = [w_1, \dots, w_{k_w}]$ with kernel size k_w and dilation size d . The Toeplitz matrix is of size $T \times T$, and the matrix is derived by setting the value of $p_n, n = 1, \dots, T - 1$ as the following:

$$p_n = 0, \forall n \neq r * 2^d, r = 0, \dots, k_w - 1,$$

$$p_{r*2^d} = w_{k_w-r}, r = 0, \dots, k_w - 1.$$

For example, we set the kernel size $k_w = 3$ and the dilation size $d = 1$ for a functional input of length $T = 6$. The Toeplitz matrix is derived from convolution kernel $[w_1, w_2, w_3]$ as:

$$\mathbf{H}_{(c)}^{(l)} = \begin{bmatrix} w_3 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_3 & 0 & 0 & 0 & 0 \\ w_2 & 0 & w_3 & 0 & 0 & 0 \\ 0 & w_2 & 0 & w_3 & 0 & 0 \\ w_1 & 0 & w_2 & 0 & w_3 & 0 \\ 0 & w_1 & 0 & w_2 & 0 & w_3 \end{bmatrix}.$$

It has to be noted that the formula for $\mathbf{h}_{(c)}^{(l)}(t)$ does not consider the padded value of $\mathbf{x}(t)$. Therefore, the Toeplitz matrix of the convolution layer $\mathbf{h}_{(c)}^{(l)}(t)$ is an approximation of the convolution operation. The approximation can be inaccurate when the padded length is considerable under a large dilation size d . Therefore, we apply the partial convolution (Liu *et al.*, 2018) technique to deeper layers to alleviate the effect of padding. In each convolution operation, an element-wise multiplication is performed between the input $\mathbf{u}_{(c)}^{(l-1)}$ and a binary masking layer \mathbf{M} , where the binary mask is zero if the corresponding region is a padded value and the symbol " \odot " denotes the Hadamard product.

$$\mathbf{u}_{(c)}^{(l)} = \mathbf{B}_{(c)}^{(l)}(t) + \mathbf{H}_{(c)}^{(l)} \left(\mathbf{M} \odot \mathbf{u}_{(c)}^{(l-1)} \right) \frac{\|\mathbf{1}\|_1}{\|\mathbf{M}\|_1} \mathbf{u}_{(c)}^{(l-1)}$$

Moreover, we add a causal constraint to the convolution layers to accept only the padded value at the beginning of each input function. The padded values for $\mathbf{x}(t)$ when $t > T$ are not involved because only the thread engagement phase at the beginning of each function is incomplete. Thus, each Toeplitz matrix is of the same size $T \times T$ and can be multiplied to obtain a matrix $\mathbf{H}_{(c)}$ representing all layers.

$$\mathbf{H}_{(c)} = \prod_{l=0}^{D-1} \mathbf{H}_{(c)}^{(l)},$$

$$\langle \boldsymbol{\beta}_{(c)}(t), \mathbf{x}(t) \rangle = \mathbf{B}_{(c)} + \mathbf{H}_{(c)} \mathbf{X}.$$

The $\mathbf{H}_{(c)}$ is also a $T \times T$ matrix and can be adapted to input \mathbf{X} with any length T . $\mathbf{H}_{(c)}$ can be viewed as a matrix basis which is adaptive to the input function $\mathbf{x}(t)$. In this way, the input $\mathbf{x}(t)$,

each channel $\beta_{(c)}(t)$ of $\beta(\cdot)$, and output $\langle \beta_{(c)}(t), \mathbf{x}(t) \rangle$ are all functional data. A total of C_1 functional data will be generated from $\beta(\cdot)$ given the input $\mathbf{x}(t)$.

$$\beta(\mathbf{x}(t)) = [\langle \beta_{(1)}(t), \mathbf{x}(t) \rangle, \dots, \langle \beta_{(C_1)}(t), \mathbf{x}(t) \rangle]$$

Therefore, the basis layer $\beta(\cdot)$ is a functional neural network that can accept and adapt to a functional input of any length without specifying the domain range.

A.2 Visualization of the Learned Bases through Functional Neural Network

The C_1 functional bases $\beta_{(c)}(t)$ in the functional neural network $\beta(\cdot)$ can be visualized. By solving the following equation, we can derive the observations of the functional basis $\beta'_{(c)}$ and visualize the learned basis $\beta_{(c)}(t)$.

$$\beta'_{(c)} \odot \mathbf{X} = \mathbf{H}_{(c)} \mathbf{X}.$$

For example, we select the functional neural network learned in the first fold of the 5-fold cross-validation on the threaded pipe dataset. The $\beta(\cdot)$ consists of $D = 5$ convolution layers with $C_1 = 40$ channels. The Toeplitz matrices $\mathbf{H}_{(c)}$ are derived using Equation (7) and the functional basis $\beta'_{(c)}$ are calculated given the functional input $\mathbf{x}(t)$.

We randomly selection one sample of multichannel functional data and the learned basis is shown in Figure A.2-1.

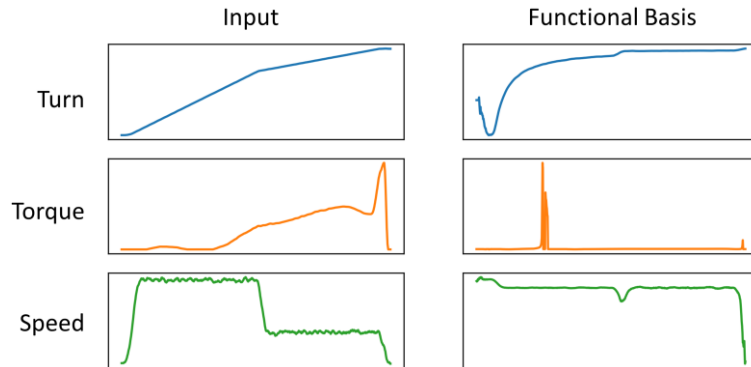


Figure A.2-1. Visualization of the Learned Basis.

Through the weights of the following neural network $g(\cdot)$, we can further identify the most important functional bases. A more straightforward way is to add weights to each channel $\beta_{(c)}(t)$, such as the channel-wise attention layers (Wang *et al.*, 2020).

A.3 Classification Results of Each Fold

The classification results on the steel pipe dataset are shown from Figure A.3-1 to Fig. A.3-4. In each subplot, we plot the confusion matrix of the classification. Each row in the matrix represents the actual label, and each column represents the predicted label.

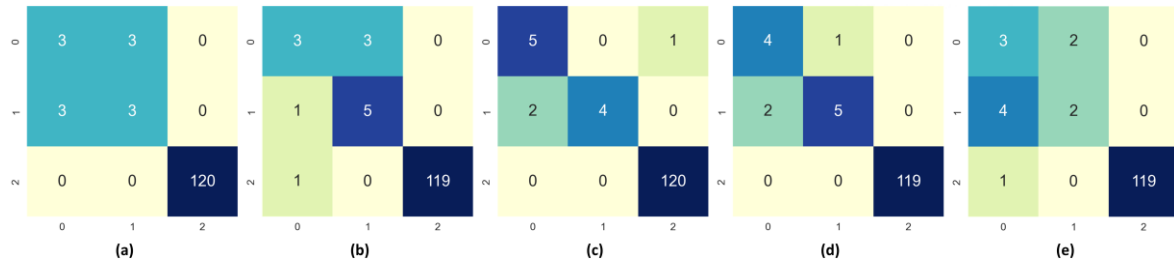


Figure A.3-1. Classification results of DTW. (a) 1st fold; (b) 2nd fold; (c) 3rd fold; (d) 4th fold; (e) 5th fold.

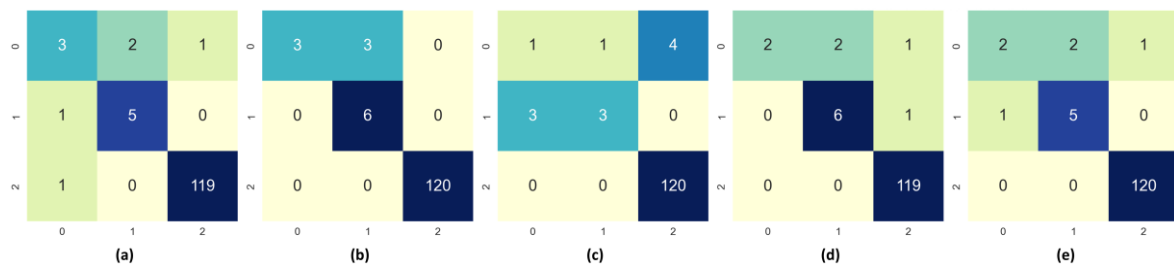


Figure A.3-2. Classification results of POIFD. (a) 1st fold; (b) 2nd fold; (c) 3rd fold; (d) 4th fold; (e) 5th fold.

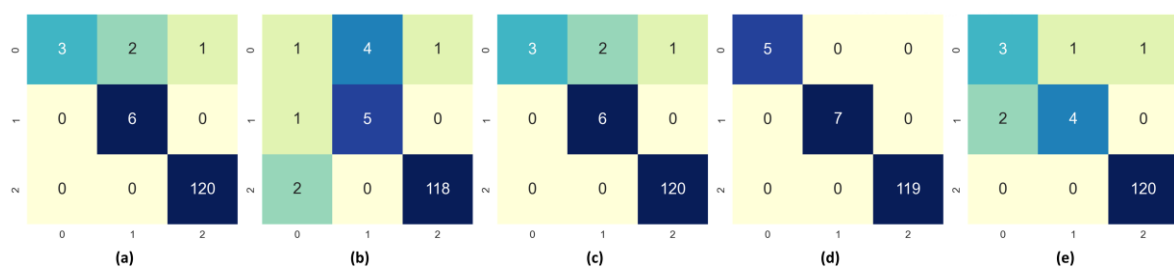


Figure A.3-3. Classification results of SFLDA. (a) 1st fold; (b) 2nd fold; (c) 3rd fold; (d) 4th fold; (e) 5th fold.

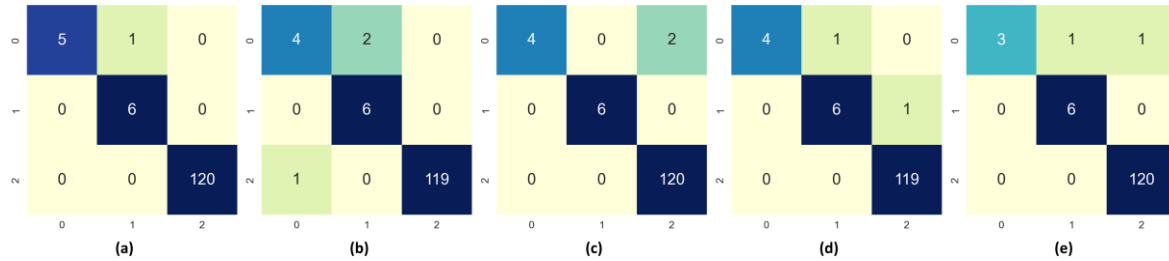


Figure A.3-4. Classification results of DeepFunction. (a) 1st fold; (b) 2nd fold; (c) 3rd fold; (d) 4th fold; (e) 5th fold.

References

- Chawla, N.V., Bowyer, K.W., Hall, L.O. and Kegelmeyer, W.P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, **16**, 321-357.
- Cuesta-Albertos, J.A., Febrero-Bande, M. and Oviedo De La Fuente, M. (2017). The DD G-classifier in the functional setting. *Test*, **26**(1), 119-142.
- Delaigle, A. and Hall, P. (2013). Classification using censored functional data. *Journal of the American Statistical Association*, **108**(504), 1269-1283.
- Du, J., Zhang, X. and Shi, J. (2017). Pairwise critical point detection using torque signals in threaded pipe connection processes. *Journal of Manufacturing Science and Engineering*, **139**(9).
- Elías, A., Jiménez, R., Paganoni, A.M. and Sangalli, L.M. (2022). Integrated depths for partially observed functional data. *Journal of Computational and Graphical Statistics*, 1-12.
- Elkan, C. (2001). The foundations of cost-sensitive learning, in *International Joint Conference on Artificial Intelligence*, pp. 973-978, Lawrence Erlbaum Associates Ltd.
- Franceschi, J.-Y., Dieuleveut, A. and Jaggi, M. (2019). Unsupervised scalable representation learning for multivariate time series. *Advances in Neural Information Processing Systems*, **32**.
- Guangjie, Y., Zhenqiang, Y., Qinghua, W. and Zhentong, T. (2006). Numerical and experimental distribution of temperature and stress fields in API round threaded connection. *Engineering Failure Analysis*, **13**(8), 1275-1284.
- Heinrichs, F., Heim, M. and Weber, C. (2023). Functional Neural Networks: Shift invariant models for functional data with applications to EEG classification, in *International Conference on Machine Learning*, pp. 12866-12881, PMLR.
- Honglin, X., Taihe, S. and Zhi, Z. (2014). Theoretical analysis on makeup torque in tubing and casing premium threaded connections. *Journal of Southwest Petroleum University (Science & Technology Edition)*, **36**(5), 160.
- James, G.M. and Hastie, T.J. (2001). Functional linear discriminant analysis for irregularly sampled curves. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **63**(3), 533-550.
- Khosla, P., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., Maschinot, A., Liu, C. and Krishnan, D. (2020). Supervised contrastive learning. *Advances in Neural Information Processing Systems*, **33**, 18661-18673.
- Kneip, A. and Liebl, D. (2020). On the optimal reconstruction of partially observed functional data. *The Annals of Statistics*, **48**, 1692-1717, 3.

- Kraus, D. (2015). Components and completion of partially observed functional data. *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, 777-801.
- Kraus, D. and Stefanucci, M. (2019). Classification of functional fragments by regularized linear classifiers with domain selection. *Biometrika*, **106**(1), 161-180.
- Li, J., Cuesta-Albertos, J.A. and Liu, R.Y. (2012). DD-classifier: Nonparametric classification procedure based on DD-plot. *Journal of the American Statistical Association*, **107**(498), 737-753.
- Liu, G., Reda, F.A., Shih, K.J., Wang, T.-C., Tao, A. and Catanzaro, B. (2018). Image inpainting for irregular holes using partial convolutions, in *Proceedings of the European Conference on Computer Cision (ECCV)*, pp. 85-100.
- Mayne, R. and Margolis, S. (1982). *Introduction to Engineering*.
- Mou, S., Cao, M., Bai, H., Huang, P., Shi, J. and Shan, J. (2023). PAEDID: Patch Autoencoder-based Deep Image Decomposition for pixel-level defective region segmentation. *IISE Transactions*, 1-15.
- Oord, A.V.D., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. and Kavukcuoglu, K. (2016). Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.
- Park, J., Ahn, J. and Jeon, Y. (2022). Sparse functional linear discriminant analysis. *Biometrika*, **109**(1), 209-226.
- Perdices, D., De Vergara, J.E.L. and Ramos, J. (2021). Deep-FDA: Using functional data analysis and neural networks to characterize network services time series. *IEEE Transactions on Network and Service Management*, **18**(1), 986-999.
- Radford, A., Metz, L. and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.
- Ramsay, J., Ramsay, J. and Silverman, B. (2005). *Functional Data Analysis*. Springer Science & Business Media.
- VAM book. (2023). <https://www.vamservices.com/assets/downloads/VAM%C2%AE%20Book.pdf>
- Van Der Maaten, L. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, **9**(11).
- Wang, P., Han, K., Wei, X.-S., Zhang, L. and Wang, L. (2021). Contrastive learning based hybrid networks for long-tailed image classification, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 943-952.
- Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W. and Hu, Q. (2020). ECA-Net: Efficient channel attention for deep convolutional neural networks, in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11534-11542.
- Wang, Q., Zheng, S., Farahat, A., Serita, S., Saeki, T. and Gupta, C. (2019). Multilayer perceptron for sparse functional data, in *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1-10, IEEE.
- Yao, J., Mueller, J. and Wang, J.-L. (2021). Deep Learning for Functional Data Analysis with Adaptive Basis Layers, in *International Conference on Machine Learning*, pp. 11898-11908, PMLR.
- Yue, Z., Wang, Y., Duan, J., Yang, T., Huang, C., Tong, Y. and Xu, B. (2022). Ts2vec: Towards universal representation of time series, in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 8980-8987.

Zhu, J., Wang, Z., Chen, J., Chen, Y.-P.P. and Jiang, Y.-G. (2022). Balanced contrastive learning for long-tailed visual recognition, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6908-6917.